

UNIVERSITY OF TECHNOLOGY SYDNEY
Faculty of Engineering and Information Technology

**NON-IID RECOMMENDER SYSTEMS:
A MACHINE LEARNING APPROACH**

by

Liang Hu

A THESIS SUBMITTED
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE

Doctor of Philosophy

Sydney, Australia

2018

Certificate of Original Authorship

I, Liang Hu declare that this thesis, submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the Advanced Analytics Institute at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise reference or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This thesis is the result of a research candidature conducted jointly with Shanghai Jiao Tong University as part of a collaborative Doctoral degree.

This document has not been submitted for qualifications at any other academic institution.

This research is supported by an Australian Government Research Training Program Scholarship.

Production Note:
Signature removed
Signature: prior to publication.

Date: **March 28, 2019**

ABSTRACT

NON-IID RECOMMENDER SYSTEMS: A MACHINE LEARNING APPROACH

by

Liang Hu

A recommender system (RS) comprises the core software, tools, and techniques that effectively and efficiently cope with information overload as well as locate information that is genuinely required. As one of the most widely used artificial intelligence (AI) systems, RSs have been integrated into daily life over the past two decades. In recent decade, the machine learning approach has dominated AI research in almost all areas. Therefore, modeling advanced RSs using the machine learning approach forms the basic methodology of this thesis.

Current RSs suffer from many problems, such as data sparsity and cold start, because they fail to consider the non-IIDness in data, which includes the heterogeneities and coupled relations within and between users and items, as well as their interactions. Thus, we propose non-IID recommender systems by modeling the non-IIDness in recommendation data with the machine learning approach. Specifically, we study non-IID RS modeling techniques from three perspectives: users, items, and interactions. This research not only promotes the design of new machine learning models and algorithms in theory, but also extensively influences the evolution of technology and society.

To construct the non-IID RS from a user perspective, we jointly model two aspects: (1) the heterogeneities of users and (2) the coupling between users. Specifically, we study the non-IID user modeling in two representative RSs: (1) a group-based RS (GBRS) and (2) a social network-based RS (SNRS). First, we perform an in-depth analysis of existing GBRSs and demonstrate their deficiencies in modeling

the heterogeneity and coupling between group members for making group decisions. A deep neural network is designed to learn a group preference representation, which jointly considers all members' heterogeneous preferences. Second, we model an SNRS by modeling the influential contexts that embed the influence of relevant users and items, because a user's selection is largely influenced by other users with social relationships.

To construct the non-IID RS from an item perspective, we target two modeling aspects: (1) the heterogeneities of items and (2) the coupling between items. Specifically, we study the non-IID item modeling in two representative RSs: (1) a cross-domain RS (CDRS) and (2) a session-based RS (SBRS). First, existing CDRSs may fail to conduct cross-domain transfer because of domain heterogeneity; thus, we propose an irregular tensor factorization model, which can more effectively capture the coupling between heterogeneous domains with learning the domain factors for each domain. Second, we construct an effective and efficient personalized SBRS to more effectively capture the couplings between items by modeling intra- and inter-session contexts.

To construct the non-IID RS from an interaction perspective, we target two modeling aspects: (1) the heterogeneities of interactions and (2) the coupling between interactions. Specifically, we study the non-IID interaction modeling in two representative RSs: (1) a multi-objective RS (MORS) and (2) an attraction-based RS (ABRS). First, we study an MORS to tackle the challenges of recommendation for users and items in the long tail. Subsequently, a coupled regularization model is proposed to jointly optimize two objectives: the credibility and specialty. Existing content-based RSs can recommend new content according to similarity; however, they are not capable of interpreting the attraction points in user-item interactions. Therefore, to construct an interpretable content-based RS, we propose attraction modeling to learn and track user attractiveness.

In the last section, we summarize the contributions of our work and present the future directions that can improve and extend the non-IID RS.

Dedication

To my parents, my wife, and my son.

Acknowledgements

Firstly, I would like to express my sincere gratitude to my supervisor Prof. Longbing Cao for the continuous support of my Ph.D. study and related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of my research papers and this thesis. His advice on both research as well as on my career have been priceless. Prof. Longbing Cao shows strong motivation, ambition, and dedication in his work and research, which impressed me very much. Moreover, I'd like to thank my co-supervisor Prof. Guandong Xu. With his invitation, I first time came to Australia and visited Advanced Analytics Institute (AAi), University of Technology Sydney. His strong sociability shows me a great example of career development.

I would also like to thank my committee members, Professor Ling Chen, Professor Dong Xu, and Professor Jinyan Li for serving as my committee members even at hardship. I also want to thank you for letting my candidate assessment be an enjoyable moment, and for your brilliant comments and suggestions, thanks to you.

A special thanks to my family. Words cannot express how grateful I am to my mother, and father for all of the sacrifices that you have made on my behalf. Your prayer for me was what sustained me thus far. In the end, I would like express appreciation to my wife who was always my strongest back support. Thanks for her great efforts to and take care of my son. This thesis would not have been possible without their warm love, continued patience, and endless support.

I would like to thank all my lab mates for their continued support. I appreciate the discussion with Shoujin Wang, visiting Ph.D. student Xin Li and Sheng Luo, and visiting Professor Wenpeng Lu. Moreover, I thank all my coauthors in the lab, including Shoujin Wang, Qianqian Chen and Wei Cao. Especially, I'd like to thank Songlei Jian for coauthoring top-rank papers and co-tutoring in top-rank

conferences. I am impressed by her brilliance, diligence, optimism and great comprehension, and we got a lot of inspirations from our discussion. I hope all of them have a happy life and successful career.

Last but not least, there are still many other people that I would like to express my gratitude to them for their contribution during my Ph.D. study.

Liang Hu
Sydney, Australia, 2018.

List of Publications

Journal Papers

- J-1. Hu, L., Chen, Q., Jian, S., Cao, L., and Cao, J. Neural Cross-session Filtering: Next-item Prediction under Intra- and Inter-session Context. *IEEE Intelligent System*, vol. 33, pp. 57-67, 2018.
- J-2. Hu, L., Cao, L., Cao, J., Gu, Z., Xu, G., and Wang, J. Improving the Quality of Recommendations for Users and Items in the Tail of Distribution. *ACM Trans. Inf. Syst.* 35, 3, 1-37, 2017.
- J-3. Hu, L., Cao, L., Cao, J., Gu, Z., Xu, G., and Yang, D. Learning Informative Priors from Heterogeneous Domains to Improve Recommendation in Cold-Start User Domains. *ACM Trans. Inf. Syst.* 35, 2, 1-37, 2016.
- J-4. Hu, L., Cao, J., and Gu, Z. . Service Discovery and Recommendation in Rough Hierarchical Granular Space. *Advanced Science Letters*, 7(1), 135-139, 2012.
- J-5. Yang, D., Guo, J., Wang, Z.-J., Wang, Y., Zhang, J., Hu, L., Yin, J., and Cao, J. FastPM: An approach to pattern matching via distributed stream processing. *Information Sciences* 453, 263-280, 2018.
- J-6. Cao, J., Xu, W., Hu, L., Wang, J. and Li, M.. A social-aware service recommendation approach for mashup creation. *International Journal of Web Services Research (IJWSR)*, 10(1), pp.53-72, 2013.
- J-7. Hu, L., Jian, S., Cao, L., and Chen, Q.. Attraction Modeling: Enhancing Content-based Recommender Systems with Statistical Interpretability. (To be submit to AIJ)

Conference Papers

- C-1. Hu, L., Jian, S., Cao, L., Chen, Q., and Z. Gu, "HERS: Modeling Influential Contexts with Heterogeneous Relations for Sparse and Cold-start Recommendation," in AAAI-19, 2019.
- C-2. Hu, L., Jian, S., Cao, L., and Chen, Q. Interpretable Recommendation via Attraction Modeling: Learning Multilevel Attractiveness over Multimodal Movie Contents. In Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18, 3400-3406, 2018.
- C-3. Hu, L., Cao, L., Wang, S., Xu, G., Cao, J., and Gu, Z. Diversifying Personalized Recommendation with User-session Context. In Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17, 1858–1864, 2017.
- C-4. Hu, L., Cao, W., Cao, J., Xu, G., Cao, L., and Gu, Z. Bayesian Heteroskedastic Choice Modeling on Non-identically Distributed Linkages. In Data Mining (ICDM), 2014 IEEE International Conference on, 851-856, 2014.
- C-5. Hu, L., Cao, J., Xu, G., Cao, L., Gu, Z., and Cao, W. Deep Modeling of Group Preferences for Group-Based Recommendation. In Twenty-Eighth AAAI Conference on Artificial Intelligence, 1861-1867, 2014.
- C-6. Hu, L., Cao, J., Xu, G., Cao, L., Gu, Z. and Zhu, C. Personalized recommendation via cross-domain triadic factorization. In Proceedings of the 22nd international conference on World Wide Web (pp. 595-606). ACM, 2013.
- C-7. Hu, L., Cao, J., Xu, G., Wang, J., Gu, Z. and Cao, L. Cross-domain collaborative filtering via bilinear multilevel analysis. In International Joint Conference on Artificial Intelligence. IJCAI-13, 2013.
- C-8. Hu, L., Cao, J., Xu, G. and Gu, Z.. Latent Informative Links Detection. In KES (pp. 1233-1242), 2012.
- C-9. Hu, L., Cao, J. and Gu, Z.. Service Discovery and Recommendation in Rough Hierarchical Granular Computing. In Services Computing Conference (APSCC), 2010 IEEE Asia-Pacific (pp. 191-198). IEEE, 2010, December.

- C-10. Jian, S., Hu, L., Cao, L., Lu, K., and Gao H., "Evolutionarily Learning Multi-aspect Interactions and Influences from Network Structure and Node Content," in AAAI-19, 2019.
- C-11. Jian, S., Hu, L., Cao, L., and Lu, K. Metric-Based Auto-Instructor for Learning Mixed Data Representation. In AAAI-18, 2018.
- C-12. Wang, S., Hu, L., Cao, L., Huang, X., Lian, D. and Liu, W.. Attention-based transactional context embedding for next-item recommendation. AAAI, 2018.
- C-13. Wang, S., Hu, L., and Cao, L. Perceiving the Next Choice with Comprehensive Transaction Embeddings for Online Recommendation. In Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18-22, 2017.
- C-14. Cao, W., Hu, L., and Cao, L. Deep Modeling Complex Couplings within Financial Markets. In Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, 2518-2524, 2015.
- C-15. Chen, Q., Hu, L., Xu, J., Liu, W., and Cao, L. Document similarity analysis via involving both explicit and implicit semantic couplings. In Data Science and Advanced Analytics (DSAA), 2015. 36678 2015. IEEE International Conference on, 1-10, 2015.
- C-16. Xu, W., Cao, J., Hu, L., Wang, J. and Li, M., June. A social-aware service recommendation approach for mashup creation. In 2013 IEEE 20th International Conference on Web Services (pp. 107-114). IEEE, 2013.
- C-17. You, Y., Huang, G., Cao, J., Chen, E., He, J., Zhang, Y. and Hu, L.. Geam: A general and event-related aspects model for twitter event detection. In International Conference on Web Information Systems Engineering (pp. 319-332). Springer, Berlin, Heidelberg, 2013, October.
- C-18. Shangguan, Q., Hu, L., Cao, J. and Xu, G.. Book Recommendation Based on Joint Multi-relational Model. In Cloud and Green Computing (CGC), 2012 Second International Conference on (pp. 523-530). IEEE, 2012, November.

Contents

Certificate	ii
Abstract	iii
Dedication	v
Acknowledgments	vi
List of Publications	viii
List of Figures	xxi
Abbreviation	xxvi
Notation	xxviii
I Research Background	1
1 Introduction	2
1.1 Background	3
1.1.1 Recommender Systems	4
1.1.2 Recommendation Data Modeling with Machine Learning	8
1.2 Challenges in Modeling RSs	11
1.2.1 Challenges from Data Characteristics	11
1.2.2 Challenges from User Behavior	12
1.2.3 Challenges from Information Coupling	14
1.3 Research Objectives	15
1.3.1 Non-IID Modeling Aspects on Recommendation Data	16

1.3.2	Non-IID RS Modeling	19
1.4	Thesis Organization	28
2	Literature Survey	31
2.1	Basic Recommendation Techniques	31
2.1.1	CF	31
2.1.2	CBF	34
2.1.3	Hybrid CF and CBF	38
2.2	Modeling Implicit Feedback and Long-tail Distribution	40
2.2.1	Implicit Feedback	40
2.2.2	Long Tail	41
2.3	SNRSs	42
2.3.1	Summary of the Challenges to Target in an SNRS	43
2.3.2	Social Media Data	44
2.3.3	Trust Relation Modeling	44
2.3.4	Deep Learning Models	45
2.4	GBRSs	46
2.4.1	Summary of the Challenges to Target in GBRSs	46
2.4.2	Aggregation Approach	46
2.4.3	Deep Learning Approach	48
2.5	CDRSs	48
2.5.1	Summary of the Challenges to Target in CDRSs	49
2.5.2	Latent Factor Models	50
2.5.3	Deep Learning Models	51
2.6	SBRSSs	52

2.6.1	Summary of the Challenges to Target in SBRs	53
2.6.2	Markov Models	53
2.6.3	Deep Learning Models	54
2.7	MORs	55
2.7.1	Summary of the Challenges to Target in MORs	56
2.7.2	Multi-criteria Ratings	56
2.7.3	Multi-objective Ranking	57
3	Preliminaries	60
3.1	Neighborhood Method	60
3.2	Latent Factor Models	60
3.2.1	Matrix Factorization	60
3.2.2	Tensor Factorization	62
3.3	Deep Learning Models	64
3.3.1	Restricted Boltzmann Machines	64
3.3.2	Recurrent Neural Networks	65
3.4	Evaluation Metrics	68
3.4.1	Rating Prediction	68
3.4.2	Ranking Prediction	69
3.4.3	Diversity	70
3.5	Baseline Methods for Experiments	70
3.5.1	Heuristic Methods	71
3.5.2	CF Methods	71
3.5.3	Factorization Machines	71
3.5.4	Content-based Methods	71

3.5.5	Relation-based Method	72
3.5.6	Session Methods	72
3.5.7	Cross-domain Methods	72
II Non-IID RS: Modeling Non-IIDness on Users		73
4	A Group-based Recommender System for Modeling Group Preference over Member Decisions	74
4.1	Introduction	74
4.2	Problem Formulation	76
4.3	Model and Inference	77
4.3.1	Disentangling Collective and Individual Embedding	77
4.3.2	A Comprehensive Representation of Group Preferences	80
4.3.3	Recommendation for a Group	83
4.4	Experiments	83
4.4.1	Data Preparation	84
4.4.2	Comparison Methods	84
4.4.3	Results	85
4.5	Summary of Contributions	89
5	A Social Network-based Recommender Systems for Modeling User and Item Influential Contexts	90
5.1	Introduction	90
5.2	Problem Formulation	93
5.3	Model and Learning	94
5.3.1	Architecture	94

5.3.2	Influential-context Aggregation Unit	95
5.3.3	User’s Influential Context Embedding	96
5.3.4	Item’s Influential Context Embedding	99
5.3.5	User-item Interaction Ranking	100
5.3.6	Training Procedure	101
5.4	Experiments	101
5.4.1	Data Preparation	102
5.4.2	Experimental Settings	103
5.4.3	Recommendation Performance	104
5.4.4	Recommendation for Cold-start Users and Items	106
5.4.5	Visualization and Interpretation	108
5.5	Summary of Contributions	108
III	Non-IID RS: Modeling Non-IIDness on Items	111
6	A Cross-domain Recommender System for Modeling the Couplings over Heterogeneous Item Domains	112
6.1	Introduction	112
6.1.1	Leveraging Cross-Domain Information	113
6.1.2	Modeling Domain Factors	116
6.1.3	Irregular Triadic Relation	116
6.2	Problem Formulation	118
6.2.1	Weighted Regularized Matrix Factorization	118
6.2.2	Learning Priors from Cross-Domain Feedback	119
6.2.3	Learning Posterior on Target Domain for Fine Tuning	120

6.3	Weighted Irregular Tensor Factorization	121
6.3.1	Transformation	121
6.3.2	Parameter Learning	125
6.3.3	Weight Matrix Configuration	128
6.4	Remarks	131
6.4.1	Tricks on Sparse Weight Matrices for Complexity Reduction .	131
6.4.2	Training with Additional Noisy Examples for Improving Generalization	132
6.4.3	Post-Learning	134
6.5	Experiments	135
6.5.1	Comparison Methods	135
6.5.2	Evaluation Metric	137
6.5.3	Rating Prediction on Epinions.com	137
6.5.4	Click Prediction on Tmall.com	144
6.5.5	Time Period as Domain on Movie Rating Prediction	153
6.6	Summary of Contributions	158
7	A Session-based Recommender System for Modeling the Intra- and Inter-sessions Context	161
7.1	Introduction	161
7.2	Problem Formulation	163
7.3	Neural Cross-session Filtering	164
7.3.1	Historical Session Encoding	165
7.3.2	Current Session Encoding	166
7.3.3	Intra- and Inter-session Context Encoding	167

7.3.4	Objective Function	167
7.3.5	Learning and Prediction	168
7.4	Empirical Study	169
7.4.1	Data Preparation	169
7.4.2	Comparison Methods	170
7.4.3	Diversity Evaluation	173
7.5	Summary of Contributions	176

IV Non-IID RS: Modeling Non-IIDness on Interactions177

8	A Multi-objective Recommender System for Modeling Specialty and Credibility for Long-tail Recommendation	178
8.1	Introduction	178
8.1.1	Challenges of Tail Users and Items	179
8.1.2	Optimizing Specialty and Credibility	182
8.2	Preliminaries	185
8.2.1	Reputation Modeling	185
8.2.2	Explicit and Implicit Rating	189
8.3	Model and Learning	190
8.3.1	Overview of RMRM	190
8.3.2	Learning Regularized C-HMF Model	194
8.3.3	Learning Regularized S-HMF Model	203
8.4	Algorithm and Prediction	206
8.5	Discussion	209
8.5.1	Multi-objective Optimization	209

8.5.2	Social Regularization from PoGE Perspective	210
8.6	Experiments	211
8.6.1	A Comparison of the State-of-the-Art Methods	212
8.6.2	Explicit Rating Data Evaluation	213
8.6.3	Implicit Rating Data Evaluation	223
8.7	Summary of Contributions	228
9	Attraction-based Recommender Systems for Capturing and Interpreting User Attraction in Content	231
9.1	Introduction	231
9.2	Attraction Model	234
9.2.1	Content Representation Module	235
9.2.2	Attraction Filter Module	236
9.2.3	Attractiveness Score Module	237
9.3	Hierarchical Attraction Model on Textual Content	238
9.3.1	Multilevel Attraction	239
9.3.2	Bidirectional Gated Recurrent Units	239
9.3.3	Model Architecture	240
9.3.4	Multilevel Attraction Filters	241
9.3.5	Hierarchical Attractive Content Encoder	243
9.3.6	Objectives and Parameter Learning	246
9.4	Multimodal Attraction Model on Movies	247
9.4.1	Multimodal Attraction	249
9.4.2	Model Architecture	249
9.4.3	Story Attraction Module	250

9.4.4	Cast Attraction Module	253
9.4.5	Multimodal Movie Attraction Scoring	254
9.4.6	Objective and Parameter Learning	254
9.5	Experiments	255
9.5.1	Comparison Methods	256
9.5.2	Attraction on Books for Classification	257
9.5.3	Attraction on Academic Papers for Ranking	261
9.5.4	Multimodal Attraction on Movies	266
9.6	Summary of Contributions	272
V	Summary and Prospect	273
10	Conclusion	274
10.1	Non-IID RS Modeling on Users	274
10.1.1	GBRS Modeling: Learning Comprehensive Group Preference Representation	274
10.1.2	SNRS Modeling: Learning Comprehensive Group Preference Representation	275
10.2	Non-IID Recommender Systems Modeling on Items	275
10.2.1	CDRS Modeling: Leveraging Knowledge Across Heterogeneous Domains	275
10.2.2	SBRS Modeling: Modeling Selection with Influential Users and Items	276
10.3	Non-IID RS Modeling on Complex Interaction	277
10.3.1	MORS Modeling: Optimizing Credibility and Novelty for Long-tail Recommendation	277

10.3.2	ABRS Modeling: Capturing and Interpreting Attraction	
	Points in Content	278
11	Open Challenges and Future Directions	279
11.1	New Evaluation Methods and Metrics	279
11.2	SNRSs	280
11.3	GBRSs	281
11.4	CDRSs	282
11.5	SBRSSs	283
11.6	MORSs	284
11.7	ABRSs	285
11.8	Unified and Ubiquitous RSs	286
	Bibliography	288

List of Figures

1.1	Overview of the non-IID RSs targeted in this thesis. Each modeling item for these systems is marked with one or two icons. Each icon corresponds to one non-IID modeling aspect shown in the right-hand section.	20
1.2	Overview of machine learning methods used to model non-IID RSs in this thesis. Each listed item for these systems is marked with one or two icons. Each icon corresponds to one machine learning approach listed in the right-hand section.	21
3.1	The demonstration of CP factorization on a third-order tensor	63
3.2	The structure of restricted Boltzmann machine	65
3.3	Long short-term memory [161]	66
3.4	Gated recurrent unit	67
4.1	Left: Overview of the two-layer collective DBN used to disentangle high-level collective and individual embeddings. Right: More detailed structure of the collective RBM at the top layer where the collective embedding is connected to the member embedding w.r.t. each member.	78
4.2	A dual-wing RBM is placed on the top of DBN, which jointly models the group choices and collective embedding to learn the comprehensive embedding of group preference.	81

4.3	MAP w.r.t. 2-member groups vs. 2 ⁺ -member groups	88
5.1	A multi-relational RS consists of user-user relation, user-item relation and item-item relation. Each user-item interaction is influenced by the user context and the item context.	91
5.2	The architecture of ICE-MRS for modeling user's and item's influential contexts	94
5.3	Influential-context Aggregation Unit (ICAU): A two-stage aggregation model to construct influential context embedding (ICE)	97
5.4	Item recommendation for cold-start users of Delicious and Lastfm.	107
5.5	User recommendation for cold-start items of Delicious and Lastfm.	108
5.6	The visualization of influential contexts of a sampled user selection on an artist in the Lastfm dataset. The artists in item network are labeled by their names and the anonymous users in user network are labeled with IDs. The thickness of edges specifies the significance of influence.	109
6.1	Demonstration of the occurrence of the blinder-transfer issue in CDMF	114
6.2	The feedback from the majority of users in each domain is scant due to the power law distribution, and users have different unfamiliar domains due to differences in interests.	115
6.3	Irregular triadic relation and regular triadic relation.	117
6.4	WITF transforms the slices with heterogeneous domain-specific items into a regular third-order tensor containing an identical virtual item set.	122
6.5	Rating distributions of items on Kids & Family and Hotel & Travel show obvious long tails.	138
6.6	RMSEs comparison over user groups with different number of ratings.	141

6.7	RMSEs comparison over different number of noisy examples for each user.	143
6.8	Mean APs over different values of confidence parameter c	149
6.9	Recall@5~20 of comparison methods on target domain D1 and D2.	151
6.10	Time period as domain to model the heterogeneities of user preferences on movies	154
6.11	MAE comparison with α_k increasing.	157
7.1	The motivation of this work: (a) Classic recommendation without considering session context; (b) Two sequences adding milk and bread into the cart; (c) Rigid-order session vs. relaxed-order session; (d) First-order Markov chain model vs. recurrent model; (e) Next item prediction based on both intra- and inter-session contexts.	162
7.2	The architecture of NCSF. In the left part of split line, historical sessions are used to build inter-session context. In the right part of split line, the items in the current session are used to build intra-session context.	164
7.3	REC@5 - REC@50 on test cases <i>LAST</i> and <i>LOO</i>	173
7.4	F1 scores on test cases <i>LAST</i> and <i>LOO</i>	175
8.1	Items (left) and users (right) are ranked by the number of their ratings (truncated from 0 to 100) on Rich Epinions Dataset; they are both clearly distributed with short heads and long tails.	180
8.2	A recurrent mutual regularization process couples S-HMF and C-HMF using the user and item-factors learned from one another as the empirical priors to couple the objectives <i>specialty</i> and <i>credibility</i>	185

8.3	The screen snapshots of reviews from Amazon.com (left) and Ciao.com (right), where the red, dotted boxes show the helpfulness score.	188
8.4	The graphical representation of the RMRM framework, where S-HMF and C-HMF are recurrently regularized by the empirical priors, induced from one another.	191
8.5	The geometric illustration of the recurrent mutual regularization process, where the estimates of S-HMF and C-HMF are recurrently regularized by the empirical priors induced from one another.	193
8.6	Long-tail distributions for the number of ratings of items and users (truncated from 0 to 500).	214
8.7	The distributions for the number of helpful scores w.r.t. items and users (truncated from 0 to 200).	215
8.8	MAEs of rating prediction for the long-tail item distribution.	217
8.9	MAEs of rating prediction for the long-tail user distribution.	218
8.10	MAEs varying the numbers of involved (a) user trusters, and (b) system experts.	221
8.11	MAEs for head items and tail items with shilling attack.	223
8.12	Long-tail distributions over the number installations w.r.t. users and items (truncated).	224
8.13	Recall@20-50 of tail-item recommendations for users.	227
8.14	Recall@20-50 of tail-user recommendations for users.	228
9.1	Users with different background may have quite different attractive points on an article [199]	232
9.2	Architecture of Attraction Model	235
9.3	The architecture of hierarchical attraction model	241

9.4	The architecture of hierarchical attraction model over movies with two modalities: Cast (left) and Story (right)	250
9.5	Attractiveness visualization on the abstract of <i>Invitation to the Game</i> w.r.t. sentences and words in the most attractive sentences for two users. The larger size and deeper color of font denotes the larger attractiveness weight is assigned.	260
9.6	Recall@5-50 on the Published Paper testing set and the New Paper testing set	264
9.7	Demonstration of the different of group attraction on the abstract of the paper <i>Exploring complex networks</i> . We list the most attractive word in each sentence for each group to illustrate their difference. . .	266
9.8	Demonstration of two exemplary users' most attractive papers in the testing set. We list the most attractive word in each sentence of the recommended papers for interpretation.	266
9.9	R@5-50 on the Released Movies and the New Movies	270
9.10	Two comparative testing samples of <i>User 182</i> : the left movie <i>Wild America</i> obtains a high attraction score because of the cast members in red appear in user's watched movies while the cast members of <i>Bogus</i> never appear in user's movie list.	270
9.11	Statistical attractiveness on movie <i>Election (1999)</i> w.r.t. sentences, words in the most attractive sentences and cast members from the perspectives of User 156 and User 2163. The larger size and deeper color of font denote the larger attractiveness weight is assigned. . . .	271

Abbreviation

RS - Recommender System

AI - Artificial Intelligence

CF - Collaborative Filtering

CBF - Context-based Filtering

NLP - Natural Language Processing

CV - Computer Vision

CNN - Convolutional Neural Networks

RNN - Recurrent Neural Networks

TSA - Time Series Analysis

CDRS - Cross-domain Recommender Systems

SNRS - Social Network-based Recommender Systems

CARS - Context-aware Recommender Systems

MORS - Multi-objective Recommender Systems

SBRS - Session-based Recommender Systems

GBRS - Group-based Recommender Systems

ABRS - Attraction-based Recommender Systems

MF - Matrix Factorization

TF - Tensor Factorization

RBM - Restricted Boltzmann Machines

GRU - Gated Recurrent Units

LSTM - Long Short-Term Memory

MAE - Mean Absolute Error

RMSE - Root Mean Square Error

AP - Average Precision

MRR - Mean Reciprocal Rank

nDCG - Normalized Discounted Cumulative Gain

AUC - Area Under the ROC Curve

Nomenclature and Notation

$(.)^\top$ denotes the transpose operation.

\mathbf{I} is the identity matrix.

\mathbb{R} , \mathbb{R}^+ denote the field of real numbers, and the set of positive reals, respectively.

\mathcal{X} denotes a tensor (boldface script letter)

$\mathcal{X}^{I \times J \times K}$ denotes The dimensionality of each mode

\mathbf{X} denotes a matrix (boldface capital letter)

$\mathbf{X}_{i,:}$ denotes the i th row of matrix \mathbf{X}

$\mathbf{X}_{:,j}$ denotes the j th column of matrix \mathbf{X}

$\mathbf{X}_{i,j}$ denotes the entry (i, j) of matrix \mathbf{X}

\mathbf{x} denotes a vector (boldface lower-case letter)

$\mathbf{X}_{(n)}$ denotes mode- n matricization of a tensor

\otimes denotes Kronecker product

\odot denotes Khatri-Rao product

\cdot^* denotes Hadamard product

$\ ./$ denotes element-wise division

\circ denotes outer product

$\|\mathcal{X}\|$ denotes the norm of a matrix or a tensor \mathcal{X}

$\mathcal{X} = \llbracket \mathbf{A}, \mathbf{B}, \mathbf{C} \rrbracket$ denotes the factorization form of a tensor

$\text{vec}\mathbf{X}$ denotes vectorization of matrix

Part I

Research Background

Chapter 1

Introduction

With the rapid development of Internet technology and artificial intelligence (AI), various information, products, and other resources have become much easier to be published and shared than ever before. Different from the centralized information-publishing mode in the Web 1.0 age, everyone is a source of social media and plays the role of information publisher. As a result, new information, products, and other resources continually appear. The main challenge in this era no longer focuses on how to share and search for information but how to effectively and efficiently locate and extract the information which is really needed.

Researchers have argued that: we are leaving the “Information Age” and entering the “Recommendation Age” [7]. Especially, recommender systems (RSs) are the core software, tools, and techniques providing suggestions in this age. The suggestions provided by a recommender system are aimed at supporting their users in various decision-making processes, such as what items to buy, what music to listen, or what news to read. Recommender systems are valuable tools for online users to cope with information overload and help them make better choices. Recommender systems are now one of the most powerful and popular information discovery tools on the web. Many techniques for recommendation have been proposed, and during the last decade, many of them have also been successfully deployed in commercial environments [184].

1.1 Background

Advanced RSs can improve customers' user experience and make more profits for enterprises. The most well-known companies have tightly integrated an RS into their main business. For example, E-commerce companies such as Amazon (<http://www.amazon.com/>), eBay (<http://www.ebay.com/>), and Alibaba (<http://www.alibaba.com/>); the online social websites like Facebook (<http://www.facebook.com/>), Twitter (<http://www.twitter.org/>), and Weibo (<http://www.weibo.com/>); search service providers such as Google (<http://www.google.com/>), Bing (<http://www.bing.com/>), and Baidu (<http://www.baidu.com/>); and even public services, have paid more attention to RSs. Modeling an RS is a multidisciplinary effort that involves experts from various fields, such as AI, human-computer interaction, data mining, statistics, decision support systems, marketing, and user behavior [183].

Furthermore, the renaissance of AI in the last decade has attracted much attention from every corner of the world. In this golden age of AI, global funds and efforts have been unprecedentedly devoted to AI research. In particular, machine learning approaches [18] have dominated AI research in almost all areas, including natural language processing (NLP), machine translation (MT), computer vision (CV), and game playing. Ever since the world champion players of Go, an ancient strategy game, were beaten by Google's AlphaGo AI in 2017 [198, 200], machine learning has taken center stage. The abovementioned companies have embraced AI especially machine learning techniques and most have evolved into AI-first companies to extend their business. RSs, as one of the most widely used AI systems, have been integrated into daily life for a long time. Moreover, the machine learning approach plays the most significant role in RS evolution.

1.1.1 Recommender Systems

RSs are software, tools, and techniques that provide suggestions for items that are most likely to be of interest to a particular user [184]. A variety of techniques have been proposed as the basis for RS, such as collaborative filtering (CF) and content-based filtering (CBF), as well as knowledge-based and demographic techniques. Therein, knowledge-based RSs require heavy expertise and demographic techniques involve privacy [184]; these factors heavily limit their development and deployment. Therefore, we only introduce CF and CBF, the two most widely used approaches in current RSs.

Collaborative Filtering (CF)

CF [150,206] is the most widely used technique in RSs. Briefly, CF is a method of making automatic predictions (filtering) about the interests of a user by collecting preferences or taste information from many users (collaborating). The underlying assumption of the CF approach is that if person A has the same opinion as person B on a topic, then person A is more likely to have a similar opinion to person B than a randomly chosen person. The research and development company Xerox PARC took the first step in this direction when they incorporated user actions and opinions into a message database and search system called Tapestry [58]. This model is known as pull-active CF because it is the responsibility of the user who desires recommendations to actively pull the recommendations from the database [194]. Resnick et al. [181] designed the GroupLens system, which is mainly used to help readers filter the content in which they are interested; they must then score ratings over this filtered news. The basic assumption is that if a reader is interested in certain topics, she or he will be interested in those topics again in the future. The major difference between Tapestry and GroupLens is that Tapestry filters information according to the trusted users specified by a target user, whereas GroupLens releases

this limitation and can match similar users automatically. Nowadays, CF is widely used in many areas, such as online shopping, spam filtering, and movie recommendation; therefore, the convenience generated by the CF technique can be enjoyed by the general public. In contrast to traditional information retrieval techniques, CF possesses the following advantages:

- **User coupling:** CF can effectively use feedback from similar users to learn the preferences of a target user, which relieves the data insufficiency problem of a single user.
- **Content independence:** CF can detect similarity over heterogeneous products without involving their content (e.g., descriptions, attributes, and images).
- **Serendipity:** CF can find potentially required items that are dissimilar to the content of users' historically chosen items; thus, it may find interests that users were unaware they have.

Although CF methods have been successfully applied to many areas, they still possess some disadvantages, which include [206]:

- **Cold start:** The CF approach often requires a large amount of existing data on a user to make accurate recommendations.
- **Scalability:** In many real-world environments, millions of users and items exist. Thus, large amounts of computational power and storage are often necessary to run recommendations.
- **Sparsity:** The number of items in a real-world RS is often extremely large. The most active users will only have rated a small subset of the overall items. Thus, even the most popular items have very few ratings.

Content-based Filtering (CBF)

Another common type of RS is CBF [43]. CBF is based on the attributes or descriptions of items and a profile of users' preferences. In CBF, the designed algorithms attempt to recommend items that are similar to those that a user has liked in the past (or is examining presently). In particular, various candidate items are compared with items previously selected by the user; the most similar items are recommended. Basically, these methods use an item profile that characterizes the item within the system. The system creates a content-based profile of users based on selected item profiles. Machine learning techniques such as Bayesian classifiers, cluster analysis, decision trees, and artificial neural networks can be employed to estimate the probability of how a user tends to like an item. The CBF approach is often used in areas with rich content features, such as news recommendation. Pandora Radio is a popular example of a content-based RS that plays music with similar characteristics to that of a song provided by the user as an initial seed. Moreover, numerous content-based RSs are aimed at providing movie recommendations, including Rotten Tomatoes and Internet Movie Database. When compared with CF techniques, CBF possesses the following advantages [43]:

- **User Independence:** CF methods require ratings from other users to locate the nearest neighbors (i.e., users deemed to have similar tastes because they rated the same items similarly). Then, only the items that are most liked by the neighbors of the active user are recommended. By contrast, CBF solely exploits feedback provided by target users to build the user profile.
- **Interpretability:** Interpretations of how RSs make recommendations can be provided by explicitly listing content features or descriptions that resulted in an item appearing in the recommendation list. These features are indicators for deciding whether to trust a recommendation. By contrast, CF-based RSs

are black boxes, because the only explanation for an item recommendation is that unknown users with similar tastes have liked that item.

- **New item:** : CBF methods are capable of recommending items not yet rated by any user. Consequently, they do not suffer from the aforementioned cold start problem, which affects the CF approach because CF relies solely on users' preferences for making recommendations. Therefore, until an item is rated by a substantial number of users, CF methods would not be able to recommend it.

Nonetheless, CBF approach also has several disadvantages [43]:

- **Limited content analysis:** CBF has a natural limit to the number and type of features that are associated, whether automatically or manually, with the objects they recommend [138]. Domain knowledge is often required; for example, for movie recommendations, the system must know the actors and directors. No content-based RS can provide suitable suggestions if the analyzed content does not contain enough information to discriminate items that users like or do not. Some representations capture only certain aspects of the content, but there are many others that would influence a user's experience. For instance, often, not enough information exists in the word frequency to model user interests in jokes or poems, whereas techniques for affective computing would be most appropriate. Again, for Web pages, feature extraction techniques from text completely ignore aesthetic qualities and additional multimedia information.
- **Over-specialization:** CBF has no inherent method for finding something unexpected. As a result, CBF-based RSs tend to suggest items that are highly matched against the user profile; therefore, users will have items recommended to them that are similar to those already rated. This drawback is also called a

lack of serendipity, which highlights the tendency of content-based systems to produce recommendations with a limited degree of novelty. CBF rarely finds anything novel, which limits its range of useful applications.

- **New user:** CBF requires sufficient user feedback before it can genuinely understand user preferences and provide accurate recommendations. Therefore, when no user feedback is available, a CBF-based RS cannot provide reliable recommendations for new users.

1.1.2 Recommendation Data Modeling with Machine Learning

RSs are information processing systems that actively gather various types of data to build recommendations. Data are primarily on the items to be recommended and the users who will receive these recommendations. However, because the data and knowledge sources available for RSs can be highly diverse, whether they can ultimately be exploited depends on the recommendation technique [184]. Almost all current machine learning methods are built on data, and the quality of parameter estimation is largely dependent on the amount and structure of the data. This is why the machine learning-based AI approach achieves tremendous success in the age of big data. An RS is a typical AI application built on various types of data, including ratings, attributes, text, images, networks, videos, audio, and sequences. Numerous corresponding machine learning methods are available for managing these data types to model an RS:

- **Rating:** Ratings are the most elementary data in RSs. Explicit ratings, such as five-star ratings, express user preferences over products with different levels, which are the most frequently used data in the CF approach. However, explicit ratings are not always available, whereas implicit ratings such as click logs and dwell time, are the more easily obtained. For instance, an RS may consider navigating to a particular product page an implicit sign of preference for the

items shown on that page. Explicit ratings are often viewed as numeric values or categorical values that can be modeled using bilinear models, such as matrix factorization (MF) [117], and neural models, such as a restricted Boltzmann machine (RBM) [189]. Implicit ratings do not contain explicit preference levels; thus, they are often modeled using order relation [178]. Machine learning methods, including learning to rank [23], and heteroscedastic models [89], have been employed to model implicit ratings.

- **Attributive data:** Attributes are the most frequently used features in data analytics. Item attributes such as price and weight, and user attributes such as age, sex, and occupation, can be used to construct a content-based and demographic RS. Machine learning methods, such as linear models [148], factor analysis [101], and clustering [237], are available to manage these attribute data. Moreover, attributes often consist of numerical and categorical data; therefore, we proposed more advanced neural models [98] to manage such mixed data.
- **Textual data:** Reviews, comments, and descriptions are typical textual data in RSs. CBF and sentimental analysis are two representative techniques that consume these textual data. In machine learning, topic models such as latent semantic analysis (LSA) [120] and latent Dirichlet allocation (LDA) [19], and more word embedding models such as continuous bag-of-words (CBOW) [152] and skip-gram [153], are powerful tools for modeling and analyzing textual data. Moreover, most current NLP approaches are based on machine learning models, which can be employed to manage these textual data.
- **Image data:** Pictures of an item are the most straightforward element used to attract users. Visual sentimental analysis [245] has become an emerging research topic for analyzing user preferences. However, understanding the

semantics of an image was not an easy task a decade ago. With the rise of deep learning, machines have achieved comparable performance against humans in some specific tasks, such as the classification task on ImageNet [118]. Therein, convolutional neural network (CNN)-based models have dominated the areas of image processing and CV. In particular, VGGNet [201] and ResNet [71] are two of the most representative CNN models of recent years.

- **Relational data:** RSs have a primary relation over users and items. Additional relations, including social relation, trust relation, and item relation, are often incorporated into RSs to manage the cold-start problem and improve recommendation accuracy. For example, social RSs [67] aim to present the most relevant and attractive data to the target users by coping with their social relationships. The machine learning methods for relational data have been well studied, including relational learning [56], random walk [51], and network representation learning [68].
- **Sequential data:** User selections in a specific time period are often not independent. For example, a user tends to select different but relevant products in a transaction. As a result, session-based CF aims to model the dependency and transition over the item sequence within the session context, and then calculates the scores for potential next items to generate the rank. Time series analysis (TSA) [21] is the most classic method of modeling sequential data. Markov chain (MC) is another straightforward method of modeling the first-order transition over sequential data [32]. In deep learning, recurrent neural networks (RNNs) are the most effective method of modeling sequential data. Recently, an RNN was successfully applied for session-based recommendation [76].

Obviously, RSs are tightly coupled with all types of data, and the machine learning

approach has become the primary technique for modeling RSs in this recommendation age driven by the exploitation of complex data.

1.2 Challenges in Modeling RSs

To develop the research on existing and emerging RSs, numerous challenges must be confronted. In this thesis, we present these challenges from three aspects: (1) data characteristics; (2) user behavior; and (3) information coupling.

1.2.1 Challenges from Data Characteristics

Because RSs are a type of AI system, data are fundamental for making an RS learn user preferences and item features. Therefore, the quality of data has a direct impact on the recommendation performance. However, several typical challenges exist that must be addressed in real-world recommendation data.

- **Sparsity and imbalance:** In real-world RSs, user data often follow a power-law distribution (i.e., a short head and long tail); that is, the majority of users do not possess sufficient data for their preferences to be accurately learned. Many commercial RSs are based on large datasets. As a result, the user-item matrix used can be extremely large and sparse, which generates challenges in the recommendation. Similarly, items have the same problem. The majority of items do not receive sufficient feedback from users before they can be recommended to users with similar tastes.
- **Cold start:** Long-tail distributions imply that the majority of users and items are cold-start in nature. Cold-start users usually have provided little feedback or sometimes none at all. For a cold-start user, neighborhood-based methods do not have sufficient data to find suitable neighbors. For a model-based method such as MF, long-tail users have provided very little feedback on

popular items; therefore, the learned factors of these users tend to be similar; for this reason, they are not able to clearly represent personal preferences. Similarly, cold-start items usually receive little feedback or sometimes none at all. As a result, the item profile cannot be learned well from user feedback.

- **Heterogeneity:** In RSs, items may belong to different categories and fields, and thus they often have different features. As a result, they cannot be modeled with the same feature specification or distribution. Similarly, user behavior possesses heterogeneity because of the differences in age, sex, occupation, and characteristics. Most current RSs are modeled under an identical distribution assumption over all items and users, which often fails to represent the heterogeneity over users, items, and domains. As a result, they often lead to poor recommendation effects. Therefore, a more effective modeling method must be sought to represent heterogeneous information.
- **Scalability:** As the increase in feedback from old users and continually joining users and items, the data in an RS becomes increasingly large. Traditional RSs seldom respond in real time; therefore, how to quickly and efficiently deal with these data in a commercial RS is a challenge. Thus, one of the key elements for a practical application is how to implement an efficient algorithm that can significantly reduce time or space complexity and work in a parallel mode.

1.2.2 Challenges from User Behavior

In RSs, users are the ultimate targets to provide recommendations to. In contrast to items, users possess subjective feelings and complex social relationships, which lead to quite different user behaviors. Because users are one of the core elements in an RS, we list the typical challenges generated by user behavior as follows.

- **Social behavior:** In a society, people are often influenced by various social

factors when making decisions. Therefore, social information is extremely helpful for improving the recommendation results and relieving the data sparsity and cold-start problems. In addition, people are constantly engaged in social activities (online or offline), because social activities usually involve multiple people, and each person's preferences are different; therefore, how to rationally represent group preferences is one of the great challenges in designing group-based RSs.

- **Malicious attack:** Because of the open environment of the Internet, RSs must confront false information and malicious attacks. For example, a shilling attack refers to a group of spam users intentionally providing fake feedback (e.g., much higher or lower ratings than a true rating to bias the ratings and recommendations for them). In particular, the information of tail items in a long-tail distribution is limited by the lack of feedback from users. As a result, they suffer attacks much more easily from a few fake ratings. Therefore, designing RSs that are robust to malicious attacks is an inevitable challenge for real applications.
- **Privacy breach:** One approach to enhance the modeling of user profiles is to integrate user demographics, such as age, sex, occupation, and income. However, this engenders certain risks that may limit the uptake of an RS, one of which is a privacy breach. Privacy risk is mainly caused by the need for an RS to collect and store personal information about their users. Indeed, to provide personalized recommendations, an RS must possess information about its users encapsulated in user models. This information serves as the basis for generating recommendations, and generally, the quality of the recommendations is correlated with the amount, richness, and freshness of the underlying user data. Therefore, the sensitivity of user information and missing information

are inevitable challenges in designing an RS.

1.2.3 Challenges from Information Coupling

Traditional RSs only consider a single type of data, such as ratings. However, a single type often cannot provide sufficient information, as presented in the challenges of data characteristics. To construct more advanced RSs, multiple types of information from multiple data sources must be coupled to obtain more comprehensive knowledge. Some typical challenges of information coupling are listed as follows.

- **Cross-domain sharing:** A key concern with RS modeling is whether the system is able to learn user preferences from users' actions regarding one domain and use them across other domains. When the system is limited to recommending items in the domain that the user is already involved in, the value from the RS is significantly less than recommending items from other domains that the user is not involved in. For example, recommending news articles based on news browsing is useful, but would be much more useful if music, videos, products, discussions, and other sources from various services can be recommended based on news browsing habits. How to share information between domains and deal with the heterogeneity between domains are the key challenges.
- **Context awareness:** Traditionally, RSs only deal with two types of entity, users and items, and do not put them into a context when generating recommendations. These RSs focus on recommending the most relevant items to individual users and do not consider any contextual information, such as time, place, and the company of other people (e.g., for watching movies or dining out). However, in many applications, it is also crucial to incorporate the contextual information into the recommendation process to recommend items to users under certain circumstances. For example, on weekdays, a user might

prefer to read world news in the morning and the stock market report in the evening, and on weekends she or he may prefer to read movie reviews. The key challenge is how to precisely capture user interactions in various contexts.

- **Multi-information integration:** Users have different points of interest and the hotspots of Internet activity are dissimilar; thus, the amount of data generated in a variety of activities also differs greatly. In addition, users tend to be involved in multiple systems across multiple areas, such as online shopping, real estate, stocks, and games. Therefore, how to effectively connect these distributed data sources and integrate multi-information from multiple sources and multiple systems are the key challenges.
- **Multi-criteria consideration:** Instead of developing recommendation techniques based on a single criterion, such as the overall preference of a user for an item, multi-criteria systems attempt to predict a rating by exploiting preference information with multiple criteria that affect the overall preference value. The additional information provided by multiple criteria could help to improve the quality of recommendations, because it would be able to represent more complex preferences of each user. Increasing numbers of researchers propose building RSs with multi-criteria considerations. The main challenges are how to jointly model multiple relevant but inconsistent objectives as well as how to find the optimal multi-objective solution for the best recommendation results.

1.3 Research Objectives

This thesis is mainly devoted to modeling non-IID recommender systems using machine learning approaches, including the study of how to deal with the aforementioned critical problems and challenges in current RSs, as well as the design of an advanced RS with non-IID learning techniques. This research does not only promote

the design of new models and algorithms in theory, but also improves user experience and profit models, and even extensively influences the evolution of technology and society.

More specifically, this thesis focuses on studying the non-IIDness from multiple aspects of recommendation problems, including the non-IID modeling heterogeneity and coupling on users, items, and their interactions. Moreover, some prototypes of non-IID RSs are constructed to assess the feasibility, effectiveness, and performance.

1.3.1 Non-IID Modeling Aspects on Recommendation Data

In the literature, non-IID data has two forms : not identically distributed and not independently distributed. The first form generally refers to the *heterogeneity* in complex data. The second refers to the *coupling* relationships between and within values, attributes, and objects. Therefore, to study non-IID RSs, we first present the non-IID modeling aspects of recommendation data.

Heterogeneity

Heterogeneity is built into various aspects; neither users or items are identically distributed. The following list explores research scenarios of the heterogeneity of users and items as well as the heterogeneity between users and items [31].

- **Heterogeneity of users:** Each user shares her or his own attributes, characteristics, preferences, behaviors, and intents in the ratings. Simply treating all users as identically distributed may cause failure in understanding the personalized characteristics of each user, and her or his personalized demand and intent in the recommendation.
- **Heterogeneity of items:** One item differs from another in terms of type, attributes, categories, domains, and so forth. Specific item characteristics form various attractive points to different users and user ratings.

- **Heterogeneity of user/item attributes:** Each user and item attribute is different. Each user attribute describes one respective aspect of user demographics, characteristics, group, preference, behavior, and intent. Similarly, each item attribute draws a picture of a respective aspect of item categories, types, characteristics, domains, and so on. Each user/item attribute is not identically distributed; it follows its own distribution, and thus, must be handled accordingly.
- **Heterogeneity between users and items:** Users are highly different from items, and cannot be assumed to follow the same distributions as usually assumed in link prediction. Assuming that they adopt similar latent matrices or can be modeled in the same manner fails to capture the specific features of users and items.

Many existing methods do not consider the abovementioned discussions about heterogeneity in their recommendations, which may result in meaningless outcomes and misleading recommendations. Moreover, it would not be possible to provide truly personalized recommendations when the personalized characteristics are ignored in the modeling.

Coupling

Heterogeneity modeling is a critical step forward in recommendation research to capture the characteristics and complexities in RSs. Another critical matter is capturing the explicit and implicit coupling relationships. Here, couplings refer to any relationships or interactions that connect two or more aspects, which could be between inputs or between inputs and outputs [30]. Couplings in recommendation problems represent implicit or explicit connections between users, between items, and between users and items, for any reason or in any respect [31].

- **User-user couplings:** This refers to the couplings both within and between users, which are further embodied through: (1) intra-user attribute couplings, showing the relationships between the values of a user attribute, such as couplings between user preferences, groups, domains, behaviors, or social relationships; (2) inter-user attribute couplings, showing the connections between user attributes, such as user ages and their positions; and (3) user couplings between users or between user groups.
- **Item-item Couplings:** Similar to user-user couplings, these are couplings within and between items that consist of: (1) intra-item attribute couplings, (2) inter-item attribute couplings, and (3) item couplings between items or between item categories/domains.
- **User-item couplings:** These refer to the couplings within and between user-item pairs or clusters, which are embodied through the following aspects: (1) explicit user-item couplings indicated by user ratings and comments on items; and (2) implicit user-item couplings, showing the influence of or connections between a user's attributes and the user-rated item attributes.
- **Hierarchical Couplings:** In addition to the abovementioned types (aspects) [30] of couplings, couplings are often presented in terms of certain hierarchies. Hierarchical couplings exist in attribute values, attributes (for both users and items), objects (users and items), and object groups (user groups or item categories).

This thesis studies the modeling of users and items, and the tight connection between the ratings given by a user to an item and the characteristics of users and items. This involves an in-depth understanding of the nature of recommendation data; that is, the heterogeneity and couplings.

1.3.2 Non-IID RS Modeling

As previously described, many classic RSs experience significant challenges, whereas few breakthroughs and systematic innovations under an IID assumption have been made. This thesis aims to design a new generation of RSs by modeling the non-IIDness over users, items, and their interactions. In particular, this thesis employs machine learning approaches to the non-IID modeling.

Figure 1.1 depicts the six representative RSs, namely the group-based RS (GBRS), social network-based RS (SNRS), cross-domain RS (CDRS), session-based RS (SBRs), multi-objective RS (MORS), and attraction-based RS (ABRS), to study the non-IID modeling from the abovementioned three perspectives. To model these non-IID RSs, we employ various machine learning techniques. For each non-IID RS, we list the main modeling items with reference to the non-IID modeling aspects, as presented in Section 1.3.1.

As illustrated in Figure 1.2, this thesis applies the machine learning approach to modeling the abovementioned non-IID RSs, where we list the key components for each non-IID RS. The underlying machine learning techniques used for modeling these components are listed in the right-hand part of the diagram.

Modeling Non-IID RSs from a User Perspective

Users are one of the most elementary modeling targets in RSs. Modeling non-IIDness on users aims to capture interactions, connections, and influences between users in terms of users, user attribute values, user attributes, and user groupings. In this thesis, we study two representative non-IID RSs that mainly consider the non-IID modeling from a user perspective, namely a GBRS and an SNRS.

GBRS Because of the social nature of human beings, various group activities are observed throughout life, such as seeing a movie or planning a trip. Recently, the RS

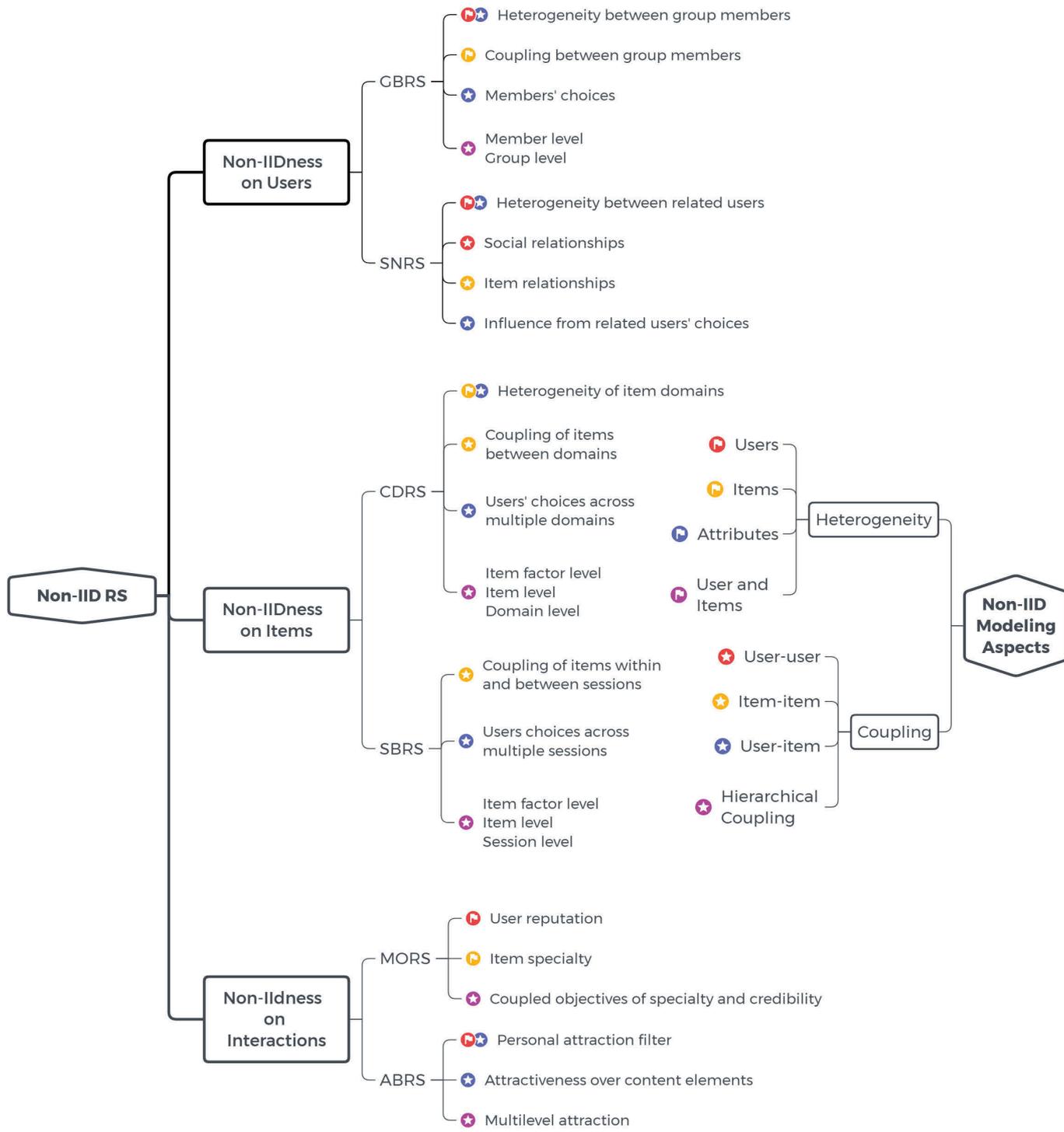


Figure 1.1 : Overview of the non-IID RSs targeted in this thesis. Each modeling item for these systems is marked with one or two icons. Each icon corresponds to one non-IID modeling aspect shown in the right-hand section.

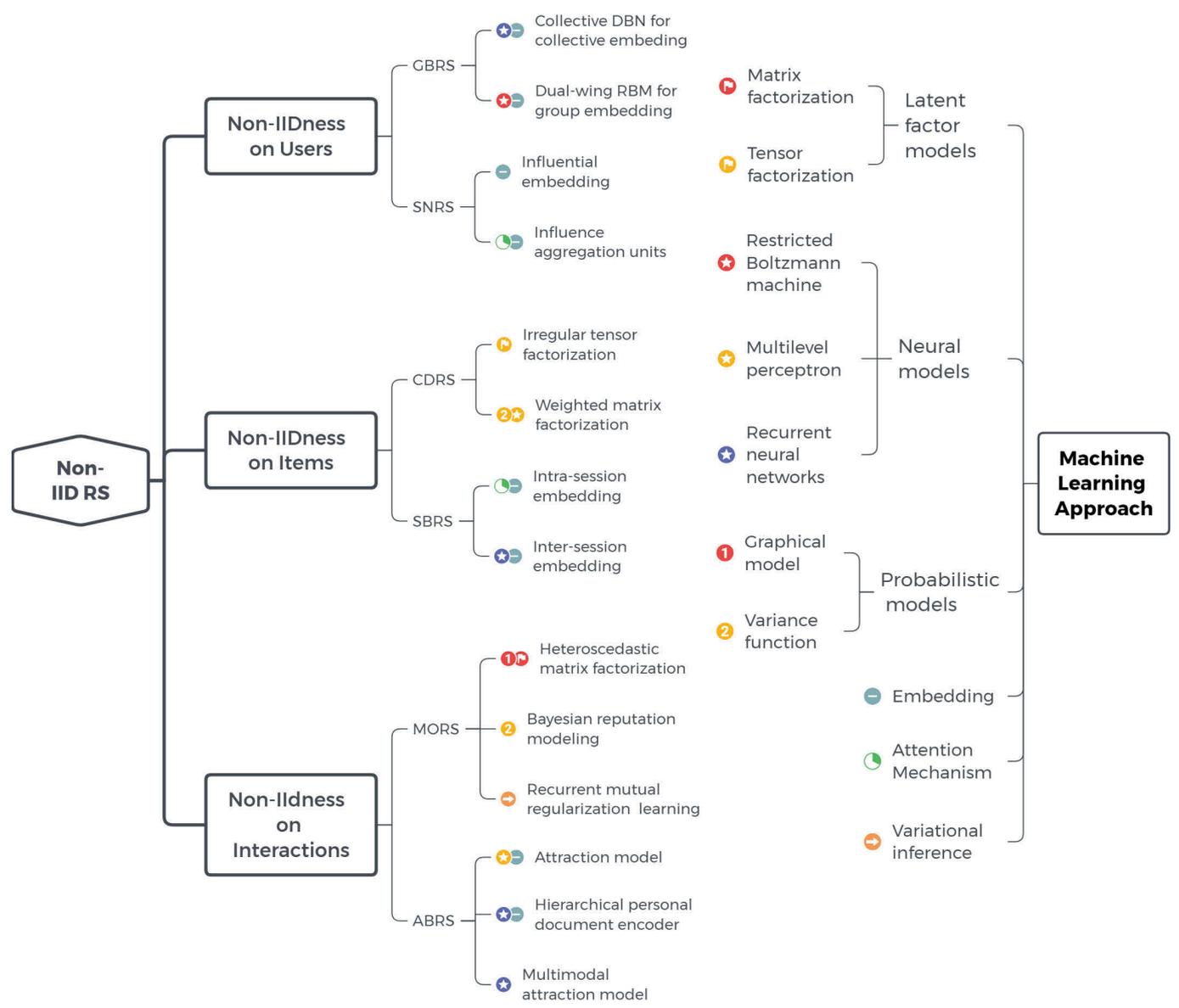


Figure 1.2 : Overview of machine learning methods used to model non-IID RSs in this thesis. Each listed item for these systems is marked with one or two icons. Each icon corresponds to one machine learning approach listed in the right-hand section.

community has begun to study group behavior to make group recommendations [95]. In particular, each member of a group often has different opinions on the same items; therefore, the main challenge with a GBRS is to satisfy most group members with diverse preferences. Obviously, this cannot be achieved through individual-based recommendation methods. The mainstream approaches of GBRS attempt to aggregate group information from individual user models [146]. These methods can be classified into two types of model that are differentiated by the timing of data aggregation. The first is called group preference aggregation (GPA), which first aggregates all members' ratings into a group profile, and then any individual-based CF approach can be used if it regards groups as multiple virtual individual users. By contrast, the second type of model is called individual preference aggregation (IPA), which first predicts the individual ratings over candidate items, and then aggregates the predicted ratings of members within a group using predefined strategies to represent group ratings.

In this thesis, we provide an in-depth analysis of existing GBRSs and show their deficiencies in integrating individual ratings or models. The key concern is that current GBRSs still model each group member independently instead of modeling the coupling between them leading to the group selection. Therefore, to improve current GBRSs, the key points are modeling the heterogeneity between group members and the coupling relationship between group members leading to the final group decision, as illustrated in Figure 1.1. In this thesis, restricted Boltzmann machine (RBM)-based models, as illustrated in Figure 1.2, are employed, which aims to model the heterogeneity between group members and learn a group preference representation from the group-level feedback data.

SNRS Social network sites [22], such as Facebook and Twitter, are proliferating and attract millions of users to write blogs, post messages, and share photos. This

rapid growth intensifies the phenomenon of social overload, where users of social media are exposed to a huge amount of information and participate in vast numbers of interactions [67].

The marriage between social media and RS has many potential benefits for both sides. For example, social media introduces many new types of data and metadata, such as tags and explicit online relationships, which can be used by an RS to enhance their effectiveness. Furthermore, RSs are crucial for social media websites to enhance the adoption and engagement by their users, and thus play a crucial role in the overall success of social media [67]. Each user selection is influenced by other relevant users with social relationships, which forms influential user contexts. In recommendation problems, the coupling between users, items naturally form a multi-relational learning problem.

However, most current SNRSs fail to appropriately measure the strength of influence from different users as well as fail to appropriately model all the influence related to users' selections. As illustrated in Figure 1.1, modeling the influence of coupled users is the key problem in designing an SNRS. In this thesis, we aim to build an SNRS to capture the high-order influence from relevant users, items, and their interactions. Specifically, we build user and item influential contexts to embed influence from relevant users and items, as depicted in Figure 1.2.

Modeling Non-IID RSs from an Item Perspective

Items are another of the most elementary modeling targets in RSs. Modeling the non-IIDness on items aims to capture the connections and influence between items, item attributes, item attribute values, and item categorization. In this thesis, we study two representative non-IID RSs that mainly consider the non-IID modeling from an item perspective, namely a CDRS and an SBRS.

Modeling CDRSs A user usually has sufficient experience in some focused domains but lacks experience in other domains. An RS that can recommend potentially desirable items to users is more useful in unfamiliar domains. However, the cold-start problem will inevitably be encountered in unfamiliar domains where almost no available data exists to learn user preferences. Because users have different interests, their rich data domains and deficient data domains also differ, and therefore, it is sensible to leverage users’ feedback data over multiple domains to enable the inference of user preferences in unfamiliar domains. Based on this idea, CDRSs have emerged as an important research topic in recent years.

Current CDRSs explicitly or implicitly assume the IIDness of the items in different domains, which may lead to some problems when transferring knowledge from one domain to another because of the domain heterogeneity. As illustrated in Figure 1.1, modeling the heterogeneity between domains and leverage knowledge from auxiliary domains in terms of their intrinsic hierarchical couplings are the key problems in designing a CDRS. In this thesis, we propose an improved solution to model domain heterogeneity and their couplings. In particular, as depicted in Figure 1.2, we design an irregular tensor factorization model to deal with both explicit and implicit feedback data over multiple domains in case the standard tensor factorization models cannot be directly used for the different numbers of items in different domains.

SBRSS Most classic RSs do not consider the session context. As a result, they may recommend items that are quite irrelevant to the current context. Moreover, these RSs tend to repeatedly recommend similar items to users because of the relevance of historical choice. Factor models, such as MF, and neighborhood methods [117], such as item-based CF [193], are the two most prevalent approaches in RSs. However, they are not immediately applicable to an SBRSS because they do

not consider sequential relevance over user choices. As a result, these RSs tend to produce homogeneous recommendations similar to their historical profiles or recent purchases. In practice, users tend to have different requirements in the context of changing sessions; therefore, the homogeneous recommendation will greatly degrade user satisfaction and business benefits.

Sequential recommendation typically aims to predict the next user action. Early approaches for predicting the next user actions were based on sequential pattern mining techniques [244]. Later, more sophisticated methods based on Markov models were proposed and successfully applied to the real problem [32]. Most recently, the use of deep learning approaches based on neural networks [76] was explored as another solution. All of these methods assume a rigid-order sequence that does not fit many real-world cases. Moreover, next-item recommendation depends not only on the current session context, but also on historical sessions, which are often neglected by current SBRSs.

Therefore, we propose modeling the coupling of user choices within a session and the coupling of user choices across sessions, as specified in Figure 1.1. To deal with the deficiencies of recommendation in traditional RSs without considering the couplings of the choices within a session and between the previous sessions and current session, this thesis studies modeling a cross-session context to build a more effective and efficient personalized SBRS. As illustrated in Figure 1.2, we construct an SBRS that jointly models intra- and inter-session context for recommending the next item.

Modeling Non-IID RSs from an Interaction Perspective

The most interesting and complicated modeling of non-IID RSs is the implicit interactions. This is highly complicated because there may be multiple objectives and hierarchical non-IIDness embedded in these interactions. In this thesis, we

study two representative non-IID RSs that mainly consider the non-IID modeling from an interaction perspective, namely an MORS and an ABRs.

MORSs In most RSs, a single objective is often set up for optimization; for example, an overall evaluation or rating of an item by a user. In some recent studies, this assumption has been considered as limited [4], because the suitability of the recommended item for a particular user may depend on more than one aspect when making the choice [5]. Particularly in systems where recommendations are based on the opinion of others, the incorporation of multiple criteria that can affect the users' opinions may lead to more accurate recommendations.

To date, most research for recommender systems has focused on improving the accuracy of RS. However, simply improving the accuracy by one or two percent will not result in an improved user experience or greater business benefit. Recommendation accuracy may not always completely align with recommendation usefulness; thus, researchers have proposed several alternative measures, including coverage, diversity, novelty, serendipity, and more, to evaluate the performance of RSs. As a result, modern RS implementation may use multiple performance criteria when deciding on the final recommendation results. Therefore, it is demanding to design an MORS that can simultaneously optimize multiple recommendation performance objectives.

According to the abovementioned analysis, the key points for designing an MORS are modeling the heterogeneous but related objective for each aspect, as well as the coupling between these objectives, which lead to the final decision. In this thesis, we study the recommendation problem for users and items in the long tail to tackle the challenges of popularity bias and shilling attacks. To tackle these challenges, we set two modeling goals: one goal is to model the credibility of each choice in terms of user reputation and the other is to model the specialty of each choice over

items, as specified in Figure 1.1, to improve the quality of recommendations for items and users in the tail of a distribution. Accordingly, a coupled mutual regularization model is designed to jointly optimize these two objectives (i.e., credibility and specialty) on user choices, as illustrated in Figure 1.2.

ABRSs The CF approach generally cannot work in the case of new items, namely on the especially articles that have not been followed by any other users. To some extent, CF actually recommends second-hand information for a target user, because it can only recommend articles when other users have read them. To overcome this deficiency of CF for new articles, the CBF approach finds articles through the semantic similarity. In fact, users often read an article because they are attracted by a very small point, such as the name of a person in a news story, an interesting keyword in a paper, or some touching words in a song. However, current CBF approaches cannot interpret the most attractive point leading to user selection. Moreover, attraction modeling is not limited to textual data as the most common data in content-based RSs. For other more complex content types, such as music and images, a CNN can be applied to learn their representations and then learn the attraction over them.

In this thesis, we propose an ABRS, which aims to model and infer personal attraction on content. This raises the following question: what is the specialty of attraction that needs to be modeled? First, the attraction is the highlights that largely lead to a person's selection and decision. For example, people often cannot recite a whole poem, but can always recall some impressive sentences; similarly, people may not remember a whole song, but can hum some touching lyrics. These highlights make a person attracted to a poem or song. Second, the attraction is a subjective feeling that often differs from person to person.

Existing CBRSs can recommend new content according to similarity, but they

are not capable of interpreting why users are attracted to the content. To build an interpretable CBRS, we propose attraction modeling to learn and track user attractiveness. As depicted in Figure 1.1, the key point for designing an ABRS is to model the filter of personal attraction over content, which often requires modeling hierarchical coupling between the user and content. Specifically, we construct a hierarchical attraction model to capture users’ multilevel attractiveness on textual content and a multimodal attraction model to capture users’ multi-aspect attractiveness on movies, as illustrated in Figure 1.2.

1.4 Thesis Organization

This thesis is organized into five parts:

- *Part I:* This part introduces the background and preliminaries of the research, and consists of the following three chapters.
 - *Chapter 1:* This chapter presents the research background and research objective for this thesis.
 - *Chapter 2:* This chapter provides a literature review of the existing RS models and algorithms that are most directly related to the content studied in this thesis to elucidate the machine learning modeling techniques used in subsequent chapters.
 - *Chapter 3:* This chapter presents the necessary prerequisite knowledge, including basic machine learning models and evaluation metrics for recommendation results, to facilitate illustrating the models and algorithms in subsequent chapters, as well as the evaluation methods of experiments.
- *Part II:* This part studies the non-IID RS modeling from a user perspective, in which we model two representative non-IID RSs, namely a GBRS and SNRS:

- *Chapter 4*: This chapter provides an in-depth analysis of existing GBRs and shows their deficiencies in modeling heterogeneity and coupling between group members for making group decisions. A deep neural network is designed to learn a group preference representation, which jointly considers all members’ heterogeneous preferences.
- *Chapter 5*: This chapter builds an SNRS to capture the high-order coupling relationships between users and items. Specifically, we build user and item influence-aware contexts to embed the influence from relevant users and items.
- *Part III*: This part studies the non-IID RS modeling from an item perspective, in which we model two representative non-IID RSs, namely a CDRS and SBRS:
 - *Chapter 6*: This chapter studies a CDRS modeled by irregular tensor factorization to more effectively capture the coupling between heterogeneous domains while learning the domain factors for each domain.
 - *Chapter 7*: This chapter studies modeling a cross-session context to build a more effective and efficient SBRS. In particular, this SBRS jointly considers intra- and inter-session contexts when recommending the next item.
- *Part IV*: This part studies the non-IID RS modeling from an item perspective, in which we model two representative non-IID RSs, namely an MORS and ABRS:
 - *Chapter 8*: This chapter studies an MORS to improve the quality of recommendations for items and users in the tail of a distribution. We propose a coupled heteroscedastic MF model to jointly optimize two objectives: credibility and specialty.
 - *Chapter 9*: This chapter proposes attraction modeling for interpreting

recommendations. We construct two ABRs: one is modeled to capture users' multilevel attractiveness on textual content and the other is modeled to capture users' multi-aspect attractiveness on movies.

- *Part V*: A brief summary of the thesis contents and its contributions are provided in the final part. In addition, recommendations for future studies are given.
 - *Chapter 10*: This chapter summarizes the research contributions of this thesis.
 - *Chapter 11*: The final chapter indicates some open challenges in current non-IID RSs. Accordingly, the potential future directions for improving existing non-IID RSs are presented.

Chapter 2

Literature Survey

To facilitate understanding the non-IID RS modeling techniques proposed in subsequent chapters, this chapter provides a literature review of the existing RS models and algorithms that are most directly related to the content studied in this thesis.

2.1 Basic Recommendation Techniques

Because CF and CBF are two of the most extensively studied approaches in RSs, this section provides a brief overview of each. Moreover, because both approaches have some limitations (as presented in Section 1.1.1), we also introduce some hybrid RSs that simultaneously apply CF and CBF for recommendation. These basic techniques will be used and extended to construct the non-IID RS in subsequent chapters.

2.1.1 CF

The CF approach can be divided into neighborhood- and model-based methods. In particular, latent factor models such as MF and deep learning models such as multilayer neural networks, are the two dominant methods for model-based CF.

Neighborhood Methods

The origin of CF [194] is found in the neighborhood-based approach [206], which has been successfully applied in a number of real-world commercial systems [193].

The underlying assumption of the CF approach is that if person A has the same opinion as person B on an issue, then person A is more likely to have a similar

opinion with person B than a randomly chosen person. In general, it can be subdivided into two categories: user-based nearest neighbor and item-based nearest neighbor [206]. Tapestry [58] developed at Xerox PARC, took the first step in this direction by incorporating user actions and opinions into a message database and search system. This model is known as pull-active CF, because it is the responsibility of the user who desires recommendations to actively pull the recommendations from the database. Resnick et al. [181] designed the GroupLens system, which is mainly used to help readers filter the content in which they are interested, and they must score ratings over this filtered news. The basic assumption is that if a reader is interested in some topics, she or he will be interested in those topics again in the future. However, this approach does not work well when the data is sparse. In fact, the majority of users provide very little data because of long-tail distribution [89], therefore, it is often impossible to obtain feedback from neighbors on unpopular items and inactive users to generate accurate prediction results.

Latent Factor Models

With the rapid development of machine learning, model-based approaches have become increasingly popular in recent years. Therein, latent factor models (LFMs) are one of the most effective and efficient approaches for CF.

MF gained dominance in the area of recommendations and demonstrated its superiority over neighborhood-based techniques when it won the Netflix Prize competition [117]. The basic idea of MF methods is to fit the user-item rating matrix using low-rank approximations and use it for prediction. To date, many MF methods have been proposed, such as probabilistic matrix factorization (PMF) [188] and maximum-margin MF (MMMMF) [205]. In addition to the MF approach, other models have achieved success in recommendations. Choice modeling [214] is somewhat related to the recommendation problem; Hu et al. [89] proposed a latent feature-

based Bayesian heteroscedastic choice model to represent the heterogeneities between users and items.

The MF-based approach is able to model a collection of dyadic relations, but it has limitations in representing high-order relations (i.e., multi-way interactions). A natural approach to modeling a high-order relation is to use a tensor to extend the matrix (second order) model to a higher order space (i.e., tensor-based models). Correspondingly, a high-order tensor can also be factorized in a collection of low-rank factor matrices [114], namely tensor factorization. Karatzoglou et al. [106] studied context-aware CF using a high-order SVD (HOSVD) model to represent the relation of user-item-context, wherein each attribute of the context is modeled as a mode of a regular tensor. Tag recommendation is a widely studied problem in Web 2.0; Rendle et al. proposed BPR-based tensor factorization models on user-item-tag to learn optimal ranking [179, 180]. Time periods can be regarded as generalized domains [47] that organize temporal observations according to time-period slices to conduct temporal link prediction.

Deep Learning Approach

With the prevalence of deep learning techniques [14], RBMs, as the pioneer of deep learning models, have been successfully applied in CF [55], and achieved comparable performance to MF in the Netflix Prize competition [190]. He et al. [74] presented a general framework named neural CF (NCF) by replacing the inner product in MF with a neural architecture that can learn an arbitrary function from data. NCF is generic and can express and generalize MF under its framework. To supercharge NCF modeling with nonlinearities, the authors proposed leveraging a multilayer perceptron to learn the user-item interaction function. Autoencoder [112, 219] is a very popular building block in deep learning. Autoencoder-based CF (ACF) [163] is the first autoencoder-based collaborative recommendation model.

Instead of using the original partial observed vectors, it decomposes them by integer ratings [233]. Deep CF [131] is a general framework for unifying deep learning approaches with a CF model. This framework makes it easier to utilize deep feature learning techniques to aid collaborative recommendation. Based on this framework, the authors proposed the marginalized denoising autoencoder-based [34] based CF model (mDA-CF) to save the computational costs.

Attention mechanisms have been proven effective in various tasks such as machine translation [8] and image captioning [221]. Its underlying assumption is that one only focuses on selective parts where needed. Chen et al. [33] proposed an attentive CF model by introducing a two-level attention mechanism to a latent factor model. The attention model is an MLP consisting of item- and component-level attention. The item-level attention is used to select the most representative items to characterize users, whereas the component-level attention aims to capture the most informative features from multimedia auxiliary information for each user.

2.1.2 CBF

Another common type of RS is CBF [43]. CBF is based on the attributes or the description of items and a profile of users' preferences. In CBF, the designed algorithms attempt to recommend items that are similar to those that a user liked in the past. In particular, various candidate items are compared with items previously selected by the user and the most similar items are recommended. Basically, these methods use an item profile that characterizes the item within the system. The system creates a content-based profile of users based on selected item profiles. Machine learning techniques such as Bayesian classifiers, cluster analysis, decision trees, and artificial neural networks can be employed to estimate the probability of how a user tends to like an item. The CBF approach is often used in many areas with rich content features, such as news recommendation. For example, Pandora

Radio (<http://www.pandora.com>) is a popular example of a content-based RS that plays music with similar characteristics to that of a song provided by the user as an initial seed. Furthermore, numerous content-based RSs exist that are aimed at providing movie recommendations, including Rotten Tomatoes and Internet Movie Database.

Text Content Representation

Textual content is the most straightforward data used in CBF RSs. To obtain insight into learning user preferences, an inevitable task is to find a good representation of the textual content. The vector space model (VSM) [192] is an algebraic model for representing text documents (and any objects in general) as vectors of identifiers, such as index terms. In the classic VSM, the term-specific weights in the document vectors are products of local and global parameters. The model is known as the term frequency-inverse document frequency (TF-IDF) model [192]. The bag-of-words (BoW) assumption simplifies and represents a document as the bag of its words, disregarding grammar and word order. Distributional semantic modeling (DSM) is generally based on the distributional hypothesis that linguistic items with similar distributions have similar meanings. Latent semantic analysis (LSA) [120] and its successor, latent Dirichlet allocation (LDA) [19], are typical DSMs based on the BoW assumption.

With the prevalence of deep learning, neural network-based methods have shown power in terms of representation. Word2vec [152, 153] and Glove [171] are highly successful word embedding models of recent years, which originated from neural language modeling [16]. In many NLP architectures, word embedding representation is close to fully replacing DSM representations. Word embedding is a word-level representation, but it has inspired research work to create higher level embeddings, such as sentence embedding [113] and document embedding [41]. In particular,

Kim et al. [110] proposed character-level text modeling in terms of CNNs to learn the article representation. These methods focus on content representation without involving personal factors, whereas we aim to construct personalized content representation per personal attraction. To capture user attraction in various granularities of a document, such as word, sentence, and document levels, a hierarchical representation of an article is necessary. Furthermore, the content in each level must be highlighted to interpret the attraction to users. Yang et al. [242] proposed hierarchical attention networks for document classification, wherein two levels of attention mechanisms are respectively applied at the word and sentence levels, enabling it to attend to more or less important content when constructing the document representation. Denil et al. [46] used a CNN to transform word embeddings in each sentence into an embedding for the entire sentence, wherein at the document level, another CNN is used to transform sentence embeddings into a document embedding vector. However, these methods mainly aim to determine salient words or sentences from documents instead of consider personalized preferences.

Classic Methods

Because the success of document retrieval in VSMs depends on query construction by selecting a set of representative keywords, methods that help users to incrementally refine queries based on previous search results have been the focus of much research. These methods are commonly referred to as relevance feedback; therein, Rocchio’s formula [191] is one of the most well-known algorithms adopted in CBF. Some researchers have used a variation of Rocchio’s algorithm in a machine learning context; for example, for learning a user profile from unstructured text [10, 170]. The goal in these applications is to automatically induce a text classifier that can distinguish between classes of documents. The use of a latent semantic index (LSI) for CBF has already been investigated in several research works [53], and it has been

demonstrated to outperform other techniques, regardless of the application domain.

News recommendation [129] is a typical application of CBF with NLP techniques. The initial attempts [154] are based on text mining. Kompan et al. [115] presented an approach for fast content-based news recommendation by cosine-similarity search and effective representation of the news. Apart from textual content, such as news and Web pages, CBF can work with other types of feature. Pandora Internet radio (<http://www.pandora.com>) is a well-known music recommendation engine, which associates tracks with features extracted from the Music Genome Project. Examples of such features of tracks could be “feature trance roots”, “synth riffs”, “tonal harmonies”, or “straight drum beats”. Users can initially specify a single example of a track of their interest to create a “station”. Starting with this single training example, similar songs are played for the user. Jhanwar et al. [96] presented a technique for content-based image retrieval using the motif co-occurrence matrix (MCM). The MCM is derived using a motif transformed image, which are computationally inexpensive but sensitive to translation in image retrieval tasks.

Deep Learning Approach

A deep semantic similarity model (DSSM) [92] is a deep neural network widely used in information retrieval. It is highly suitable for top-n recommendation [49]. DSSM projects different entities into a common low-dimensional space, and computes their similarities with cosine function. DSSM-based personalized recommendation (DSPR) [238] is a tag-aware personalized RS where each user and item are represented by tag annotations and mapped into a common tag space. Cosine similarity and softmax function are applied to decide the relevance of items and users.

A CNN is capable of capturing the global and local features as well as significantly enhancing efficiency and accuracy. CNNs have been successfully applied in many CV tasks. In CBF RSs, CNNs also play a critical role in learning features for images,

text, music, and positions. Lei et al. [123] proposed a comparative deep learning model combining CNN and MLP for image recommendation. This network consisted of two CNNs used for image representation learning and an MLP for user preference modeling. It learns to rank user preferences in terms of two contrastive images (one positive image that a user likes and one negative image that the user dislikes) against a user. Wang et al. [227] investigated the influences of visual features on point-of-interest (POI) recommendation, and proposed a visual content-enhanced POI RS by employing a CNN to extract image features. The recommendation model is built on PMF by exploring the interactions between: (1) visual content and latent user factor, and (2) visual content and latent location factor. Van et al. [162] proposed using CNN to extract features from music signals. The convolutional kernels and pooling layers allow operations at multiple timescales. Gong et al. [59] built an attention-based CNN system for hashtag recommendation in microblog. In particular, they treat hashtag recommendation as a multi-label classification problem. The proposed model consisted of a global channel and local attention channel, wherein the global channel comprised convolution filters and max-pooling layers and all words were encoded in the input of the global channel.

2.1.3 Hybrid CF and CBF

As discussed in Chapter 1, both CF and CBF have advantages and disadvantages. Recent research has demonstrated that a hybrid approach combining CF and CBF could be more effective in some cases. Hybrid approaches can be implemented in several manners: by making content-based and collaborative-based predictions separately and then combining them; by adding content-based capabilities to a collaborative-based approach (and vice versa); or by unifying the approaches into one model [3]. The side information about users and items would be beneficial if it were incorporated into CF models. Bayesian MF with side information (BMFSI) [172]

is extended from Bayesian probabilistic MF. As suggested by the name, BMFSI is modeled as a combination of MF and regression against the side information of users and items. The regression-based latent factor model (RLFM) [6] uses a different strategy to incorporate side information. It assumes that the user latent factor matrix is generated from the provided features (i.e., side information) of users through regression, whereas the item latent factor matrix is generated from the provided features of items through regression. Following this, both the user and the item latent factor matrices are used as MF.

Lu et al. [140] proposed the content-based CF (CCF) approach for news topic recommendation in Bing. By utilizing rich contexts and focusing on long-tail users, the proposed CCF combined the advantages of the CBF approach and the features of the CF approach. Wang et al. [222] proposed collaborative topic regression (CTR) for scientific article recommendation, which is a hybrid method that combines probabilistic MF and LDA. Factorization machine (FM) [175] is designed to parsimoniously capture interactions between features within high dimensional sparse datasets. In content recommendation problems, the words of an article can be treated as the features fed into an FM. Musto et al. [157] proposed learning word embeddings from Wikipedia and represented a user profile as the centroid of the embedding vectors of the items that a user previously liked.

To relieve the workload of editors for selecting articles, Wang et al. [229] proposed a Dynamic Attention Deep Model (DADM) to recommend articles, where each article is represented by a vector using character-level text modeling [110]. These methods focus on article recommendation without considering the personal attractiveness in an article. Collaborative deep learning Collaborative deep learning (CDL) [223] is a hierarchical Bayesian model that integrates a stacked denoising autoencoder into probabilistic MF, which can be viewed as a deep version of CTR [222] over item text information (e.g., abstracts of articles and plots of movies).

2.2 Modeling Implicit Feedback and Long-tail Distribution

Most current RSs are built on explicit feedback, such as ratings, to differentiate users' preferences. However, explicit feedback is not always available in the real world; implicit feedback, such as click logs and dwell time, can be obtained more easily. In this thesis, we involve implicit feedback; therefore, a brief review is provided on the current works that deal with implicit feedback.

Moreover, in the real world, the data in RSs often follow a long-tail distribution (i.e., a short head and long tail). That is, the majority of users and items do not have sufficient data, which leads to many challenges for recommendations, as presented in Section 1.2. To improve the recommendations for long-tail users and items is one of the goals to build a non-IID RS. Therefore, we also briefly review studies on long-tail recommendation.

2.2.1 Implicit Feedback

Apart from rating-oriented systems modeled with regression and classification methods, recent research has paid more attention to modeling RSs with the learning to rank approach [23] for modeling implicit feedback. EigenRank [135] addresses the item ranking problem directly by modeling user preferences derived from ratings. It ranks items based on the preferences of similar users, wherein the similarity is measured using the correlation between users' rankings of items rather than rating values. ListCF [226] is a memory-based CF method that directly predicts a total order of items for each user based on similar users' probability distributions over permutations of the items. One-class implicit feedback [82, 165] is often indicated by binary values; that is, 1 for observed choices and 0 for others. These unobserved choices have zero values, but they do not mean true negative instances. Therefore, a strategy that is often used is to assign a larger confidence level to the observed choices to represent the high certainty of users' explicit likes, whereas a much smaller

confidence level is assigned to unobserved choices to represent the small certainty of dislikes [82, 91, 165]. For instance, Bayesian personalized ranking (BPR) [178] assumes that users show a stronger liking for their chosen items than for unobserved ones. Hence, a preference ordering relationship can be constructed for each pair of items. As a result, BPR learns the utility of choosing an item from the ordering relationships. He et al. [72] designed a visual BPR (VBPR) algorithm by incorporating visual features (learned through CNN) into MF using a BPR framework. Zheng et al. [249] extended a neural autoregressive distribution estimator for CF tasks (CF-NADE) [250] to incorporate implicit feedback to overcome the sparsity problem of the rating matrix. CDAE [233] is principally used for ranking prediction. The input of CDAE is users' partially observed implicit feedback. It can also be regarded as a preference vector that reflects users' interest toward items.

2.2.2 Long Tail

The long tail was popularized by Anderson in 2004, who reported that Amazon, Apple, and Yahoo! apply this strategy to realize significant profits when selling items in the tail [7]. However, the cold start problem for long-tail items and users greatly decreases the quality of recommendations. To tackle the cold-start challenge, additional information must often be incorporated. The additional side information about the users and the items would be expected to be beneficial if it were incorporated into a model. However, side information is not always available because of privacy and security concerns. As a result, long-tail items can easily suffer from shilling attacks [85]. Park and Tuzhilin [169] observed this difficulty when recommending long-tail items with very few ratings. Thus, they proposed splitting the whole item set into head and tail parts, and then grouping the tail items into clusters. As a result, the clusters of the tail items are treated as virtual items, which have relatively more ratings than just a single item. Levy and Bosteels [125] studied

music recommendations in the long tail using a conventional, item-based CF approach. However, these methods cannot work well when the data are too sparse to find similar items. Lu et al. [140] proposed the content-based CF (CCF) approach for news topic recommendation on Bing. By utilizing rich contexts and focusing on long-tail users, the proposed CCF combined both the advantages of the CBF approach and the features of the CF approach.

Some other research has targeted long-tail recommendations from the perspective of improving certain other metrics beyond accuracy [54, 149], such as diversity and serendipity [75]. Yin et al. [243] proposed a graph-based algorithm for long-tail recommendation. To improve recommendation diversity and accuracy, they extended the hitting time algorithm and proposed an efficient absorbing time algorithm to help users find their favorite long-tail items. Vargas and Castells [217] presented a formal framework for the definition of novelty and diversity metrics that unified and generalized several state-of-the-art metrics. They identified three essential ground concepts at the roots of novelty and diversity: choice, discovery, and relevance, upon which the framework was built. To improve the diversity, Bai et al. [9] presented a deep learning framework for the recommendation of long-tail Web services.

2.3 SNRSs

Social network sites [22], such as Facebook and Twitter, are proliferating and attract millions of users who author content, post messages, share photos with their friends, and engage in many other types of activity. This rapid growth intensifies the phenomenon of social overload, where users of social media are exposed to a huge amount of information and participate in vast amounts of interactions [67]. The “marriage” between social media and RS has many potential benefits for both sides. For example, social media introduces many new types of data and metadata, such as tags and explicit online relationships, which can be used in a unique manner by

Table 2.1 : Challenges to target in SNRSs

<i>Sparsity and Imbalance</i>	Referring to users with social connections
<i>Cold Start</i>	Referring to users with social connections
<i>Heterogeneity</i>	Preferences of related users
<i>Scalability</i>	Huge social relationships
<i>Social Behavior</i>	Social influence on user decisions
<i>Malicious Attack</i>	Modeling trust relationships
<i>Privacy Breach</i>	Exposure of social relationships
<i>Cross-domain Sharing</i>	Not focused on
<i>Context Awareness</i>	Not focused on
<i>Multi-information Integration</i>	Social influence
<i>Multi-criteria Consideration</i>	Not focused on

RS to enhance their effectiveness. Moreover, RSs are crucial for social network sites to enhance adoption and engagement by their users, and thus, they play a crucial role in the overall success of social media [67]. However, SNRSs should not infringe upon user privacy [48] when benefiting from large portions of social media data.

2.3.1 Summary of the Challenges to Target in an SNRS

SNRSs aim to incorporate social relationships for additional information to improve the recommendation performance. In Section 1.2, we list the main challenges in current RSs. Table 2.1 reports the challenges that are targeted for modeling SNRSs.

2.3.2 Social Media Data

Social media, such as blogs, microblogs, news, and videos, are used often in daily life. Many RSs have incorporated social interactions over these social media for recommendation. Lerman [124] described the personalized RS implemented for the web site Digg as being based on friends and “diggers like me”. Recommendations for another popular news website, Google Reader, were described by Liu et al. [134]. They combined CF techniques with “individual filtering” techniques. Evaluation, based on a live trial, indicated that the hybrid approach performed best and improved by 38% over a popularity-based baseline. Davidson et al. [42] stated that users should understand why a video was recommended to them; thus, explanations should be incorporated in the YouTube RS. As described in their paper, YouTube recommendations are based on users’ personal activity on the site and are expanded by a variant of CF over the co-visitation graph.

2.3.3 Trust Relation Modeling

Spam data can be found everywhere on the Internet (e.g., E-commerce and social networking sites). Hence, trust and reputation systems (TRSs) [103, 104] have been designed specifically to foster trusted behavior in these domains. In recent years, with increasing attention placed on RSs, researchers in the TRS area have proposed incorporating trust into RSs [102], where a trust score between two users serves as the weight to conduct the conventional neighbor-based method. However, the neighbor-based method has its limitations when the data is highly sparse. People in the area of RSs are also aware of the importance of trust. Some social network-based recommendation algorithms employ random walks to compute recommendation ratings [93, 212]. Jamali and Ester [93] proposed the TrustWalker, which performs a random walk in online social networks to query a user’s direct and indirect friends’ ratings for the target item as well as similar items. SoRec [142]

incorporates trust networks into RSs, where joint factorization is conducted over two matrices: user-item ratings and user-user trust relationships. In recent years, researchers have utilized trust relations to perform regularization for users. SocialMF [94] and SoReg [141] are two representative models that regularize the user factor vector of a target user via the user factor vectors of their trusters. Such regularization plays the role of borrowing the preferences of the trusters to deal with the cold-start problem. Yang et al. [241] proposed circle-based recommendations in online social networks. It is obvious that a user’s social life, whether online or offline, is intrinsically multifaceted. Intuitively, a user trusts different subsets of friends in different domains. Yang et al. [240] proposed Trust-CF-ULF to incorporate social network information into top-k RSs. The Trust-CF-ULF approach is a combination of a user latent feature space-based CF approach (CF-ULF) and a social network-based approach.

2.3.4 Deep Learning Models

Wang et al. [228] extended neural CF (NCF) [74] to cross-domain social recommendations (i.e., recommending items of information domains to potential users of social networks), and presented a neural social collaborative ranking RS. Jia et al. [97] proposed an RBM-based generative model for event recommendation. By introducing hidden variables into content-based learning, the model can successfully capture the latent mapping relationships from the user and event features to the output of participation. Deng et al. [45] combined the deep learning technique with MF (DLMF) for trust-aware recommendation. The DLMF approach contains two phases of learning: in the first phase, it utilizes a deep autoencoder to learn the initial values of the latent feature vectors of users and items; and in the second phase, it learns the final latent feature vectors of users and items in terms of MF.

2.4 GBRSSs

Human beings are of a social nature; therefore, various types of group activities are observed throughout life, such as seeing a movie or planning a trip. Recently, the RS community has begun to study group behavior to make group recommendations [95]. In particular, members of a group often possess different opinions on the same items; therefore, the main challenge in GBRSSs is to satisfy most group members with diverse preferences.

2.4.1 Summary of the Challenges to Target in GBRSSs

GBRSSs aim to conduct recommendations for a group by considering all group members' preferences. According to the main challenges in RSs listed in Section 1.2. Table 2.2 reports the corresponding challenges that are focused on for modeling GBRSSs.

2.4.2 Aggregation Approach

PolyLens [164] is an early GBRSS wherein users can create groups and ask for recommendations. Berkovsky and Freyne [17] studied recipe recommendations for families where all members eat a meal together. Because each member of a group may have different opinions on the same items, the main challenge of satisfying most group members with diverse preferences cannot be achieved through an individual-based CF method.

To date, the existing mainstream approaches of GBRSS try to aggregate group information from individual user models [95, 145]. In general, these methods can be classified into two types of model that differ in the timing of data aggregation. The first type of model is called group preference aggregation (GPA), which first aggregates all members' ratings into a group profile, and then any individual-based CF approach can be used if it regards groups as virtual individual users. By con-

Table 2.2 : Challenges to target in GBRs

<i>Sparsity and Imbalance</i>	Not focused on
<i>Cold Start</i>	Not focused on
<i>Heterogeneity</i>	Preferences between group members
<i>Scalability</i>	Not focused on
<i>Social Behavior</i>	Group-based choice
<i>Malicious Attack</i>	Not focused on
<i>Privacy Breach</i>	Exposure of group relationships
<i>Cross-domain Sharing</i>	Not focused on
<i>Context Awareness</i>	Group members as the context
<i>Multi-information Integration</i>	Member preferences
<i>Multi-criteria Consideration</i>	Multi-objective optimization for all members

trast, the second type of model is called individual preference aggregation (IPA), which first predicts the individual ratings over candidate items, and then aggregates the predicted ratings of members within a group through predefined strategies to represent group ratings. GBRs measure group satisfaction by means of aggregating members' information using some aggregation models, such as GPA and IPA. In fact, quite a few heuristic strategies have been designed to work with the aggregation models. In particular, average and least misery are the two most prevalent strategies [145], thus, they are employed in this paper. For example, PolyLens [164] uses the least misery strategy, which assumes that a group tends to be as happy as its least happy member. Yuet al. [246] took the average strategy to recommend television programs for groups. Moreover, Berkovsky and Freyne [17] compared these two strategies for recipe recommendations for families. In this paper, we study a case of movie recommendations for households, which was sponsored by CAMRa2011 [187].

Some recent work [60,90] have examined this problem using some aggregation models. Therein, Hu et al. [90] tested the MF method under GPA and IPA models with various strategies. However, such methods are heavily dependent on the input data, which often fail to learn the representation of group preferences when the data is slightly inconsistent with the aggregation assumption.

2.4.3 Deep Learning Approach

To our knowledge, very few GBRSs have been built with deep learning models. Only two very recent studies were presented for group-based recommendation by applying the attention mechanism. Vinh et al. [220] developed the attentive group recommendation (AGR) model to solve the group recommendation problem through a deep learning approach. The AGR model utilizes the attention mechanism to learn the influence weight of each user in a group to make group recommendations. Specifically, given a group of users, the model creates a sub-network attention model for each user to learn their influence (i.e., the impact weight) on the group’s final decisions. Cao et al. [29] addressed the fundamental problem of preference aggregation in group recommendation by learning the aggregation strategy from data. They contributed a solution called AGREE (short for “Attentive Group REcommEndation”), based on the recent developments of attention network and NCF. Specifically, they adopted the attention mechanism to adapt the representation of a group and learn the interaction between groups and items from data under the NCF framework.

2.5 CDRSs

Because users have different interests, their rich data domains and deficient data domains also differ, and therefore, it is sensible to leverage users’ feedback data over multiple domains to enable the inference of user preferences in unfamiliar domains. Based on this idea, CDRSs have emerged as a critical research topic. In particular,

Table 2.3 : Challenges to target in CDRSs

<i>Sparsity and Imbalance</i>	Transfer knowledge from related domains
<i>Cold Start</i>	Transfer knowledge from related domains
<i>Heterogeneity</i>	Heterogeneity of items in different domains
<i>Scalability</i>	Huge number of users and items in multiple
<i>Social Behavior</i>	Not focused on
<i>Malicious Attack</i>	Not focused on
<i>Privacy Breach</i>	Exposure of user identity in multiple domains
<i>Cross-domain Sharing</i>	Sharing user preference across domains
<i>Context Awareness</i>	Related domains as context
<i>Multi-information Integration</i>	Multi-domain knowledge
<i>Multi-criteria Consideration</i>	Multi-domain learning with multiple criteria

Cantador et al. [28] defined domains from four different levels: (1) the item attribute level: the same type of items with different values of a certain attribute (e.g., movies with different genres); (2) the item type level: similar types of items (e.g., movies and videos); (3) the item level: different types of items (movies and books); and (4) the system level: same type of items on different systems (e.g., movies on IMDb.com and Netflix.com).

2.5.1 Summary of the Challenges to Target in CDRSs

CDRSs leverage the knowledge between multiple domains to deal with the challenges in traditional single-domain RSs caused by limited information. According to the main challenges in RSs listed in Section 1.2. Table 2.3 reports the corresponding challenges that are focused on for modeling CDRSs.

2.5.2 Latent Factor Models

With the prevalence of the MF approach in RSs, LFMs have been extended to conduct recommendation over multiple domains. Codebook Transfer [127] assumes that cluster-level rating patterns, which are represented by a codebook, can be found between the rating matrices in two related domains. The rating-matrix generative model [128] extends this idea with a probabilistic model to solve collective transfer learning problems. In reality, there are many cold-start users for most domains due to the power law. Therefore, it is always out of the question to use this model in unfamiliar domains because no data is available to match common patterns with auxiliary domains. Coordinate system transfer (CST) [167] is a typical CDMF model that first learns the user-factor matrix \mathbf{U}_A from an auxiliary rating matrix, and then the user-factor matrix \mathbf{U}_T of the target domain is then updated based on \mathbf{U}_A , with the regularization in terms of penalizing the divergence between \mathbf{U}_A and \mathbf{U}_T . Dual transfer learning (DTL) [136] was formulated as an optimization problem of joint non-negative matrix trfactorizations. It exploits the duality between the marginal distribution and conditional distribution to achieve effective transfer. Given the observations of source and target domains, marginal distribution is associated with the common latent features over all domains, and conditional distribution is associated with the domain-specific latent features. Because CST and DTL are modeled on the problem of transferring knowledge from a source domain to a target domain, they cannot be directly applied to multiple domains (more than two), as studied in this paper. Because user preference is not exclusive to a single domain, a straightforward method is to transfer knowledge through the user-factor matrix.

Collective MF (CMF) [202] couples the target domain matrix and all auxiliary domain matrices in the user dimension, to share the user-factor matrix across all domains. Moreover, CMF assigns a weight to the loss of fitting each domain matrix; thus, it can control the amount of influence from each domain; however, it does not

provide a mechanism for finding an optimal weight assignment. Loni et al. [137] presented an approach that encodes rating matrices from multiple domains as real-valued feature vectors $x[u, i; s_2(u), \dots, s_D(u)]$, where u, i index of a rating given by user u on item i of the target domain and $s_d(u)$ indexes all rescaled ratings on auxiliary domain d . With these vectors, an algorithm based on FMs [175] was employed to detect patterns between target and auxiliary domains. However, for a cold-start user without any rating on a target domain, no data is available to encode such a feature vector x for this user to match patterns. Therefore, this method is unsuitable for application in a cold-start environment such as that studied in this paper. Moreover, this model is built on rating data, whereas how to deal with implicit preference data is not defined.

However, we cannot employ a regular tensor factorization model to deal with the CDCF problems analyzed in the Introduction section. Inspired by PARAFAC2 [114, 158], Hu et al. [83] proposed a cross-domain triadic factorization (CDTF) model, which transforms the optimization problem on domain slices into an equivalent tensor factorization problem. When dealing with missing values, both CDTF and PARAFAC2 use an imputation strategy [204] to restore missing values by predicting ones for each iteration, and therefore, they need a full matrix to store the reconstructed data on each domain. As a result, the space complexity of CDTF and PARAFAC2 tends to be even larger when more domains are involved.

2.5.3 Deep Learning Models

Deep learning is well suited to transfer learning because it learns high-level abstractions that disentangle the variation of different domains. Domain adaptation [13] considers the setting in which the data sampled from the target and the source domain have different distributions. In particular, deep learning based domain adaptation has been successfully applied in many tasks, e.g. sentimental

analysis [34, 57] in NLP and cross-modality classification in computer vision [216].

Lian et al. presented the cross-domain content-boosted CF neural network (CC-CFNet) [132], which embeds a cross-domain model into a unified multi-view neural network to perform cross-domain recommendations. The basic composite of CC-CFNet is a dual network (for users and items, respectively), which models the user-item interactions in the last layer with dot product. To embed content information, the authors further decomposed each network of the dual net into two components: CF factor and content information. Multi-view deep neural network (MV-DNN) [49] is designed for cross-domain recommendation. It treats users as the pivot view and each domain as the auxiliary view, where a similarity score is computed for each user domain. MV-DNN is similar to CCCFNet, but CCCFNet does not involve any similarity and posterior probability estimation. He et al. [228] extended the Neural Collaborative Filtering (NCF) [74] to cross-domain social recommendations, such as recommending items of information domains to potential users of social networks. Wang et al. [228] proposed cross-domain social recommendation and developed a generic neural social collaborative ranking (NSCR) model based on NCF, which seamlessly integrates user-item interactions of the information domain and user-user social relations of the social domain.

2.6 SBRs

Time-aware RSs (TARs) [25] and SBRs are context-aware RSs (CARs), but they target different goals. TARs consider temporal factors when making recommendations; for example, how time information (e.g., weekday vs. weekend) affects the recommendation [11]. By contrast, SBRs do not require precise timestamps, because they focus on learning the relevance between items in a session scope.

Table 2.4 : Challenges to target in SBRS

<i>Sparsity and Imbalance</i>	Learning item dependency from all users' sessions
<i>Cold Start</i>	Learning item dependency from all users' sessions
<i>Heterogeneity</i>	Not focused on
<i>Scalability</i>	Huge number of sessions
<i>Social Behavior</i>	Not focused on
<i>Malicious Attack</i>	Not focused on
<i>Privacy Breach</i>	Not focused on
<i>Cross-domain Sharing</i>	Sharing user preference across user sessions
<i>Context Awareness</i>	Intra- and inter-session context
<i>Multi-information Integration</i>	Cross-user session knowledge
<i>Multi-criteria Consideration</i>	Intra-session objective and inter-session objective

2.6.1 Summary of the Challenges to Target in SBRSs

SBRSs model the dependency and transition over items in a session context. According to the main challenges in RSs listed in Section 1.2. Table 2.4 reports the corresponding challenges that are focused on for modeling SBRSs.

2.6.2 Markov Models

Session-based CF [168] firstly finds the k -nearest neighbors (k nn) for the current session and then calculates the score for each potential item to generate the rank. The number of historical sessions is huge; therefore, finding the k -nearest sessions online is unfeasible for real RSs. Factorized personalized Markov chains (FPMCs) [179] combine the power of MF and MC to model personalized sequential behavior. Identical to MF, FPMCs easily suffer from the data sparsity problem.

Markov decision processes (MDPs) [197] are an early approach to provide recommendations in a session-based manner. Purchase sequence prediction based on the hidden Markov model (HMM) and purchase intervals was introduced in [65]. Because the number of items is large, a problem encountered when applying MDP and HMM is that the state space quickly becomes unmanageable when evaluating all possible sequences over all items. The model in [35] playlists as an MC, and proposes representing songs using logistic Markov embedding (LME). Personalized metric embedding (PME) [232] and personalized ranking metric embedding (PRME) [50] extend LME by adding personalization. LME, PME, and PRME are first-order MC models built on rigid sequential data to model the transition between consequent choices.

2.6.3 Deep Learning Models

In recent years, SBRs have begun to embrace deep learning technology. Recurrent Neural Networks (RNN) have been devised to model variable-length sequence data, wherein the internal state of the network allows it to exhibit dynamic temporal behavior. Hidasi et al. [76] applied RNN consisting of gated recurrent units as the key component to construct an SBR, namely GRU4Rec. Twardowski et al. [215] used a similar RNN for Ad click prediction. Quadrana et al. [173] added an extra component to an RNN-based SBR to capture users' profiles and information between sessions. The idea was to use a hierarchical architecture that models cross-sessions by adding another RNN over user sessions, which is capable of capturing the evolution of the user across sessions and providing personalization capabilities. Loyola et al. [139] proposed an encoder-decoder architecture for the problem of predicting the next item within a session and whether he has the intention to purchase an item or not. Inspired by recent encoder-decoder approaches in MT, the proposed model has a bidirectional RNN as the encoder and a normal RNN as the decoder;

the authors added alignment (passing labels as well as predictions to the next time state) and attention (encapsulating expressive portions of the source sequence in a separate block) mechanisms as well.

Although language modeling in natural language processing (NLP) is not literally related to SBRSs, we find that they share common underlying problem; that is, learning the probability distribution over sequences of words is similar to learning it over sequences of items. Word embedding models [155, 156], especially word2vec [152, 153], have achieved great success in NLP. Moreover, because of the large vocabulary size, word embedding models do not evolve to a deep structure; by contrast, they become shallow and wide [61] to more efficiently adapt large-class data. Hu et al. [88] designed an efficient SBRS with a Shallow Wide-In-Wide-Out network (SWIWO). In view of the large number of items in RSs, SWIWO incorporates the subsampling mechanism to improve the efficiency of model learning. Specifically, SWIWO does not assume a relaxed order of items in a session. In SWIWO, the authors applied a heuristic rule to assign larger weights to the items that are closer to the target item. Wang et al. [225] designed an effective attention-based transaction embedding model (ATEM) for context embedding to weight each observed item in a transaction without assuming order. In particular, ATEM extends the SWIWO model by building an attention model over the session context to remove the weight assignment heuristics.

2.7 MORSSs

To date, most research on RSs has focused on improving their accuracy. However, simply improving the accuracy by one or two percent will hardly result in an improved user experience or greater business benefit. Recommendation accuracy may not always completely align with recommendation usefulness; therefore, researchers have proposed several alternative measures, including coverage, diversity, novelty,

and serendipity [217], to evaluate the performance of RSs. As a result, modern RS implementations may use multiple performance criteria when deciding on the final set of recommendations. In most RSs, a single objective is often set for optimization (e.g., an overall evaluation or rating of an item by a user). In some recent studies, this assumption has been considered as limited [4], because the suitability of the recommended item for a particular user may depend on more than one aspect when making the choice [5]. Particularly in systems where recommendations are based on the opinions of others, the incorporation of multiple criteria that can affect users' opinions may lead to more accurate recommendations.

2.7.1 Summary of the Challenges to Target in MORSSs

MORSSs jointly model multiple objectives to generate more considerable recommendation results with multi-perspective considerations. In Section 1.2, we list the main challenges in the current RSs. Table 2.5 reports the challenges that are targeted when modeling an MORSS.

2.7.2 Multi-criteria Ratings

The importance of studying MORSSs has been highlighted as a separate strand in RS literature [4, 144], and recently, several RSs have adopted multiple criteria ratings instead of traditional single-criterion ratings. Neighborhood-based CF techniques can consider multi-criteria ratings using the large number of design options [144]. Similarly, some model-based methods stem from their single-criterion counterparts. The aggregation function approach assumes that the overall rating r_0 serves as an aggregate of multi-criteria ratings [4]. Given this assumption, this approach finds aggregation function $r_0 = f(r_1, \dots, r_k)$ that represents the relationship between overall and multi-criteria ratings. Li et al. [130] utilized the multilinear singular value decomposition (MSVD) technique, which is a particular realization of the MF approach in multi-criteria rating settings. Zhang et al. [248] extended

Table 2.5 : Challenges to target in MORS

<i>Sparsity and Imbalance</i>	As an objective to model
<i>Cold Start</i>	As an objective to model
<i>Heterogeneity</i>	Heterogeneity between objectives
<i>Scalability</i>	Time complexity of multi-objective optimization
<i>Social Behavior</i>	Not focused on
<i>Malicious Attack</i>	As an objective to model
<i>Privacy Breach</i>	Not focused on
<i>Cross-domain Sharing</i>	Not focused on
<i>Context Awareness</i>	Multiple objectives as the context
<i>Multi-information Integration</i>	Multi-objective Integration
<i>Multi-criteria Consideration</i>	The major targeting problem

the probabilistic latent semantic analysis (PLSA) approach into multi-criteria rating settings. However, among alternative approaches, there have been several sophisticated hybrid recommendation approaches developed in recent years, and some could potentially be adopted for multi-criteria rating RSs.

2.7.3 Multi-objective Ranking

Multi-objective optimization problems (MOPs) have been extensively studied in the literature [44], although not in the context of RSs. One study [119] focused on RSs that work with implicit feedback in dynamic scenarios providing online recommendations, such as news articles and ad recommendation in Web portals. In these dynamic scenarios, user feedback to the system is given through clicks, and feedback must be quickly exploited to improve subsequent recommendations. In this scenario, the authors propose an algorithm named multi-objective ranked bandits

to recommend lists of items for weighing accuracy, diversity and novelty [119].

Data envelopment analysis (DEA), often also called “frontier analysis”, is commonly used to measure the productive efficiency of decision-making units (DMUs) in operations research. DEA computes the efficiency frontier, which identifies the items that are the “best performers” overall when all criteria are considered. While DEA has not been directly used in multi-criteria rating RSs, the multi-criteria recommendation problem without overall ratings can also be formulated as a data query problem in the database field using similar motivation [122]. Lee and Teng [122] utilized skyline queries to find the best restaurants across multiple criteria with respect to food, service, and cost. Lin et al. [234] presented several semantics of the individual utility and proposed two concepts of social welfare and fairness for modeling the overall utilities and balance between group members. Given the multi-objective nature of the fairness-aware group recommendation problem, the authors provided an optimization framework for fairness-aware group recommendation from the perspective of Pareto efficiency.

Evolutionary algorithms have great advantages over other heuristic approaches in dealing with MOPs. They can simultaneously deal with a series of possible solutions, which allows finding several members of the Pareto optimal set in a single run of the algorithm, and can easily deal with discontinuous and concave Pareto fronts. Multi-objective evolutionary algorithms (MOEA) are introduced to deal with problems with two or more contradictory objectives and return a set of Pareto optimal solutions [224]. Zuo et al. designed a multi-objective evolutionary algorithm to evolve the population to maximize the two objectives of accuracy and diversity. The accuracy was predicted by the ProbS [252] method, whereas the diversity was measured by recommendation coverage. NSGA-II [44] was adopted to solve the modeled MOP for personalized recommendation. Ribeiro et al. [182] showed that existing recommendation algorithms do not perform uniformly well when evaluated in terms of

accuracy, novelty, and diversity, and thus, they proposed approaches that exploited the Pareto efficiency concept to combine such recommendation algorithms in a manner in which a particular objective is maximized without significantly hurting the other objectives. As a result, the authors designed a Pareto-efficient hybridization recommendation approach, where MOEA was used to optimize a vector of weights assigned to different recommendation methods.

Chapter 3

Preliminaries

Since the methodology of this thesis to construct Non-IID RSs is based on the machine learning approach, we first present some preliminaries of the machine learning models which are related to the modeling in the following chapters. In this chapter, we will introduce latent factor models, deep learning models and evaluation metrics for recommendation results.

3.1 Neighborhood Method

k -nearest neighbor (k NN) is a representative, memory-based collaborative filtering approach. The simple form of user-based k NN can be given by:

$$\hat{Y}_{i,j} = \frac{\sum_{n \in \Gamma_i} w_{i,n} Y_{n,j}}{\sum_{n \in \Gamma_i} w_{i,n}} \quad (3.1)$$

where $\hat{Y}_{i,j}$ is the predictive rating of user i on item j ; $Y_{n,j}$ is the observation on j from a neighbor of i ; Γ_i denotes the neighbor set of user i ; and $w_{i,n}$ is the weight between user i and user n . Typically, $w_{i,n}$ can be computed by Pearson correlation, cosine, or Jaccard similarity [26].

3.2 Latent Factor Models

3.2.1 Matrix Factorization

Probabilistic matrix factorization (PMF) [188] is a typical model to illustrate the MF approach from the probabilistic view. Given a data matrix $\mathbf{Y} \in \mathbb{R}^{N \times M}$ with the index of each observed choice $(i, j) \in \mathbf{O}$ on N users and M items, we can obtain

the following distributions with the d -dimensional latent factors $\mathbf{U}_i \in \mathbb{R}^d$ of users and $\mathbf{V}_j \in \mathbb{R}^d$ of items:

$$P(\mathbf{U}_i) = \mathcal{N}(\mathbf{U}_i | \mathbf{0}, \sigma_U^2 \mathbf{I}) \quad P(\mathbf{V}_j) = \mathcal{N}(\mathbf{V}_j | \mathbf{0}, \sigma_V^2 \mathbf{I}) \quad (3.2)$$

$$P(Y_{ij} | \mathbf{U}_i, \mathbf{V}_j) = \mathcal{N}(Y_{ij} | \mathbf{U}_i^\top \mathbf{V}_j, \sigma^2) \quad (3.3)$$

$$P(\mathbf{U}, \mathbf{V} | \mathbf{Y}) \propto P(\mathbf{Y}, \mathbf{U}, \mathbf{V}) = \prod_{i,j \in \mathbf{O}} P(Y_{ij} | \mathbf{U}_i, \mathbf{V}_j) \prod_i P(\mathbf{U}_i) \prod_j P(\mathbf{V}_j) \quad (3.4)$$

where $\mathbf{U} = [\mathbf{U}_1, \dots, \mathbf{U}_N]$ is the user factor matrix; $\mathbf{V} = [\mathbf{V}_1, \dots, \mathbf{V}_M]$ is the item factor matrix; and $\sigma_U^2, \sigma_V^2, \sigma^2$ are the variance parameters of the Gaussian distributions. We learn the user factors and the item factors through a maximum a posteriori (MAP) estimate. According to the Bayesian theorem, we have the posterior $P(\mathbf{U}, \mathbf{V} | \mathbf{Y}) \propto P(\mathbf{Y}, \mathbf{U}, \mathbf{V})$ given in Eq. 3.4. The following objective function can then be obtained by minimizing the negative log-posterior. Without loss of generality, we easily obtain the classic objective of an MF model [117, 188], when we set $\sigma^2 = 1$ and denote $\lambda = \sigma_U^{-2} = \sigma_V^{-2}$ as the regularization parameter:

$$J = \operatorname{argmin}_{\mathbf{U}, \mathbf{V}} \frac{1}{2} \left[\sum_{i,j \in \mathbf{O}} (Y_{ij} - \mathbf{U}_i^\top \mathbf{V}_j)^2 + \lambda (\|\mathbf{U}_i\|_2^2 + \|\mathbf{V}_j\|_2^2) \right] \quad (3.5)$$

We can easily write the partial derivative $\partial J / \partial \mathbf{V}_j$ w.r.t. each \mathbf{V}_j . The optimization w.r.t. \mathbf{V}_j is convex when \mathbf{U} is fixed. A close-form update equation for \mathbf{V}_j can be obtained by setting $\partial J / \partial \mathbf{V}_j$ to zero [188]:

$$\mathbf{V}_j \leftarrow \left(\lambda \mathbf{I} + \sum_{i \in \mathbf{O}_j} \mathbf{U}_i \mathbf{U}_i^\top \right)^{-1} \sum_{i \in \mathbf{O}_j} Y_{ij} \mathbf{U}_i \quad (3.6)$$

where \mathbf{O}_j indexes those users who have chosen item j . Similarly, the optimization w.r.t. \mathbf{U}_i is convex when \mathbf{V} is fixed, and thus, we can obtain:

$$\mathbf{U}_i \leftarrow \left(\lambda \mathbf{I} + \sum_{j \in \mathbf{O}_i} \mathbf{V}_j \mathbf{V}_j^\top \right)^{-1} \sum_{j \in \mathbf{O}_i} Y_{ij} \mathbf{V}_j \quad (3.7)$$

where \mathbf{O}_i indexes the items chosen by user i .

3.2.2 Tensor Factorization

Tensor factorization (TF) models generalize MF in terms of higher order low-rank approximation.

Notations and Operations

The order of a tensor is the number of dimensions, also known as ways or modes. In this thesis, tensors are denoted by calligraphic capital letters, e.g. \mathcal{X} . Matrices are denoted by boldface capital letters, e.g. \mathbf{X} . Vectors are denoted by boldface lowercase letters, e.g. \mathbf{x} . In addition, we denote the i th row of a matrix \mathbf{X} as $\mathbf{X}_{i,:}$, the j th column as $\mathbf{X}_{:,j}$ and $\mathbf{X}_{i,j}$ for the entry (i, j) . $\mathbf{X}_{(n)}$ denotes the n th mode matricizing operation which maps a tensor into a matrix, e.g., $\mathbf{X}_{(2)}$ represents the mapping $\mathcal{X}^{I \times J \times K} \rightarrow \mathbf{X}_{(2)}^{J \times IK}$ [114]. \otimes denotes the Kronecker product and \odot denotes the Khatri-Rao product [114], e.g. we have $\mathbf{X} \odot \mathbf{Y} = [\mathbf{X}_{:,1} \otimes \mathbf{Y}_{:,1}, \mathbf{X}_{:,2} \otimes \mathbf{Y}_{:,2}, \dots, \mathbf{X}_{:,R} \otimes \mathbf{Y}_{:,R}]$. \cdot^* denotes the Hadamard (element-wise) product. $\langle \mathcal{X} \cdot^* \mathcal{Y} \rangle = \sum_{i,j,k} \mathcal{X}_{i,j,k} \mathcal{Y}_{i,j,k}$ stands for the inner product and the norm of a tensor is defined as $\|\mathcal{X}\| = \sqrt{\mathcal{X} \cdot^* \mathcal{X}}$.

Regularized CP Model

There are a number of different TF models in the literature, such as the CP model (canonical decomposition / parallel factor analysis (PARAFAC)) and the Tucker model [114]. Here, we mainly present the CP model, which is the most widely used TF model, because it has a concise factorization form.

As shown in Figure 3.1, the CP model decomposes a tensor into a sum of rank-one components. For instance, given a third-order tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$, the factorization form of \mathcal{X} can be written as $\mathcal{X} = [[\mathbf{A}, \mathbf{B}, \mathbf{C}]] = \sum_{r=1}^R \mathbf{A}_{:,r} \circ \mathbf{B}_{:,r} \circ \mathbf{C}_{:,r}$, where \mathbf{A} , \mathbf{B} and \mathbf{C} are R -dimensional latent factor matrices and \circ denotes the outer product, i.e. the entries are computed $\mathcal{X}_{i,j,k} = \sum_{r=1}^R \mathbf{A}_{i,r} \circ \mathbf{B}_{j,r} \circ \mathbf{C}_{k,r}$. We can formulate

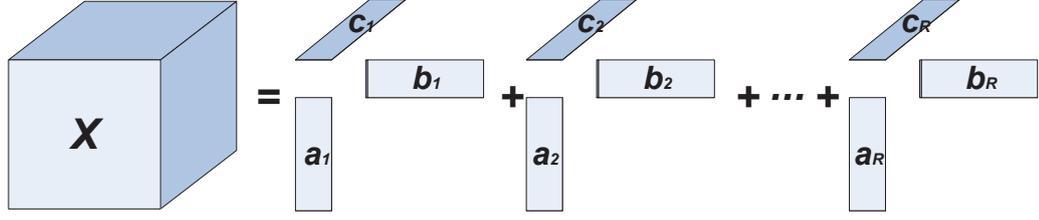


Figure 3.1 : The demonstration of CP factorization on a third-order tensor

the problem of fitting \mathcal{X} as a least squares optimization problem with the following objective function [1]:

$$J = \operatorname{argmin}_{\mathbf{A}, \mathbf{B}, \mathbf{C}} \frac{1}{2} [\|\mathcal{X} - \llbracket \mathbf{A}, \mathbf{B}, \mathbf{C} \rrbracket\|^2 + \lambda_A \|\mathbf{A}\|_F^2 + \lambda_B \|\mathbf{B}\|_F^2 + \lambda_C \|\mathbf{C}\|_F^2] \quad (3.8)$$

where regularization terms are added to avoid overfitting, $\|\cdot\|_F$ is the Frobenius norm and $\lambda_A, \lambda_B, \lambda_C$ are regularization parameters. From Eq. 3.8, the following gradient w.r.t. $\lambda_A, \lambda_B, \lambda_C$ can be derived:

$$\frac{\partial J}{\partial \mathbf{A}} = -\mathbf{X}_{(1)}(\mathbf{C} \odot \mathbf{B}) + \mathbf{A}(\mathbf{B}^\top \mathbf{B} \cdot * \mathbf{C}^\top \mathbf{C} + \lambda_A \mathbf{I}) \quad (3.9)$$

$$\frac{\partial J}{\partial \mathbf{B}} = -\mathbf{X}_{(2)}(\mathbf{C} \odot \mathbf{A}) + \mathbf{B}(\mathbf{C}^\top \mathbf{C} \cdot * \mathbf{A}^\top \mathbf{A} + \lambda_B \mathbf{I}) \quad (3.10)$$

$$\frac{\partial J}{\partial \mathbf{C}} = -\mathbf{X}_{(3)}(\mathbf{A} \odot \mathbf{B}) + \mathbf{C}(\mathbf{A}^\top \mathbf{A} \cdot * \mathbf{B}^\top \mathbf{B} + \lambda_C \mathbf{I}) \quad (3.11)$$

Similar to learning MF using alternating least square [117], CP model alternatively learn the parameters $\{\mathbf{A}, \mathbf{B}, \mathbf{C}\}$, i.e. update one of the parameters by fixing others [83]. Setting the above equation equal to zero and the property of pseudo-inverse of Khatri-Rao product [114] yields:

$$\mathbf{A} = \mathbf{X}_{(1)}(\mathbf{C} \otimes \mathbf{B})(\mathbf{B}^\top \mathbf{B} \cdot * \mathbf{C}^\top \mathbf{C} + \lambda_A \mathbf{I})^\dagger \quad (3.12)$$

$$\mathbf{B} = \mathbf{X}_{(2)}(\mathbf{C} \otimes \mathbf{A})(\mathbf{A}^\top \mathbf{A} \cdot * \mathbf{C}^\top \mathbf{C} + \lambda_B \mathbf{I})^\dagger \quad (3.13)$$

$$\mathbf{C} = \mathbf{X}_{(3)}(\mathbf{A} \otimes \mathbf{B})(\mathbf{A}^\top \mathbf{A} \cdot * \mathbf{B}^\top \mathbf{B} + \lambda_C \mathbf{I})^\dagger \quad (3.14)$$

3.3 Deep Learning Models

3.3.1 Restricted Boltzmann Machines

A Restricted Boltzmann Machine (RBM) [78] is a Markov random field over a vector of binary visible units $\mathbf{v} \in \{0, 1\}^D$ and hidden units $\mathbf{h} \in \{0, 1\}^F$, where the connections only exist between \mathbf{v} and \mathbf{h} , as illustrated in Figure 3.2. The distribution of an RBM is defined through an energy function $E(\mathbf{v}, \mathbf{h}; \boldsymbol{\theta})$:

$$P(\mathbf{v}, \mathbf{h}; \boldsymbol{\theta}) = \frac{\exp(-E(\mathbf{v}, \mathbf{h}; \boldsymbol{\theta}))}{Z(\boldsymbol{\theta})} \quad (3.15)$$

$\boldsymbol{\theta} = \{\mathbf{W}, \mathbf{b}, \mathbf{d}\}$ are the model parameters, where $\mathbf{W} \in \mathbb{R}^{D \times F}$ encodes the visible-hidden interaction, $\mathbf{b} \in \mathbb{R}^D$ and $\mathbf{d} \in \mathbb{R}^F$ encodes the biases of \mathbf{v} and \mathbf{h} . The pattern of such interaction can be formally specified through the energy function:

$$E(\mathbf{v}, \mathbf{h}; \boldsymbol{\theta}) = -\mathbf{v}^T \mathbf{W} \mathbf{h} - \mathbf{b}^T \mathbf{v} - \mathbf{d}^T \mathbf{h} \quad (3.16)$$

The conditional distributions w.r.t. visible units and hidden units are factorial [14], which can be easily derived from Eq. 3.3.1:

$$P(v_i = 1 | \mathbf{h}; \boldsymbol{\theta}) = s(b_i + \sum_{j=1}^D W_{ij} h_j) \quad (3.17)$$

$$P(h_j = 1 | \mathbf{v}; \boldsymbol{\theta}) = s(d_j + \sum_{i=1}^K v_i W_{ij}) \quad (3.18)$$

where $s(\cdot)$ is a sigmoid function. In particular, the RBM has been generalized to Gaussian RBM (GRBM) to work with real-value data. The energy of the GRBM is defined by:

$$E(\mathbf{v}, \mathbf{h}; \boldsymbol{\theta}) = \sum_{i=1}^D \frac{(v_i - b_i)^2}{2\sigma_i^2} - \sum_{i=1}^F d_j h_j - \sum_{i=1}^D \sum_{j=1}^F \frac{v_i W_{ij} h_j}{\sigma_i} \quad (3.19)$$

where the Gaussian visible units $\mathbf{v} \in \mathbb{R}^D$, the hidden units $\mathbf{h} \in \{0, 1\}^F$ and $\boldsymbol{\theta} = \{\mathbf{W}, \mathbf{b}, \mathbf{d}, \boldsymbol{\sigma}\}$ are the model parameters. Accordingly, the conditional distributions w.r.t. each visible unit and each binary hidden unit are given by:

$$P(v_i | \mathbf{h}; \boldsymbol{\theta}) = \mathcal{N}(b_i + \sigma_i \sum_{j=1}^F W_{ij} h_j, \sigma_i^2) \quad (3.20)$$

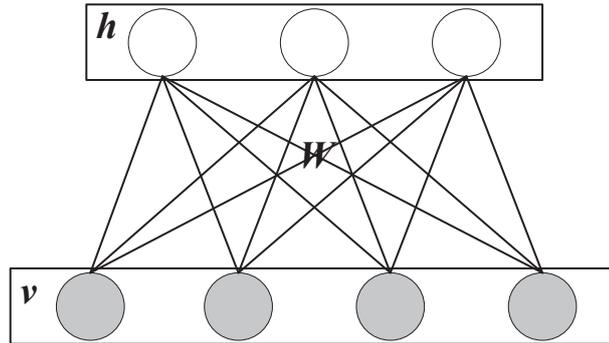


Figure 3.2 : The structure of restricted Boltzmann machine

$$P(h_j = 1 | \mathbf{v}; \boldsymbol{\theta}) = s(d_j + \sum_{i=1}^D \frac{v_i W_{ij}}{\sigma_i}) \quad (3.21)$$

Each model parameters $\theta_k \in \boldsymbol{\theta}$ can be estimated using gradient descent to minimize the negative log-likelihood:

$$-\frac{\partial \log p(\mathbf{v}; \boldsymbol{\theta})}{\partial \theta_k} = \mathbb{E}_{P(\mathbf{h}|\mathbf{v})} \left(\frac{\partial E(\mathbf{v}, \mathbf{h}; \boldsymbol{\theta})}{\partial \theta_k} \right) - \mathbb{E}_{P(\mathbf{v}, \mathbf{h})} \left(\frac{\partial E(\mathbf{v}, \mathbf{h}; \boldsymbol{\theta})}{\partial \theta_k} \right) \quad (3.22)$$

The first term, a.k.a. data-dependent expectation, is tractable but the second term, a.k.a. model-dependent expectation, is intractable and must be approximated [14]. In practice, contrastive divergence (CD) [77] is a successful algorithm which approximates the expectation with a short k-step Gibbs chain (often $k = 1$), denoted as CD_k . Moreover, Tieleman [213] proposed an improved CD algorithm, namely persistent CD.

3.3.2 Recurrent Neural Networks

A recurrent neural network (RNN) is a class of artificial neural network where connections between nodes form a directed graph along a sequence. This allows it to exhibit dynamic temporal behavior for a time sequence. RNN can use their internal state (memory) to process sequences of inputs. In practice, RNN is not capable of handling “long-term dependencies”. The problem was explored in depth

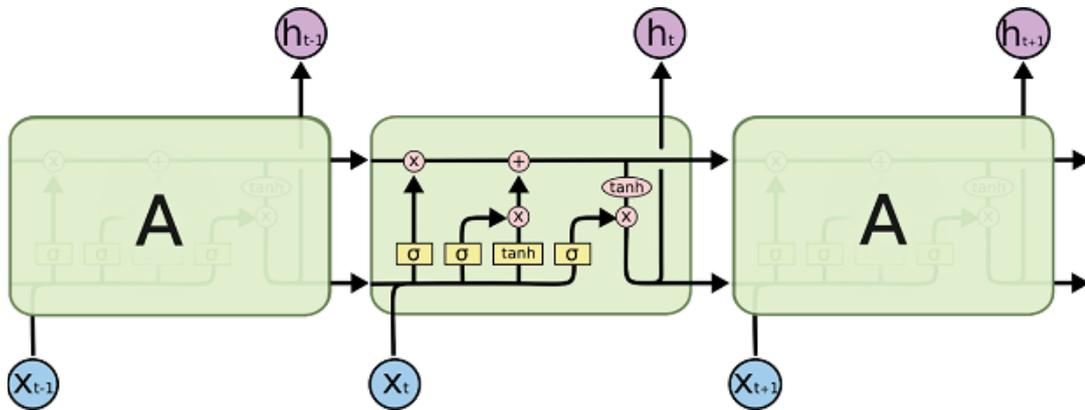


Figure 3.3 : Long short-term memory [161]

by [80], who found some pretty fundamental reasons that the accumulation of error gradients on long sequences results in the gradient vanishing or exploding.

Long Short-Term Memory

Long short-term memory (LSTM) network extends the memory module of standard RNN. Therefore, it is well suited to learn from important experiences that have very long time lags in between. There are several architectures of LSTM units. A common architecture is composed of a memory cell, an input gate, an output gate and a forget gate. Figure 3.3 depicts the architecture of this LSTM, and the corresponding update rule is given as follows:

$$\mathbf{f}_t = \sigma(\mathbf{W}_f[\mathbf{h}_{t-1}, \mathbf{x}_t])$$

$$\mathbf{i}_t = \sigma(\mathbf{W}_i[\mathbf{h}_{t-1}, \mathbf{x}_t])$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_o[\mathbf{h}_{t-1}, \mathbf{x}_t])$$

$$\tilde{\mathbf{c}}_t = \tanh(\mathbf{W}_c[\mathbf{h}_{t-1}, \mathbf{x}_t])$$

$$\mathbf{c}_t = \mathbf{f}_t * \mathbf{c}_{t-1} + \mathbf{i}_t * \tilde{\mathbf{c}}_t$$

$$\mathbf{h}_t = \mathbf{o}_t * \tanh(\mathbf{c}_t)$$

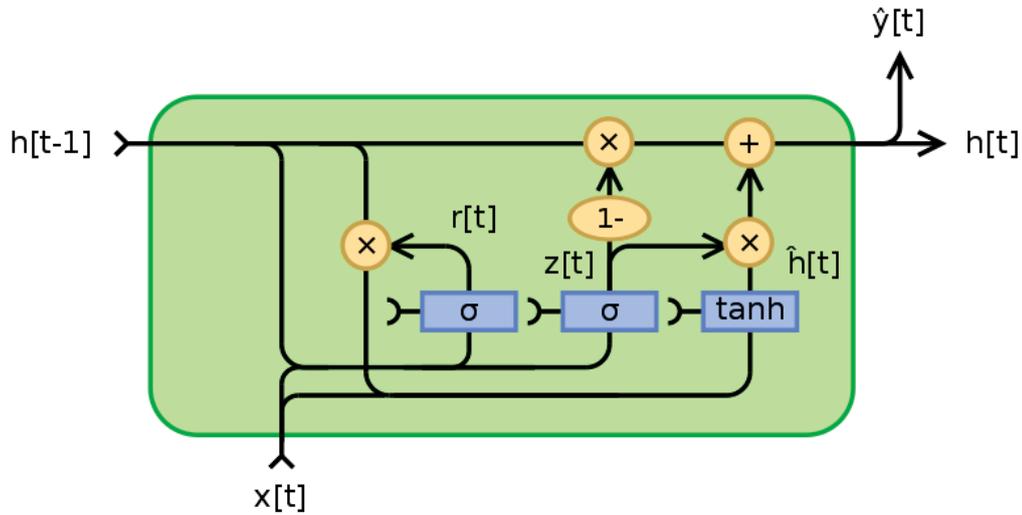


Figure 3.4 : Gated recurrent unit

where $\sigma(x) = 1/(1+\exp(-x))$ is sigmoid function, \mathbf{x}_t denotes input gate's vector, $\mathbf{f}_t \in \mathbb{R}^h$ denotes forget gate's activation vector, $\mathbf{i}_t \in \mathbb{R}^h$ denotes input gate's activation vector, $\mathbf{o}_t \in \mathbb{R}^h$ denotes output gate's activation vector, \mathbf{c}_t denotes cell state vector and \mathbf{h}_t denotes output vector, $\{\mathbf{W}_f, \mathbf{W}_i, \mathbf{W}_o, \mathbf{W}_c\}$ are the weight matrices.

Gated Recurrent Unit

Gated recurrent units (GRU) [36] are a gating mechanism in recurrent neural networks (RNN). It combines the forget and input gates into a single “update gate” and merges the cell state and hidden state. The resulting model is simpler than standard LSTM models and has been growing increasingly popular. Figure 3.4 depicts the architecture of GRU, and the corresponding update rule is given as follows:

$$\mathbf{z}_t = \sigma(\mathbf{W}_z[\mathbf{h}_{t-1}, \mathbf{x}_t])$$

$$\mathbf{r}_t = \sigma(\mathbf{W}_r[\mathbf{h}_{t-1}, \mathbf{x}_t])$$

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}[\mathbf{r}_t * \mathbf{h}_{t-1}, \mathbf{x}_t])$$

$$\mathbf{h}_t = (1 - \mathbf{z}_t) * \mathbf{h}_{t-1} + \mathbf{z}_t * \tilde{\mathbf{h}}_t$$

where $\sigma(x) = 1/(1 + \exp(-x))$ is sigmoid function, \mathbf{z}_t denotes update gate vector, \mathbf{r}_t denotes reset gate vector, \mathbf{x}_t denotes input vector and \mathbf{h}_t denotes output vector, $\{\mathbf{W}_z, \mathbf{W}_r, \mathbf{W}, \}$ are the weight matrices.

3.4 Evaluation Metrics

We conducted experiments with all the comparison methods on explicit feedback data, e.g. ratings and implicit data, e.g. clicks. The rating prediction metrics and the ranking prediction metrics are used to evaluate the performance of the accuracy of the recommendation. However, people have realized the harmfulness of evaluating RSs only using accuracy metrics [149, 186]. Therefore, in some experiments, we additionally evaluate our model and other comparison methods from the perspectives of diversity to evaluate if a model can generate a diverse recommendation result.

3.4.1 Rating Prediction

To measure the accuracy of rating prediction, we utilized the most widely used evaluation metrics, namely Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) [75].

- MAE:

$$MAE = \frac{\sum_{i=1}^N abs(Y_i - \tilde{Y}_i)}{N} \quad (3.23)$$

- RMSE:

$$RMSE = \frac{\sqrt{\sum_{i=1}^N (Y_i - \tilde{Y}_i)^2}}{N} \quad (3.24)$$

where Y_i denotes a true rating in testing set, \tilde{Y}_i is the predicted rating, and N denotes the number of ratings in the testing set.

3.4.2 Ranking Prediction

The most common way to assess the prediction performance on implicit feedback data is to measure whether relevant items are placed in the top positions of a recommendation list. Information retrieval metrics are therefore often employed to evaluate the ranking performance of a recommender system. Here, we denote $rel(k) = 1$ if the item at position k is relevant and $rel(k) = 0$ otherwise.

- **Recall@K**: Considers the fraction of relevant items over all N relevant items:

$$Recall@K = \frac{\sum_{k=1}^K rel(k)}{N} \quad (3.25)$$

- **Precision@K**: Considers the fraction of relevant items over top K recommended items:

$$Precision@K = \frac{\sum_{k=1}^K rel(k)}{K} \quad (3.26)$$

- **AP@K**: Average Precision is the average result over Precision@1~K, which is defined as:

$$AP@K = \frac{\sum_{k=1}^K rel(k) \cdot Precision@k}{\min(K, N)} \quad (3.27)$$

- **MRR@K**: Mean Reciprocal Rank evaluates any process that produces a list of possible responses to N testing query samples, ordered by probability of correctness., which is defined as:

$$MRR@K = \frac{1}{N} \sum_{k=1}^K rel(k) \frac{1}{k} \quad (3.28)$$

- **nDCG@K**: Normalized Discounted Cumulative Gain [23] is a measure of ranking quality which places strong emphasis on relevant items:

$$nDCG@K = \frac{DCG@K}{IDCG@K} \quad (3.29)$$

where IDCG means ideal DCG and

$$DCG@K = \sum_{k=1}^K \frac{2^{rel(k)} - 1}{\log_2(k + 1)}, \quad IDCG@K = \sum_{k=1}^K \frac{1}{\log_2(k + 1)}$$

- **AUC**: Area Under the ROC Curve measures the probability that the rank of relevant items \mathbf{T}^+ is higher than irrelevant items \mathbf{T}^- :

$$AUC = \frac{\sum_{i \in \mathbf{T}^+} \sum_{k \in \mathbf{T}^-} \sigma(\text{rank}(i) < \text{rank}(k))}{N \times M} \quad (3.30)$$

where $\sigma(\text{rank}(i) < \text{rank}(k))$ returns 1 if $\text{rank}(i) < \text{rank}(k)$ and 0 otherwise.

3.4.3 Diversity

Since accuracy and diversity often cannot be optimized simultaneously [251], we design the following metrics to jointly consider accuracy and diversity:

- **DIV@K**: This diversity measures the mean non-overlap ratio between each pair of recommendations $\langle \mathbf{R}_i, \mathbf{R}_j \rangle$ over all N top- K recommendations (note that the number of all possible pairs is $N(N-1)/2$).

$$DIV@K = \frac{2}{N(N-1)} \sum_{i \neq j} \left(1 - \frac{|\mathbf{R}_i \cap \mathbf{R}_j|}{K}\right) \quad (3.31)$$

- **F1@K**: The traditional F1 score is the harmonic mean of recall and precision. Here, we replace precision with diversity to jointly consider accuracy and diversity metrics.

$$F1_{MRR-DIV@K} = \frac{2(MRR@K \times DIV@K)}{MRR@K + DIV@K} \quad (3.32)$$

$$F1_{REC-DIV@K} = \frac{2(REC@K \times DIV@K)}{REC@K + DIV@K} \quad (3.33)$$

3.5 Baseline Methods for Experiments

In this section, we list the methods that will be used as the baselines to compare with our non-IID RSs introduced in following chapters.

3.5.1 Heuristic Methods

- *MostPop*: This method applies the simple strategy to rank items by their popularity (measured by the number of observations associated with items).
- *kNN*: The neighborhood-based method recommends items according to the feedback from the top-k nearest neighbors (e.g. k=10).

3.5.2 CF Methods

- *PMF* [188]: The probabilistic matrix factorization model learns the user and the item factors of from a rating matrix.
- *WRMF* [91, 165]: The weighted regularized matrix factorization model is designed to deal with implicit feedback.
- *BPR-MF* [178]: It uses Bayesian Personalized Ranking (BPR) to optimize the matrix factorization model.

3.5.3 Factorization Machines

- *FM* [175]: Factorization machines are designed to parsimoniously capture interactions between features within high dimensional sparse datasets.
- *NFM* [73]: Neural factorization machines implement FM model with neural network and has multiple hidden layers to learn non-linear interactions.

3.5.4 Content-based Methods

- *CENTROID* [157]: It creates user profiles using the centroid of all content feature from the users' articles.
- *CTR* [222]: Collaborative topic regression performs regression for users over the latent topic distribution of each article learned from LDA.

3.5.5 Relation-based Method

- *SoRec* [142]: This method jointly models the trust-link matrix and a user-item rating matrix, which shares user factors to propagate the interaction between two matrices.
- *SoReg* [141]: This method utilizes the trust relationships to construct the regularizer to learn user factors with matrix factorization.
- *SocialMF* [94]: This method is very similar to SoReg. The main difference lies in the setting of similarities for trusters.
- *CMF* [203]: Collective matrix factorization is a CDMF model. In the experiments, we couple the rating matrix of each domain on user dimension.

3.5.6 Session Methods

- *FPMC* [179]: It combines MF and first-order MC as a session-based model, which uses personalized MC for sequential prediction.
- *GRU4Rec* [76]: This is an SBRS consisting of a deep RNN with the GRU cells.
- *SWIWO* [88]: This model simulates the CBoW to model the user-session context with a shallow network architecture.

3.5.7 Cross-domain Methods

- *PARAFAC2* [70]: It is a TF model to deal with inconsistently sized matrices.
- *CDTF* [83]: This is an extended PARAFAC2 model which tunes the weight to control the influence between auxiliary domains and target domain by a genetic algorithm.

Part II

Non-IID RS: Modeling

Non-IIDness on Users

Chapter 4

A Group-based Recommender System for Modeling Group Preference over Member Decisions

4.1 Introduction

Various RSs have been applied to capture personalized requirements and offer tailored services for better user experience and new business opportunities. However, human beings are of a social nature, so various kinds of group activities are observed throughout life, e.g. seeing a movie, or planning a travel. Recently, the RS community has begun to study group behavior to make group recommendations [95]. For instance, PolyLens [164] is an early GBRS where users can create groups and ask for recommendations. Moreover, Berkovsky and Freyne [17] studied recipe recommendations for families where all members eat a meal together.

Each member of a group may have different opinions on the same items, so the main challenge in GBRS is to satisfy most group members with diverse preferences. Obviously, this is not achieved through an individual-based recommendation method. To date, the existing mainstream approaches of GBRS try to aggregate group information from individual user models [95,145]. In general, these methods can be classified into two types of models which differ in the timing of data aggregation. The first type of model is called Group Preference Aggregation (GPA), which firstly aggregates all members' ratings into a group profile, and then any individual-based recommendation approach can be used if it regards groups as virtual individual users. In contrast, the second type of model is called Individual

Preference Aggregation (IPA) which firstly predicts the individual ratings over candidate items, and then aggregates the predicted ratings of members within a group via predefined strategies to represent group ratings.

However, both these two aggregation models have their deficiencies. Quite often, only a few members will give ratings to the same items used by a group. Hence, we can hardly construct a representative group profile because each group rating is often generated merely from a single member. As a result, the recommendations may be biased towards some of the members in a group. The IPA model aggregates group recommendations using the predictive results produced by individual models. However, the recommendations made by individual models fail to consider individual behavior when she/he makes choices in the role of a group member.

In fact, quite a few heuristic strategies have been designed to work with the aggregation models. In particular, *Average* and *Least Misery* are the two most prevalent strategies [145], so they will be evaluated in this chapter. Some recent work [60, 90] studied this problem using some aggregation models. Therein, Hu et al. [90] tested the MF method under GPA and IPA models with various strategies. However, such methods are heavily dependent on the input data, which often fail to learn the representation of group preference when the data is slightly inconsistent with the aggregation assumption. In essence, these models lack the capability to build a good representation of the group preference, which we believe crucial to the success of GBRS.

The above discussions disclose the need for a GBRS to satisfy each member and the group as a whole. In this chapter, we attempt to address this challenge by employing the deep-learning technique that has been proved effective to learn high-level features [14]. Since the hypothesis behind our approach is that the individual choices are governed by collective factors when she/he acts as a group member,

we design a collective Deep Belief Network (DBN [78]) to disentangle collective embedding w.r.t. a group, according to the choices of each member. Furthermore, each group choice can be regarded as a joint decision by all members, so we can take advantage of collective embedding as the priors to model the probability of making each group choice. Accordingly, we design a dual-wing Restricted Boltzmann Machine (RBM [78]) at the top level to learn the representation of group preferences by jointly modeling group choices and collective embedding.

4.2 Problem Formulation

This chapter is aimed to learn an expressive representation of the group preferences so as to make appropriate recommendations to groups. Especially, we address the typical case of movie recommendation for households which was sponsored by CAMRa2011 Challenge [187]. Before introducing our model, we first need to give some definitions to clarify the following presentation.

- ***Collective Embedding***: it represents compromised preferences of a group, which are shared among all members and can be disentangled from the *Member Embedding*.
- ***Individual Embedding***: it represents independent individual-specific preference, which can be disentangled from the *Member Embedding* w.r.t. this user.
- ***Member Embedding***: it represents the individual preference of a user when she/he makes choices as a group member, which can be regarded as a mixture of *Collective Embedding* and *Individual Embedding*.

In particular, we study a case of movie recommendations for households, which was sponsored by CAMRa2011 [187]. To avoid the vulnerability in the group-based

recommendation, we design a deep model to represent group preference using high-level features that are learned from lower-level features. Such a deep model can effectively remove the sensitivity from data [14].

4.3 Model and Inference

Most current GBRSs are built on GPA or IPA models, so they are vulnerable to the data. To address this issue, we need to learn high-level and abstract features to replace the shallow features that are directly coupled in data.

To learn high-level features, we build a multi-layer model in terms of a deep learning technique. Using such a model, we can recover low-level features accounting for the data, and then pool low-level features to form higher-level invariant features [14]. In particular, we employ DBN and RBM as the building blocks to construct a collective DBN, where the term “collective” signifies that this DBN jointly model all members in a group as a whole. This collective DBN is capable of disentangling the collective embedding from low-level member embedding. Such collective embedding is an abstract representation of group preference, which avoids the deficiency of direct aggregation on the individual ratings. Furthermore, we design a dual-wing RBM on the top of the DBN to learn the comprehensive embedding w.r.t. each group, where one wing is connected to the group profile and the other is connected to the collective embedding learned from the collective DBN. Such a deep-structure design jointly models group choices and collective embedding, so that it can produce high-level features to represent the group preference so as to overcome the vulnerabilities in current shallow models.

4.3.1 Disentangling Collective and Individual Embedding

In individual-based RS, users independently make decisions on choosing which items, whereas in GBRS, each member needs to consider other members’ preferences

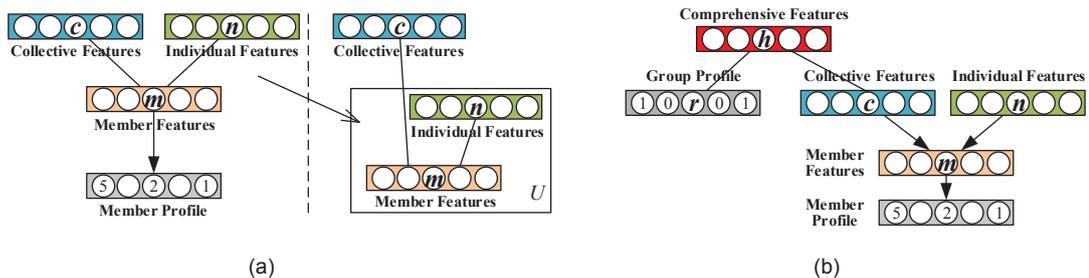


Figure 4.1 : Left: Overview of the two-layer collective DBN used to disentangle high-level collective and individual embeddings. Right: More detailed structure of the collective RBM at the top layer where the collective embedding is connected to the member embedding w.r.t. each member.

when she/he makes choices. That is, each choice is a mixed individual and collective decision. Therefore, we need to disentangle the individual and collective factors leading to the decisions.

To achieve this goal, we first learn the low-level member embedding from the member profile, i.e. the ratings given by the member, through the bottom-layer model depicted on the left of Figure 4.1. Then, we disentangle the high-level collective and individual embedding from the member embedding using the top-layer model. In particular, the top-layer model is a collective RBM as illustrated on the right of Figure 1, where the plate notation is used to represent the repeated individual and member embedding of a group, and the collective embedding is coupled with all member embeddings. To date, the most effective approach to learn the parameters of a DBN is through greedy layer-wise training using a stack of RBMs [15, 79]. In our model, the bottom-layer model is a GRBM w.r.t. each user. Such a user-based RBM model has been studied in the literature for individual-based CF [55, 189]. We simply use the same method to learn the member embedding, denoted $\mathbf{m}_u \in \mathbb{R}^D$, w.r.t. each member u , where the conditional distributions used for CD have been

given by Eq. 3.20 and 3.21.

When the member embedding is learned, we take them as the visible units to learn higher-level features. In particular, it is possible to disentangle the collective and individual embedding from the member embedding since they represent a compromised preference among all members. As shown in Figure 4.1, we construct a collective RBM for each group, where the collective embedding, denoted $\mathbf{c} \in \mathbb{R}^K$, are connected to the member embedding w.r.t. each member. Then, we can write the following energy function to describe the interaction pattern of this collective RBM.

$$E(\mathbf{m}, \mathbf{n}, \mathbf{c}; \boldsymbol{\theta}) = -\mathbf{f}^T \mathbf{c} - \sum_{u=1}^U (-\mathbf{m}_u^T \mathbf{W} \mathbf{n}_u - \mathbf{m}_u^T \mathbf{X} \mathbf{c} - \mathbf{b}^T \mathbf{m}_u - \mathbf{d}^T \mathbf{n}_u) \quad (4.1)$$

where U denotes the number of members in the group and $\boldsymbol{\theta} = \{\mathbf{W}, \mathbf{X}, \mathbf{b}, \mathbf{d}, \mathbf{f}\}$ are the model parameters. $\mathbf{W} \in \mathbb{R}^{D \times F}$ encodes the interaction between member embedding and individual embedding and $\mathbf{X} \in \mathbb{R}^{D \times K}$ encodes the interaction between member embedding and collective embedding.

Similar to the conditional distributions for a standard RBM, we can easily derive the conditional distribution w.r.t. each member embedding $m_{u,i}$, each individual embedding $n_{u,j}$ and each collective embedding c_k .

$$P(m_{u,i} = 1 | \mathbf{c}, \{\mathbf{n}_u\}; \boldsymbol{\theta}) = s(b_i + \sum_{j=1}^F W_{ij} n_j + \sum_{k=1}^K X_{ik} c_k) \quad (4.2)$$

$$P(n_{u,j} = 1 | \mathbf{m}; \boldsymbol{\theta}) = s(d_j + \sum_{i=1}^D m_{u,i} W_{ij}) \quad (4.3)$$

$$P(c_k = 1 | \mathbf{m}; \boldsymbol{\theta}) = s(f_k + \sum_{u=1}^U \sum_{i=1}^D m_{u,i} X_{ik}) \quad (4.4)$$

With these conditional distributions in hand, we can learn each parameter $\theta_i \in \boldsymbol{\theta}$ using CD. For example, the stochastic gradient descent update using CD_k is given by:

$$\theta_i \leftarrow \theta_i - \alpha \left(\frac{\partial E(\mathbf{m}^0, \mathbf{n}^0, \mathbf{c}^0; \boldsymbol{\theta})}{\partial \theta_i} - \frac{\partial E(\mathbf{m}^k, \mathbf{n}^k, \mathbf{c}^k; \boldsymbol{\theta})}{\partial \theta_i} \right) \quad (4.5)$$

where \mathbf{m}^0 are the visible data, \mathbf{n}^0 and \mathbf{c}^0 are respectively sampled from Eq. 4.3.1 and 4.3.1. $\mathbf{m}^k, \mathbf{n}^k, \mathbf{c}^k$ are the k -step sample from a Gibbs chain with the initial values $\mathbf{m}^0, \mathbf{n}^0, \mathbf{c}^0$. When the model parameters are learned, we set the value of collective embedding c_k using its expectation, i.e. $\hat{c}_k = s(f_k + \sum_{m=1}^M \sum_{i=1}^D m_{m,i} X_{ij})$, instead of a stochastic binary value to avoid unnecessary sampling noise (Hinton 2012).

4.3.2 A Comprehensive Representation of Group Preferences

GPA models create group profiles by aggregating individual ratings but, as discussed in the introduction, the recommendations may be biased towards a minority of members' taste based on such group profiles. To avoid such deficiency, the group profiles used in our approach simply consist of the group choices over items. Formally, we denote the group choices using binary ratings: $r_{gi} = 1$ indicates item i which was chosen by group g and $r_{gi} = 0$ otherwise. Given such group profiles, we can run an individual-based CF method for making recommendations by taking each group as a virtual user. However, only using such group profiles may lead to learning less expressive features because we cannot distinguish the degree of like on the same items between groups due to the identical ratings.

Each group choice is a joint decision made by all members whereas collective embedding exactly represents compromised preference of a group according to members' choices. Hence we can take advantage of the collective embedding to model the degree of like on an item, more formally, the probability of making that choice. As a result, we design a dual-wing RBM (DW-RBM) on the top of our model as illustrated in Figure 4.2, where one wing of the DW-RBM is connected to the group profile, and the other wing is connected to the collective embedding layer of the collective DBN. Under such a construction, it learns a set of comprehensive embedding that jointly models the group choices and the collective embedding. In fact, our approach can be viewed as a transfer learning model in which the collective em-

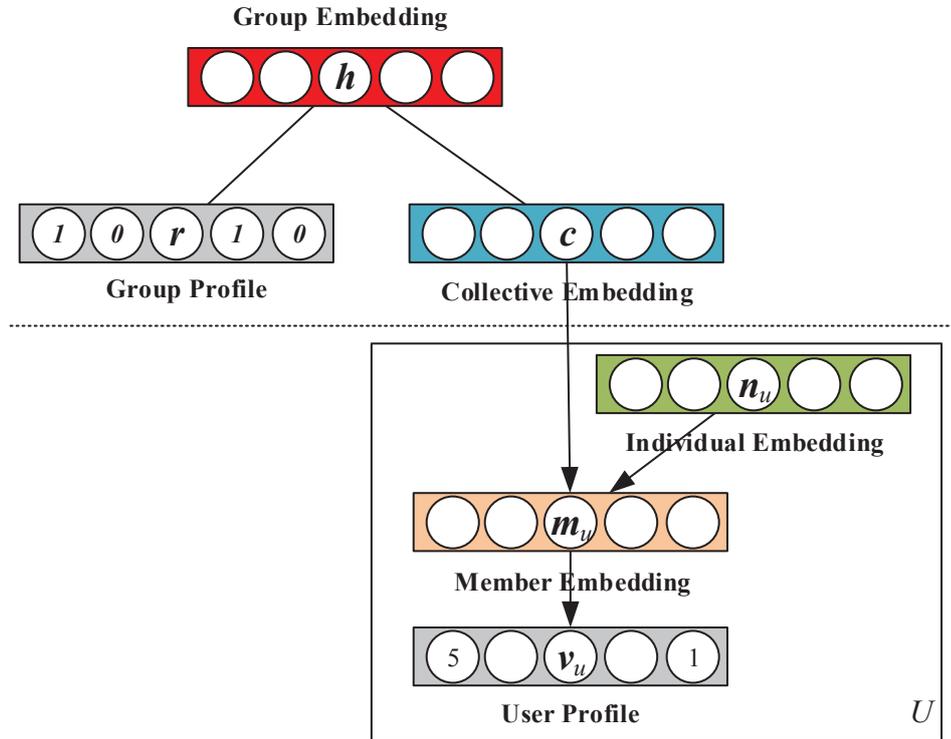


Figure 4.2 : A dual-wing RBM is placed on the top of DBN, which jointly models the group choices and collective embedding to learn the comprehensive embedding of group preference.

bedding learned from the low-level collective DBN are transferred to the high-level DW-RBM model so as to learn a more comprehensive representation of the group preferences.

For any item with a one-rating, i.e. $r_{gi} = 1$, we can say that group g is explicitly interested in item i . However, it is not certain that group g is not interested in item i or is unaware of it if $r_{gi} = 0$. Therefore, we cannot simply treat unchosen items as true-negative instances. Thus, it is a so-called “one-class” or “implicit feedback” CF problem [91, 165] as presented in Section 1.1.2. These methods use a weighted matrix factorization approach where it assigns a relatively large weight

to apply a higher penalty on the loss on fitting one-ratings and a much smaller weight to apply a lower penalty on the loss on fitting zero-ratings [91, 165]. Equally, it can be interpreted from a probabilistic view [222]: the one-rating is generated from an informative distribution with a high confidence level whereas the zero-rating is generated from a less informative distribution with a low confidence level. Following the same idea [222], we define a concentrated distribution governed by a small variance parameter for one-ratings whereas a diffuse distribution governed by a large variance parameter for zero-ratings.

$$\begin{cases} \sigma_{gi}^2 = \alpha f(g, i), & \text{if } r_{gi} = 1 \\ \sigma_{gi}^2 = \beta, & \text{if } r_{gi} = 0 \end{cases} \quad (4.6)$$

where $\beta > \alpha f(i) > 0$, α, β are constants and the function $f(g, i)$ can simply be a constant 1, or a more sophisticated form to retrieve the group satisfaction measured by some aggregation strategy. For example, if we take the *Least Misery* strategy, we can define $f(g, i) = 1/lm(g, i)$, where $lm(g, i)$ returns the least member rating on item i . That is, larger group satisfaction means smaller variance.

Following the setting of one-class CF [222], we model the group profile using Gaussian visible units with different variance parameters. Under such a construction, the energy function for the DW-RBM can be defined as follows (note that we omit the subscript g for concise, since each DW-RBM models a single group):

$$E(\mathbf{r}, \mathbf{c}, \mathbf{h}; \boldsymbol{\theta}) = \sum_{i=1}^D \frac{(r_i - b_i)^2}{2\sigma_i^2} - \sum_{k=1}^K f_k c_k - \sum_{j=1}^Y d_j h_j - \sum_{i=1}^M \sum_{j=1}^Y \frac{r_i W_{ij} h_j}{\sigma_i} - \sum_{k=1}^K \sum_{j=1}^Y c_k X_{kj} h_j \quad (4.7)$$

where $\mathbf{r} \in \{0, 1\}^M$ are the group ratings, $\mathbf{h} \in \{0, 1\}^Y$ are the comprehensive embedding and $\boldsymbol{\theta} = \{\mathbf{W}, \mathbf{X}, \mathbf{d}, \mathbf{f}\}$ are the model parameters. $\mathbf{W} \in \mathbb{R}^{M \times Y}$ encodes the interaction between \mathbf{r} and \mathbf{h} and $\mathbf{X} \in \mathbb{R}^{D \times F}$ encodes the interaction between \mathbf{c} and \mathbf{h} . According to this energy function, we can respectively obtain the conditional distribution w.r.t. each rating r_i , each collective embedding c_k , and each

comprehensive embedding h_j :

$$P(r_i|\mathbf{h}; \boldsymbol{\theta}) = \mathcal{N}(b_i + \sigma_i \sum_{j=1}^F W_{ij} h_j, \sigma_i^2) \quad (4.8)$$

$$P(c_k = 1|\mathbf{y}; \boldsymbol{\theta}) = s(f_k + \sum_{j=1}^Y X_{kj} h_j) \quad (4.9)$$

$$P(h_j = 1|\mathbf{v}, \mathbf{c}; \boldsymbol{\theta}) = s(d_j + \sum_{i=1}^M \frac{r_i W_{ij}}{\sigma_i} + \sum_{k=1}^K c_k X_{kj}) \quad (4.10)$$

Then, the model parameters $\boldsymbol{\theta}$ can be estimated by CD as demonstrated in the previous subsection.

4.3.3 Recommendation for a Group

The one-class CF approach [91, 165] ranks the items for recommendation according to the reconstructed ratings. Given a zero-rating item, the reconstructed rating tends to be relatively large if this item meets a user's preference, otherwise, it tends to be small.

In the same way, we can reconstruct a group profile using the DW-RBM. In particular, we perform a one-step mean-field reconstruction [230] instead of a stochastic reconstruction to avoid sampling noise.

$$\hat{h}_j = s(d_j + \sum_{i=1}^M \frac{r_i W_{ij}}{\sigma_i} + \sum_{k=1}^K c_k X_{kj}) \quad (4.11)$$

$$\hat{r}_i = \mathbb{E}[\mathcal{N}(b_i + \sigma_i \sum_{j=1}^F W_{ij} \hat{h}_j, \sigma_i^2)] = b_i + \sigma_i \sum_{j=1}^F W_{ij} \hat{h}_j \quad (4.12)$$

Then, we can rank the recommendation items \mathbf{C} for a group by sorting their reconstructed ratings $\{\hat{r}_i\}_{i \in \mathbf{C}}$.

4.4 Experiments

Many studies on GBRs were evaluated using synthetic group preferences created from individual profiles due to the lack of available data on group preferences.

Table 4.1 : Statistics of the evaluation data

Data	# Users/# Groups	# Ratings	Density
<i>Train_{user}</i>	602	145,069	0.0313
<i>Train_{group}</i>	290	114,783	0.051
<i>Eval_{group}</i>	286	2,139	/

However, such synthetic datasets cannot truly reflect the characteristics of group behaviors because all the individual choices are made independently. To overcome this deficiency, CAMRa2011 [187] released a real-world dataset containing the movie watching records of households and the ratings on each watched movie given by some group members. Following track 1 of CAMRa2011, we evaluated our approach and other comparison methods to compare the performance of movie recommendation for households.

4.4.1 Data Preparation

The dataset for track 1 of CAMRa2011 has 290 households with a total of 602 users who gave ratings (on a scale 1-100) over 7,740 movies. This dataset has been partitioned into a training set and an evaluation set. The training set contains 145,069 ratings given by those 602 members and 114,783 movie choice records from the view of 290 groups. That is, only 1.26 members give the rating to a watched movie. The evaluation set contains 286 groups with 2,139 group-based choices. Some statistical information is provided in Table 4.1.

4.4.2 Comparison Methods

To compare our approach with state-of-the-art methods, we evaluate the following methods, which are extended from the methods introduced in Section 3.5 with

the following settings:

- *kNN*: This is a baseline method to recommend movies watched by the top-k most similar groups.
- *MF-GPA*: This method performs PMF on the group ratings that are aggregated from individual ratings through a specified strategy.
- *MF-IPA*: This method performs PMF on individual ratings, and then aggregates the predicted ratings as the group ratings, using a specified strategy.
- *WRMF*: This method performs WRMF on the binary group ratings where the weights are set according to a specified strategy.
- *OCRBM*: This simply uses an RBM over the group choices without a connection to collective embedding. The variance parameters are set the same as the DW-RBM.
- *DLGR*: This is our deep learning approach, where the variance parameters of the DW-RBM (cf. the previous section) are set according to a specified strategy.

The evaluation metrics mean Average Precision (MAP) (cf. Eq. 3.27) and AUC (cf. Eq. 3.30) to evaluate above models over all testing groups.

4.4.3 Results

To perform a comprehensive comparison, we evaluated all comparison methods using the two most prevalent, *Average* and *Least Misery*, aggregation strategies (if applicable), in addition to the evaluation without using any strategy. Specially, we set $\beta = 1$ and $\alpha = 0.5$ (cf. Eq. 4.3.2) for OCRBM and DLGR when no strategy is used, and we set $\alpha = 1$ and $f(g, i) = 1/(1 + \log s(g, i))$ when a strategy $s(\cdot)$ is used. Also, we used similar settings for the weights of WRMF.

Table 4.2 : MAP of all comparison models with different strategies

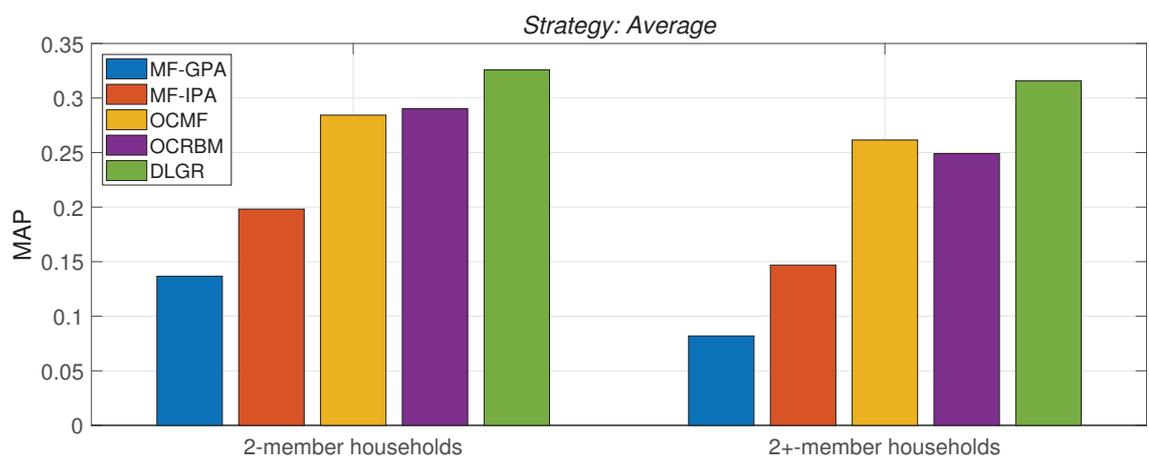
Model/Strategy	MAP		
	<i>No Strategy</i>	<i>Average</i>	<i>Least Misery</i>
kNN (k=5)	0.1595	N/A	N/A
MF-GPA	N/A	0.1341	0.0628
MF-IPA	N/A	0.1952	0.1617
WRMF	0.2811	0.2858	0.2801
OCRBM	0.2823	0.2922	0.2951
DLGR	0.3236	0.3252	0.3258

Table 4.3 : Mean AUC of all comparison models with different strategies

Model/Strategy	AUC		
	<i>No Strategy</i>	<i>Average</i>	<i>Least Misery</i>
kNN (k=5)	0.9367	N/A	N/A
MF-GPA	N/A	0.9535	0.9297
MF-IPA	N/A	0.9635	0.9503
WRMF	0.9811	0.9813	0.981
OCRBM	0.9761	0.9778	0.9782
DLGR	0.988	0.9892	0.9897

The results of MAP and mean AUC are reported in Table 4.2 and 4.3. The baseline method kNN does not achieve a good performance because it is hard to find a set of groups with identical taste over a sparse dataset. For the similar reason, MF-IPA and MF-GPA also do not perform very well. Note that MF-IPA outperforms MF-GPA. The main reason for this is that most movies are rated by only one instead of most members, so the GPA model aggregates a biased group profile. WRMF and OCRBM perform much better than MF-IPA and MF-GPA because they construct their models on the group choices instead of the aggregated ratings but use them in a more subtle way. Moreover, it is easy to see that our model DLGR marginally outperforms any other method regardless of using an aggregation strategy or not using a strategy. The main reason is that all these methods except DLGR try to directly learn a good representation of the group preference from the data. However, they may fail to learn the expressive features based on such a shallow structure. In contrast, DLGR can learn a high-level representation from low-level features through deep architecture which removes the vulnerabilities of data. In particular, DLGR outperforms its sub-model OCRBM. This is because OCRBM makes no use of the individual member choices which contain useful features determining the group choices. In comparison, DLGR provides a more robust solution which not only models the group choices but also takes advantage of the collective embedding learned from all members' choices.

In general, a group with more members implies more different preferences, so it is harder to find recommendations satisfying all members. In our problem, each household may contain 2-4 members in this dataset. A house-hold with 2 members, e.g., a couple, may easily agree on choosing a movie, whereas a household with more than 2 members, e.g., parents and children, may have different tastes due to the generation gap. Therefore, we additionally evaluated the MAP w.r.t. 2-member households and the 2^+ -member households under *Average* and *Least Misery* strategies.



(a) MAP with *Average* strategy



(b) MAP with *Least Misery* strategy

Figure 4.3 : MAP w.r.t. 2-member groups vs. 2⁺-member groups

Figure 4.3 plots the MAP w.r.t. the above two cases. We can see that DLGR outperforms all other comparison methods and the performance difference between the two cases is relatively small. Such a result proves that DLGR is still effective to represent group preference even when there are more members with different preferences. In comparison, other comparison models are constructed in a shallow manner and are more sensitive to data hence they cannot learn the best features to represent group preference when the group becomes larger.

4.5 Summary of Contributions

In this chapter, we model a GBRS with deep neural networks. In particular, the non-IID technique is focused on modeling the coupling relationships between group members for making group choices. Our main contributions are summarized as follows:

- We propose a deep architecture model to learn a high-level representation of group preferences, which avoids the vulnerability of data in traditional approaches.
- We design a collective DBN over all member profiles of a group so as to disentangle the high-level collective embedding from the low-level member embedding.
- We devise a dual-wing RBM at the top level to learn a comprehensive representation of group preferences which jointly both collective embedding and group choices.
- We conducted empirical evaluations on a real-world data set. The results prove the superiority of our approach in comparison with state-of-the-art methods.

Chapter 5

A Social Network-based Recommender Systems for Modeling User and Item Influential Contexts

5.1 Introduction

RSs essentially involve multiple relations over users, items, and their interactions. They are heterogeneous and coupled within and between each other [31], forming multiple relations which can be formalized as user-user relation, item-item relation, and user-item relation. It is increasingly recognized that leveraging these multi-relational data is important for addressing challenges in RSs, including social recommendation, cross-domain problems, sparsity and cold-start problem [20, 86, 211]. We present a multi-relational RS (MRS) which considers not only social relationships but also item relationships in this chapter. Note that the item-item relations are often defined with social information, e.g. social tagging systems. Therefore, the MRS studied in this chapter can be viewed as a generalized SNRS.

We take an example to introduce the motivation of building multi-relational RS (MRS). Figure 5.1 demonstrates a multi-relational recommendation problem over the user-user relation, e.g., friends in a social network, the item-item relation, e.g., a product line, and the user-item relation, e.g., users' purchase on items. Given a user u_t in this social network, her/his selections are often influenced by her/his friends [52]. Moreover, the influences from friends' friends transitively influence u_t 's choice. Those users whose affect u_t 's choices on item i_t form u_t 's influential context. Similarly, u_t 's selection on an item i_t is also influenced by i_t 's relevant items which form i_t 's influential context. For example, we can infer that u_t more

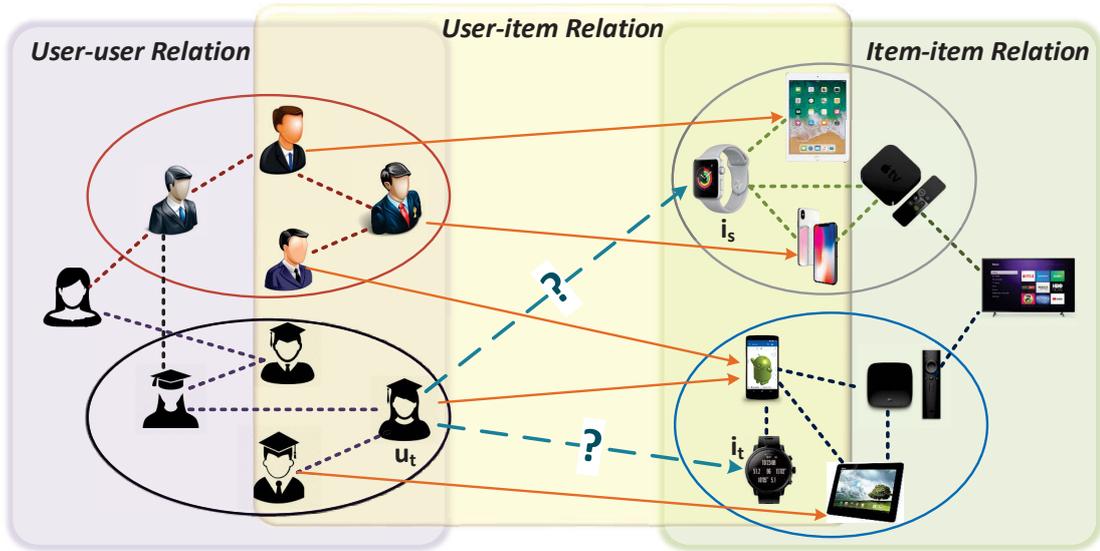


Figure 5.1 : A multi-relational RS consists of user-user relation, user-item relation and item-item relation. Each user-item interaction is influenced by the user context and the item context.

probably selects i_t (Android watch) than i_s (Apple watch) if we consider influential contexts of users and items, i.e., the selection of u_t 's friends and the compatibility of electronic products. According to this example, the incorporation of influential contexts of users and items makes recommendation more accurate and interpretable. Moreover, it is helpful to overcome the sparsity of user-item interactions and address both user and item cold-start problem by referring to the influential users and items in the contexts.

In fact, it needs to deal with various challenges to achieve this MRS. In this work, we focus on two major ones: (1) *How to model the user's/item's influential context in terms of one user-item interaction in multi-relational data?* (2) *How to learn the strength of influence from different users or items in different contexts?* These two questions are important and challenging because the influential contexts for different user-item interactions are different, and the influence from each user/item

in one context is also different.

The representative RSs which leverage user-user relation and user-item relation are SNRSs [85, 99, 211]. The main strategy they adopt is to combine collaborative filtering (CF) and social relation analysis [209, 218]. In general, there are two representative ways for social recommendation based on matrix factorization (MF): one is to co-factorize rating information and social information [94, 142] and the other is to add regularization term in optimization function [143]. However, these methods need to specify the relevance between users, which can hardly learn the high-order influence from indirect users. Moreover, SNRSs only consider user interaction and do not consider the influence of relevant items.

A possible workaround is to employ factorization machine (FM) [176] to represent multi-relational data with a design matrix [177]. However, using a single design matrix for all relations implicitly assumes the homogeneity of these relations. Obviously, user-user, item-item, and user-item relations are quite different, which may degrade the recommendation performance due to their heterogeneities. Moreover, FM is unable to model the strengths of influence from the same users/items in different influential contexts w.r.t. different target users/items.

In this work, we design Influential-context Aggregation Units (ICAU) to aggregate all user/item influences in a context into an embedding, namely influential context embedding (ICE). Taking ICAUs as the building blocks, we construct an Influential Context Embedded Multi-relation Recommender System (ICE-MRS) which considers both user’s and item’s influential contexts when conducting recommendations.

5.2 Problem Formulation

As illustrated in Figure 5.1, we build an MRS with user-user, user-item and user-item relations for more effective recommendation. Let u and $\mathcal{U} = \{u_1, u_2, \dots, u_{|\mathcal{U}|}\}$ denote a user and the whole user set. $\mathcal{R}_{\mathcal{U}}$ denotes the user-user relation. Let i and $\mathcal{I} = \{i_1, i_2, \dots, i_{|\mathcal{I}|}\}$ denote an item and the whole item set. $\mathcal{R}_{\mathcal{I}}$ denotes the item-item relation. The user-item interactions are denoted as $\mathcal{R}_{\mathcal{UI}} = \{y_{u,i} | u \in \mathcal{U}, i \in \mathcal{I}\}$, and $y_{u,i}$ could be explicit feedback, e.g., ratings or implicit feedback, e.g., clicks. Generally, each user-item interaction $y_{u,i}$ is not only decided by user u and item i but also other users and items in the influential context [31]. Hence, we formally define a user's and an item's influential contexts to model the user-item interactions.

Definition 5.1: User Influential Context (UIC): Given a target user u , the UIC denotes $\mathcal{C}_u = \{\mathcal{U}_u, \mathcal{R}_u\}$, where $\mathcal{U}_u = \{u, \mathcal{U}_u^c\}$ consists of target user u and all influential users relevant to u , \mathcal{R}_u denotes the user relationships over \mathcal{U}_u .

Definition 5.2: Item Influential Context (IIC): Given a target item i , the IIC denotes $\mathcal{C}_i = \{\mathcal{I}_i, \mathcal{R}_i\}$, where $\mathcal{I}_i = \{i, \mathcal{I}_i^c\}$ consists of target item i and all influential items relevant to i , \mathcal{R}_i denotes all the item relationships over \mathcal{I}_i .

Interaction Score Decomposition: Each user-item interaction $y_{u,i}$ can be measured by a score function s in terms of the UIC \mathcal{C}_u and the IIC \mathcal{C}_i ; formally, $s : s(\mathcal{C}_u, \mathcal{C}_i, y_{u,i}) \mapsto s_{\langle \mathcal{C}_u, \mathcal{C}_i \rangle}$. According to Definitions 5.1 and 5.2, the overall interaction score $s_{\langle \mathcal{C}_u, \mathcal{C}_i \rangle}$ can be decomposed into four scores:

$$s_{\langle \mathcal{C}_u, \mathcal{C}_i \rangle} = \lambda_1 s_{\langle u, i \rangle} + \lambda_2 s_{\langle u, \mathcal{I}_i^c \rangle} + \lambda_3 s_{\langle \mathcal{U}_u^c, i \rangle} + \lambda_4 s_{\langle \mathcal{U}_u^c, \mathcal{I}_i^c \rangle} \quad (5.1)$$

where $s_{\langle u, i \rangle}$ scores user u 's preference on item i ; $s_{\langle u, \mathcal{I}_i^c \rangle}$ scores u 's preference on influential items \mathcal{I}_i^c ; $s_{\langle \mathcal{U}_u^c, i \rangle}$ scores relevant users' preference on i , and $s_{\langle \mathcal{U}_u^c, \mathcal{I}_i^c \rangle}$ scores the subsidiary preference between influential users and influential items. $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ are the scale parameters for weighing these scores.

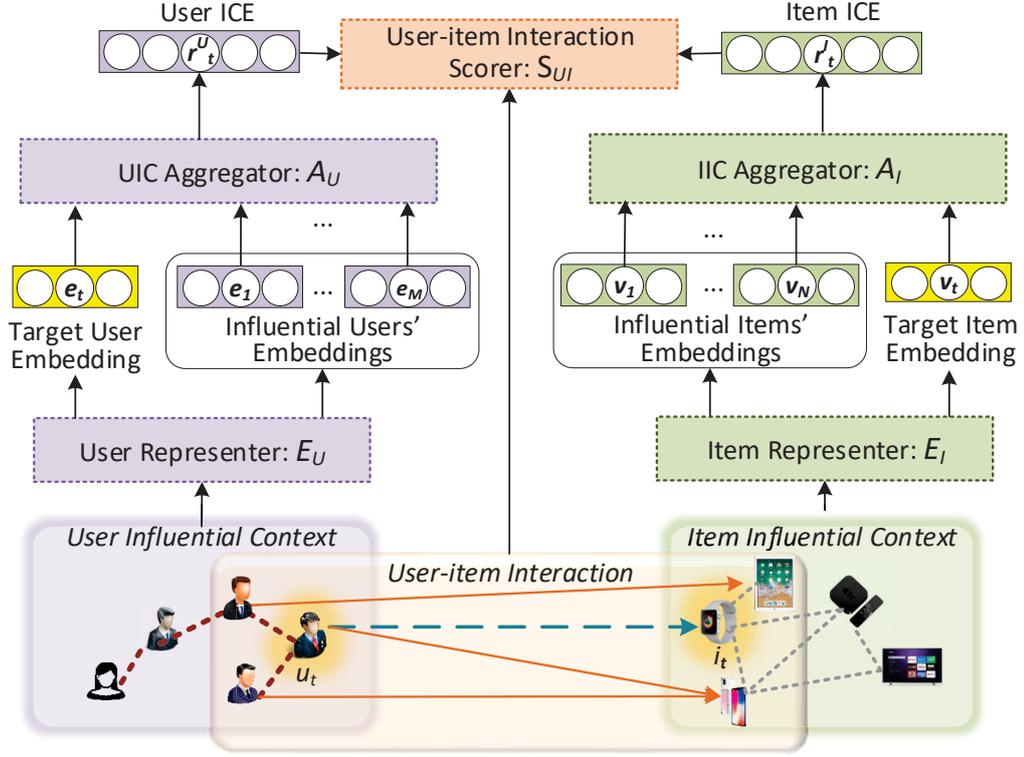


Figure 5.2 : The architecture of ICE-MRS for modeling user's and item's influential contexts

5.3 Model and Learning

In this section, we present an Influential Context Embedded MRS (ICE-MRS) with the neural network model. Specially, we design the influential-context aggregation units (ICAU) to learn ICEs as the core component of this ICE-MRS.

5.3.1 Architecture

The architecture of the ICE-MRS is illustrated in Figure 5.2, which consists of five components: User Representer E_U , UIC Aggregator A_U , IIC Representer, IIC Aggregator A_I and user-item interaction scorer S_{UI} . Given a target user u_t and the corresponding UIC \mathcal{C}_{u_t} , a target item i_t and the corresponding IIC \mathcal{C}_{i_t} :

- User Representer E_U : it maps target user u_t and its influential users in UIC to the corresponding user embeddings, i.e., $E_U(\mathcal{U}_{u_t}) \mapsto \mathcal{E}_{u_t}$ where $\mathcal{E}_{u_t} = \{\mathbf{e}_t, \mathbf{e}_1, \dots, \mathbf{e}_M\}$.
- Item Representer E_I : it maps target item i_t and its influential items in IIC to the corresponding item embeddings, i.e., $E_I(\mathcal{I}_{i_t}) \mapsto \mathcal{E}_{i_t}$ where $\mathcal{E}_{i_t} = \{\mathbf{v}_t, \mathbf{v}_1, \dots, \mathbf{v}_N\}$.
- UIC Aggregator A_U : it learns a representation \mathbf{r}_t^U for the influential context \mathcal{C}_{u_t} , namely influential context embedding (ICE). Formally, we have $A_U(\mathcal{C}_{u_t}, \mathcal{E}_{u_t}) \mapsto \mathbf{r}_t^U$.
- IIC Aggregator A_I : it learns i_t 's ICE by aggregating the influential context \mathcal{C}_{i_t} , that is, $A_I(\mathcal{C}_{i_t}, \mathcal{E}_{i_t}) \mapsto \mathbf{r}_t^I$.
- User-item Interaction Scorer S_{UI} : it learns to score the interaction strength between the target user-item pair $\langle u_t, i_t \rangle$ in terms of the user ICE \mathbf{r}_t^U and the item ICE \mathbf{r}_t^I , namely $S_{UI}(\mathbf{r}_t^U, \mathbf{r}_t^I, y_{u_t, i_t}) \mapsto s_{\langle \mathcal{C}_u, \mathcal{C}_i \rangle}$ (cf. Eq. 5.1).

This architecture can be implemented with many concrete methods, e.g. the mixture model. In this work, we implement it by neural network models which have been proved most effective and efficient in recent years.

5.3.2 Influential-context Aggregation Unit

The ICAUs aim to aggregate the user embeddings \mathcal{E}_{u_t} or the item embeddings \mathcal{E}_{i_t} in a context into an ICE according to the strength of influence from each user or item. Figure 5.3 demonstrates an ICAU for aggregating user embeddings \mathcal{E}_{u_t} , which consists of a two-stage aggregation: S1 and S2.

S1: This stage outputs the subsidiary influence embedding \mathbf{c}_t through an aggre-

gation function $h(\cdot)$ over the influential users' embeddings:

$$\{\alpha_1, \dots, \alpha_K\} = a(\mathbf{e}_1, \dots, \mathbf{e}_K) \quad (5.2)$$

$$\mathbf{c}_t = h(\mathbf{e}_1, \dots, \mathbf{e}_K | \alpha_1, \dots, \alpha_K). \quad (5.3)$$

where α_i denotes the influential strength modeled by a function $a(\cdot)$.

S2: This stage generates the ICE by aggregating the subsidiary influence context embedding \mathbf{c}_t and the target embedding \mathbf{e}_t through a gate function $f(\cdot)$:

$$g = f(\mathbf{c}_t, \mathbf{e}_t) \quad (5.4)$$

$$\mathbf{r}_t = g\mathbf{c}_t + (1 - g)\mathbf{e}_t \quad (5.5)$$

In ICAU, h and f could be any linear or non-linear functions. In this work, we implement h by multilayer neural networks in terms of the attention mechanism. And f is implemented with a gate neural network. Note that the ICAUs can be used in cascade to aggregate higher order ICEs, which is presented in the next subsection.

5.3.3 User's Influential Context Embedding

Given a user influential context $\mathcal{C}_{u_t} = \{\mathcal{U}_{u_t}, \mathcal{R}_{u_t}\}$ (cf. Definition 5.1), \mathcal{U}_{u_t} consists of the target user u_t and her/his first-order influential neighbors $\{u_{t,m}\}_{1 \leq m \leq M}$, and each $u_{t,m}$'s neighbors $\{u_{t,m,k}\}_{1 \leq k \leq K_m}$, i.e., second-order influential neighbors of u_t , according to the relationships \mathcal{R}_{u_t} .

To learn the ICE from the UIC \mathcal{C}_{u_t} , we first employ the User Representer E_U to extract the user embeddings for all users in \mathcal{U}_{u_t} . The embedding for a first-order neighbor $u_{t,m}$ is denoted as $\mathbf{e}_{t,m}$, and the embedding for a second-order neighbor $u_{t,m,k}$ is denoted as $\mathbf{e}_{t,m,k}$. To generate the ICE for \mathcal{C}_{u_t} , we construct an ICAU-based two-level tree-like model as the UIC Aggregator A_U to recursively embed both second-order and first-order neighbors' influences. Firstly, the second level ICAUs are used to generate the ICE $\mathbf{r}_{t,m}$ w.r.t. each first-order neighbor $u_{t,m}$. Then, the first

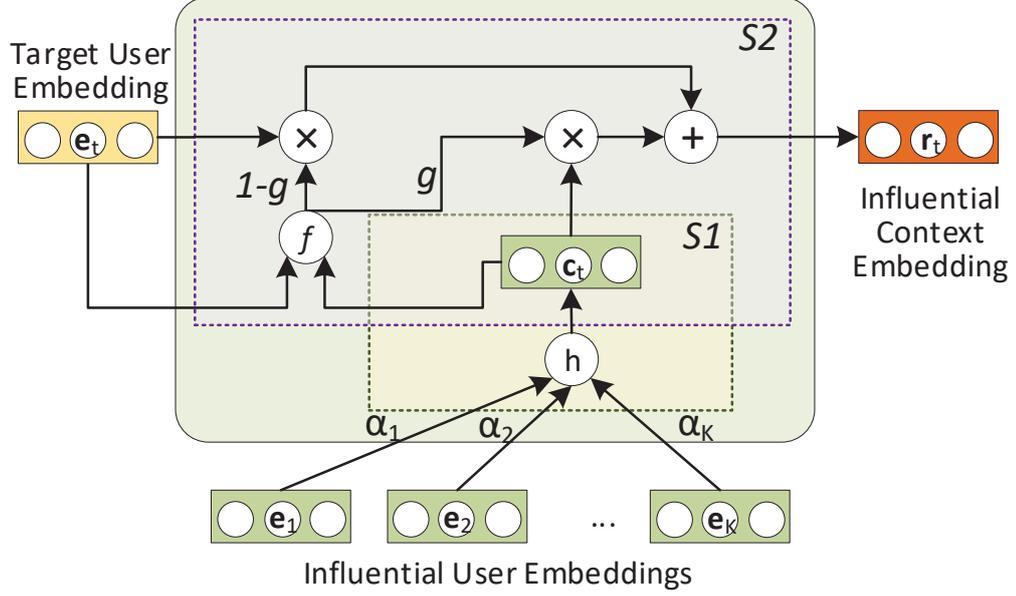


Figure 5.3 : Influential-context Aggregation Unit (ICAU): A two-stage aggregation model to construct influential context embedding (ICE)

level ICAU recursively generates the target ICE \mathbf{r}_t^U by aggregating these first-order ICEs $\{\mathbf{r}_{t,m}\}$. More details are presented as follows.

The Second Level ICAUs

Taking a first-order influential neighbor $u_{t,m}$ of target user u_t as an example, we present how to implement an ICAU with neural networks for generating the ICE $\mathbf{r}_{t,m}$ w.r.t. $u_{t,m}$ in two stages.

S1: We adopt the attention mechanism to model the influential strength for each neighbor $u_{t,m,k}$ of $u_{t,m}$. Specifically, we construct a three-layer attention neural network to weight the influence according to the user embedding $\mathbf{e}_{t,m,k}$. First, $\mathbf{e}_{t,m,k}$ is projected into hidden units by a tanh layer to capture nonlinear interaction:

$$\mathbf{h}_{t,m,k} = \tanh(\mathbf{W}^{(1)}\mathbf{e}_{t,m,k} + \mathbf{b}) \quad (5.6)$$

where the weight matrix $\mathbf{W}^{(1)} \in \mathbb{R}^{L \times L}$ and we omit the bias term \mathbf{b} in the following equations for concision.

Then, the normalized influence for each neighbor is scored by the softmax(x_k) = $e^{x_k} / \sum_j e^{x_j}$ function:

$$\alpha_{t,m,k}^U = \text{softmax}\left(f_{ISR(\theta)}(\mathbf{W}^{(2)}\mathbf{h}_{t,m,k})\right) \quad (5.7)$$

where the weight matrix $\mathbf{W}^{(2)} \in \mathbb{R}^{1 \times L}$ and $f_{ISR(\theta)}$ is an inverse square root unit which is defined as follows:

$$f_{ISR(\theta)}(x) = \frac{x}{\sqrt{1 + \theta x^2}} \quad (5.8)$$

where θ is the parameter which decides the range of ISR function. A larger θ results in a smaller range of ISR. In general, softmax tends to enlarge the difference of input values, so shrinking the range of input values with the ISR function could learn the influence from each input. Besides, ISR function can bound the inputs for softmax to avoid overflow of the exponential function.

Further, the subsidiary influence embedding $\mathbf{c}_{t,m}$ is aggregated from the influential neighbors' embeddings $\{\mathbf{e}_{t,m,k}\}$ according to their influence strengths $\{\alpha_{t,m,k}^U\}$:

$$\mathbf{c}_{t,m} = \sum_{k=1}^{K_m} \alpha_{t,m,k}^U \mathbf{e}_{t,m,k} \quad (5.9)$$

S2: The ICE $\mathbf{r}_{t,m}$ w.r.t. the first-order user $u_{t,m}$ is calculated as follows:

$$\mathbf{r}_{t,m} = g_{t,m}\mathbf{c}_{t,m} + (1 - g_{t,m})\mathbf{e}_{t,m} \quad (5.10)$$

where g measures the influence strength from the second-order neighbors. The influential gate g is modeled by a gate neural network:

$$g_{t,m} = \sigma\left(f_{ISR(\theta)}(\mathbf{W}^{(3)} \tanh(\mathbf{W}^{(4)}\mathbf{c}_{t,m} + \mathbf{W}^{(5)}\mathbf{e}_{t,m}))\right) \quad (5.11)$$

where $\sigma(z) = 1/(1 + e^{-z})$, $\mathbf{W}^{(4)}, \mathbf{W}^{(5)} \in \mathbb{R}^{L \times L}$ and $\mathbf{W}^{(3)} \in \mathbb{R}^{1 \times L}$.

The First Level ICAU

When we obtain the ICEs $\{\mathbf{r}_{t,m}\}$ w.r.t. all first-order influential neighbors, another ICAU at the first level is used to learn the target ICE:

$$\mathbf{c}_t^U = \sum_{m=1}^M \alpha_{t,m}^U \mathbf{r}_{t,m} \quad (5.12)$$

where $\alpha_{t,m}^U$ denotes the influence strength w.r.t. the first-order ICE $\mathbf{r}_{t,m}$, which is calculated by another three-layer attention network having the same form of Eq. 5.6 and 5.7. Then, we get the ICE \mathbf{r}_t^U w.r.t. the target user u_t :

$$\mathbf{r}_t^U = g_t^U \mathbf{c}_t^U + (1 - g_t^U) \mathbf{e}_t \quad (5.13)$$

where g_t^U is learned by a gate neural network which has the same structure with Eq. 5.11.

5.3.4 Item's Influential Context Embedding

Given an IIC $\mathcal{C}_i = \{\mathcal{I}_i, \mathcal{R}_i\}$, \mathcal{R}_i is often built with the item relevance. Different from the indirect influence in user-user relation modeling, a user normally only consider those items directly relevant to the target item when they make a choice. Therefore, we only consider the first-order influential neighbors of a target item for modeling IIC. Given $\{i_{t,n}\}_{1 \leq n \leq N}$ w.r.t. target item i_t , their corresponding embeddings \mathbf{v}_t and $\{\mathbf{v}_{t,n}\}_{1 \leq n \leq N}$ are retrieved by Item Representer E_I . As a result, we use an ICAU to learn the item ICE. The subsidiary influence embedding \mathbf{c}_t^I is calculated as:

$$\mathbf{c}_t^I = \sum_{n=1}^N \alpha_{t,n}^I \mathbf{v}_{t,n} \quad (5.14)$$

where the influential strength $\alpha_{t,n}^I$ of $\mathbf{v}_{t,n}$ is calculated by a three-layer attention network as in the UIC Aggregator.

$$\alpha_{t,n}^I = \text{softmax}\left(f_{ISR(\theta)}\left(\mathbf{W}^{(6)} \tanh\left(\mathbf{W}^{(7)} \mathbf{v}_{t,n}\right)\right)\right) \quad (5.15)$$

where $\mathbf{W}^{(6)} \in \mathbb{R}^{L \times L}$ and $\mathbf{W}^{(7)} \in \mathbb{R}^{1 \times L}$.

Subsequently, the ICE \mathbf{r}_t^I w.r.t. i_t is obtained by aggregating the subsidiary influence embedding \mathbf{c}_t^I and the target item embedding \mathbf{v}_t :

$$\mathbf{r}_t^I = g_t^I \mathbf{c}_t^I + (1 - g_t^I) \mathbf{v}_t \quad (5.16)$$

where the influential gate g_t^I is also learned from a three-layer gate neural network as in Figure 5.3.

$$g_t^I = \sigma\left(f_{ISR(\theta)}\left(\mathbf{W}^{(8)} \tanh\left(\mathbf{W}^{(9)} \mathbf{c}_t^I + \mathbf{W}^{(10)} \mathbf{v}_t\right)\right)\right) \quad (5.17)$$

where $\mathbf{W}^{(8)}, \mathbf{W}^{(9)} \in \mathbb{R}^{L \times L}$ and $\mathbf{W}^{(10)} \in \mathbb{R}^{1 \times L}$.

5.3.5 User-item Interaction Ranking

Each user-item connection denotes a user's selection on an item. User-item connections can be regarded as one-class preference data [86] which cannot differentiate user preferences. To handle the one-class problem, we treat the learning on the user-item interactions as a ranking problem [178]. Given a user u_t , we construct a contrastive item pair to specify the preference order. A positive item i_p is the one which has an observed connection to u_t in user-item relationships, i.e. a user-selected item, while a pseudo-negative item i_n refers to the one without connection to u_t . Then, we have the preference order $\langle u_t, i_p \rangle \succeq \langle u_t, i_n \rangle$. Accordingly, we have $S_{\langle u_t, i_p \rangle} \geq S_{\langle u_t, i_n \rangle}$ where $S_{\langle u_t, i \rangle}$ denotes the preference score on item i by the inner product of \mathbf{r}_t and \mathbf{r}_i :

$$S_{\langle u_t, i_t \rangle} = \mathbf{r}_t^{u \top} \mathbf{r}_t^I \quad (5.18)$$

Then, we use the max-margin loss [121] to optimize the ranking order over pairs:

$$L_{\langle u_t, i_p \rangle \succeq \langle u_t, i_n \rangle} = \max\{0, m - S_{\langle u_t, i_p \rangle} + S_{\langle u_t, i_n \rangle}\} \quad (5.19)$$

where $m = 10$ is set as the maximum margin in this chapter.

Remark for Scoring Model: if we expand Eq. 5.18, we obtain the following form by using Eqs. 5.13 and 5.16:

$$\begin{aligned}
S_{\langle u_t, i_t \rangle} &= (g_t^U \mathbf{c}_t^U + (1 - g_t^U) \mathbf{e}_t)^\top (g_t^I \mathbf{c}_t^I + (1 - g_t^I) \mathbf{v}_t) \\
&= (1 - g_t^U)(1 - g_t^I) \mathbf{e}_t^\top \mathbf{v}_t + g_t^U (1 - g_t^I) \mathbf{c}_t^{u\top} \mathbf{v}_t \\
&\quad + (1 - g_t^U) g_t^I \mathbf{e}_t^\top \mathbf{c}_t^I + g_t^U g_t^I \mathbf{c}_t^{u\top} \mathbf{c}_t^I
\end{aligned} \tag{5.20}$$

According to Eq. 5.1, we find that (i) $s_{\langle u, i \rangle}$ is modeled by $\mathbf{e}_t^\top \mathbf{v}_t$; (ii) $s_{\langle u, \mathcal{I}_i^c \rangle}$ is modeled by $\mathbf{e}_t^\top \mathbf{c}_t^I$; (iii) $s_{\langle U_u^c, i \rangle}$ is modeled by $\mathbf{c}_t^{u\top} \mathbf{v}_t$, and (iv) $s_{\langle U_u^c, \mathcal{I}_i^c \rangle}$ is modeled by $\mathbf{c}_t^{u\top} \mathbf{c}_t^I$. Correspondingly, $\lambda_1 = (1 - g_t^U)(1 - g_t^I)$, $\lambda_2 = g_t^U(1 - g_t^I)$, $\lambda_3 = (1 - g_t^U)g_t^I$ and $\lambda_4 = g_t^U g_t^I$ are learned to weigh these scores. Therefore, the above-expanded terms provide an insight into how the influences in the UIC and the IIC are embedded in our model to affect the final recommendation.

5.3.6 Training Procedure

For each user selection $\langle u_t, i_p \rangle$, we can construct a triplet $\langle u_t, i_p, i_n \rangle$ to optimize the ranking loss $L_{\langle u_t, i_p \rangle \succeq \langle u_t, i_n \rangle}$. Then the loss of a mini-batch \mathcal{B} for training is given as:

$$\mathcal{L} = \frac{1}{|\mathcal{B}|} \sum_{\langle u_t, i_p, i_n \rangle \in \mathcal{B}} L_{\langle u_t, i_p \rangle \succeq \langle u_t, i_n \rangle} \tag{5.21}$$

To learn the parameters, we adopt a gradient decent-based algorithm over $\partial \mathcal{L} / \partial \mathbf{W}$ w.r.t. each weight matrix \mathbf{W} in our model. We implement our model using Keras [37] with Tensorflow GPU version as backend. We use Adam [111] as the gradient optimizer and the mini-batch size is set to 200.

5.4 Experiments

In this section, we conduct experiments on two real datasets to compare the recommendation quality of our approach with other state-of-the-art recommendation methods.

5.4.1 Data Preparation

Most public datasets for recommendation only consider user-item relation, so a dataset contains multiple relations, i.e., user-user, user-item, and item-item, is not easy to find. Fortunately, two datasets, Delicious* and Lastfm†, provided by RecSys Challenge 2011 [27] can satisfy our requirement.

The Delicious dataset contains social network, bookmarking, and tagging information from Delicious social bookmarking system. A contact relation is identified between two users when they belong to a mutual fan relation in Delicious, which is used as the user-user relation. The user-item relation is constructed from users and their bookmarked items. The item-item relation is built on the common tags between items. Given a target item, we assign links to top 10 items which have the most common tags with.

The Lastfm dataset contains social network, tagging, and music artist information from Last.fm online music system. The friend relationships between users are used as the user-user relation. The items are artists which are connected with users if the artists are listened by these users. The listening relationships between users and artists are served as the user-item relation. The item-item relation is built through the tags using the same method as Delicious. The statistics of the datasets are summarized in Table 5.1 which contains the number of entities (i.e., users, items and users+items), number of links in the user-user, the item-item and the user-item relations and the sparsity in each type of relation. Since both datasets are very sparse, they will benefit from the additional information from the social network and item-item relation for recommendation.

*<http://www.delicious.com>

†<http://www.last.fm>

Table 5.1 : Statistics of the datasets: Delicious and Lastfm

	Property	User-user	Item-item	User-Item
Delicious	#Entity	1,892	17,632	1,892+17,632
	#Links	25,434	199,827	104,799
	Sparsity	0.0142	0.0012	0.0031
Lastfm	#Entity	1,867	69,226	1,867+69,226
	#Links	15,328	682,314	92834
	Sparsity	0.0288	0.0002	0.0007

5.4.2 Experimental Settings

To evaluate the ranking accuracy, we adopt two metrics: *Mean Average Precision* (MAP) and *normalized Discounted Cumulative Gain* (nDCG), to measure the quality of preference ranking and top-N recommendation.

Comparison Methods

In addition to ICE-MRS, the baseline methods are extended from the methods introduced in Section 3.5 with the following settings:

- *BPR-MF*: It uses BPR optimization based matrix factorization on the user-item relation.
- *SoRec*: It jointly factorizes the social relationship matrix and a user-item interaction matrix.
- *SoicalMF*: It adds regularization into MF according to the social network.
- *SoReg*: It leverage social relationships to regularize the users latent factors.

- *CMF*: It is model with three coupled matrices for the user-user, the user-item and the item-item relation respectively.
- *FM*: It embeds features into a latent space and models the interactions between each two features. We integrate user-user, user-item, item-item relations as the features.
- *NFM*: It extends FM model with the deep neural networks.

Parameter Settings

The lengths of user/item embeddings and context embeddings are set to 128. To accelerate the model, we choose top 10 first-order neighbors for each target user and 10 second-order neighbors for each first-order user, and 10 neighbors for each item in the item-item relation ranked by the influence weights, cf. Eq. 5.7. The θ in Eq. 5.19 is set to 16.

5.4.3 Recommendation Performance

We construct the testing set by holding out 20% user-item interactions as the ground truths, accompanying with ten times user-item pairs without interaction as the contrastive samples, i.e., i_n . The remaining data of user-item relationships are used for training, together with user-user and item-item relationships.

Overall Comparison

Table 5.2 reports MAP and nDCG at 5 and 10 over all testing users. Among all methods, our method achieves the best performance in terms of all metrics on both datasets. Specifically, ICE-MRS demonstrates an approximate 20% improvement over the second-place method NFM on Delicious and 6.1% improvement over the second-place method CMF on Lastfm in terms of MAP@5.

Table 5.2 : Item recommendation for test users of Delicious and Lastfm

Delicious				
	MAP@5	MAP@20	nDCG@5	nDCG@20
<i>BPR-MF</i>	0.4157	0.3225	0.4318	0.3744
<i>SoRec</i>	0.4174	0.3390	0.4476	0.3965
<i>SocialMF</i>	0.4181	0.3409	0.4520	0.4017
<i>SoReg</i>	0.4239	0.3444	0.4577	0.4056
<i>CMF</i>	0.4375	0.3507	0.4739	0.4158
<i>FM</i>	0.4246	0.3363	0.4522	0.3896
<i>NFM</i>	0.4565	0.3754	0.4924	0.4347
<i>ICE-MRS</i>	0.5477	0.4200	0.6064	0.5273
Lastfm				
<i>BPR-MF</i>	0.5154	0.4586	0.6252	0.6334
<i>SoRec</i>	0.5350	0.4775	0.6412	0.6457
<i>SocialMF</i>	0.5489	0.4907	0.6544	0.6575
<i>SoReg</i>	0.5495	0.4878	0.6548	0.6541
<i>CMF</i>	0.5530	0.4928	0.6549	0.6749
<i>FM</i>	0.5366	0.4837	0.6453	0.6723
<i>NFM</i>	0.5462	0.4885	0.6516	0.6702
<i>ICE-MRS</i>	0.5865	0.5302	0.6913	0.7021

Effect of User-user Relation Modeling

BPR-MF more easily suffers from data sparsity than other comparison methods, because it cannot borrow information from the social relation. As a result, it achieves the worst performance. In comparison, other methods benefit from the information

from social network. Especially, ICE-MRS outperforms all other methods, which should thank to ICAU for precisely weighing the influence from different users and aggregating high-order influence.

Effect of Item-item Relation Modeling

CMF, FM and NFM consider item-item relation in addition, which makes the recommendation more effective than social relation only methods. Compared with MF and FM-based methods, ICE-MRS demonstrates its superiority on integrating multiple relational data with the influence aggregation modeling.

5.4.4 Recommendation for Cold-start Users and Items

The cold-start problem is ubiquitous in RS which can be categorized into cold-start users and cold-start items. In the user cold-start problem, for new users, we recommend items to them. The recommended items' ranking is regarded as the evaluation criteria. In the item cold-start problem, for new items, we recommend them to different users. The ranks over the recommended users are used to evaluate the performance. In this section, we show the ability of comparison methods for handling user and item cold-start problem respectively.

Cold-start Users

To test the recommendation for cold-start users, we randomly remove 20% users and all their links from the user-item relation as the training set. BPR-MF only considers the user-item relation, so it cannot be used in the cold-start scenario. The performance compared with other methods are shown in Figure 5.4. ICE-MRS significantly outperforms other methods on both datasets. On Delicious, in terms of MAP@5, the relative improvement of ICE-MRS over SoRec, Social MF, SoReg, CMF, FM and NFM is 42.3%, 32.2%, 29.4%, 21.6%, 19.7% and 12.6% respectively. On Lastfm, in terms of MAP@5, the relative improvement of ICE-

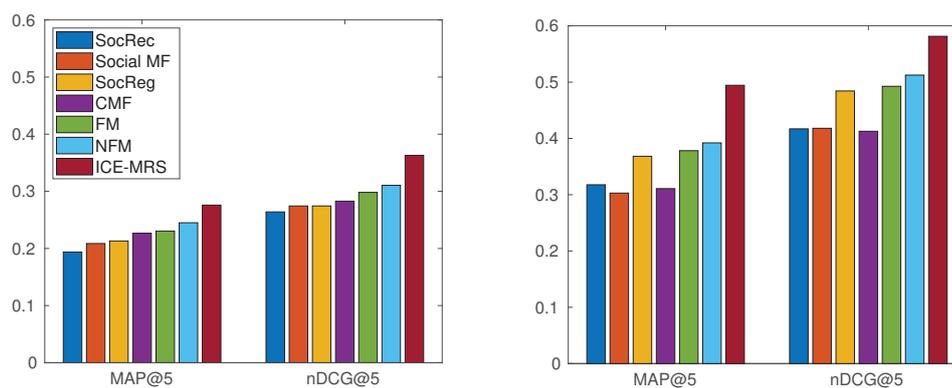


Figure 5.4 : Item recommendation for cold-start users of Delicious and Lastfm.

MRS over SoRec, Social MF, SoReg, CMF, FM and NFM is 55.1%, 58.2%, 36.6%, 54.2% 30.7% and 26.1% respectively. Eq. 5.20 gives the insight into why ICE-MRS can handle this cold-start problem more effectively. It is because the ICEs of users have embedded the information from their neighbors in the influential context, even when no historical user selection is observed.

Cold-start Items

To test the case of cold-start items, we randomly remove 20% items and their all connected edges from user-item relationships. For each testing item, we rank the predictive scores over users. The MAP and nDCG results are shown in Figure 5.5. Since *SoRec*, Social MF and SoReg only model user relation, they cannot handle cold-start items. Compared with CMF, FM and NFM, ICE-MRS achieves the best performance on both datasets. Note that ICE-MRS's performance on cold-start items is not significant as that for cold-start users. This can be interpreted that the impact from other users are more influential than the impact from relevant items when users' making selection. However, the embedded influential item context can still provide useful information when no selection is available for a new item.

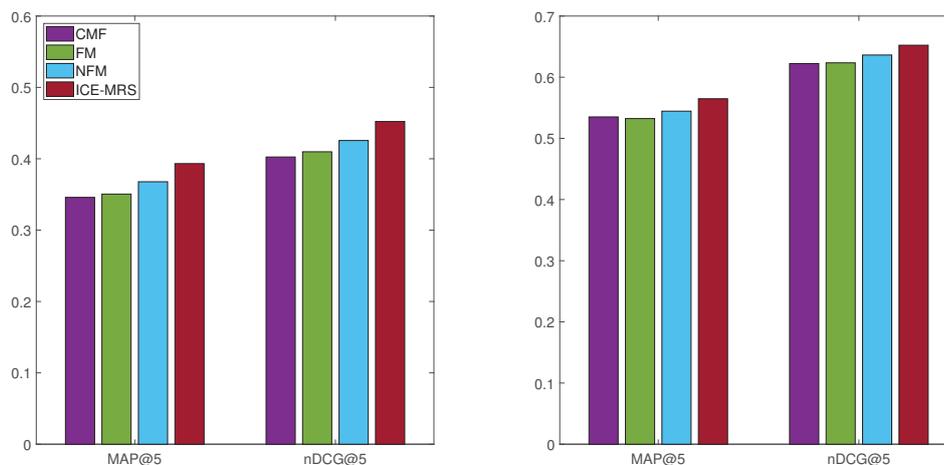


Figure 5.5 : User recommendation for cold-start items of Delicious and Lastfm.

5.4.5 Visualization and Interpretation

To interpret the influence from influential users and items when users making a selection, we randomly choose a user and her/his selection on an artist (User 41 with Raised Fist) from the LastFM dataset as a study case. We visualize the influential context w.r.t. the target user and the target item in Figure 5.6 by differentiating the influence with different edge thickness. We can find that the influence from different users/items is quite different, for example, User 1808 and User 184 have the common neighbor 1626 but the influence of User 1626 on User 184 is much larger than on User 1808. In the item network, we observe similar cases from the influential context w.r.t. Raised Fist. Therefore, these influential contexts can provide the interpretation of how the target users and the target items are influenced by the influential users and items to form the connection.

5.5 Summary of Contributions

In this chapter, we model a multi-relational RS with embedding user and item influence from social network and item network. In particular, the non-IID technique

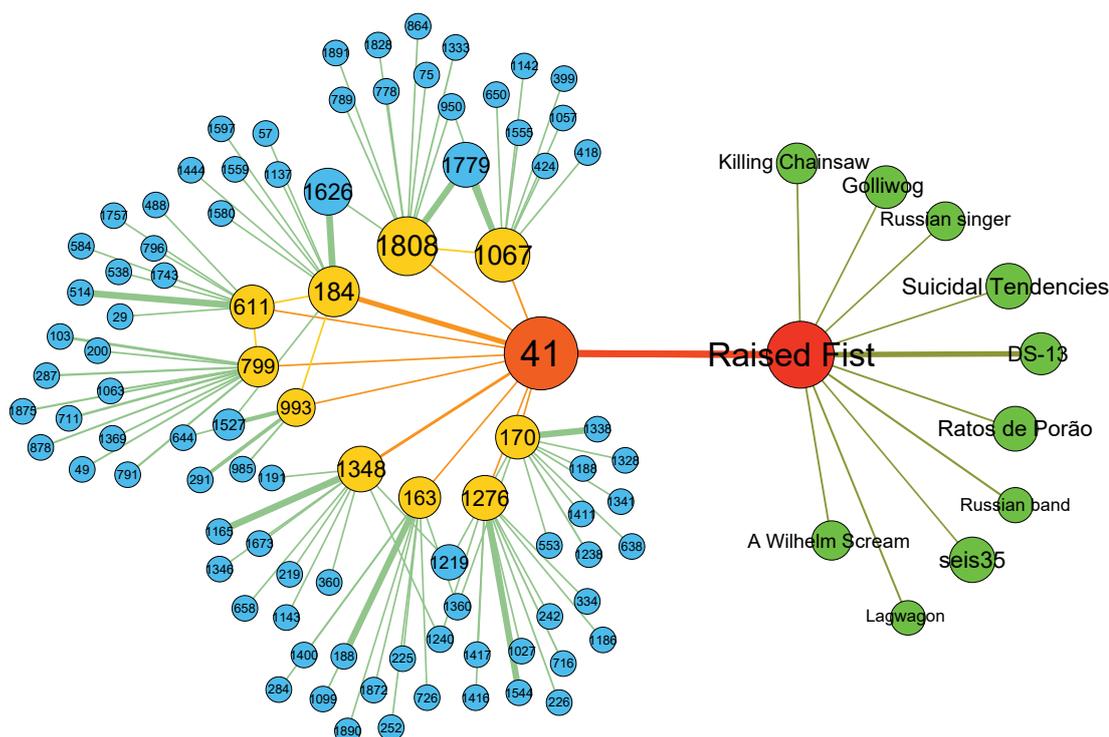


Figure 5.6 : The visualization of influential contexts of a sampled user selection on an artist in the Lastfm dataset. The artists in item network are labeled by their names and the anonymous users in user network are labeled with IDs. The thickness of edges specifies the significance of influence.

is focused on modeling the coupling relationships between users and items in terms of influential contexts. Our main contributions are summarized as follows:

- We propose a framework to model the multi-relational data in recommendation tasks with considering the user influential context and the item influential context.
- We design an influence aggregation unit (ICAU) and apply ICAUs to construct the *ICE-MRS* as an instantiation of the proposed framework.
- The ICAUs based *ICE-MRS* empower the interpretability on recommendation results in terms of the quantitative influence from relevant users and items.

- We conduct experiments on two real-world multi-relational datasets and the results show the effectiveness of our model on recommendation quality and superiority in terms of handling both user and item cold-start problems. In addition, we visualize learned contexts of a user and item and interpret the influence.

Part III

Non-IID RS: Modeling

Non-IIDness on Items

Chapter 6

A Cross-domain Recommender System for Modeling the Couplings over Heterogeneous Item Domains

6.1 Introduction

In the era of Big Data, the huge and rapidly increasing amount of information has penetrated every corner of our life. However, it is easy to become overwhelmed by so much information and it can be difficult to find what one is looking for. When we follow events on Facebook, buy books on Amazon or install apps on a smartphone, we are encouraged by the underlying systems to provide feedback, e.g., a rating or a comment. This is because modern RSs can predict personalized preferences for unconsumed items based on the feedback collected from like-minded users. CF has been widely studied as a core component of RSs, but in fact, users do not always provide feedback for various personal reasons, e.g., privacy. As a result, CF methods tend to suffer from two common challenges: data sparsity [206] and cold-start [117, 195] as presented in Section 1.2. Some real-world applications naturally suffer from the data sparsity problem; for instance, users who have recently bought a new car may not have a new car purchase plan for another five years, therefore the lack of feedback data becomes a major barrier to the use of current CF methods.

A user usually has sufficient experience in some focused domains (rich data domains) but lacks experience in other domains (deficient data domains). An RS that can recommend potentially desirable items to users is therefore more useful in unfamiliar domains. However, it is inevitable that the cold-start issue will be en-

countered in unfamiliar domains where there is almost no data from which to learn user preferences. Since users have different interests, their rich data domains and deficient data domains also differ, and it is therefore sensible to leverage users' feedback data over multiple domains to find like-minded users to enable the inference of user preferences in unfamiliar domains. Based on this idea, Cross-Domain Collaborative Filtering (CDCF) has emerged as an important research topic in recent years [126]. Most current CDCF approaches focus on the product domain [127, 128, 167], but although the term “domain” usually refers to product domains, it may apply to more generalized references, such as time domains and spatial domains.

6.1.1 Leveraging Cross-Domain Information

CF methods can generally be sub-divided into two categories: neighborhood-based (a.k.a. memory-based) and model-based [81, 181, 206]. Therein, the model-based approach, such as MF [91, 117], have gained dominance in recent years. In fact, we can construct an integrated item set containing the items from all domains so that traditional CF approaches can be directly applied as naive CDCF approaches. In this chapter, we refer to neighborhood-based methods and MF-based methods running on such integrated item sets as kNN-CDCF and MF-CDCF, respectively. The item factors affecting user preferences in one domain may be quite different from those in another, but taking the integrated item set as input implicitly assumes the homogeneity of items as a single domain. As a result, the naive CDCF approach may lead to poor prediction due to the failure of representing heterogeneities between domains.

A number of refined CDCF methods have recently been proposed, such as cross-domain MF (CDMF) models [167, 203]. CDMF is based on transfer learning, whose underlying idea is illustrated in Figure 6.1: the user factor matrix \mathbf{U} serves as a bridge to transfer knowledge from the auxiliary domain (A) to the target domain

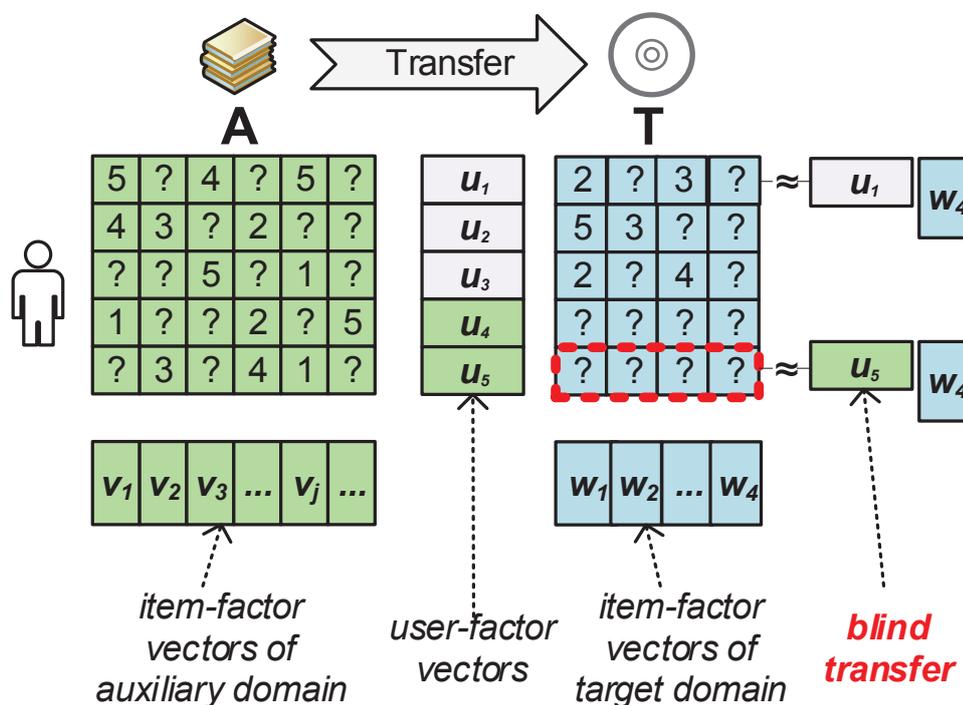


Figure 6.1 : Demonstration of the occurrence of the blinder-transfer issue in CDMF

(T). CDMF models assume that auxiliary data is relatively dense for all users and all items [167]. However, we argue that this assumption is not always true. Our argument is based on the well-known power law (long-tail) distribution, as illustrated in Figure 6.2, where the minority of users and items provide sufficient data while the majority of users and items provide only a little data. This has an impact on the hypothesis of traditional CDCF approaches, which results in the deterioration of prediction performance.

A worse, unavoidable problem of CDCF may be caused by the cold-start issue in some domains. Since users have quite different interests, a user is usually active in certain domains but silent in other domains. As shown in Figure 6.1, users always have different unfamiliar domains which may negatively reduce the recommendation

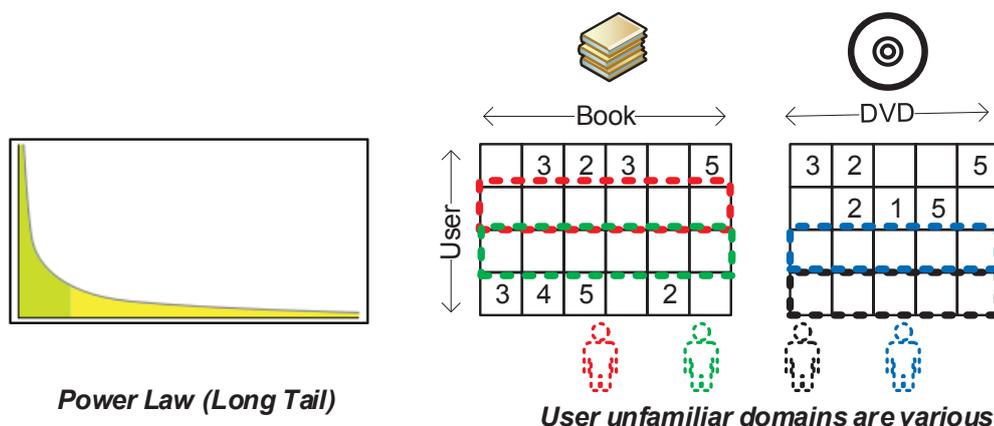


Figure 6.2 : The feedback from the majority of users in each domain is scant due to the power law distribution, and users have different unfamiliar domains due to differences in interests.

performance of CDMF models due to the heterogeneities between domains. If we take a close look at how this happens in terms of Figure 6.2, we see that CDMF aims to improve recommendation on the target domain (T) by utilizing the features of user preferences (i.e. the factor matrix \mathbf{U}) learned from the auxiliary domain (A). A new user factor matrix \mathbf{U}' , which models the user preferences on the target domain (T), is updated based on the transferred matrix \mathbf{U} using the data on (T) [167, 203]. Therefore, the user-factor vectors for users are co-determined by the feedback in the auxiliary and target domains. If no data is available for a user in the target domain (marked with a red box), the user factor vector \mathbf{u}_i is simply transferred from the auxiliary domain without updating. As a result, the prediction on the target domain tends to yield poor recommendation results using this \mathbf{u}_i because of the heterogeneities between two domains. In this thesis, we call this a “*blind-transfer*” issue. Recall that the data associated with the majority of users are insufficient and even absent in a domain, so the above CDCF approaches commonly suffer from such

blind-transfer issues for realistic online data.

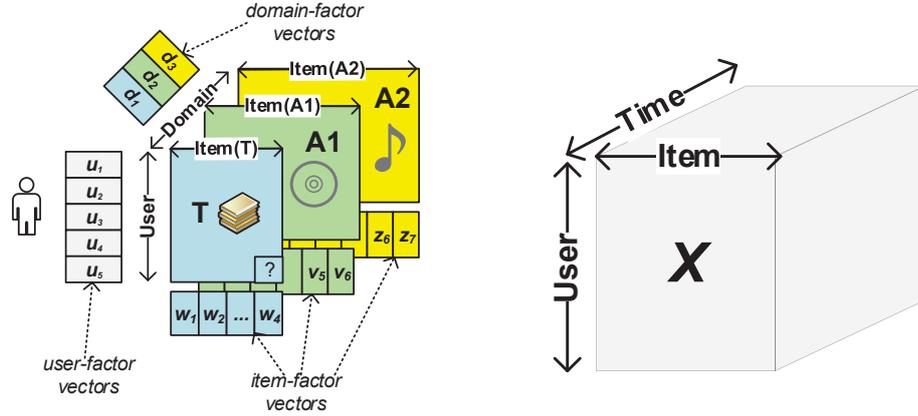
6.1.2 Modeling Domain Factors

The major cause of the blind-transfer issue is the fact that CDMF deals with a set of user-item data over multiple domains in the traditional MF manner so that the rating given by user i on item j is only determined by user and item factors, i.e., \mathbf{u}_i and \mathbf{v}_j , but it does not contain the factors to model the difference between multiple domains. For this reason, CDMF cannot escape the blind-transfer issue, especially when the data is extremely sparse. We argue that domain factors are an essential element for representing each domain in the “*cross-domain*” problem, so CDCF should take into consideration the domain characteristics to reveal domain-specific user preferences on items in depth, rather than only user factors and item factors modeled in CDMF.

As illustrated in Figure 6.3a, our approach allows an exclusive item-factor matrix for each domain to express heterogeneities. In addition, user-factor matrix U is used to model general users’ concerns across all domains, and domain-factor matrix D carries the information to express the traits of each domain. Hence, each observation can be viewed as the result of the three-way interaction among user, item and domain factors. In addition, we can interpret that the domain-specific user factors are generated by the interaction between domain factors and general user factors as shown in Figure 6.3a. Since the domain-factor vector reflects the characteristics of a domain and it is always available, thus the domain-specific user preferences can be obtained to avoid the blind-transfer issue.

6.1.3 Irregular Triadic Relation

According to above analysis, MF-based cross-domain methods, i.e. CDMF, can only model the dyadic interaction between users and items, so they are inevitable



(a) Irregular triadic relation for model- (b) A regular triadic relation “user-item-
 time” represented by a 3D-tensor.
 ing heterogeneities between domains.

Figure 6.3 : Irregular triadic relation and regular triadic relation.

to suffer from the blind-transfer issue. As a result, we propose to model three-way interaction among user, item and domain to avoid this issue. A natural approach to model high-order interaction, e.g. user-item-tag, user-item-time, is in terms of tensor factorization (TF) models [180, 236]. For such regular triadic relations, let us denote the user set \mathcal{U} , the item set as \mathcal{I} and \mathcal{D} is the set for the third dimension, i.e. a triadic relation $\mathcal{U} \times \mathcal{I} \times \mathcal{D}$, so they can be naturally represented by a 3D-tensor as demonstrated in Figure 6.3b. Unfortunately, we cannot use a regular tensor to deal with CDCF problems, since each domain d has a different domain-specific item set, \mathcal{I}_d as shown in Figure 6.3a. Therefore, we cannot obtain a regular triadic relation over the sets, \mathcal{U}, \mathcal{D} and $\{\mathcal{I}_d\}_{d \in \mathcal{D}}$. As a result, regular TF models become not applicable.

In this chapter, we design an irregular TF model, named Weighted Irregular Tensor Factorization (WITF), to model the irregular triadic relation. As implied by its name, WITF couples a set of $\{\mathcal{U} \times \mathcal{I}_d\}_{d \in \mathcal{D}}$ relations among multiple domains

and derives an optimization form consisting of a TF equivalent component which enables to capture the irregular triadic interaction and learn the domain factors.

6.2 Problem Formulation

In recent years, matrix factorization-based methods [91, 117] have gained dominance in RSs. However, single-domain MF methods are not able to provide a good solution to deal with the cold start issue. In the following sections, we will take a close look at how MF methods suffer from this issue and propose our CDCF solution.

6.2.1 Weighted Regularized Matrix Factorization

Let us consider the MF model from a probabilistic view. Given a data matrix \mathbf{Y} with the entries indexed by (i, j) , we can obtain the following joint distribution with the R -dimensional Gaussian user factor vector \mathbf{u}_i for each user i and item factor vector \mathbf{v}_j for each item j :

$$P(\mathbf{u}_i) = \mathcal{N}(\mathbf{u}_i | \boldsymbol{\mu}_i, \tau_U^{-1} \mathbf{I}) \quad P(\mathbf{v}_j) = \mathcal{N}(\mathbf{v}_j | \boldsymbol{\mu}_j, \tau_V^{-1} \mathbf{I}) \quad (6.1)$$

$$P(Y_{ij}, \mathbf{u}_i, \mathbf{v}_j) = \mathcal{N}(Y_{ij} | \mathbf{u}_i^\top \mathbf{v}_j, w_{ij}^{-1}) \quad (6.2)$$

$$P(\mathbf{Y}, \mathbf{U}, \mathbf{V}) = \prod_{ij} P(Y_{ij}, \mathbf{u}_i, \mathbf{v}_j) \prod_i P(\mathbf{u}_i) \prod_j P(\mathbf{v}_j) \quad (6.3)$$

$$P(\mathbf{U}, \mathbf{V} | \mathbf{Y}) = \frac{P(\mathbf{Y}, \mathbf{U}, \mathbf{V})}{P(\mathbf{Y})} \propto P(\mathbf{Y}, \mathbf{U}, \mathbf{V}) \quad (6.4)$$

where \mathbf{I} denotes an identity matrix, $\mathbf{U} = [\mathbf{u}_1, \dots, \mathbf{u}_N]$ is the user factor matrix, $\mathbf{V} = [\mathbf{v}_1, \dots, \mathbf{v}_M]$ is the item factor matrix, and τ_U, τ_V, w_{ij} are the precision parameters of the Gaussian distributions. We can learn the user factors and the items factors through maximum a posteriori (MAP) estimate. According to the Bayesian theorem, we have the posterior $P(\mathbf{U}, \mathbf{V} | \mathbf{Y}) \propto P(\mathbf{Y}, \mathbf{U}, \mathbf{V})$ given in Eq. 6.4. The following objective function can then be obtained by minimizing the negative log-posterior.

$$J = \underset{\mathbf{U}, \mathbf{V}}{\operatorname{argmin}} \frac{1}{2} \left[\sum_{ij} w_{ij} (Y_{ij} - \mathbf{u}_i^\top \mathbf{v}_j)^2 + \tau_U \sum_i \|\mathbf{u}_i - \boldsymbol{\mu}_i\|^2 + \tau_V \sum_j \|\mathbf{v}_j - \boldsymbol{\mu}_j\|^2 \right] \quad (6.5)$$

Thus, we obtain a Weighted Regularized Matrix Factorization (WRMF) model, in which the precision parameter w_{ij} serves as the weight w.r.t. each entry. Typically, if we set $\boldsymbol{\mu}_i = 0, \boldsymbol{\mu}_j = 0$ (i.e., zero-mean priors), $w_{ij} = 1$ for observed rating while $w_{ij} = 0$ otherwise, and $\lambda = \tau_U = \tau_V$ (i.e., regularization parameter), then we immediately obtain the objective of the classical probabilistic MF (PMF) model [188].

We can estimate user factors and item factors by the gradient-based method using a coordinate descent strategy. First, the gradient w.r.t. \mathbf{u}_i and \mathbf{v}_i can be easily derived from the objective 6.5:

$$\frac{\partial J}{\partial \mathbf{u}_i} = \sum_j w_{ij} \mathbf{v}_j \mathbf{v}_j^\top \mathbf{u}_i - \sum_j w_{ij} Y_{ij} \mathbf{v}_j - \tau_U \boldsymbol{\mu}_i + \tau_U \mathbf{u}_i \quad (6.6)$$

$$\frac{\partial J}{\partial \mathbf{v}_i} = \sum_i w_{ij} \mathbf{u}_i \mathbf{u}_i^\top \mathbf{v}_j - \sum_i w_{ij} Y_{ij} \mathbf{u}_i - \tau_V \boldsymbol{\mu}_j + \tau_V \mathbf{v}_i \quad (6.7)$$

When \mathbf{V} is fixed, the optimization w.r.t. each \mathbf{u}_i is convex. Therefore, it yields a close-form update equation for \mathbf{u}_i by setting Eq. 6.6 to zero:

$$\mathbf{u}_i \leftarrow \left(\tau_U \mathbf{I} + \sum_j w_{ij} \mathbf{v}_j \mathbf{v}_j^\top \right)^{-1} \left(\tau_U \boldsymbol{\mu}_i + \sum_j w_{ij} Y_{ij} \mathbf{v}_j \right) \quad (6.8)$$

Similarly, we can obtain the update equation for each \mathbf{v}_j with \mathbf{U} being fixed:

$$\mathbf{v}_j \leftarrow \left(\tau_V \mathbf{I} + \sum_i w_{ij} \mathbf{u}_i \mathbf{u}_i^\top \right)^{-1} \left(\tau_V \boldsymbol{\mu}_j + \sum_i w_{ij} Y_{ij} \mathbf{u}_i \right) \quad (6.9)$$

Now, let us consider a fully cold-start case where user i has no data, i.e. $w_{ij} = 0$ for all j , and the user factor vector \mathbf{u}_i is always the prior mean $\boldsymbol{\mu}_i$, since no data is available to update it. In such a cold-start case, the prediction on the preferences of user i is fully dependent on the given prior. For PMF, the prior is assumed to be zero-mean so it is unavailable to conduct recommendation for fully cold-start users.

6.2.2 Learning Priors from Cross-Domain Feedback

A straightforward way to resolve a cold-start problem when few data are available from the viewpoint of the Bayesian probabilistic model is to find more informative

priors. Since we argue that it is possible to learn user preference by leveraging information from other domains, we propose to learn the priors over the cross-domain feedback data.

As analyzed previously, we argue that CDCF should take domain factors into consideration, thus the triadic user-item-domain relation needs to be modeled. Since MF methods can only model two-way interactions between users and items, they cannot capture high-order interaction across domains. We therefore need to construct a higher order latent factor model to capture the three-way interaction user-item-domain. Intuitively, the tensor factorization (TF) model [114] is a good candidate for representing this type of third-order interaction. A regular tensor requires each domain slice to have identical items. However, as illustrated in Figure 6.3, each domain slice has a domain-specific item set in the CDCF problem, therefore it cannot form a regular tensor for factorization. To work with this problem, we propose a weighted irregular tensor factorization (WITF) model which relaxes the constraint that dictates that the same item set should be employed for all domains. As a result, WITF respectively learns an item factor matrix for each domain, as shown in Figure 6.3.

6.2.3 Learning Posterior on Target Domain for Fine Tuning

When WITF is learned, we can obtain the general user factors \mathbf{u}_i for each user, the domain factors \mathbf{d}_k , and item factors \mathbf{v}_j^k for each item in domain k , as shown in Figure 6.3. Although these factors can be used to reconstruct the entries for each domain [83], they may not perfectly predict the user preference in a target domain for recommendation because these estimates are retrieved by fitting the data over all domains. Therefore, we need a fine-tuning procedure to adjust those learned factors by tightly refitting the evidence in the target domain.

In Bayesian statistics, the posterior probability is the conditional probability that

is assigned after the relevant evidence is taken into account. As presented previously, WRMF learns parameters from the data in a given domain, and the factors learned from WITF clearly serve as good informative priors for WRMF. Therefore, we can obtain the refined user factors and item factors in a target domain k by MAP estimation (c.f. Eq. 6.4, 6.5, 6.6, 6.7). The prior mean of user factors can be constructed in terms of the general user factors \mathbf{u}_i and the domain factors \mathbf{d}_k , and the item factors \mathbf{v}_j^k can directly serve as the prior mean of item factors of WRMF (cf. Eq. 6.1):

$$\boldsymbol{\mu}_i = \text{diag}(\mathbf{d}_k)\mathbf{u}_i \quad \boldsymbol{\mu}_j = \mathbf{v}_j^k \quad (6.10)$$

$\text{diag}(\mathbf{d}_k)$ generates a diagonal matrix with the diagonal elements \mathbf{d}_k . This can be regarded as a transfer learning procedure which transfers the factors estimated from WITF as the priors to regularize the user factors and item factors when learning WRMF (cf. Eq. 6.5).

6.3 Weighted Irregular Tensor Factorization

6.3.1 Transformation

As previously discussed, we need to model the three-way interaction between *user-item-domain* in the CDCF problem to capture heterogeneities between domains. An intuitive way to capture this three-way interaction is in terms of a third-order tensor, where each frontal slice in the cube corresponds to a feedback data matrix over users and items for each domain. However, as illustrated in the left-hand image of Figure 6.4, each domain consists of a different number of domain-specific items, so it cannot form a regular tensor. Therefore, we need to transform the set of heterogeneous domain matrices into a regular tensor for the purpose of applying the TF. To work with this challenge, we designed a novel weighted irregular TF model which conducts a transformation over heterogeneous domain matrices into a regular tensor consisting of virtual items and virtual data, as illustrated in

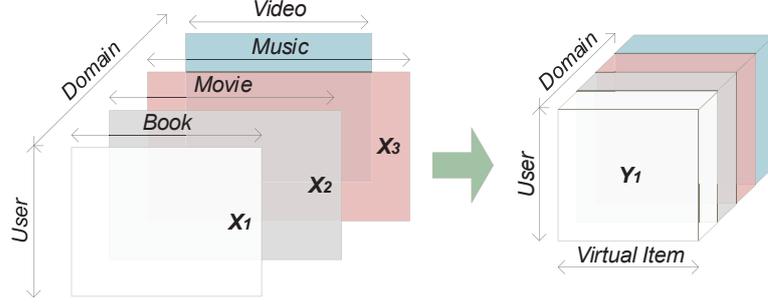


Figure 6.4 : WITF transforms the slices with heterogeneous domain-specific items into a regular third-order tensor containing an identical virtual item set.

Figure 6.4.

Loss Function: Let $\{\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_K\}$ denote the data matrices of all domains, where each matrix, \mathbf{X}_k , has the size $N \times M_k$, N is the number of users and M_k is the number of items in domain k . We denote $\mathbf{U} \in \mathbb{R}^{N \times R}$ as the user factor matrix ($\mathbf{U}_{i,:}$ refers to the factor vector of user i), $\mathbf{V}_k \in \mathbb{R}^{M_k \times R}$ is the item factor matrix of domain k ($\mathbf{V}_{k,j,:}$ refers to the factor vector of item j), and $\Sigma_k = \text{diag}(\mathbf{C}_{k,:})$ is an $R \times R$ diagonal matrix where $\mathbf{C}_{k,:}$ refers to the factor vector of the domain k . Given the user factor vector $\mathbf{U}_{i,:}$, the item factor vector $\mathbf{V}_{k,j,:}$ and the domain factor vector $\mathbf{C}_{k,:}$, the likelihood of entry $\mathbf{X}_{k,i,j}$ is given by:

$$P(\mathbf{X}_{k,i,j} | \mathbf{U}_{i,:}, \mathbf{V}_{k,j,:}, \mathbf{C}_{k,:}) = \mathcal{N}(\mathbf{X}_{k,i,j} | \mathbf{U}_{i,:} \Sigma_k \mathbf{V}_{k,j,:}^T, w_{k,i,j}^{-1}) \quad (6.11)$$

Then, the likelihood over the entries over all domains can be given by:

$$P(\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_K | \mathbf{U}, \mathbf{V}_1, \mathbf{V}_2, \dots, \mathbf{C}) = \prod_{k=1}^K \prod_{i=1}^N \prod_{j=1}^{M_k} P(\mathbf{X}_{k,i,j} | \mathbf{U}_{i,:}, \mathbf{V}_{k,j,:}, \mathbf{C}_{k,:}) \quad (6.12)$$

We can easily obtain the following weighted loss function by minimizing the above negative log-likelihood:

$$J = \underset{\mathbf{U}, \mathbf{V}, \mathbf{C}}{\text{argmin}} \frac{1}{2} \sum_{k=1}^K \|\mathbf{W}_k \cdot^* (\mathbf{X}_k - \mathbf{U} \Sigma_k \mathbf{V}_k^T)\|_F^2 \quad (6.13)$$

where the entry of the weight matrix is $\mathbf{W}_{k,i,j} = \sqrt{w_{k,i,j}}$. That is, each domain matrix \mathbf{X}_k has the factorization form $\mathbf{U}\Sigma_k\mathbf{V}_k^\top$. However, this factorization form is not unique without additional constraints [107]. For example, $\mathbf{U}\Sigma_k\mathbf{V}_k^\top = (\mathbf{U}\Sigma_k\mathbf{A}^{-1}\mathbf{F}^{-1})\mathbf{F}(\mathbf{V}_k\mathbf{A})^\top = \mathbf{U}'\mathbf{F}\mathbf{V}'^\top$ where \mathbf{F} is a diagonal matrix and $\mathbf{U}' = \mathbf{U}\Sigma_k\mathbf{A}^{-1}\mathbf{F}^{-1}$, $\mathbf{V}' = (\mathbf{V}_k\mathbf{A})^\top$, so we get a new factorization form w.r.t. \mathbf{X}_k . To improve the uniqueness property, Harshman [70] proposed the imposition of a constraint whereby the cross product $\mathbf{V}_k^\top\mathbf{V}_k$ is an invariant matrix over k , i.e., $\Phi = \mathbf{V}_1^\top\mathbf{V}_1 = \mathbf{V}_2^\top\mathbf{V}_2 = \dots = \mathbf{V}_K^\top\mathbf{V}_K$. Following this suggestion [70], we impose a column-wise orthonormal matrix $\mathbf{P}_k \in \mathbb{R}^{M_k \times R}$ (i.e. $\mathbf{P}_k^\top\mathbf{P}_k = \mathbf{I}$) and a square matrix $\mathbf{V} \in \mathbb{R}^{R \times R}$ that does not vary by slice. We then find that the cross-product constraint is enforced implicitly since

$$\mathbf{V}_k^\top\mathbf{V}_k = (\mathbf{P}_k\mathbf{V})^\top(\mathbf{P}_k\mathbf{V}) = \mathbf{V}^\top\mathbf{V} = \Phi \quad (6.14)$$

Accordingly, the loss function of Eq. 6.13 can be rewritten as

$$J = \underset{\mathbf{U}, \mathbf{V}, \mathbf{C}}{\operatorname{argmin}} \frac{1}{2} \sum_{k=1}^K \|\mathbf{W}_k \cdot^* (\mathbf{X}_k - \mathbf{U}\Sigma_k(\mathbf{P}_k\mathbf{V})^\top)\|_F^2 \quad s.t. \mathbf{P}_k^\top\mathbf{P}_k = \mathbf{I} \quad (6.15)$$

Weighting Influence over Domains: From Eq. 6.15, user factor matrix \mathbf{U} is shared across all domains, i.e. learning \mathbf{U} is affected by the loss of data in all domains. However, the amount of data in each domain is quite different. Some domains may have many items and extensive feedback while other domains may only have a few items and little feedback. As a result, \mathbf{U} tends to be mainly determined by domains with a large amount of data. Hence, we can assign a weight $\omega_k > 0$ to the loss in each domain to control the penalty of this loss.

$$J = \underset{\mathbf{U}, \mathbf{V}, \mathbf{C}}{\operatorname{argmin}} \frac{1}{2} \sum_{k=1}^K \omega_k \|\mathbf{W}_k \cdot^* (\mathbf{X}_k - \mathbf{U}\Sigma_k(\mathbf{P}_k\mathbf{V})^\top)\|_F^2 \quad s.t. \mathbf{P}_k^\top\mathbf{P}_k = \mathbf{I} \quad (6.16)$$

Intuitively, if we assign a large weight to the loss in a domain, then factor matrix \mathbf{U} is largely learned from the factorization over this domain. Note that a change of \mathbf{U}

will update other factor matrices $\mathbf{V}, \mathbf{C}, \mathbf{P}_k$ in turn during the process of factorization. Therefore, we can control the learning result of all factor matrices by tuning the weight assigned to each slice. In fact, ω_k can be absorbed into \mathbf{W}_k , that is, we can rewrite the entry of the weight matrix as

$$\mathbf{W}_{k,i,j} = \sqrt{\omega_k w_{k,i,j}} \quad (6.17)$$

Thus, we still have the same form of loss function as given by Eq. 6.15. In addition, we denote $\ddot{\mathbf{W}} = \mathbf{W}_k \cdot^* \mathbf{W}_k$ as the element-wise squared \mathbf{W}_k , which is involved in most of the following computations.

$$\ddot{\mathbf{W}}_{k,i,j} = \omega_k w_{k,i,j} \quad (6.18)$$

Equivalence to TF: The loss function given by Eq. 6.15 is the summation of loss over the matrix \mathbf{X}_k of each domain with different sizes, which still cannot capture the interaction across domains. Unfortunately, Eq. 6.15 cannot be transformed into a TF problem in terms of PARAFAC2 [107] to calculate the weighted loss over each domain, due to the Hadamard product. To enable it to capture cross-domain correlation, we give the following theorem to transform Eq. 6.15 into an equivalent TF problem. The proof of this theorem can be found in the appendix of [85].

Theorem 6.1: Minimizing the weighted loss given by Eq. 6.15 is equivalent to minimizing

$$J = \underset{\mathbf{U}, \mathbf{V}, \mathbf{C}}{\operatorname{argmin}} \frac{1}{2} \left[\|\mathcal{Y} - \llbracket \mathbf{U}, \mathbf{V}, \mathbf{C} \rrbracket\|^2 + \sum_k \|\hat{\mathbf{X}}_k \cdot^* \mathbf{H}_k\|_F^2 \right] \quad s.t. \mathbf{P}_k^\top \mathbf{P}_k = \mathbf{I} \quad (6.19)$$

where $\mathcal{Y} \in \mathbb{R}^{N \times R \times K}$ is a third-order tensor with K slices: $\mathbf{Y}_k = \mathbf{Z}_k \mathbf{P}_k$, $\hat{\mathbf{X}}_k = \mathbf{U} \Sigma_k (\mathbf{P}_k \mathbf{V})^\top$ and $\mathbf{H}_k = \sqrt{(\ddot{\mathbf{W}}_k - \mathbf{1} \cdot \mathbf{1}^\top)}$, i.e. $\mathbf{H}_{k,i,j} = \sqrt{\omega_k w_{k,i,j} - 1}$. Here, we denote $\mathbf{Z}_k = \ddot{\mathbf{W}}_k \cdot^* \mathbf{X}_k$.

The first term of 6.19 is the unweighted loss on fitting \mathcal{Y} , and the second term is derived from the transformation from Eq. 6.15 to Eq. 6.19 by eliminating the

weight matrices $\{\mathbf{W}_k\}$ from the weighted loss over $\{\mathbf{X}_k\}$ (see the derivation in the Appendix); it can be interpreted as weighted loss compensation to the unweighted loss on fitting \mathcal{Y} to keep the equivalence between Eq. 6.15 and Eq. 6.19. Therefore, we can now use a TF approach to capture the high-order interaction between domains.

We obtain the final objective by appending the regularization terms to avoid overfitting.

$$J = \underset{\mathbf{U}, \mathbf{V}, \mathbf{C}}{\operatorname{argmin}} \frac{1}{2} \left[\underbrace{\left(\|\mathcal{Y} - \llbracket \mathbf{U}, \mathbf{V}, \mathbf{C} \rrbracket\|^2 + \lambda_U \|\mathbf{U}\|_F^2 + \lambda_V \|\mathbf{V}\|_F^2 + \lambda_C \|\mathbf{C}\|_F^2 \right)}_{\text{1:Regularized TF Model}} \right. \quad (6.20)$$

$$\left. + \underbrace{\sum_k \|\hat{\mathbf{X}}_k \cdot^* \mathbf{H}_k\|_F^2}_{\text{2:Loss Compensation}} \right] \quad s.t. \mathbf{P}_k^\top \mathbf{P}_k = \mathbf{I}$$

6.3.2 Parameter Learning

Given the above objective function, we designed a constrained optimization algorithm to learn the parameters $\Theta = (\mathbf{U}, \mathbf{V}, \mathbf{C}, \{\mathbf{P}_k\})$. This algorithm consists of two sub-procedures: one is to learn $\{\mathbf{P}_k\}$ with the column-wise orthonormal constraint; the other is to learn a TF w.r.t. $\{\mathbf{U}, \mathbf{V}, \mathbf{C}\}$.

Finding Constrained $\{\mathbf{P}_k\}$: When $\{\mathbf{U}, \mathbf{V}, \mathbf{C}\}$ are fixed, learning \mathbf{P}_k is conducted to solve the following constrained weighted least squares (WLS) problem on the data of each domain k (cf. Eq. 6.15)

$$J = \underset{\{\mathbf{P}_k\}}{\operatorname{argmin}} \frac{1}{2} \sum_{k=1}^K \|\mathbf{W}_k \cdot^* (\mathbf{X}_k - \mathbf{U} \Sigma_k (\mathbf{P}_k \mathbf{V})^\top)\|_F^2 \quad s.t. \mathbf{P}_k^\top \mathbf{P}_k = \mathbf{I} \quad (6.21)$$

This corresponds to a WLS-based Orthogonal Procrustes Problem [62]. Now, let $\mathbf{M} = \mathbf{U} \Sigma_k \mathbf{V}^\top$, and we can use an iterative approach [64, 108] to find \mathbf{P}_k by minimizing

$$\|[\mathbf{M} \hat{\mathbf{P}}_k^\top - \overline{\mathbf{W}}_k \cdot^* (\mathbf{M} \hat{\mathbf{P}}_k^\top) - \mathbf{X}_k] - \mathbf{M} \mathbf{P}_k^\top\|_F^2 \quad (6.22)$$

where $\overline{\mathbf{W}}_k = \check{\mathbf{W}}_k \mathbf{D}^{-1}$ and $\mathbf{D} = \text{diag}(\max(\check{\mathbf{W}}_{k,i,:}))$ is a diagonal matrix whose diagonal elements are the maximum elements of the rows of $\check{\mathbf{W}}_k$, and $\hat{\mathbf{P}}_k$ denotes the current estimate of \mathbf{P}_k . Now, let $\mathbf{G} = \mathbf{M}\hat{\mathbf{P}}_k^\top - \overline{\mathbf{W}}_k \cdot^* (\mathbf{M}\hat{\mathbf{P}}_k^\top) - \mathbf{X}_k$, and the above function leads to the orthogonal Procrustes problem of $\|\mathbf{G} - \mathbf{M}\mathbf{P}_k^\top\|_F^2$, which has a close-form solution [38, 64]:

$$\mathbf{G}^\top \mathbf{D} \mathbf{M} \approx \mathbf{A}_R \boldsymbol{\Sigma}_R \mathbf{B}_R^\top \quad (6.23)$$

$$\mathbf{P}_k = \mathbf{A}_R \mathbf{B}_R^\top \quad (6.24)$$

where $\mathbf{A}_R \boldsymbol{\Sigma}_R \mathbf{B}_R^\top$ is the truncated R -rank SVD on the matrix $\mathbf{G}^\top \mathbf{D} \mathbf{M}$. The estimated \mathbf{P}_k is then used as $\hat{\mathbf{P}}_k$ in the next iteration.

Finding $\mathbf{U}, \mathbf{V}, \mathbf{C}$: When the column-wise orthonormal matrices $\{\mathbf{P}_k\}$ are given, we need to resolve the TF-based optimization problem given in Eq. 6.20. Because of the Hadamard product in the loss compensation part of Eq. 6.20, we cannot obtain the gradient w.r.t. the matrix \mathbf{U} as a whole, but we can obtain the gradient w.r.t. each row of \mathbf{U} , i.e. the factor vector of each user. Referring to the gradients in Eq. 3.9 for TF, we have

$$\frac{\partial J}{\partial \mathbf{U}_{i,:}} = -\mathbf{Y}_{(1),i,:} (\mathbf{C} \cdot^* \mathbf{V}) + \mathbf{U} (\mathbf{C}^\top \mathbf{C} \cdot^* \mathbf{V}^\top \mathbf{V} + \lambda_U \mathbf{U}_{i,:} + \mathbf{U}_{i,:} \sum_{k=1}^K [\boldsymbol{\Sigma}_k (\mathbf{P}_k \mathbf{V})^\top \Omega_{k,i} \mathbf{P}_k \mathbf{V} \boldsymbol{\Sigma}_k]) \quad (6.25)$$

where $\Omega_{k,i} = \text{diag}(\check{W}_{k,i,:}) - \mathbf{I}$ is a diagonal matrix. We set this partial derivative to $\mathbf{0}$, and the update equation w.r.t. $\mathbf{U}_{i,:}$ is obtained.

$$\mathbf{U}_{i,:} = \mathbf{Y}_{(1),i,:} (\mathbf{C} \odot \mathbf{V}) \left(\mathbf{C}^\top \mathbf{C} \cdot^* \mathbf{V}^\top \mathbf{V} + \sum_{k=1}^K [\boldsymbol{\Sigma}_k (\mathbf{P}_k \mathbf{V})^\top \Omega_{k,i} \mathbf{P}_k \mathbf{V} \boldsymbol{\Sigma}_k] \right)^{-1} \quad (6.26)$$

$$\mathbf{C}_{k,:} = \mathbf{Y}_{(3),i,:} (\mathbf{V} \odot \mathbf{U}) \left(\mathbf{U}^\top \mathbf{U} \cdot^* \mathbf{V}^\top \mathbf{V} + \lambda_C \mathbf{I} \sum_{i=1}^N [\boldsymbol{\Sigma}_i (\mathbf{P}_k \mathbf{V})^\top \Omega_{k,i} \mathbf{P}_k \mathbf{V} \boldsymbol{\Sigma}_i] \right)^{-1} \quad (6.27)$$

where $\boldsymbol{\Sigma}_i = \text{diag}(\mathbf{U}_{i,:})$. The partial derivative w.r.t. \mathbf{V} is

$$\frac{\partial J}{\partial \mathbf{V}} = -\mathbf{Y}_{(2)} (\mathbf{C} \odot \mathbf{U}) + \mathbf{V} (\mathbf{C}^\top \mathbf{C} \cdot^* \mathbf{U}^\top \mathbf{U} + \lambda_V \mathbf{V}_{i,:} + \sum_{k=1}^K \sum_{i=1}^N \mathbf{P}_k^\top \Omega_{k,i} \mathbf{P}_k \mathbf{V} \Omega_k \mathbf{U}_{i,:}^\top \mathbf{U}_{i,:} \boldsymbol{\Sigma}_k) \quad (6.28)$$

Due to $(\mathbf{P}_k^\top \Omega_{k,i} \mathbf{P}_k \mathbf{V}) \Omega_k(\mathbf{U}_{i,:}^\top \mathbf{U}_{i,:} \boldsymbol{\Sigma}_k)$ having the form \mathbf{AVB} , we cannot simply obtain the update equation as above. According to the property of the Kronecker product, we place the vectorization operator on both sides of the above equation, and then we obtain:

$$\begin{aligned} \text{vec} \frac{\partial J}{\partial \mathbf{V}} &= -\text{vec}[\mathbf{Y}_{(2)}(\mathbf{C} \odot \mathbf{U})] + \text{vec}[\mathbf{V}(\mathbf{C}^\top \mathbf{C} .* \mathbf{U}^\top \mathbf{U}) \\ &\quad + \text{vec} \left(\sum_{k=1}^K \sum_{i=1}^N \mathbf{P}_k^\top \Omega_{k,i} \mathbf{P}_k \mathbf{V} \boldsymbol{\Sigma}_k \mathbf{U}_{i,:}^\top \mathbf{U}_{i,:} \boldsymbol{\Sigma}_k \right) \\ &= -\text{vec}[\mathbf{Y}_{(2)}(\mathbf{C} \odot \mathbf{U})] \\ &\quad + \left[(\mathbf{C}^\top \mathbf{C} .* \mathbf{U}^\top \mathbf{U} + \lambda_V \mathbf{I} + \sum_{k=1}^K \sum_{i=1}^N (\boldsymbol{\Sigma}_k \mathbf{U}_{i,:}^\top \mathbf{U}_{i,:} \boldsymbol{\Sigma}_k) \otimes (\mathbf{P}_k^\top \Omega_{k,i} \mathbf{P}_k) \right] \text{vec} \mathbf{V} \end{aligned}$$

where the vectorization of a matrix is a transformation which converts the matrix into a column vector. Accordingly, we can obtain the update equation w.r.t. $\text{vec} \mathbf{V}$:

$$\begin{aligned} \text{vec} \mathbf{V} &= \left[(\mathbf{C}^\top \mathbf{C} .* \mathbf{U}^\top \mathbf{U} + \lambda_V \mathbf{I} + \sum_{k=1}^K \sum_{i=1}^N (\boldsymbol{\Sigma}_k \mathbf{U}_{i,:}^\top \mathbf{U}_{i,:} \boldsymbol{\Sigma}_k) \otimes (\mathbf{P}_k^\top \Omega_{k,i} \mathbf{P}_k) \right]^{-1} \\ &\quad \text{vec}[\mathbf{Y}_{(2)}(\mathbf{C} \odot \mathbf{U})] \end{aligned} \quad (6.29)$$

Algorithm Summarization: Algorithm I summarizes the parameter learning procedure. In particular, we find that $\mathbf{U}_{i,:}$, $\mathbf{C}_{k,:}$ and \mathbf{P}_k are updated in a parallel scheme by taking advantage of the conditional independence. \mathbf{V} is updated as a whole, but its size is very small, only $R \times R$, and is not dependent on the number of users and items. In Eq. 6.26, 6.27 and 6.29, we need to respectively compute their matrix inversions. In fact, we can avoid computing these inversions by solving linear systems. Algorithm I consists of two parts of sub-iterations, and in practice, we find that the number of sub-iterations can be set with a small value, i.e. m and n are set as less than 5 in general. In our experiments, we set $m=n=1$ which produces a sufficiently good result. This is because too many iterations easily become stuck in poor local minima. According to the analysis, this algorithm can be executed very efficiently in a powerful parallel computing environment. Note that Steps 4

and 5 in Algorithm I constitute an optional sub-procedure for adding noisy data from neighbors to relieve overfitting and improve generalization when data is very sparse. The detail of this sub-procedure will be discussed in Section 6.4.

6.3.3 Weight Matrix Configuration

The weight matrices play an important role in the WITF model. From Eq. (18), we can find the weight matrix in each domain which consists of the weight on each entry of data matrix, $w_{k,i,j}$, and the weight of domain influence, ω_k .

Weights over Data

The data in RSs can usually be divided into two categories: n-ary preference data, such as ratings, and unary preference data, such as clicks or purchases.

N-ary Preference Data: The multilevel ratings, e.g. five-star rating data, which explicitly differentiate user preferences, are typical of n-ary feedback ($n \geq 2$). This kind of feedback is usually treated as a missing data problem. That is, we only model observed ratings and the remaining entries are treated as missing. An indicative matrix of binary weights, which has been successfully applied in MF and TF methods [2,204], is often used in this type of case. Based on this setting, we set the weights for n-ary preference data as follows

$$w_{k,i,j} = \begin{cases} 1 & (k, i, j) \text{ is an observation} \\ a & (k, i, j) \text{ is a noisy example} \\ 0 & \text{else} \end{cases} \quad (6.30)$$

Here, $a \leq 1$ is a smaller constant weight assigned to the additional noisy examples which serve as virtual data. In practice, we can simply let $a = 1$ and tune the number of imposed noisy examples. This is discussed in more detail in the following subsection. As a result, the entries of weight matrices for n-ary preference data are

given by

$$\ddot{\mathbf{W}}_{k,i,j} = \begin{cases} \omega_k & (k, i, j) \text{ is an observation} \\ a\omega_k & (k, i, j) \text{ is a noisy example} \\ 0 & \text{else} \end{cases} \quad (6.31)$$

Under this setting, we find that the zero weights eliminate the loss on fitting missing entries. Therefore, we can use a set of sparse matrices to store only observed entries for each domain.

Unary Preference Data: In real-world applications, explicit preference data is not always available but implicit feedback, e.g. clicks and purchases, is more easily obtained. For example, users rarely rate their bought items, but their purchase record is available in the system. Observed entries are usually represented by 1 and blanks by 0 for this kind of data. This implicit feedback is called unary preference rather than binary preference because the blanks are usually generated as a result of users' lack of awareness and do not necessarily indicate user dislike [75]. Unary preference data are also known as one-class data in some literature [167].

As a result, we assign a higher confidence level to entries with observed choices and a lower confidence level to entries without observed choices [82]. Recall that the confidence level is controlled by the precision parameter in a Gaussian distribution. In our model, the precision parameter of the distribution of each entry corresponds to the weight, cf. Eq. 3.9. Therefore, we can differentiate between the weights on observed choice entries and those on unobserved choice entries. A similar weighting strategy has been successfully applied in some MF models for unary feedback [91, 165]. In the WITF model, we set the weights for unary feedback as follows:

$$w_{k,i,j} = \begin{cases} c_{k,i,j} + 1 & (k, i, j) \text{ is an observation} \\ 1 & \text{else} \end{cases} \quad (6.32)$$

where $c_{k,i,j} \geq 0$ denotes the confidence parameter which can be set a single value [91]

for each domain or can be set different values for each user and item [82, 165], We find that the weights on unobserved choices are always 1 (a minimal confidence) whereas the weights on observed choices are $c_{k,i,j} + 1 \geq 1$. Hence, this guarantees that the confidence level on observed choices, i.e. true positive instances, is higher than it is on unobserved ones, i.e. uncertainly negative instances. Accordingly, the entries of the final weight matrices for unary preference data are given by

$$\ddot{W}_{k,i,j} = \begin{cases} \omega_k c_{k,i,j} & (k, i, j) \text{ is an observation} \\ \omega_k & \text{else} \end{cases} \quad (6.33)$$

Weights over Domains

As previously noted, ω_k is used to trade off the influence between domains. The optimal weight configuration can be found by some advanced search methods, e.g. genetic algorithm [83]. However, the number of possible configuration increases exponentially with the number of domains so this approach is too time-consuming in practice. Here, we present an empirical strategy to seek a suboptimal weight configuration for WITF. The number of training examples on each domain may be quite different, so learning WITF can be regarded as a problem to fit imbalance data over multiple domains. Cost-sensitive learning [251] is an often used approach to dealing with such imbalance data, where a typical process is to assign different weights to training examples of different classes (here corresponds to domains) in proportion to the misclassification costs (here corresponds to the loss of fitting all training examples of a domain). Without loss of generality, we fix ω_T for the target domain T as 1 because it does not change the optimization problem by scaling all $\{\omega_k\}$. Now, let $\#O_T$ denote the number of training examples on the target domain and $\#O_k$ denote the number of training examples on an auxiliary domain k ; we then empirically set ω_k as follows:

$$\omega_k = \alpha_k \frac{\#O_T}{\#O_k} \quad (6.34)$$

where $\alpha \geq 0$ is a proportional parameter which controls the amount of influence from auxiliary domains. Since loss \mathcal{L}_k of fitting training examples is proportional to $\#O_k$, i.e. $\mathcal{L}_k \propto \omega_k \#O_k$, we have the contribution ratio between the loss of an auxiliary domain \mathcal{L}_k and the that of the target domain \mathcal{L}_T by using Eq. 6.34:

$$\frac{\mathcal{L}_k}{\mathcal{L}_T} \propto \frac{\omega_k \#O_k}{1 \#O_T} = \alpha_k \quad (6.35)$$

That is, the amount of contribution from the auxiliary domain is controlled by α_k in terms of scaling the cost of loss on fitting the data on this auxiliary domain. A small α_k tends to bypass the influence from the auxiliary domain so the user preference are mainly learned from the target domain, whereas a large α_k borrows large amount of information from the auxiliary domain. That is, a larger α_k implies more compatible user preferences between the target domain and the auxiliary domain, therefore we can assess the heterogeneities between different auxiliary domains and target domain according to α_k . In general, an optimal α_k is often selected from a set of given values in terms of cross-validation.

6.4 Remarks

6.4.1 Tricks on Sparse Weight Matrices for Complexity Reduction

The weight matrices are only involved in the derivatives of the loss compensation term when computing the parameter updating equations: Eq. 6.26, 6.27, 6.29. In particular, we find that the term related to these weights has the form $\mathbf{A}_k^\top \Omega_{k,i} \mathbf{A}_k$, and specifically, $\mathbf{A}_k = \mathbf{P}_k \mathbf{V} \Sigma_k$ in Eq. 6.26, $\mathbf{A}_k = \mathbf{P}_k \mathbf{V}$ in Eq. 6.27 and $\mathbf{A}_k = \mathbf{P}_k$ in Eq. 6.29. Clearly, \mathbf{A}_k does not contain the subscript i , so it can be pre-computed before looping the user index i . As a result, we can design a more efficient computing strategy in consideration of the weights.

Let us now take Eq. 6.26 as the example. We can expand $\mathbf{A}_k^\top \Omega_{k,i} \mathbf{A}_k$ as $\mathbf{A}_k^\top \text{diag}(\ddot{\mathbf{W}}_{k,i,:}) \mathbf{A}_k - \mathbf{A}_k^\top \mathbf{A}_k$ for explicit preference data. Since $\text{diag}(\ddot{\mathbf{W}}_{k,i,:})$ only

contains $m_{k,i}$ non-zero entries for user i on domain k , $\mathbf{A}_k^\top \text{diag}(\ddot{\mathbf{W}}_{k,i,:})\mathbf{A}_k$ can be computed in time $O(\sum_k R^2 m_{k,i})$. Note that both R and $m_{k,i}$ are small, so the time complexity is very low. The total time complexity for looping all users is $O(N \sum_k R^2 \bar{m}_k)$ where \bar{m}_k denotes the mean of the number of observations over users on domain k . The computation on the term $\sum_{k=1}^K [\mathbf{A}_k^\top \mathbf{A}_k]$ is in time $O(\sum_k R^2 M_k)$, which can be pre-computed before looping. As a whole, the total time complexity on looping all users is $O(N \sum_k R^2 m_k + \sum_k R^2 M_k)$. In fact, it can be finished in time $O(\sum_k R^2 m_k + \sum_k R^2 M_k)$, when we loop users in a parallel scheme, as given in Algorithm 1.

The diagonal weight matrix $\Omega_{k,i} = \text{diag}(\ddot{\mathbf{W}}_{k,i,:}) - \mathbf{I}$ in the case of unary preferences can be equivalently rewritten as $[\text{diag}(\ddot{\mathbf{W}}_{k,i,:}) - \omega_k \mathbf{I}] - (1 - \omega_k)\mathbf{I}$. From Eq. 6.33, we find that the term $[\text{diag}(\ddot{\mathbf{W}}_{k,i,:}) - \omega_k \mathbf{I}]$ only contains $m_{k,i}$ non-zero diagonal entries with the value $\omega_k c$. Therefore, $\sum_{k=1}^K [\mathbf{A}_k^\top [\text{diag}(\ddot{\mathbf{W}}_{k,i,:}) - \omega_k \mathbf{I}] \mathbf{A}_k]$ can be computed in time $O(N \sum_k R^2 m_k)$. That is, it can be finished in time $O(\sum_k R^2 m_k)$ in a parallel scheme. The term $(1 - \omega_k) \sum_{k=1}^K [\mathbf{A}_k^\top \mathbf{A}_k]$ can be pre-computed in time $\sum_k R^2 M_k$. Therefore, the total time complexity is the same as the explicit feedback case, i.e. $O(N \sum_k R^2 m_k + \sum_k R^2 M_k)$ and $O(\sum_k R^2 m_k + \sum_k R^2 M_k)$ in a parallel fashion. Similar tricks can be applied to efficiently compute Eq. 6.27 and 6.29 to loop users.

6.4.2 Training with Additional Noisy Examples for Improving Generalization

As mentioned previously, users' feedback follows power-law distribution in most domains; that is to say, all users in the long tail are cold-start and have very few data. We only model the observed ratings for explicit feedback, which tends to lead to very poor generalization performance as a result of overfitting with too few data. Table 6.1 depicts statistics from a real-world explicit feedback dataset in the experiment section, from which we find that the mean number of ratings over users

Table 6.1 : Statistics of Epinions Dataset over Five Heterogeneous Domains

Domain	# Items	$\frac{\#Ratings}{\#Users}$	$\frac{\#Ratings}{\#Items}$	Sparsity
Kids & Family*	3,769	4.9309	9.9077	0.0013
Hotels & Travel*	2,545	3.921	11.6676	0.0015
Restaurants & Gourmet	2,543	3.3394	9.9446	0.0013
Wellness & Beauty	3,852	3.5481	6.9756	0.0009
Home and Garden	2,785	2.6003	7.0707	0.0009

is less than 5 in each domain, i.e., the total number of observations is less than $5N$. The data are even sparser in the real-world scenario since we removed some users and items with too few data from the experiments. However, the number of factors, R , is normally larger than 5. Therefore, the total number of parameters for a domain (all latent user factors, item factors and domain factors) is $NR + M_k R + R \gg 5N$, even if we set $R = 5$. The number of parameters is much greater than the number of observations, which leads to overfitting issues and results in very poor prediction performance.

In the case of fewer observations and more parameters, we have tried to tune the regularization parameters $\lambda_U, \lambda_V, \lambda_C$ but have failed to improve performance significantly, which may be attributable to data being too sparse data and having a rank problem of a matrix, namely many rows are all zeros. As a result, we found another effective way to tackle this issue, and to be free from tuning $\lambda_U, \lambda_V, \lambda_C$. It has been illustrated in the literature [Bishop 1995] that training by adding noise is equivalent to regularization. However, simply adding noise to observations does not change the sparsity of data. In our framework, we train WITF to fill noisy examples into randomly selected blank entries instead of merely smoothing objective functions (regularization) by adding noise onto observed data. This strategy has

been effectively applied to improve generalization [159, 219]. In detail, we fill the blanks with Gaussian noisy virtual examples as specified in Algorithm I. First, we randomly select S blank entries for each user. Then, a noisy virtual example is generated as $\tilde{X}_{kij} = \mu_k + e_{kij}$ where $e_{kij} \sim \mathcal{N}(0, \Sigma^2)$ for each selected blank entry, where μ_k denotes the mean of observed data on domain k . Accordingly, we assign small weights to these randomly generated examples (cf. Eq. 6.31) while larger weights to real observations to differentiate between their confidence levels. The setting of S is dependent on the amount of data. Generally, sparser data requires larger S , and we will compare performance using different settings of S over the data with different sparsities in the experiment section.

6.4.3 Post-Learning

When the parameters $\mathbf{U}, \mathbf{V}, \mathbf{C}, \{\mathbf{P}_k\}$ are learned from WITF, the user factor matrix is represented as $\mathbf{U}_k = \Sigma_k \mathbf{U}^\top$ and the item factor matrix is $\mathbf{V}_k = (\mathbf{P}_k \mathbf{V})^\top$ in a given target domain k . We can then immediately use $\mathbf{U}_k, \mathbf{V}_k$ for prediction, as follows:

$$\hat{\mathbf{X}}_k = \mathbf{U}_k^\top \mathbf{V}_k = \mathbf{U} \Sigma_k (\mathbf{P}_k \mathbf{V})^\top \quad (6.36)$$

$\hat{\mathbf{X}}_k$ is the predictive data matrix of domain k . The recommendation list can then be ranked according to the predictive values.

As presented in Section 6.2.3, $\mathbf{U}, \mathbf{V}, \mathbf{C}, \{\mathbf{P}_k\}$ are estimated from the data over all domains, so \mathbf{U}_k may not perfectly represent the user preference feature in the target domain and \mathbf{V}_k also may not perfectly represent the feature of items. Therefore, we use the data in the target domain to finely tune \mathbf{U}_k and \mathbf{V}_k . The factors \mathbf{U}_k and \mathbf{V}_k learned from WITF serve as a good informative prior means for the WRMF (cf. Eq. 6.1), so we have

$$\boldsymbol{\mu} = \mathbf{U}_{k, :, i} \quad \boldsymbol{\mu}_j = \mathbf{V}_{k, :, j} \quad (6.37)$$

Then, we use the MAP estimator to alternately update user factor vector \mathbf{u}_i by Eq.

6.8 and item factor vector \mathbf{v}_j by Eq. 6.9 until convergence. Thus, we obtain the refined user factor matrix $\mathbf{U} = [\mathbf{u}_1, \dots, \mathbf{u}_N]$ and item factor matrix $\mathbf{V} = [\mathbf{v}_1, \dots, \mathbf{v}_{M_k}]$.

From Eq. 6.8 and Eq. 6.9, we find that the regularization parameters τ_U and τ_V control the strength of shrinkage of \mathbf{u}_i and \mathbf{v}_j towards the prior means $\boldsymbol{\mu}_i$, i.e. $\mathbf{U}_{k,:,i}$ and $\boldsymbol{\mu}_j$, i.e. $\mathbf{V}_{k,:,j}$. In particular, it obtains $\mathbf{u}_i = \boldsymbol{\mu}_i$ and $\mathbf{v}_j = \boldsymbol{\mu}_j$ when we set large τ_U and τ_V to place huge regularization on \mathbf{u}_i and \mathbf{v}_j . Therefore, the prediction performance on finely tuned parameters will never be worse than the performance achieved directly using the parameters learned from WITF.

6.5 Experiments

In this section, we evaluate our models and other state-of-the-art methods with a set of metrics to compare prediction performance. The experiments are conducted on three real world datasets covering ratings, user clicks history, and time-period separated data.

6.5.1 Comparison Methods

In the following experiments, a group of state-of-the-art methods (cf. Section 3.5) are employed for comparison, where some of these methods are used for explicit feedback and some others are used for implicit feedback.

- *MostPop*: This applies the simplest strategy to rank items by their popularity (measured by the number of observations associated with items).
- *kNN-CDCF*: The naive user-based neighborhood CDCF model uses the integrated item set. In this experiment, we use top 10 nearest neighbors (i.e. $k=10$) with the highest similarity.
- *PMF*: It is used as a representative single-domain CF method. In the experiments, we train it only using the target domain data.

- *PMF-CDCF*: We use PMF as a naive CDMF model which takes the concatenated rating matrix over all domains as the training data.
- *MF-IF*: This can be regarded as a zero-mean based WRMF model to deal with implicit feedback. We use it as a single-domain MF model on implicit feedback data of target domain.
- *MF-CDCF-IF*: We use MF-IF as an implicit feedback based CDMF model which takes the concatenated implicit data matrix over all domains as the training data.
- *CMF*: Collective matrix factorization works as a CDMF model. In the experiments, we couple the rating matrix of each domain on user dimension.
- *PARAFAC2*: It does not have a mechanism to control the influence from each domain when applied to CDMF problem. Furthermore, it cannot deal with implicit feedback under unary preference assumption. Therefore, we run it under binary assumption when conducting experiments on implicit feedback.
- *CDTF*: This a model developed in our previous work which tunes the weight to control the influence between auxiliary domains and target domain by a genetic algorithm.
- *CDTF-IF*: This is a CDTF-based model that works with implicit feedback [83].
- *WITF*: This is an irregular TF model proposed in this chapter, which is able to deal with explicit and implicit feedback in a unified framework. In the experiments, we directly use the parameters estimated from WITF to conduct prediction.
- *WITF+WRMF*: The parameters learned from WITF serve as the priors for WRMF and are finely tuned by MAP on the target domain (cf. Section 6.4.3). Then, we use the refined parameters for prediction.

As discussed previously, we set the number of latent factors as 5, i.e. $R = 5$, which produces the best results in the following experiments, since a larger R easily leads to the overfitting on sparse datasets for all above latent factor models.

6.5.2 Evaluation Metric

We conducted experiments with all the comparison methods on rating data (the case of n-ary preference) and click data (the case of unary preference). We used rating metrics to assess performance on the rating data and ranking metrics to assess performance on the click data.

6.5.3 Rating Prediction on Epinions.com

Epinions.com, set up in 1999 and ultimately acquired by eBay in 2005, was a general product review platform where ratings, buying tips and advice were generated by consumers to help users decide on purchases. The products on Epinions.com cover a number of domains, such as electronics, movies, health, etc. The crawled dataset [210] contains 5-level ratings over products in 28 domains. In this experiment, we chose five domains to construct the cross-domain dataset: Kids & Family, Hotels & Travel, Restaurants & Gourmet, Wellness & Beauty and Home and Garden. User preferences are clearly relevant but quite heterogeneous between these domains, so this dataset is very suitable for testing which CDCF methods are able to appropriately leverage information between the target domain and heterogeneous auxiliary domains.

Data Preparation

In this experiment, we extracted those users who had rated at least two items in one of the five domains and the items which had been rated by two users from the raw dataset. As a result, we obtained 7,573 users and 15,494 items. Some evaluation statistics from this dataset are reported in Table 6.1, from which we can see that

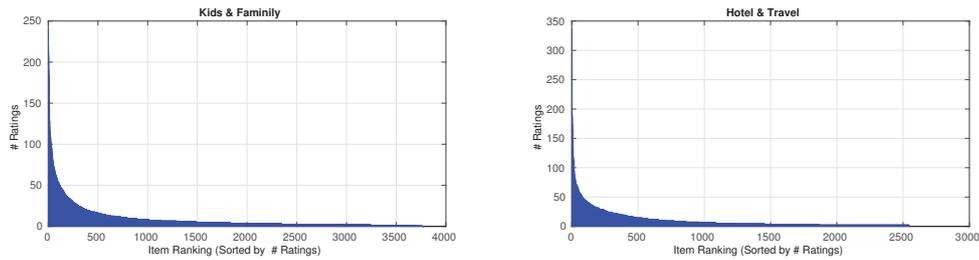


Figure 6.5 : Rating distributions of items on Kids & Family and Hotel & Travel show obvious long tails.

the mean number of ratings over users is less than 5 in each domain. That is, most users have unfamiliar domains and largely suffer from the cold-start issue.

In this experiment, we respectively chose Kids & Family and Hotels & Travel as the target domains for evaluating the rating prediction performance for all comparison methods. Figure 6.5 respectively demonstrates the rating distribution over items on each of the target domains, where we can observe obvious long tails. Therefore, it is a very suitable scenario to borrow information from other domains to improve the inference on user preferences on the target domain. Given a target domain, we constructed the training/testing sets as follows. First, we randomly withheld 20% of the rating data from the target domain as the ground truth for testing, denoted as **TS-20%**, and the remaining 80% of data, denoted as **TR-80%** were used as the training set. In the same way, we constructed a sparser training set by withholding 50% of data, denoted as **TS-50%**, for testing, and the remaining 50% of data, denoted as **TR-50%**, were used as the training set.

Rating Prediction Performance Comparison

Table 6.2 reports the RMSEs on the target domain Kids & Family and Hotels & Travel respectively over all testing users. Clearly, kNN-CDCF underperforms all comparison methods for all cases. This is because a user-based method such as this

often fails to find existing ratings on tail items from neighbors to generate effective prediction. As for model-based methods, we can see that the TF-based models, i.e. PRAFAC2, CDTF, WITF, WITF+WRMF overall achieve better performance than MF-based models. This proves that the TF methods benefit from the modeling of three-way interaction between users, items and domains, which captures higher order information that is unable to learn from MF methods. In particular, we find the performance of all comparison methods is relatively close when the data are relatively dense (**TR-80%** cases), whereas the margins between TF-based methods and other methods become much wider when the data become sparser (**TR-50%** cases). Comparing CMF with PMF-CDCF, we find that CMF overall outperforms PMF-CDCF. The reason is that PMF-CDCF learns user preference on an integrated item set which ignores the heterogeneity between domains, especially when the amount of data in auxiliary domains is much larger than it is in the target domain. In comparison, CMF retains the rating matrix of each domain and provides a more effective way of controlling the knowledge transfer between domains.

Comparing all TF-based methods, we find that PARAFAC2 underperforms others since it is unable to trade off the amount of influence between the auxiliary domains and the target domain. WITF lags slightly behind CDTF because WITF uses an empirical suboptimal weights configuration in domains (cf. Section 6.3.3) while CDTF uses a search procedure to find an optimal weights configuration by genetic algorithm (GA) [83]. Note that WITF can also employ GA to search the optimal configuration of weights. However, running a GA-based search procedure is very time- and space-consuming because it is necessary to rerun the whole learning algorithm under each possible configuration generated by GA. In comparison, WITF uses a much more economical strategy to find a suboptimal weights configuration, and the parameters learned from WITF are also learned by MAP using WRMF. As a result, we obtained smaller RMSEs when the parameters learned from CDTF

Table 6.2 : RMSEs of Comparison CDCF Methods on Epinions Dataset ($-$ denotes baseline, $*$ means $p < 0.01$, $**$ denotes smallest p)

Target Domain	Kids & Family		Hotels & Travel	
	TR-80%	TR-50%	TR-80%	TR-50%
kNN-CDCF	1.2562	1.3016	1.1605	1.3338
PMF-CDCF	1.1719 $-$	1.3547 $-$	1.1260 $-$	1.2925 $-$
CMF	1.1312 $*$	1.2908 $*$	1.0805 $*$	1.2457 $*$
PARAFAC2	1.1102 $*$	1.1458 $*$	1.0647 $*$	1.0891 $*$
CDTF	1.0968 $*$	1.1219 $*$	1.0351 $*$	1.0585 $*$
WITF	1.1043 $*$	1.1293 $*$	1.0375 $*$	1.0619 $*$
WITF+WRMF	1.0563$**$	1.0835$**$	0.9983$**$	1.0284$**$

were finely tuned by WRMF. This proves the effectiveness of using the user, item and domain factors learned from WITF as the informative priors for WRMF. Overall, WITF+WRMF is a more efficient and practical approach than all comparison methods.

Moreover, we performed significance test by choosing PMF-CDCF as the baseline to compare with another model through their paired prediction errors. In particular, we used the sign test [66] to measure if a comparison method can significantly outperform the baseline with the p-value < 0.01 (marked with $*$ in Table 6.2). In particular, the smallest p-value is returned from the test on comparing the outputs between WITF-WRMF and baseline, i.e. WITF+WRMF achieves the most significant improvement among all comparison methods.

Table 6.3 : Statistics of Testing Users Grouped by the Number of Ratings

Target Domain	# Ratings	# testing users in TR-50%	
		Kids & Family	Hotels & Travel
Experienced	> 20	120	55
Little Experienced	6 ~ 20	816	517
Cold-Start	1 ~ 5	2,260	2,807
Fully Cold-Start	0	695	1,072

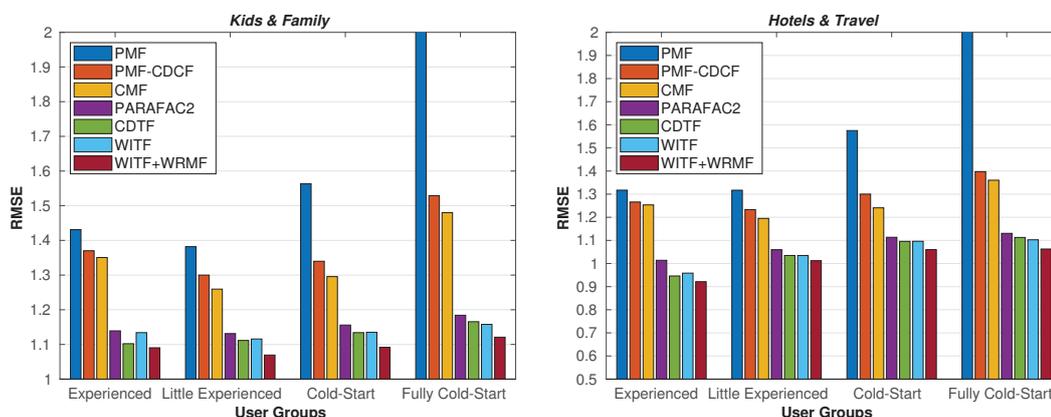


Figure 6.6 : RMSEs comparison over user groups with different number of ratings.

The Prediction Performance over Different Numbers of Training Ratings

To evaluate the performance with different amounts of training data, we split testing users into four groups according to the number of ratings they have in the training set **TR-50%**. Table 6.3 shows statistics of these four groups in the target domain Kids & Family and Hotels & Travel respectively. From the statistics, we find that most users are cold-start, which follows the typical long-tail distribution in the real world.

Figure 6.6 depicts the comparable RMSE results over the four user groups in the target domain Kids & Family and Hotels & Travel respectively. We find that the performance on non-cold-start user groups (Experienced and Little Experienced) between PMF and PMF-CDCF is close. This is because it is not necessary to borrow much information from other domains when user feedback in the target domain is relatively adequate, so the single domain PMF achieves comparable performance with PMF-CDCF in this case. In comparison, the performance on cold-start user groups of single-domain PMF is very poor because insufficient data is available to learn user preferences. As analyzed in Section 3.1, PMF cannot deal with fully cold-start users who do not have any data, so we only show the truncated RMSE for PMF for the case of Fully Cold-Start. Obviously, CMF and PMF-CDCF lag well behind TF-based methods, especially in the case of Fully Cold-Start. This is because CMF and PMF-CDCF only model dyadic interaction between users and items so they cannot model domain factors to represent heterogeneities among domains. As a result, CMF and PMF-CDCF inevitably suffer from the blind transfer issue when no data are available in the target domain, i.e. fully cold-start. In comparison, TF-based methods are capable of representing the heterogeneities by domain factors and therefore achieve more stable performance over all four user groups. In all TF-based methods, PRAFAC2 underperforms all others due to the lack of a mechanism to control influence between domains. WITF+WRMF achieves the best performance in all cases. In particular, the fine-tuning procedure on WITF+WRMF leads to apparent margins from WITF.

Adding Gaussian Noisy Examples for Regularization

As discussed in Section 6.4.2, adding Gaussian noisy examples acts as a regularizer to avoid overfitting and improve generalization ability. The number of noisy examples added for each user is controlled by the parameter S . Intuitively, too small

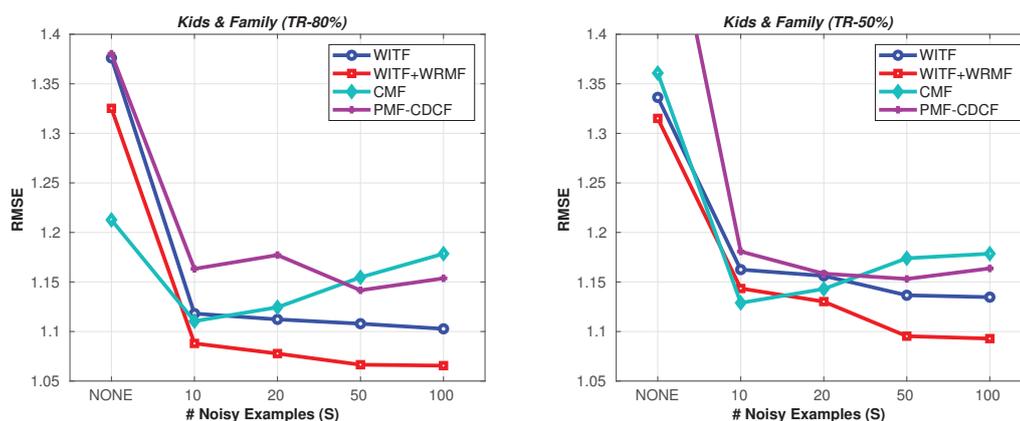


Figure 6.7 : RMSEs comparison over different number of noisy examples for each user.

S generates insufficient data to survive from overfitting while too large S not only increases the time and space complexity but may overwhelm observed data. We therefore evaluated the prediction performance over different settings of S in this experiment. Apart from our WITF model, we also impose noisy examples to the baseline methods, PMF-CDCF and CMF, as the regularization method to test if it can improve their generalization ability.

Figure 6.7 illustrates the rating prediction performance in the target domain Kids & Family using the training sets **TR-80%** and **TR-50%** respectively. From Table 6.1, we easily find that the mean number of ratings over users is less than 4 with **TR-80%** and less than 2.5 with **TR-50%** in the target domain, and even fewer in auxiliary domains. As analyzed in Section 6.4.2, the models become overfitting even if we use a small number of latent factors. Here, we demonstrate the change in RMSEs when 0, 10, 20, 50, 100 noisy examples are added. When no noisy examples ($S = 0$) are imposed, the performance of all comparison methods is evidently poor due to lack of regularization. When a small number of noisy examples ($S = 10$) are added, the improvement is significant because overfitting is relieved (in this case, the total number of real and virtual observations becomes larger than the number

of parameters).

As expected, too much noisy examples may over-regularize the parameter learning. CMF achieves the best performance when 10 noisy examples are imposed, whereas the performance drops when more noisy examples are added. Similar situation is observed w.r.t. PMF-CDCF. In comparison, WITF needs more noisy examples to overcome overfitting since it has more model parameters than those of CMF and PDF-CMF. For WITF and WITF-WRMF, the improvement in the case of **TR-80%** is relatively small when $S \geq 20$. In comparison, the improvement in **TR-50%** is more apparent than that it is in **TR-80%** when $S \geq 20$, since **TR-50%** is sparser than **TR-80%** and needs more noisy data for regularization. In conclusion, we only need to add a small number of noisy examples when training, which is effective to improve the generalization ability.

6.5.4 Click Prediction on Tmall.com

Merchants usually need to personalize the layout of items to attract potential buyers. It is well known that in the field of online advertising, customer targeting is extremely challenging, especially for new buyers. By targeting these potential customers, merchants can greatly reduce the cost of promotion and enhance the return on investment. Unfortunately, the explicit feedback from buyers is usually sparse and may not even be available for buyers in new domains. To alleviate this problem, it is valuable for merchants to utilize implicit feedback, e.g., the user click log recorded by the backend system. In this experiment, we use our models to solve this problem with the click log accumulated by Tmall.com* which is the largest B2C platforms in China.

*<http://www.tmall.com>

Table 6.4 : Statistics of Tmall.com Click-Log Dataset over Four Anonymous Domains (* Each of these domains is chosen as the respective target domain for evaluation)

Domain	#Items	#Clicks/#Users	#Clicks/#Items	Sparsity
D1*	8,179	23.2003	19.717	0.0028
D2*	6,940	18.5455	18.5749	0.0027
D3	5,561	22.5005	28.1246	0.004
D4	6,145	16.0606	18.1671	0.0026

Data Preparation

The dataset provided by Tmall.com contains anonymized user click logs for six months[†] covering 1,627 fine-grained anonymous domains, including clothes, furniture, books, food, and so on. In this raw dataset, all domains are anonymized, so we selected the four domains with the largest number of click records for evaluation, and denoted them as D1, D2, D3 and D4. First, we extracted 7,000 users who had at least 15 clicks in any two domains so that at least one focused domain for each user was available to serve as the auxiliary domain. Then, we trimmed the items accounting for fewer than 5 clicks. The statistics of this preprocessed dataset for evaluation are illustrated in Table 6.4. In this experiment, D1 and D2 are respectively chosen as the target domains for evaluation. For each target domain, we applied the same strategy as the previous experiment to construct two training sets, **TR-80%** and **TR-50%**, and used the withheld data as the testing sets **TS-20%** and **TS-50%**.

[†]<http://tianchi.aliyun.com/datalab/introduction.htm?id=1>

Table 6.5 : The Mean AP@5,10 and nDCG@5,10 on Tmall.com Dataset with D1 as the target domain (− denotes baseline, * means $p < 0.01$, ** denotes smallest p)

Target Domain	TR-80%				TR-50%			
	AP@		nDCG@		AP@		nDCG@	
	5	20	5	20	5	20	5	20
MostPop	0.0161 [−]	0.0175 [−]	0.0269 [−]	0.0382 [−]	0.0322 [−]	0.0223 [−]	0.0567 [−]	0.0577 [−]
N-CDCF	0.0252 [*]	0.0240 [*]	0.0441 [*]	0.0465 [*]	0.0352 [*]	0.021	0.0604 [*]	0.0534
MF-IF	0.0263 [*]	0.0293 [*]	0.0432 [*]	0.0631 [*]	0.0455 [*]	0.0324	0.0813 [*]	0.0854 [*]
MF-IF-CDCF	0.0242 [*]	0.0258 [*]	0.0399 [*]	0.0552 [*]	0.0431 [*]	0.0296	0.0763 [*]	0.0775 [*]
PARAFAC2	0.0213 [*]	0.0226 [*]	0.0350 [*]	0.0476 [*]	0.0395 [*]	0.0267	0.0691 [*]	0.0687 [*]
CDTF-IF	0.0258 [*]	0.0276 [*]	0.0425 [*]	0.0587 [*]	0.0423 [*]	0.0294	0.0758 [*]	0.0767 [*]
WITF	0.0267 [*]	0.0285 [*]	0.0451 [*]	0.0623 [*]	0.0484 [*]	0.034	0.0849 [*]	0.0872 [*]
WITF+WRMF	0.0271^{**}	0.0290^{**}	0.0462^{**}	0.0643^{**}	0.0486^{**}	0.0343^{**}	0.0851^{**}	0.0879^{**}

Ranking Prediction Performance Comparison

In this experiment, we evaluated the prediction performance on clicks using the metrics AP@5, AP@20, nDCG@5 and nDCG@20. Table 6.5 and 6.6 reports the mean results with the sign test over all testing users on the target domain D1 and D2 respectively. Predicting a new item from thousands of items that a user has never clicked is a very difficult task, so the AP and nDCG of all comparison methods are relatively low. We find that the overall results for **TR-50%** are higher than the results for **TR-80%**. This is because 50% of the withheld data is used as the testing set for **TS-50%** so the hit probability is naturally higher than it is for the 20% of withheld testing data **TS-20%**.

The baseline method, MostPop, underperforms all comparison methods. This is because popular items only account for a small proportion (a short head) and most users have their personally preferred items in the long tail. Anderson’s well-known

Table 6.6 : The Mean AP@5,10 and nDCG@5,10 on Tmall.com Dataset with D2 as the target domain (− denotes baseline, * means $p < 0.01$, ** denotes smallest p)

Target Domain	TR-80%				TR-50%			
	AP@		nDCG@		AP@		nDCG@	
	5	20	5	20	5	20	5	20
MostPop	0.0175 [−]	0.0194 [−]	0.0288 [−]	0.0424 [−]	0.0297 [−]	0.0231 [−]	0.0530 [−]	0.0591 [−]
N-CDCF	0.0281*	0.0261*	0.0435*	0.0520*	0.0228	0.0243*	0.038	0.0357
MF-IF	0.0320*	0.0354*	0.0528*	0.0747*	0.0501*	0.0370*	0.0872**	0.0924**
MF-IF-CDCF	0.0240*	0.0262*	0.0397*	0.0563*	0.0380*	0.0285*	0.0675	0.0724*
PARAFAC2	0.0215*	0.0234*	0.0356*	0.0506*	0.0327*	0.0251*	0.0589*	0.0638*
CDTF-IF	0.0326*	0.0337*	0.0526*	0.0662*	0.0454*	0.0316*	0.0761*	0.0750*
WITF	0.0338*	0.0363*	0.0552*	0.0753*	0.0538*	0.0383*	0.0905*	0.0909*
WITF+WRMF	0.0343**	0.0369**	0.0556**	0.0758**	0.0542**	0.0386**	0.0907**	0.0915*

research in economics proved this phenomenon, and Anderson [7] suggested that future business would obtain more profit from long-tail selling. MostPop is unable to find the personally preferred items in the long tail, which leads to the poorest performance. N-CDCF also does not perform well, and the reason is similar to MostPop, that is, the preferred long-tail items for each user are quite different, so using neighbors' clicked long-tail items as the prediction result deviates significantly from users' true preferences.

The single domain method, MF-IF, achieves relatively good performance because it exploits both the clicked items and the unclicked ones for each user to model their implicit preferences, i.e. the observed data as positive instances and unobserved data as uncertainly negative instances. Thus, it not only resolves the unclassifiable issue on unary (one-class) preference data [165] but also relieves the data sparsity issue due to the small amount of observed data. Although MF-IF-CDCF also exploits the

unobserved data, its results are worse than those of MF-IF, which can be mainly attributed to the heterogeneities between domains. That is, much more data (both observed and unobserved ones) from heterogeneous auxiliary domains overwhelm the learning of user factors in the target domain. PARAFAC2 is based on binary assumption which treats the unclicked items as true negative instances and assigns too strong dislike weights to them. In addition, PARAFAC2 cannot control the influence from auxiliary domains. These two deficiencies result in it even underperforming MF-IF-CDCF. In comparison, CDTF-IF surpasses MF-IF-CDTF since it controls the mutual influence between domains. However, the design of CDTF-IF is defective when assigning different confidence levels [83] on clicked and unclicked items. In fact, the different confidence levels do not take effect when training the model. In this chapter, WITF fixes this defect and we devise a more efficient algorithm to learn the parameters, with the result that it outperforms CDTF-IF. After fine-tuning, the model WITF+WRMF achieves a level of improvement over WITF.

The Impact of Confidence Parameter

Click records are typical unary preference data so we cannot directly differentiate user preference levels over them. As presented in related work, both observed data and unobserved data should be exploited to resolve this problem. In Section 6.3.3, the confidence parameter $c_{k,i,j}$ is introduced to emphasize the observed clicks. Although $c_{k,i,j}$ can be tuned for each user and item respectively [82, 165], it is beyond the main topic of this chapter. In this experiment, we use a single confidence parameter c for all domains [91] since we mainly focus on evaluating the impact of c using different values.

We increase the value of c from 0 to 4 to evaluate the change of mean AP for WITF. Figure 6.8 plots the change of performance in the target domain D1 and D2 respectively. It degenerates to the case of binary preference data when $c = 0$ is

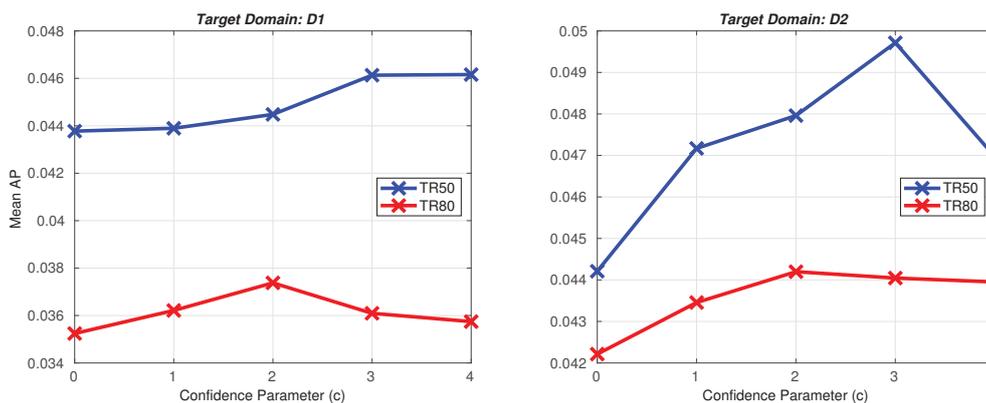


Figure 6.8 : Mean APs over different values of confidence parameter c .

set, i.e. the weights on observed clicks (positive instances) and unobserved clicks (negative instances) are the same. As presented previously, those unclicked items in the training sets **TR-80%** and **TR-50%** do not necessarily indicate user dislike; moreover, a part of truly observed clicks has been held out for testing. Therefore, a lot of unclicks in these training sets are not true negative instances. Hence, it is improper to learn model parameters under binary preference assumption. As a result, we find that the left-hand plots and the right-hand ones in Figure 6.8, $c = 0$ underperforms the other settings. When c is increased, we find that the performance is improved. Specially, the best performance is achieved over **TR-80%** when c is around 2 while the best performance is achieved over **TR-50%** when c is around 3. It can be interpreted that the amount of observed clicks in **TR-50%** is less than that in **TR-80%**, hence a larger c is required to emphasize these positive instances in **TR-50%**. Intuitively, too large c on observed clicks will overwhelm the information learned from unclicks. As expected, the performance drops when we continue increasing c to a larger value.

Table 6.7 : Statistics of Testing Users Grouped by the Number of Ratings

Target Domain	# Clicks	# testing users in TS-50%	
		D1	D2
Experienced	> 10	1,000	617
Cold-Start	1 ~ 10	3,779	3,552

Recall over Different User Groups

The metric $\text{recall}@k$ effectively assesses if a RS has successfully generated a personalized item list for which a user has shown a positive preference. In this experiment, we separated the testing users in the target domain into two groups, namely the Cold-Start User Group where users had fewer than 10 clicks and the Experienced User Group where users had clicked more than 10 items. Table 6.7 shows statistics of these two groups in the target domain D1 and D2 respectively.

We evaluated $\text{recall}@5\sim 20$ over all comparison methods and Figure 6.9 illustrates the results. MostPop does not achieve a high recall and the margin between it and other methods widens as K increases, which illustrates that users have different long-tail items of interest apart from the most popular items. The performance of N-CDCF is even worse than that of MostPop, especially for the Cold-Start User Group. This is because the similarity of neighbors is computed from the data of all domains, hence the neighbors are largely determined by the data in auxiliary domains when a user is cold start in the target domain. However, user preferences in each domain are quite different, and each neighbor of a user tends to have personally preferred long-tail items in the target domain. Accordingly, the clicked items from neighbors are quite varied, which makes the prediction from N-CDCF aimless. MF-IF-CDCF, PARAFAC2 and CDTF-IF underperform MF-IF, for the reason explained in the

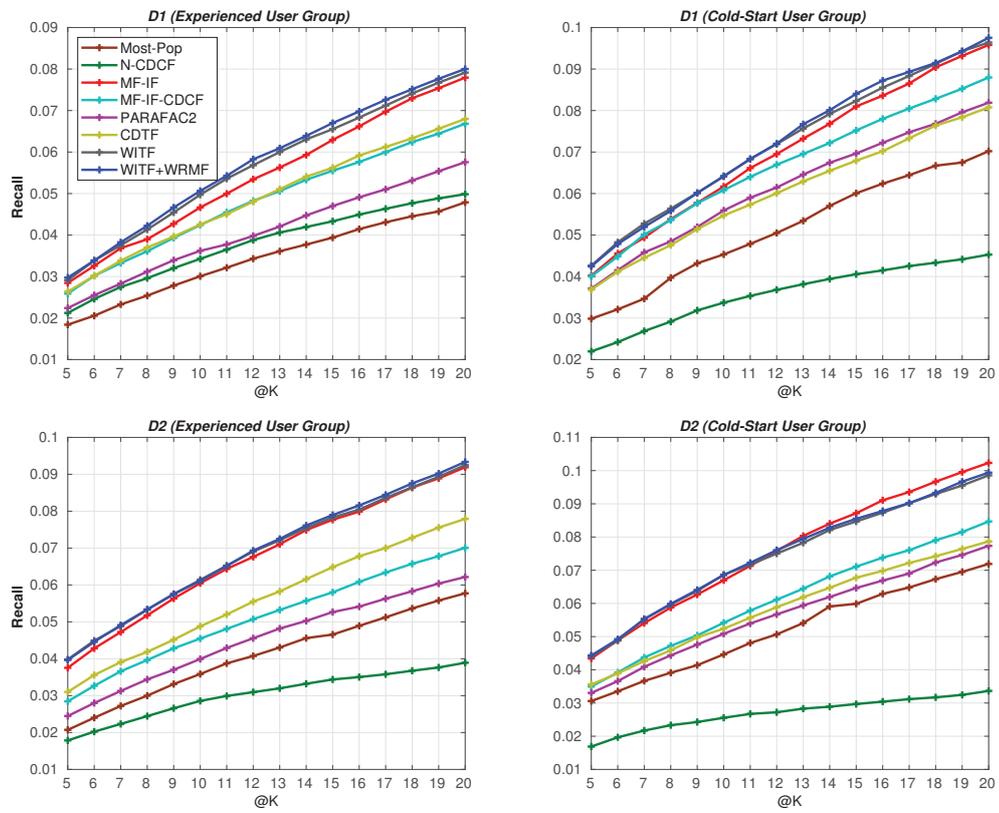


Figure 6.9 : Recall@5~20 of comparison methods on target domain D1 and D2.

previous experiment, i.e. blindly borrowing the knowledge of user preferences from other domains may even hurt preference learning from the target domain due to the heterogeneities between domains. In comparison, as shown in all four subfigures of Figure 6.9, the plots of WITF and WITF+WRMF are above those of other methods with apparent margins. Therefore, we conclude that WITF provides an effective way to model the unary preference data, such as clicks, over multiple domains, which enables personal preferences for target domain items to be better captured by this method than by other methods.

Training Time Comparison

Time cost is a critical problem for taking into account when a model is deployed in a real-world production environment. In this chapter, we designed two strategies to reduce the time cost: (1) a parallelizable parameter learning algorithm (cf. Algorithm 1); (2) the trick to reduce time complexity (cf. Section 6.3.3).

We trained our models on a cluster with 12 Intel Xeon CPUs and 94G memory. Our parallel parameter learning algorithm is implemented with MATLAB Parallel Computing Toolbox. Table 6.8 illustrates the average time to run an iteration under using different number of CPUs, where WITF (RAW) denotes the model without taking the time complexity trick (cf. Section 6.3.3). Obviously, WITF using the time complexity trick is almost twice faster than WITF (RAW), and such improvement will be more significant when more items are involved in a real-world production environment. The time cost reduces when more CPUs are used, which proves the effectiveness of our parallel learning algorithm. The speed can be further improved when the model is implemented with some industrial parallel technique in a production environment. Moreover, the time cost of WRMF for post-learning takes few seconds, and it only needs to be run a few iterations on the basis of the well-estimated parameters from WITF.

Table 6.8 : The average time (in seconds) to run an iteration

# CPUs	1 CPU	4 CPUs	8 CPUs	12 CPUs
WITF (RAW)	50.83	28.64	24.89	22.53
WITF	21.78	17.05	15.27	13.31
WRMF	2.91	2.26	1.87	1.58

6.5.5 Time Period as Domain on Movie Rating Prediction

Watching movies is one of the most frequent recreational activities. People usually see newly released movies because of a movie’s popularity or shared tastes with friends, whereas they may watch old movies for nostalgia, or their relation to new movies. That is, the attractiveness of new movies vs. old movies is quite different. This suggests that movies should be grouped by different time periods to better represent such heterogeneities. As shown in Figure 6.10, the movies in our experiment are grouped into four slices from New to Old according to their release period. This construction implies a cross-domain recommendation problem in which time periods serve as domains. Therefore, we can apply our WITF methods to better capture user preferences from the high-order interaction between users, movies and time periods.

Data Preparation

In this experiment, we used the MovieLens20M dataset which contains the latest movies released prior to 04/2015. We grouped the movies into four domains according to their release date, as reported in Table 6.9. We removed those users who had watched fewer than three new movies from the raw dataset so that we extracted 5,726 users for evaluation. Table 6.9 summarizes the statistics for each domain. In this experiment, we use New and Old as the respective target domains

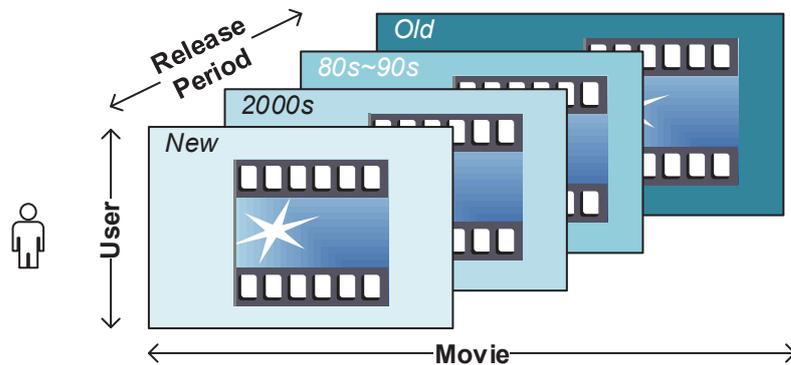


Figure 6.10 : Time period as domain to model the heterogeneities of user preferences on movies

Table 6.9 : Statistics of the MovieLens Dataset over Different Release Time Periods (* Each of these domains is chosen as the respective target domain for evaluation)

Domain	Time Period	#Movies	$\frac{\# \text{ Ratings}}{\# \text{ Users}}$	$\frac{\# \text{ Ratings}}{\# \text{ Movies}}$	Sparsity
Old*	~ 1979	6,750	98.8418	83.8471	0.0146
80s90s	1980 ~ 1999	3,385	106.5423	180.2248	0.0315
2000s	2000 ~ 2012	7,850	210.2548	153.3655	0.0268
New*	2013 ~	1,138	18.603	93.6037	0.0163

for evaluation. For each target domain, we create a training set **TR-50%** and a testing set **TS-50%**, just as in previous experiments.

Rating Prediction Performance Comparison

Table 6.10 reports the MAEs and RMSEs on the target domain New and Old respectively over all testing users. It is interesting to observe that the prediction performance on the target domain Old is overall better than the performance on the

Table 6.10 : MAEs and RMSEs of Comparison Methods on MovieLens Dataset (- denotes baseline, * means $p < 0.01$, ** denotes smallest p)

Target Domain	New (TR-50%)		Old (TR-50%)	
	MAE	RMSE	MAE	RMSE
kNN-CDCF	0.8552	1.1107	1	1.0395
PMF	0.6758	0.9153	0.6277	0.8159
PMF-CDCF	0.5995 ⁻	0.8047 ⁻	0.5844 ⁻	0.7745 ⁻
CMF	0.5996	0.8111	0.5903	0.7891
PARAFAC2	0.6012	0.8069	0.5723*	0.7638*
CDTF	0.5986*	0.8087*	0.5655*	0.7572*
WITF	0.5978*	0.8043*	0.5622*	0.7510*
WITF+WRMF	0.5976**	0.8041**	0.5620**	0.7508**

target domain New. This phenomenon can be interpreted as follows. A user tends to watch new movies due to the movie's popularity, promotion and social relationships, that is, the feedback on new movies is more or less influenced the opinions of others. Old movies, by comparison, are no longer hot topics, so users tend to watch them and give feedback only if they are personally really interested in these movies. The randomness of feedback on new movies is therefore much higher than it is on old movies, and this is why most approaches more easily capture the user preferences on movies in the Old domain and consequently make better predictions.

From Table 6.10, we find that the margins between the different methods are relatively small, and the improvement between the TF-based methods and MF-based methods is not so obvious compared to the previous experiments. This is because all the items in each domain are movies, so the heterogeneities are much smaller than

in the previous experiments. As a result, the MF-based methods, e.g. PMF-CDCF, do not suffer from significant heterogeneity issues when leveraging information from auxiliary domains. Moreover, we find the apparent margins by comparing the performance of PMF and PMF-CDCF in the New domain, which suggests that it is helpful to incorporate more data to train models, since the available user feedback within a small time period is relatively little. WITF-based models outperform the other methods thanks to the sophisticated heterogeneity representation and influence control mechanism. Table 6.10 also reports the sign tests performed between the baseline, i.e. PMF-CDCF, and other models. The results of comparison models significantly outperform the results of baseline are marked with * (p-value \leq 0.01).

The Amount of Influence from Auxiliary Domains

To determine how much amount of information to be leveraged from auxiliary domains has a direct influence on prediction performance in the target domain. In Section 6.3.3, we presented a way to trade off the influence between domains in terms of setting the proportional parameter α_k . In this experiment, we use WITF to evaluate the rating prediction performance in the target domain New and Old respectively by changing α_k . We use only a single auxiliary domain to investigate its influence on the target domain in each case.

Figure 6.11 depicts MAEs changing with the proportional parameter α_k which controls the influence of the auxiliary domains. The results show that performance changes with the value of α_k . In both cases, we find that the prediction performance reaches its best point when a certain amount of influence is borrowed from auxiliary domain. Either too large or too small α_k may degenerate the performance.

From the left figure, i.e. the case of Old movies as the target domain, the best α_k is about 0.4 when New movies serves as the auxiliary domain, and the performance gets worse when α_k increases. In comparison, the best α_k is around 0.8 when

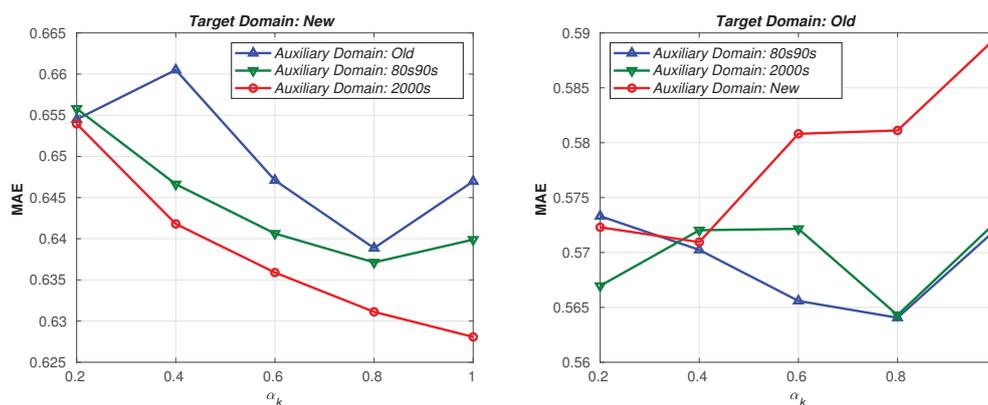


Figure 6.11 : MAE comparison with α_k increasing.

80s90s movies serves as the auxiliary domain. This proves that the user behavior on watching new movies is quite different from watching old ones, so incorporating too much influence from New movies domain may hurt the performance. Comparatively, the time gap between Old movies and 80s90s movies is naturally smaller than that between Old movies and New movies, so their heterogeneities are correspondingly smaller. As a result, the target domain Old can incorporate more information from 80s90s auxiliary domain, i.e. a relative larger α_k .

From the right figure, i.e. the case of New movies as the target domain, similar phenomenon can be observed. The best α_k is about 0.8 when Old movies serves as the auxiliary domain whereas $\alpha_k = 1$ still does not reach the best point and we can continue to set it a larger value when the auxiliary domain 2000s movies. Furthermore, we can find that the MAE using 2000s as the auxiliary domain is consistently smaller than the MAE using either Old or 80s90s as the auxiliary. This proves that the heterogeneities of user preferences between New movies and Old movies is relative larger than those between New and others. As a result, it is helpful to set a smaller α_k so as to control the knowledge learned from the domain of Old movies. In comparison, the user preferences on 2000s movies are relatively closer to those on the New movies, so a larger α_k is preferred to transfer relatively more information

learned from this auxiliary domain. Therefore, the optimal configuration of α_k can be used to quantify the degree of heterogeneity between target domain and auxiliary domain.

6.6 Summary of Contributions

In this chapter, we model a CDRS with tensor factorization. In particular, the non-IID technique is focused on modeling the heterogeneity and coupling relationships between different item domains. The main contributions of this work are summarized as follows:

- We analyze the deficiencies of current cross-domain recommendation methods caused by the heterogeneities between domains. Specially, we address the irregular triadic relation in CDCF, where each domain has different item set. As a result, it becomes unfeasible to apply regular TF models. To deal with this challenge, we design a weighted irregular tensor factorization (WITF) model to couple the data among multiple heterogeneous domains.
- Explicit preference data, e.g., ratings, are not always available in many real-world scenarios in RSs, while objective preference data, such as purchase history, browsing behavior, and click logs, are much more easily obtained. The proposed learning algorithm for WITF can deal with both explicit preference data and implicit preference data in a unified way.
- We propose a transfer learning method in terms of a Weighted regularized matrix factorization (WRMF) model, where the user factors, item factors and domain factors learned from WITF serve as informative cross-domain priors to regularize latent factor learning in WRMF. These informative priors enable WRMF to infer user preferences in cold-start domains.

- We perform a collection of experiments on three real-world cases and make comparisons with other state-of-the-art methods to test the effectiveness of our approach. These are: (1) multi-category product recommendation in an e-business site; (2) multi-category attractive item recommendation in a social networking site based on implicit feedback; (3) movie recommendation over long time periods. All the evaluation results prove that our approach significantly outperforms other comparison methods.

Algorithm 1 Weighted Irregular Tensor Factorization

Input: \mathbf{X}_k is the data matrix for each domain,

ω_k is the influence weight for each domain,

$w_{k,i,j}$ is the weight on each entry,

$\lambda_U, \lambda_V, \lambda_C$ are the regularization parameters

Output: \mathbf{U} is the factor matrix for users,

\mathbf{C} is the factor matrix for domains,

$\mathbf{V}, \{\mathbf{P}_k\}$ are the factor matrices for items

Initialization

- 1: $\ddot{\mathbf{W}}_{k,i,j} \leftarrow \omega_k w_{k,i,j}, \mathbf{V} \leftarrow \mathbf{I}$
 - 2: Randomly initialize \mathbf{U}, \mathbf{C}
 - 3: $\mathbf{P}_k \leftarrow \mathbf{A}_R \mathbf{B}_R^\top$, with the SVD: $\mathbf{X}_k^\top \mathbf{U} \Sigma_k \mathbf{V}^\top \approx \mathbf{A}_R \Sigma_R \mathbf{B}_R^\top$
 - 4: **while** Not convergence **do**

Add neighbor noisy examples (optional):

 - 5: Randomly select S blank entries for each user i
 - 6: Fill neighbor noisy examples in the selected entries
 - 7: Generate tensor \mathbf{Y} with the slice for each domain k : $\mathbf{Y}_k \leftarrow (\ddot{\mathbf{W}} \cdot^* \mathbf{X}_k) \mathbf{P}_k$
 - 8: **for** iteration $< m$ **do**
 - 9: Update $\mathbf{U}_{i,:}$ in parallel for each user i using Eq. 6.26
 - 10: Update $\mathbf{C}_{k,:}$ in parallel for each domain k using Eq. 6.27
 - 11: Update \mathbf{V} using Eq. 6.29
 - 12: **end for**
 - 13: **for** iteration $< n$ **do**
 - 14: Update \mathbf{P}_k in parallel for each domain k using Eq. 6.23
 - 15: **end for**
 - 16: Return $\mathbf{U}, \mathbf{V}, \mathbf{C}, \{\mathbf{P}_k\}$
 - 17: **end while**
-

Chapter 7

A Session-based Recommender System for Modeling the Intra- and Inter-sessions Context

7.1 Introduction

Most classic RSs do not consider the context of the current session. As a result, these RSs may recommend quite irrelevant items to the current context. Moreover, these RSs tend to repeatedly recommend similar items to users due to the relevance of historical choice. Factor models, such as MF [117], and neighborhood methods, such as item-based collaborative filtering [193], are the two most prevalent approaches in RSs. However, these two approaches are not immediately applicable to session-based RS (SBRS) because they do not consider sequential relevance over user choices. As a result, these RSs tend to produce homogeneous recommendations similar to their historical profiles or recent purchases as depicted in Figure 7.1 (a). In practice, users tend to have different requirements in the context of changing sessions; therefore, the homogeneous recommendation will greatly degrade user satisfaction and business benefits. Recently, researchers have begun to incorporate session context into RSs.

To capture relevance between items in a session, sequential pattern mining (SPM) is a simple and straightforward approach [244]. SPM extracts a set of patterns according to the co-occurrence frequency in the historical data. However, a session context may consist of an arbitrary set of items in the recommendation problem so it probably fails to match any pattern from the extracted pattern space. Moreover, SPM can hardly capture the long-term relevance and the transition over multiple

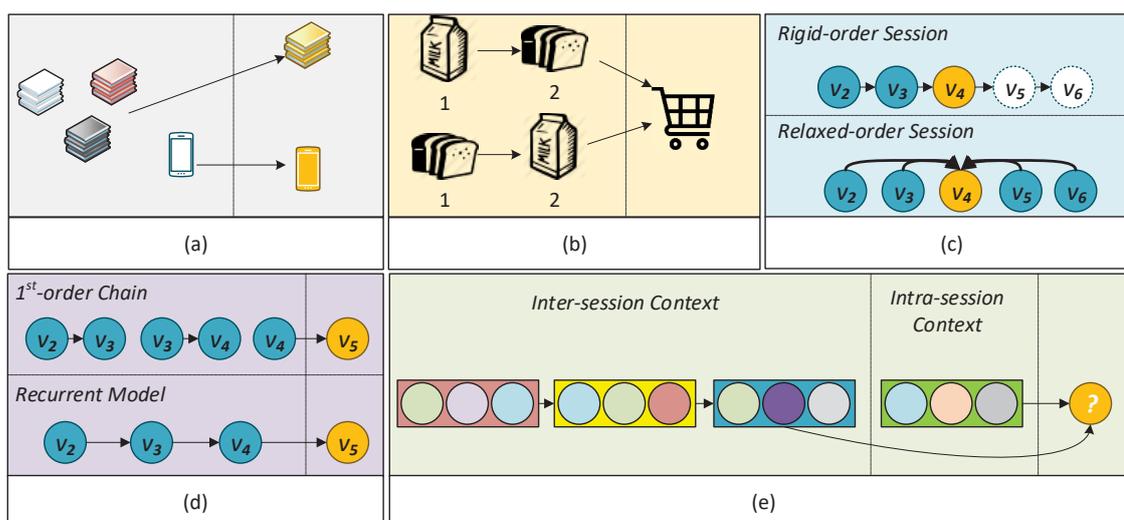


Figure 7.1 : The motivation of this work: (a) Classic recommendation without considering session context; (b) Two sequences adding milk and bread into the cart; (c) Rigid-order session vs. relaxed-order session; (d) First-order Markov chain model vs. recurrent model; (e) Next item prediction based on both intra- and inter-session contexts.

items. Markov chain (MC) is another straightforward way to model sequential data [32]. However, MC only captures the first-order dependency, i.e., it predicts the transition between a pair of items instead of that between an item and a contextual item set. Neural models have achieved tremendous success in a number of tasks, including RSs [247], with the development of deep learning. Recently, researchers [76] have applied RNN for SBRS.

Both MC and RNN are originally applied on time series data in a natural order. We argue that current SBRSs often assume a rigidly ordered sequence over data which may not fit in many real-world cases. For example, the order in which toast, milk, and ham are put into a shopping cart generally makes no difference in a real-world transaction, as shown in Figure 7.1 (b). Moreover, most real-world datasets do not provide precise timestamps on items purchased. Accordingly, we cannot obtain

the true rigidly ordered sequences over these datasets. In this work, we relax this assumption on the items in a session to allow an arbitrary order in a neighborhood window as illustrated in the bottom of Figure 7.1 (c). Moreover, the next-item recommendation depends on not only the current session context but historical sessions. For example, a user has bought a smartphone in a recent session and they do not tend to buy a new one in the next sessions in a short term. Therefore, we propose cross-session filtering to jointly consider intra- and inter-session context for the next-item recommendation.

If we regard a sequence of items for a session as a sequence of words in a sentence, and a sequence of sessions for recent purchases as a sequence of sentences for a paragraph, then we can borrow some successful experience in modern language modelings. Especially, the word2vec models, which build context from the words in a window without assuming a rigid order over the word sequence, inspire us to build SBRS over relaxed-order sessions. As a result, we propose neural cross-session filtering (NCSF) to conduct next-item recommendation under intra- and inter-context in terms of a neural network model as illustrated in Figure 7.1 (e). The empirical experiments on a real-world e-commerce dataset show the superiority of our model over the state-of-the-art methods.

7.2 Problem Formulation

Before introducing the specificity of our model, we first formulate the problem and clarify basic concepts.

In general, user set $\mathcal{U} = \{u_1, u_2, \dots, u_{|\mathcal{U}|}\}$ and item set $\mathcal{V} = \{v_1, v_2, \dots, v_{|\mathcal{V}|}\}$ are two basic elements in RSs. As to SBRSs, $\mathcal{S} = \{s_1, s_2, \dots, s_{|\mathcal{S}|}\}$ denotes session set containing all observed sessions, where each session consists of a collection of items, $s_i \in \mathcal{V}^{|s_i|}$. $\mathcal{S}^u = \{s_1^u, s_2^u, \dots, s_{|\mathcal{S}^u|}^u\}$ denotes the session set w.r.t. user u , i.e., $\mathcal{S}^u \subset \mathcal{S}$. Specially, we denote s_T^u as a current session and s_{T-1}^u as the previous

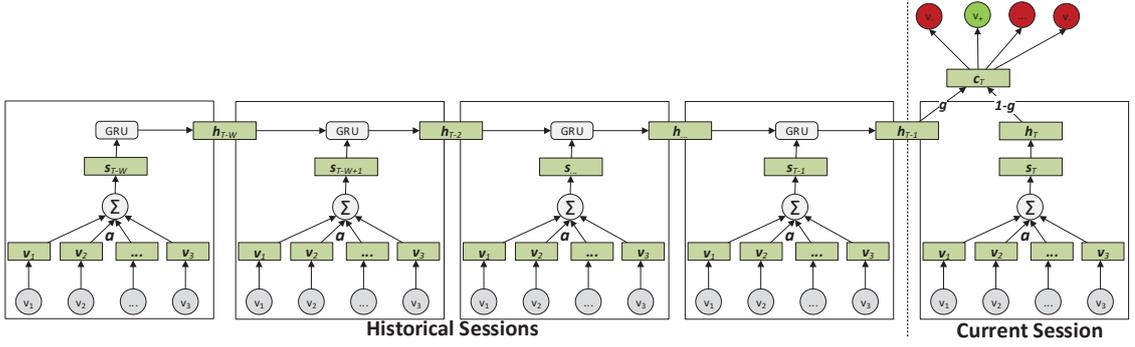


Figure 7.2 : The architecture of NCSF. In the left part of split line, historical sessions are used to build inter-session context. In the right part of split line, the items in the current session are used to build intra-session context.

session. Given a target item $v_t \in s_T^u$, a context window with the size $2L$ is used to include the relevant items before and after v_t as the intra-session context, $\mathbf{c}_{T,t}^u = \{v_{t-L}, \dots, v_{t-1}, v_{t+1}, \dots, v_{t+L}\}$. Moreover, we incorporate user's W previous sessions w.r.t. s_T^u as the inter-session context, namely $\mathbf{c}_{\langle T,W \rangle}^u = \{s_{T-1}^u, s_{T-2}^u, \dots, s_{T-W}^u\}$.

Then, the objective of NCSF is to train a classifier which maximizes the conditional distribution $P(v_t | \mathbf{c}_{T,t}^u, \mathbf{c}_{\langle T,W \rangle}^u)$ with the intra-session context $\mathbf{c}_{T,t}^u$ and inter-session context $\mathbf{c}_{\langle T,W \rangle}^u$. Correspondingly, the next-item recommendation problem is reduced to generate the ranking over all candidate items under the corresponding intra- and inter-session context $\langle \mathbf{c}_{T,t}^u, \mathbf{c}_{\langle T,W \rangle}^u \rangle$. Thus, we can obtain the rank by sorting the probabilities over all items, i.e., $\{P(v | \mathbf{c}_{T,t}^u, \mathbf{c}_{\langle T,W \rangle}^u) | \forall v \in \mathbf{V}\}$.

7.3 Neural Cross-session Filtering

Figure 7.2 illustrates the architecture of NCSF, which trains a probabilistic classifier to maximize the predictive probability on the most relevant item for a given user session context $\mathbf{c}_{T,t}^u$ and its corresponding historical sessions $\mathbf{c}_{\langle T,W \rangle}^u = \{s_{T-1}^u, \dots, s_{T-W}^u\}$. In brief, NCSF can be divided into three parts: (1) the historical session encoder as shown in the left part of Figure 7.2; (2) the current session

encoder as shown in the lower right part of Figure 7.2 and (3) the joint intra- and inter-session context encoder for item prediction as shown in the upper right part of Figure 7.2.

7.3.1 Historical Session Encoding

Specifically, we first illustrate how to encode a historical session. Given a historical user session $s_t^u \in \mathbf{c}_{\langle T, W \rangle}^u$, the input of s_t^u is all the items of this session. For each item $v_i \in s_t^u$, it is mapped into an embedding, i.e., a continuous lower-dimension vector, according to its ID, denoting $\mathbf{e}_i \in \mathbb{R}^K$. Thus, we can map all items in s_t^u into their embedding vectors $\{\mathbf{e}_1, \dots, \mathbf{e}_{|s_t^u|}\}$.

Then, we need to encode these item embeddings into an high-level embedding as the representation of this intra-session context. As illustrated in Eq. 7.1, we construct this context embeddings \mathbf{s}_t as a mixture of $\{\mathbf{e}_1, \dots, \mathbf{e}_{|s_t^u|}\}$.

$$\mathbf{s}_t = \sum_{v \in s_t^u} w_v \mathbf{e}_v \quad (7.1)$$

where $\sum_v w_v = 1$ is a normalized weight. Each w_v signifies how informative of the item in this session context. For example, Hu et al. [88] proposes a heuristic strategy to assign weights. That is, the contextual items previous and next to the target item v_t , i.e., v_{t-1} and v_{t+1} , have the largest weights, and those context items farther from v_t are assigned smaller weights.

However, above heuristic strategy implicitly assumes the order of items in the context, which is often inconsistent with the real-world cases. Furthermore, the closer item may not be the more informative item in the context. As a result, we employ attention mechanism to learn the importance of each item in the context. We use $\mathbf{f}^h \in \mathbb{R}^K$ to denote the context filter over all item embeddings in the session s_t^u to learn their importance. Then, we compute the attention score w_v in terms of

a two-layer neural network.

$$\begin{aligned} a_v &= \tanh(\mathbf{f}^h \mathbf{e}_v + b) \\ w_v &= \text{softmax}(a_v) \end{aligned} \tag{7.2}$$

where $\text{softmax}(a_v) = e^{a_v} / \sum_i e^{a_i}$ and b is the bias.

Then, we can obtain all historical session context embeddings $\{\mathbf{s}_{T-1}, \dots, \mathbf{s}_{T-W}\}$ as above. To accumulate the influence from historical sessions, we input this historical session context embedding sequence into a RNN with gated recurrent units (GRU) [36].

$$\mathbf{h}_t = GRU_{\theta^G}(\mathbf{s}_t, \mathbf{h}_{t-1}) \tag{7.3}$$

where θ^G denotes the parameters of GRU. Each GRU cell takes session embedding at time t and previous output \mathbf{h}_{t-1} as the inputs, and then integrate these two inputs into \mathbf{h}_t . At the final step of historical sessions, we obtain the output \mathbf{h}_{T-1} which stores the information of this historical session sequence.

7.3.2 Current Session Encoding

Given current user session s_T^u , the items in the context window, $\mathbf{c}_{T,t}^u$, play the most important role in predicting the target item v_t . To represent this context, similar to encode historical sessions, we employ another two-layer network to learn the attention over each item, $v \in \mathbf{c}_{T,t}^u$, in terms of their embedding \mathbf{e}_v .

$$\begin{aligned} a_v &= \tanh(\mathbf{f}^g \mathbf{e}_v + b) \\ w_v &= \text{softmax}(a_v) \end{aligned} \tag{7.4}$$

where $\mathbf{f}^g \in \mathbb{R}^K$ is the context filter.

Then, the context embedding is constructed as:

$$\begin{aligned} \mathbf{s}_T &= \sum_{v \in \mathbf{c}_{T,t}^u} w_v \mathbf{e}_v \\ \mathbf{h}_T &= \tanh(\mathbf{s}_T) \end{aligned} \tag{7.5}$$

7.3.3 Intra- and Inter-session Context Encoding

So far, we have modeled the historical session context and current session context. Since both of them contribute to the next item prediction, we need to integrate them into a joint context embedding. However, what information should be preserved from the historical session context is uncertain. Therefore, we design a gate vector, \mathbf{g} , to filter the information from historical session context embedding \mathbf{h}_{T-1} and current session context embedding \mathbf{h}_T .

We employ a two-layer neural network to compute the gate vector \mathbf{g} , which takes \mathbf{h}_{T-1} and \mathbf{h}_T as the inputs to learn how to integrate these two context information.

$$\mathbf{g} = \sigma(\mathbf{W}^g[\mathbf{h}_{T-1}; \mathbf{h}_T] + b) \quad (7.6)$$

where $\sigma(x) = 1/(1+e^{-x})$ is a sigmoid function and $\mathbf{W}^g \in \mathbb{R}^{K \times 2K}$ is a weight matrix. Then, we can integrate \mathbf{h}_{T-1} and \mathbf{h}_T into a joint context embedding \mathbf{c}_T in terms of \mathbf{g} ,

$$\mathbf{c}_T = \mathbf{g} * \mathbf{h}_{T-1} + (1 - \mathbf{g}) * \mathbf{h}_T \quad (7.7)$$

where $*$ denotes element-wise product.

7.3.4 Objective Function

Given the joint intra- and inter-session context embedding \mathbf{c}_T , we can compute the score S_{v_t} of the target item v_t in terms of the context embeddings $\langle \mathbf{h}_u, \mathbf{h}_c \rangle$:

$$S(v_t|\mathbf{c}_T) = \mathbf{W}_{t,:}^S \mathbf{h}_c \quad (7.8)$$

where $\mathbf{W}_{t,:}^S$ denotes the t -th row of $\mathbf{W}^S \in \mathbb{R}^{|\mathcal{V}| \times K}$. This score function quantifies the compatibility of the target item v_t with the joint session context. As a result, the conditional distribution $P_{\Theta}(v_t|\mathbf{c}_T)$ can be defined in terms of a softmax function.

$$P_{\Theta}(v_t|\mathbf{c}_T) = \frac{\exp(S(v_t|\mathbf{c}_T))}{Z(\mathcal{V}|\mathbf{c}_T)} \quad (7.9)$$

where $Z(\mathcal{V}|\mathbf{c}_T) = \sum_{v \in \mathcal{V}} \exp(S(v|\mathbf{c}_T))$ is the normalizing constant and Θ defines the model parameter set $\Theta = \{\{\mathbf{e}_i\}, \theta^G, \mathbf{f}^h, \mathbf{f}^g, \mathbf{W}^g, \mathbf{W}^S\}$. As a result, we obtain a probabilistic classifier for NCSF.

Given $\mathcal{D} = \{\langle \mathbf{c}_{\langle T,W \rangle}^u, \mathbf{c}_{T,t}^u \rangle | u \in \mathcal{U}, s_T^u \in \mathcal{S}^u, \mathbf{c}_{T,t}^u \subset s_T^u\}$ as all possible user current session context and historical session context as the training dataset, we easily obtain the joint probability distribution over it:

$$P_{\Theta}(\mathcal{D}) \propto \prod_{\langle \mathbf{c}_{\langle T,W \rangle}^u, \mathbf{c}_{T,t}^u \rangle \in \mathcal{D}} P_{\Theta}(v_t | \mathbf{c}_{\langle T,W \rangle}^u, \mathbf{c}_{T,t}^u) \quad (7.10)$$

Therefore, we can learn the model parameters Θ by maximizing the conditional log-likelihood (cf. Eq. 7.9):

$$\begin{aligned} L_{\Theta} &= \sum_{\langle \mathbf{c}_{\langle T,W \rangle}^u, \mathbf{c}_{T,t}^u \rangle \in \mathcal{D}} \log P_{\Theta}(v_c | \mathbf{c}_{\langle T,W \rangle}^u, \mathbf{c}_{T,t}^u) \\ &= \sum_{\langle \mathbf{c}_{\langle T,W \rangle}^u, \mathbf{c}_{T,t}^u \rangle \in \mathcal{D}} S(v_t | \mathbf{c}_T) - \log Z(\mathcal{V} | \mathbf{c}_T) \end{aligned} \quad (7.11)$$

7.3.5 Learning and Prediction

Both evaluating L_{Θ} and computing the corresponding log-likelihood gradient involves the normalizing term $Z(\mathcal{V}|\mathbf{c}_T)$, which needs to sum $\exp(S(v|\mathbf{c}_T))$ over the entire item set for each training example, cf. Eq. 7.9. This means that training this model needs to take time $|\mathcal{V}| \times |\mathcal{D}|$ to compute the normalizing constants over all the training examples for each iteration. Unfortunately, $|\mathcal{V}|$ and $|\mathcal{D}|$ are always large in real RSs, which makes the training process intractable.

The vocabulary size in NLP systems is also large so language modeling runs into similar challenge for next word prediction [153]. In this chapter, we adopt a subsampling approach, i.e., negative sampling, to approximate the softmax computation over all items. More specifically, we randomly draw a small set of items as the N noise samples $\{v_1^-, \dots, v_N^-\}$ w.r.t. the true target item v_t , where $N \ll |\mathcal{V}|$

denotes the number of noise samples. Accordingly, we approximate the softmax for the target item (cf. Eq.7.9) as follows:

$$P_{\Theta}(v_t|\mathbf{c}_T) = \frac{\exp(S(v_t|\mathbf{c}_T))}{Z(\tilde{\mathcal{V}}|\mathbf{c}_T)} \quad (7.12)$$

where $\tilde{\mathcal{V}} = \{v_t, v_1^-, \dots, v_N^-\}$.

To learn the parameters, we adopt a gradient-based algorithm over $\partial L_{\Theta}/\partial\theta$ w.r.t. each parameter $\theta \in \Theta$. We implement our model by using Keras [37] with Tensorflow GPU version as backend. We use Adam [111] as the gradient optimizer and the mini-batch size is set to 200.

When the model has been trained, we can immediately use it for next item recommendation. Given a user current session context and his/her historical session context, we can compute the scores over all candidate items according to Eq. 7.8, and then rank them accordingly.

7.4 Empirical Study

For empirical evaluation, we use the IJCAI-15 competition dataset* which is a real-world dataset collected from Tmall.com. Tmall is the largest online B2C platform in China, and it contains anonymized user shopping logs for the six months before and on the “Double 11” day (11th November).

7.4.1 Data Preparation

The raw dataset contains 600,357 users and 372,740 items. Note that this dataset only provides buying date without a specific time, so we treat a user’s shopping records in a natural day as a session. For intra-day sequence, we retain the order given in the raw data. First, we remove items which were bought less than ten times and those users who have less than three shopping sessions. Then, we remove

*<https://tianchi.shuju.aliyun.com/datalab/dataSet.htm?id=1>

Table 7.1 : Statistic of IJCAI-15 dataset for evaluation

# users:	50,197
# items:	52,206
# item/# session:	3
# training sessions:	195,866
# training examples:	399,394
# testing cases (<i>LAST</i>):	4,423
# testing cases (<i>LOO</i>):	11,852

all singleton sessions, i.e., only containing one item, from the raw data. From the six-month shopping logs, we randomly hold out 20% of the sessions from the last 30 days for testing, and the remaining data is used for training. In particular, we construct two testing sets: *LAST* means that the last item in each testing session is used as ground truth, and *LOO* means each item in a testing session was held out, in turn, to serve as ground truth, i.e., leave-one-out. The statistics of this dataset for evaluation are summarized in Table 7.1.

7.4.2 Comparison Methods

For conducting an empirical study, we compare the representative state-of-the-art methods introduced in Section 3.5 with the following settings:

- *MostPop*: This is a simple recommendation method to rank items for recommendation according to occurrence frequency without considering session context.
- *FPMC*: It combines MF and first-order MC as an SBRS, which uses personalized MC for sequential prediction.

Table 7.2 : The evaluation results for MRR and AUC

<i>LAST</i>				
Model	MRR@10	MRR@20	MRR@50	AUC
<i>MostPop</i>	0.0077	0.0086	0.0092	0.8170
<i>FPMC</i>	0.1482	0.1516	0.1537	0.8075
<i>GRU4Rec</i>	0.2601	0.2654	0.2676	0.8756
<i>SWIWO</i>	0.4038	0.4071	0.4084	0.9126
<i>NCSF</i>	0.4403	0.4440	0.4457	0.9407
<i>LOO</i>				
Model	MRR@10	MRR@20	MRR@50	AUC
<i>MostPop</i>	0.0089	0.0102	0.0109	0.8269
<i>FPMC</i>	0.1611	0.1649	0.1666	0.8211
<i>GRU4Rec</i>	0.2494	0.2556	0.2584	0.8890
<i>SWIWO</i>	0.4211	0.4242	0.4258	0.9163
<i>NCSF</i>	0.4403	0.4440	0.4457	0.9407

- *GRU4Rec*: This SBRS consists of a deep RNN with the GRU cells.
- *SWIWO*: This is relaxed-order intra-session context model with a shallow network architecture.
- *NCSF*: This is the model proposed in this chapter considering both intra- and inter-session context.

Results

Table 7.2 demonstrates the results of MRR@10, MRR@20, MRR@50 and AUC over the testing sets *Last* and *LOO*. From Table 7.1, we find the number of candidate

items is above 50,000, so it is a big challenge to predict the true next item. MostPop achieves reasonable results, which reflects common user behavior for online shopping, that is, users tend to choose items with high sales volume, i.e., popularity, but users will not always buy the same popular items because they have quite different requirement in the different context. We set the number of latent factors to 20 for training FPMC, and the results become worse when more factors are used. This is because the dataset is very sparse for MF methods where each row only contains less than two items (cf. the *avg. session length* in Table 7.1, note that one of the items needs to be used as the output) and the others are all empty (cf. *# items* in Table 7.1). Furthermore, most users only have three sessions although we remove users with too few sessions. Therefore, this constructs a very sparse matrix to train the MF model. Moreover, FPMC employs a first-order MC to learn the transitions over successive items. We set the item embedding size as 100 for GRU4Rec. The structure of GRU enables GRU4Rec to accumulate the long-step influence from relatively early selections, which results in a qualitative leap. We can find that the MRRs of GRU4Rec are above 0.25 in both testing cases.

We set 100 units for the item embeddings when training the SWIWO model. We find that SWIWO outperforms GRU4Rec by clear margins, where the MRR is above 0.4 and the AUC is above 0.91 for both testing cases. The highest MRR also proves that SWIWO can accurately list user-desired items on the first page. Moreover, SWISO has a shallow structure, which makes it efficient to recompute the scores over all candidate items when many session contexts keep updating in online SBRs. When training the NCSF model, we set 100 units for the item embeddings, window size 4 for modeling the intra-context and two historical sessions as the inter-session context. From the results, we find that NCSF outperforms SWIWO in MRRs by 5% and AUC by 3% because NCSF incorporates the inter-session context as well as the intra-session context. Another important cause leads SWIWO and NCSF to

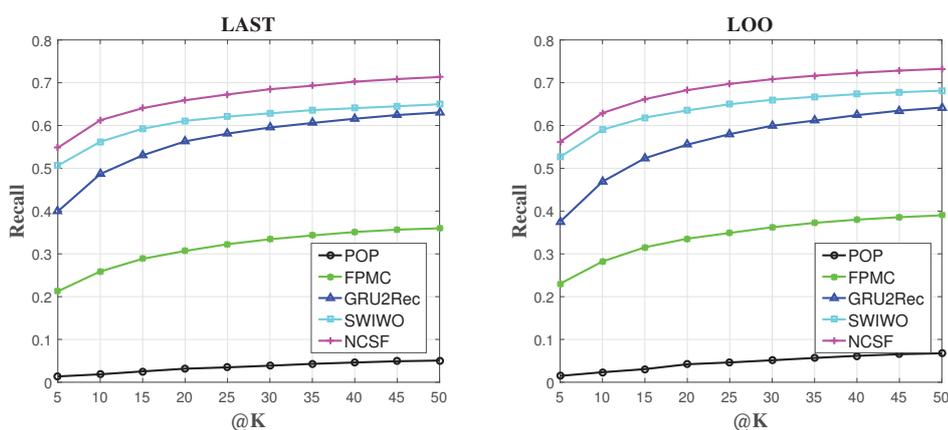


Figure 7.3 : REC@5 - REC@50 on test cases *LAST* and *LOO*

better performance than GRU4Rec is that SWIWO and NCSF apply a relaxed-order session context modeling strategy which builds a more reasonable session context to predict the next item selection. Furthermore, NCSF incorporates the attention mechanism to build both the intra-session context and inter-session context, which is advanced to the context building heuristic strategy applied by SWIWO.

Figure 7.3 depicts the recall of all comparison methods. Here, we chose $K \in [5, 50]$ because most users are only interested in viewing the recommendation on the first few pages in the real-world scenario. We find that the plots of NCSF and SWIWO are above the plots of baselines with apparent margins, i.e., NCSF and SWIWO can more accurately retrieve the next items for each user in top positions. NCSF combines the information from both the current session context and the historical session context, which leads to the best recall.

7.4.3 Diversity Evaluation

People have realized the harmfulness of evaluating RSs only using accuracy metrics [54]. Recall that we propose to diversify recommendation for different user-session context to replace similarity-based recommendation over history profile. Therefore, we respectively evaluate our model and other comparison methods

Table 7.3 : The evaluation results for diversity

<i>LAST</i>					
	MostPop	FPMC	GRU4Rec	SWIWO	NCSF
<i>DIV@10</i>	0	0.9234	0.9977	0.9978	0.9980
<i>LOO</i>					
	MostPop	FPMC	GRU4Rec	SWIWO	NCSF
<i>DIV@10</i>	0	0.9616	0.9982	0.9983	0.9983

from the perspectives of accuracy and diversity, but they often cannot be optimized simultaneously [251]. Since we aim to diversify recommendation under different contexts, we use the evaluation metrics defined in Section 3.4.3.

Results

Table 7.3 demonstrates the diversity of the top-10 recommendations over testing sets *Last* and *LOO*. MostPop always recommends the same items for all sessions so it has zero diversity. FPMC is a first-order MC model which does not consider the long influence of early choices. Accordingly, we get relatively low diversity for FPMC compared with GRU4Rec, SWIWO, and NCSF. GRU4Rec, SWIWO, and NCSF take all the items in the session context into account, therefore they all generate very high diversity recommendations under different session context.

Figure 7.4 illustrates $F1_{MRR-DIV}@10$ and $F1_{REC-DIV}@10$ testing sets *Last* and *LOO* to jointly consider accuracy and diversity. Although GRU4Rec achieves very close diversity to SWIWO and NCSF, its accuracy performance, i.e., MRR@10 and REC@10 shown in Table 7.2, are lower than those of SWIWO and NCSF. Accordingly, GRU4Rec achieves lower F1 scores than SWIWO and NCSF. In particular, from Figure 7.4, we find that NCSF achieves higher F1 scores than those generated

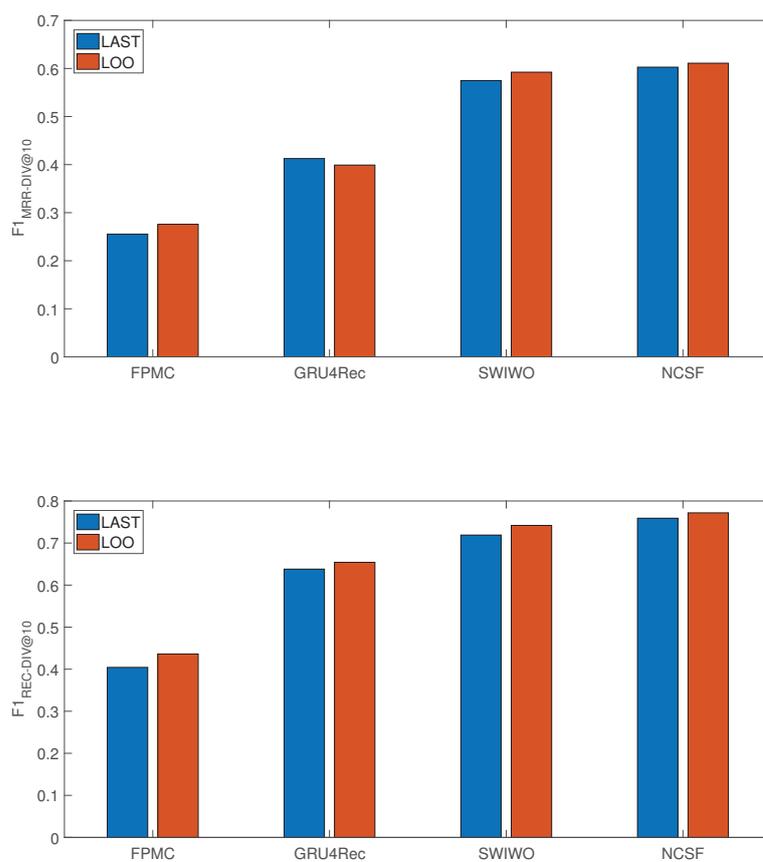


Figure 7.4 : F1 scores on test cases *LAST* and *LOO*

by SWIWO. This is because NCSF takes the influence from historical sessions into account, which makes the recommendations based on more considerable and diverse contexts. In comparison, SWIWO does not model historical sessions and it applies a heuristic strategy to model the intra-session context. As a result, we conclude that NCSF is the best model to serve for real-world SBRs, jointly considering accuracy and diversity.

7.5 Summary of Contributions

In this chapter, we model a session-based recommender system with RNN and attention mechanism. In particular, the non-IID technique is focused on modeling the coupling relationships of items within a session and between sessions. Our main contributions are summarized as follows:

- We analyze the deficiencies of recommendation in classic RSs without considering session context and SBRs with rigid order assumption. Accordingly, we relax this assumption on the items in a session to allow an arbitrary order in a neighborhood window.
- We devise a neural cross-session filtering (NCSF) framework which jointly models the coupling relationships within the intra-session context and between the inter-session context when recommending the next item.
- The empirical evaluation conducted on the real-world e-commerce dataset proves the comprehensive superiority of our approach over all other state-of-the-art methods.

Part IV

Non-IID RS: Modeling Non-IIDness on Interactions

Chapter 8

A Multi-objective Recommender System for Modeling Specialty and Credibility for Long-tail Recommendation

8.1 Introduction

We are leaving the information age and entering the recommendation age [7]. Because of this, RSs are playing an increasingly important role than ever before. CF is a core component of modern RSs; it leverages feedback from other users and items to generate recommendations for a target user. However, CF techniques are still challenged by complicated real-world data characteristics. On the one hand, some of the challenges arise from the distribution of real-world data in nature. It is known that a lot of real-world data can be observed following a long-tail, a.k.a., power-law distribution, as discussed in Section 1.2. Due to a lack of sufficient data for most users and items, data sparsity and cold start are two of the most common research issues addressed by the study of RSs. On the other hand, other challenges may be caused by human behavior. For instance, shilling attack is one such typical issue as presented in Section 1.2. Since the basic idea of CF is to predict ratings in terms of the related data associated with other users and items, insufficient data and spam data will obviously deteriorate recommendation results, especially for those users and items in the tail of distribution.

To date, most research has focused on improving the accuracy of RSs. However, simply improving the accuracy by one or two percent will hardly result in a better user experience or a greater business benefit. Here, we give an intuitive interpre-

tation from the long-tail distribution. A long-tail distribution implies skewed data that has a short head and a long tail, that is, a small number of popular items in the head part, which account for most of the data, whereas the large number of items in the tail only account for a small amount of data. Here, we use the experimental Rich Epinions Dataset (RED) [151] as a demonstration. This dataset was crawled from the well-known online review Web site epinions.com, which contains a total of 1,127,673 reviews given by 113,629 users on 317,755 items. Each review contains a user rating, and the density of this dataset is only 0.003%. The left- and right-hand sides of Figure 8.2 depict, respectively, the long-tail distribution for items and for users. We find that only a few items and users in the short head have sufficient ratings while a large number of items and users in the long tail have less than ten ratings each. Such a skewed data distribution causes RSs to learn users' preferences largely from popular items because these items account for the majority of the data. As a result, the improvement of recommendations is largely determined by popular items. However, such improvement for popular items is trivial because popular items are likely already known by most users who can make the decision to choose them or not. Moreover, Anderson's well-known research suggests that future businesses will obtain more profit from long-tail selling [7]. Motivated by these observations, we focus on improving the quality of recommendations for tail users and items, which we believe will be a great benefit for both business profits and the user's experience.

8.1.1 Challenges of Tail Users and Items

In recent years, a lot of CF techniques have been developed [206], where k NN [26] and MF [117] are two examples. In Chapter 3, we have briefly reviewed the k NN and MF models. Due to the skewness of the distributions of the users and items (cf. Figure 8.1), the data pulled from long-tail users and items is much sparser than that of short-head users and items. As a result, the k NN and MF models are more

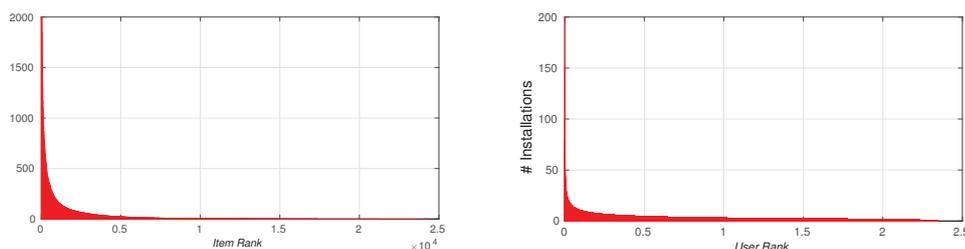


Figure 8.1 : Items (left) and users (right) are ranked by the number of their ratings (truncated from 0 to 100) on Rich Epinions Dataset; they are both clearly distributed with short heads and long tails.

vulnerable to the following challenges than long-tail users and items.

Popularity Bias: Given any two users, their choices tend to overlap more with popular items but less so with long-tail items. Hence, the neighbor set in k NN is largely constructed from popular items. Furthermore, note that the data is very sparse in the tail, i.e., each tail item is chosen by very few users. As a result, a user often cannot find any feedback on those long-tail items from neighbors, so a prediction is unavailable (cf. Eq. 3.1), which creates a situation where those items will never be recommended. As for popular items, they tend to have sufficient feedback, so k NN can more easily make predictions and recommend them. Although MF does not directly depend on neighbor data, it still suffers from the tail’s data-sparsity challenge. From Eq. 3.6 and Eq. 3.7, we find that the estimates of item factors and user-factors are largely determined by the amount of observed data w.r.t. item j and user i . For items and users in the tail, the amount of observable data is small, so the quality of the estimates for them are naturally poorer than with short-head items and users. Accordingly, the predictions for short-head items and users are much more accurate than those for long-tail items and users.

Cold Start: The long-tail distribution implies a number of users are cold start in nature. Cold-start users usually have provided little feedback on some popular

items, or sometimes no feedback at all. Given a cold-start user, k NN does not have sufficient data to find suitable neighbors. As a result, the prediction results for long-tail items or users is poor. From Eq. 3.7, we find that the factors, \mathbf{U}_i , differentiate user preferences according to the feedback they have provided on different items. However, long-tail users have provided very little feedback on popular items so the learned factors of these users tend to be similar; for this reason, they are not able to clearly represent personal preferences. In extreme cases, we find that neither k NN nor MF can work, cf. Eq. 3.1 and 3.7, when we have only cold-start users without any feedback data.

Shilling Attack: As mentioned, a shilling attack refers to a group of spam users intentionally providing fake feedback, e.g., much higher or lower ratings than a true rating to bias the ratings and the recommendations for them. Intuitively, short-head items are well known, and users either actively or passively learn information about them from many sources. For this reason, short-head items are less affected by shilling attack. In comparison, information on unpopular items is limited, and they are largely known by recommendations. Thus, these items can suffer more easily from shilling attack. Moreover, the number of ratings on head items is much more than on tail items; as a result, it is much easier to attack tail items by imputing a few fake ratings. For example, given a head item that has 1,000 ratings and a tail item that has 5 ratings, and where both have an average score of 4, if a shilling attack gives five low ratings with a score of 1 on both items, the average rating of this head item is still close to 4, but the average rating of the tail item is reduced to 2.5.

Since k NN depends on neighbors' feedback to construct a prediction, it can suffer greatly from shilling attack on tail items due to the large proportion of fake feedback. As for the MF method, each item factor vector, \mathbf{V}_j , determines how this item would be preferred by users. If a shilling attack is conducted on a tail item j , \mathbf{V}_j is learned

largely only from fake feedback (cf. Eq. 3.6). As a result, the prediction for a user’s preference for item j is biased by the fake \mathbf{V}_j .

8.1.2 Optimizing Specialty and Credibility

Users’ choices for popular items are largely influenced by others. For example, seeing a new movie with friends may not really reflect a particular user’s subjective preference. On the other hand, choices for long-tail items can better reflect a user’s taste since they are rarely due to the influence of others. To tackle the popularity-bias challenge, we need to construct a model that can emphasize the choices of long-tail items in order to learn users’ special preferences. However, only emphasizing long-tail items is not enough because, as mentioned, long-tail items suffer more easily from shilling attack. Therefore, we also need to construct a model that can weigh the credibility of each piece of feedback. Moreover, an efficient way to deal with the cold-start challenge is to borrow information from other relevant users. We argue that high-quality, relevant users should be reputable and, thus, trusted. In summary, we need to design an approach that jointly models both the objective to emphasize the *Specialty* of choices and the objective to assess the *Credibility* of the feedback for each choice.

Heteroscedastic Matrix Factorization: In Eq. 3.3, the variance parameter, σ^2 , does not vary with different observations Y_{ij} , which is so-called “homoscedasticity”. Now, if we model each observation Y_{ij} with different variance, σ_{ij}^2 , as demonstrated in Eq. 8.1, then these observations are assumed to be “heteroscedastic”. By minimizing the negative log-form of Eq. 3.4, we immediately obtain the following objective (Eq. 8.2) where the weighted squared loss $w_{ij}(Y_{ij} - \mathbf{U}_i^\top \mathbf{V}_j)^2$ is the log-form of Eq. 8.1 and $w_{ij} = \sigma_{ij}^{-2}$ serves as the weight to penalize the loss of fitting Y_{ij} . As a result, we call this variant MF model heteroscedastic MF (HMF). From a probabilistic view, the variance parameter σ_{ij}^2 controls the confidence level [82]. Specifically,

a smaller σ_{ij}^2 implies higher confidence and less uncertainty of the observation Y_{ij} , i.e. a large w_{ij} is applied to more tightly fitting Y_{ij} .

$$P(Y_{ij}|\mathbf{U}_i, \mathbf{V}_j) = \mathcal{N}(Y_{ij}|\mathbf{U}_i^\top \mathbf{V}_j, \sigma_{ij}^2) \quad (8.1)$$

$$J = \underset{\mathbf{U}, \mathbf{V}}{\operatorname{argmin}} \frac{1}{2} \left[\underbrace{\sum_{ij} w_{ij} (Y_{ij} - \mathbf{U}_i^\top \mathbf{V}_j)^2}_{\text{weight loss}} + \lambda (\|\mathbf{U}\|_F^2 + \|\mathbf{V}\|_F^2) \right] \quad (8.2)$$

Heteroscedastic Modeling: In order to emphasize the specialty of users' choices with long-tail items, we differentiate each user choice in terms of heteroscedastic modeling. Specifically, we model the variance parameter σ_{ij}^2 by a variance function $f^S(\cdot)$ to score the *specialty* of this choice. As a result, we obtain specialty-specific heteroscedastic MF (S-HMF), which more tightly fits the users' choices for long-tail items.

On the other hand, we need to model the *credibility* of users' feedback. Technically, the parameters should be estimated by tightly fitting the more credible feedback while loosely fitting the less credible feedback. Hence, we can construct another HMF model using Eq. 8.1, where the variance parameter σ_{ij}^2 is modeled by a variance function $f^C(\cdot)$, which scores the *credibility* of each review. We name this type of HMF as credibility-specific HMF (C-HMF).

Coupling Objectives using Empirical Priors: So far, we have presented S-HMF and C-HMF, which correspond to two independent objectives for optimization. However, we need to jointly consider both objectives w.r.t. *specialty* and *credibility* when learning user preference as stated previously. From the view of Bayesian modeling, MAP trades off the estimation of parameters between prior and likelihood. That is, the estimates are very close to the given prior when little data is available. In PMF, the priors of user factors and item factors are modeled by uninformative priors, i.e. zero-means as illustrated by Eq. 3.2. If we provide some informative

priors, $\boldsymbol{\mu}_i$ and $\boldsymbol{\mu}_j$, pertaining to the user factors and the items factors as Eq. (10), then we obtain the objective function illustrated by Eq. (11). In this objective, we find that the estimates of \mathbf{U}_i and \mathbf{V}_j are regularized by the given priors $\boldsymbol{\mu}_i$ and $\boldsymbol{\mu}_j$. As a result, the estimates of user factors and items factors tend to shrink towards the given informative priors.

$$P(\mathbf{U}_i) = \mathcal{N}(\mathbf{U}_i | \boldsymbol{\mu}_U, \sigma_U^2 \mathbf{I}) \quad P(\mathbf{V}_j) = \mathcal{N}(\mathbf{V}_j | \boldsymbol{\mu}_V, \sigma_v^2 \mathbf{I}) \quad (8.3)$$

$$J = \operatorname{argmin}_{\mathbf{U}, \mathbf{V}} \frac{1}{2} \left[\underbrace{\sum_{ij} w_{ij} (Y_{ij} - \mathbf{U}_i^\top \mathbf{V}_j)^2}_{\text{weight loss}} + \underbrace{\lambda (\|\mathbf{U} - \boldsymbol{\mu}_i\|_F^2 + \|\mathbf{V} - \boldsymbol{\mu}_j\|_F^2)}_{\text{regularization}} \right] \quad (8.4)$$

The user factors and the item-factors learned from C-HMF target the objective of *credibility*, so they can serve as good empirical priors for modeling the user factors and the item factors of S-HMF. Symmetrically, the user factors and the item factors learned from S-HMF target the objective of *specialty*, which contains information on the users' intrinsic preferences. As a result, the user factors and the item factors learned from S-HMF are good empirical priors for modeling the user factors and the item factors of C-HMF. Thus, as shown in Figure 8.2, it forms a mutual regularization process that couples S-HMF and C-HMF using the empirical priors learned from each other recurrently. Therefore, we name this coupled model for multi-objective optimization the RMRM.

Moreover, we need to borrow information from other users' preferences to deal with cold-start users. Since user factors represent the features of user preference, we can construct the prior using user factors from a group of relevant users. Specifically, we design a sophisticated way to combine these Gaussian-distributed user factor vectors, namely, by using the product of Gaussian experts (PoGE) [77, 231]. In fact, such a PoGE-prior can be regarded as playing the role of social regularization [141]. This will be discussed further in the following sections.

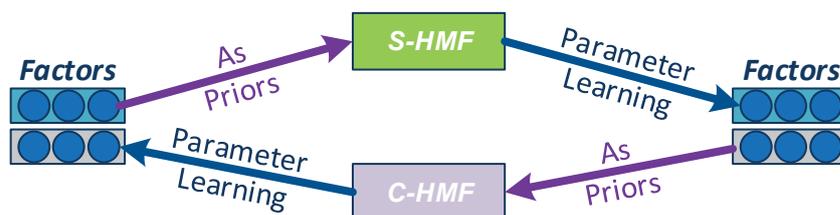


Figure 8.2 : A recurrent mutual regularization process couples S-HMF and C-HMF using the user and item-factors learned from one another as the empirical priors to couple the objectives *specialty* and *credibility*

8.2 Preliminaries

Before commencing a detailed discussion, we summarize the frequently used notations used in this chapter and their meanings in Table 8.1 to simplify the presentation in the rest of the chapter.

8.2.1 Reputation Modeling

One of the key components in our approach is modeling the reputation of users and the credibility of their feedback. Intuitively, the reviews from high-reputation users tend to be more trusted and helpful to other users, and thus, they more credibly discuss the real features of items. In other words, the reputation of a user is highly relevant to the credibility of her feedback. Bayesian reputation systems [105] have been proposed to model reputation from a probabilistic perspective, so this can be integrated easily into our framework. In particular, in this work, we employ the beta reputation model [103] to obtain the helpfulness scores for user reviews.

Table 8.1 : Summary of frequently used notations in this chapter

Symbol	Description
i	$i \in \{1, \dots, N\}$ is used to index a user
j	$j \in \{1, \dots, M\}$ is used to index an item
\mathbf{U}_i	$\mathbf{U} = [\mathbf{U}_1, \dots, \mathbf{U}_N]$ is the user factor matrix, where \mathbf{U}_i is the user factor vector of user i
\mathbf{V}_j	$\mathbf{V} = [\mathbf{V}_1, \dots, \mathbf{V}_M]$ is the item factor matrix, where \mathbf{V}_j is the item factor vector of item j
Y_{ij}	\mathbf{Y} is the data matrix, and $Y_{ij} \in \mathbf{Y}$ is an entry with the index (i, j)
\mathbf{O}_i	\mathbf{O} is the index set of all modeled data points, \mathbf{O}_i is the index set of data w.r.t. user i
\mathbf{O}_j	\mathbf{O}_i is the index set of data w.r.t. item j
σ_{ij}^2	is the variance parameter of observation \mathbf{Y}_{ij}
$\boldsymbol{\mu}_i$	$\boldsymbol{\mu}_i$ is the empirical prior placed on the user factors \mathbf{U}_i
$\boldsymbol{\mu}_j$	$\boldsymbol{\mu}_i$ is the empirical prior placed on the item factors \mathbf{V}_j
w_{ij}	\mathbf{W} is the weight matrix, and $w_{ij} \in \mathbf{W}$ is the weight to scale the loss of fitting Y_{ij}
ϕ_i	ϕ_i is the reputation score of user i
ω_{ij}	ω_{ij} is the credibility score on a user review
η_{ij}	η_{ij} is the specialty score on a user choice
\mathcal{N}_i^k	\mathcal{N}_i^k is the top-K neighbors of user i
\mathcal{N}^R	\mathcal{N}^R denotes the top-R high-reputation experts in the system
\cdot^S	Superscript to indicate S-HMF related model parameters
\cdot^C	Superscript to indicate C-HMF related model parameters
$\ x\ _2$	The 2-norm of a vector
$\text{diag}(x)$	Generate a diagonal matrix using a vector
PoGE	Product of Gaussian experts
\cdot^*	Element-wise product
\cdot^2	Element-wise square
\cdot^{-1}	Element-wise inverse

Beta Reputation Model

Let $e \stackrel{\text{def}}{=} \{r, s\}$ denote the evidence that contains r positive feedback and s negative feedback w.r.t. a target entity. The probability of evidence \mathbf{e} can be described by a group of Bernoulli events, which follows a binomial distribution:

$$P(\mathbf{e}|p) = \binom{r+s}{r} p^r (1-p)^s \quad (8.5)$$

Then, we can obtain the following definition of a reputation function given the beta-prior with the probability density function (pdf) defined by a gamma function $\Gamma(\cdot)$:

$$\text{Beta}(p|\alpha, \beta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} p^{\alpha-1} (1-p)^{\beta-1} \quad (8.6)$$

Definition 8.1: (Reputation Function) [103]

$$\varphi \stackrel{\text{def}}{=} \int P(\mathbf{e}|p) \text{Beta}(p|\alpha, \beta) dp = \text{Beta}(r + \alpha, s + \beta). \quad (8.7)$$

Obviously, the reputation function is defined as the posterior of the beta distribution, where the hyperparameters α and β can be thought of as a certain amount of pseudo positive and negative feedback; in practice, this is often set $\alpha = \beta = 1$. Next, we can obtain the expectation of this reputation function, i.e., the mean of $\text{Beta}(r + \alpha, s + \beta)$:

$$\varphi \stackrel{\text{def}}{=} \mathbb{E}[\varphi(\mathbf{e})] = \frac{r + \alpha}{r + \alpha + s + \beta} \quad (8.8)$$

We find that $\mathcal{R}(\mathbf{e})$ is bounded within (0,1), and that it approaches the upper bound 1 only if the user has a large amount of positive feedback. Clearly, we can employ such a beta reputation model to assess the reputation of a user by the score $\mathcal{R}(\mathbf{e})$ in RSs.

Data for Modeling Reputation

For most online shopping and review Web sites, a user's review of an item consists of a rating and a free message. In order to find helpful reviews and to display them

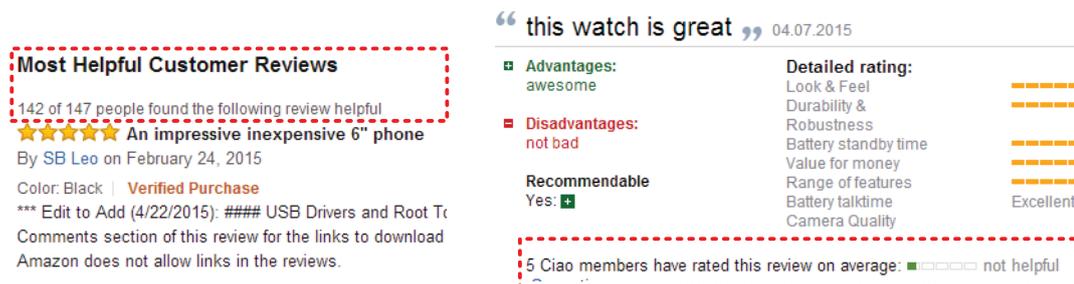


Figure 8.3 : The screen snapshots of reviews from Amazon.com (left) and Ciao.com (right), where the red, dotted boxes show the helpfulness score.

on the first page, some of the most well-known Web sites, e.g., Amazon.com^{*}, Epinions.com[†], Ciao.com[‡], have designed a scoring system to evaluate the helpfulness of each review.

Figure 8.3 demonstrates two screen snapshots of reviews found on Amazon.com and Ciao.com. These Web sites have integrated several algorithms to score the helpfulness of each review in terms of other users' feedback or experts' judgments of this review. As a result, each review is generally associated with a multilevel helpfulness score, from *Not Helpful* to *Most Helpful*, as shown in Figure 8.3. Obviously, we can employ the beta reputation model to assess the reputation of each user: if a user gives a lot of reviews that mostly receive high helpfulness scores, then this user tends to be a high-reputation user. On the contrary, the reputation score will be low if a spam user gives a lot of fake reviews. The mathematical representation of the reputation score will be discussed along with our model in the following subsections.

^{*}<http://www.amazon.com>

[†]<http://www.epinions.com>

[‡]<http://www.ciao.com>

8.2.2 Explicit and Implicit Rating

Rating data is typical feedback that represents the preferences of users. Typically, rating data can be divided into two categories: explicit and implicit.

Explicit Rating: The multilevel rating scores, e.g., five-star ratings, can explicitly differentiate user preferences, so they are typical explicit rating data. Therefore, we only model observed ratings, while the remaining entries are treated as missing. From the HMF view, we have no information on these missing entries to tell us whether users have liked the items or not, so we assign positive confidence, $c_{ij} > 0$, to the observed ratings and zeros to the remaining ones.

$$w_{ij} = \begin{cases} c_{ij} & (i, j) \text{ indexes an observation} \\ 0 & \text{otherwise} \end{cases} \quad (8.9)$$

If we set all $c_{ij} = 1$, then we obtain a binary-weight matrix \mathbf{w} [2,204]. Using this \mathbf{w} in Eq. 8.9, we can immediately obtain the traditional unweighted MF objective, as shown in Eq. 3.5, from the objective of HMF. In this case, the index set $\mathbf{O} = \{(i, j) | w_{ij} > 0\}$ only consists of observed entries.

Implicit Rating: In the real world, explicit ratings are not always provided by users, but implicit rating data, such as purchase records and number of clicks, can be obtained more easily. This implicit rating data is usually modeled as a unary preference because the blank entries do not necessarily indicate user dislike, but, instead, are a result of the users' lack of awareness [75]. Hence, we can assign a higher confidence level to observed entries and a much lower confidence level to blank entries [82,91]. Recall that the confidence level is associated with the variance parameter, i.e., the inverse of weight (cf. Eq. 8.9), when a Gaussian distribution is assumed. As a result, the weighting strategy [91,165] of implicit ratings is often

established as follows:

$$w_{ij} = \begin{cases} c_{ij} & (i, j) \text{ indexes an observation} \\ \epsilon & \text{otherwise} \end{cases} \quad (8.10)$$

where ϵ is a small constant to denote the low confidence representing users' likes or dislikes for blank entries while $c_{ij} > \epsilon$ denotes relatively higher confidence representing users' likes of observed entries. In this case, we need to model both likes and dislikes so the index set \mathbf{O} consists of all entries of the data matrix.

8.3 Model and Learning

8.3.1 Overview of RMRM

To implement more reliable recommendations for tail users and tail items, we propose to model two coupled objectives for joint optimization, namely, the *specialty* of user choices and the *credibility* of user feedback. To achieve this goal, we design a recurrent mutual regularization model (RMRM) to couple these two objectives together.

The Framework

As illustrated in Figure 8.4, the objective of *specialty* is modeled by S-HMF (the right model shown in Figure 4) while the objective of *credibility* is modeled by C-HMF (the left model shown in Figure 8.4). RMRM couples these two objective models in terms of the empirical priors induced from one another.

The C-HMF focuses on modeling the credibility of each user review. This is implemented using two means. First, C-HMF assigns different levels of confidence, i.e., variance, for each observation, Y_{ij} , where the variance is modeled by a variance function $f^C(\cdot)$, which is devised based on the Bayesian reputation model as presented in Section 3.2. As a result, the estimation of the item-factors is more dependent

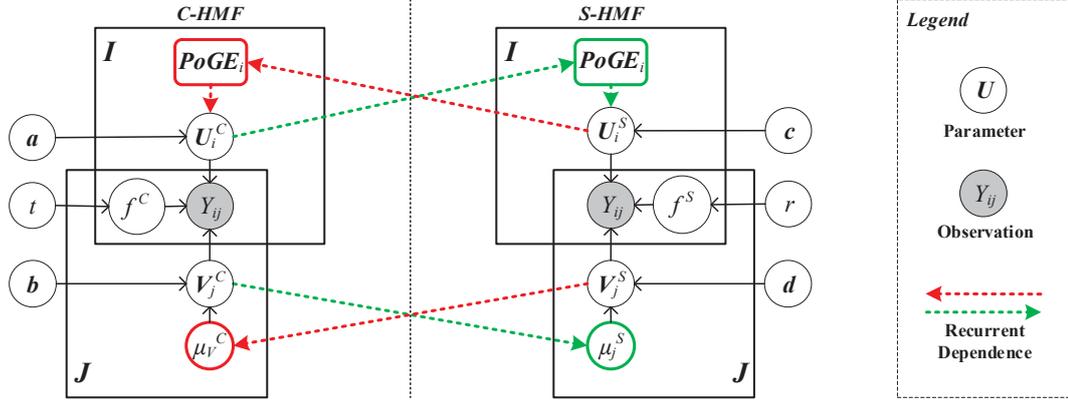


Figure 8.4 : The graphical representation of the RMRM framework, where S-HMF and C-HMF are recurrently regularized by the empirical priors, induced from one another.

on credible feedback. Second, a PoGE-prior is imposed on the user factors of each user, which plays the role of regularizing user behavior in terms of relevant, high-reputation experts. Here, such a PoGE-prior can regularize both the preference learning of cold-start users and the behavior of spam users. Therefore, the item factor vectors $\{\mathbf{V}_j^C\}$ learned from C-HMF represents more authentic features of items than those learned from classic MF models. At the same time, the user factor vectors $\{\mathbf{U}_i^C\}$ of tail users contain knowledge from relevant experts.

The S-HMF focuses on emphasizing the specialty of choices. The choices of tail items are much less influenced by others, thus they better reflect personal preferences. In S-HMF, the variance function $f^S(\cdot)$ assigns greater confidence to the choice of items in a deeper tail. As a result, S-HMF tends to fit the observations of tail items more tightly than those of head items. Therefore, the user factor vectors $\{\mathbf{U}_i^S\}$ that are estimated from C-HMF can better reflect users' personal preferences than those learned using classic MF methods.

As discussed in previous sections, long-tail items and users with little data are more easily affected by shilling attack and cold-start issues, which leads to unreliable estimates $\{\mathbf{V}_j^S\}$ and $\{\mathbf{U}_i^S\}$ learned from S-HMF. According to Bayesian probabilistic modeling, a prior plays an important role when there is limited data. Therefore, $\{\mathbf{V}_j^C\}$ and $\{\mathbf{U}_i^C\}$ learned from the credibility-oriented objective model, i.e., C-HMF, are good empirical priors to regularize S-HMF to relieve both shilling attack and cold start. In turn, $\{\mathbf{U}_i^S\}$ learned from S-HMF are refined user features so they can serve as the empirical priors for C-HMF in order to deal with popularity bias. Therefore, we designed a RMRM framework that consists of the recurrent dependencies between C-HMF and S-HMF to handle these challenges.

Geometric Illustration

To give an intuitive understanding of the working mechanism of RMRM, we provide the geometric illustration depicted in Figure 8.5. Here, we demonstrate the data fitting process of RMRM from the perspective of a given user, as well as a similar process that can be conducted from an item perspective. The axes arrange items (denoted as small circles) according to their popularity. More specifically, those items where the user provides credible feedback are marked with solid circles whereas hollow circles denote items receiving less credible feedback. The colored lines indicate fitting curves; the closer the curve is to a circle indicates the tighter the parameters to fit the choice of the corresponding item.

Figure 8.5(a) depicts S-HMF, which is regularized by the parameters learned from C-HMF. Given a user i , the top fitting curve of Figure 8.5(a) reflects the parameters learned from C-HMF, which tend to tightly fit user i 's choices with credible feedback whereas they are loosely fit with others without credible feedback. The middle fitting curve of Figure 8.5(a) reflects the parameters learned directly by maximizing the heteroscedastic likelihood of the choices of user i , i.e., minimizing the

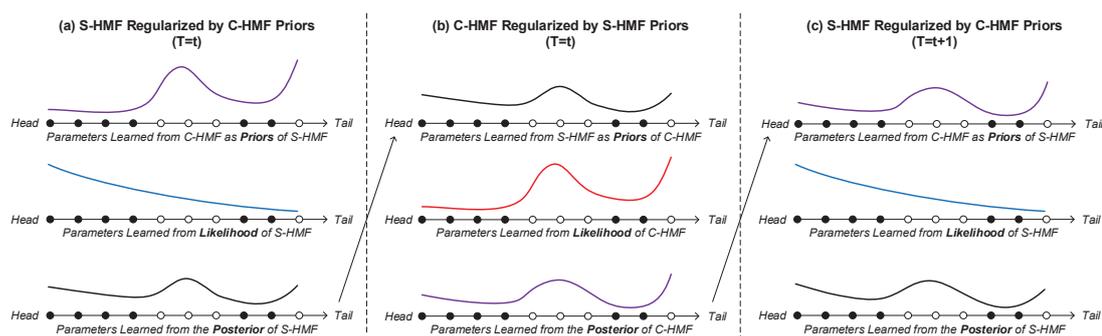


Figure 8.5 : The geometric illustration of the recurrent mutual regularization process, where the estimates of S-HMF and C-HMF are recurrently regularized by the empirical priors induced from one another.

weighted loss of fitting user i 's choices (cf. Eq. 8.9). When the parameters learned from C-HMF are employed as empirical priors for S-HMF, we obtain a regularized S-HMF model, as depicted in the right part of RMRM. The parameters can be estimated by maximizing the posterior, i.e., minimizing the objective given in Eq. 8.9), where the regularization term brings estimates closer to the given priors. As a result, the bottom fitting curve of Figure 8.5(a) represents the regularization results of the parameters, which more aggressively fit those choices in the tail with a high degree of credibility.

In turn, the parameters learned from this regularized S-HMF serve as the empirical priors to regularize C-HMF, as shown in the top curve of Figure 8.5(b). The parameters learned directly by maximizing the heteroscedastic likelihood of C-HMF tend to fit more tightly with the choices with credible feedback, as shown by the middle curve of Figure 8.5(b). When the empirical priors are imposed for regularization, the parameters learned from the posterior of C-HMF contain the information of the specialty of choices from the priors. Therefore, the bottom fitting curve shown in Figure 8.5(b) tends to fit more tightly the choices of tail items than the one produced by the maximum heteroscedastic likelihood estimation.

Now, let us move to the next iteration, as shown in Figure 8.5(c). As in the previous iteration, the parameters learned from the regularized C-HMF serve as the empirical priors to regularize S-HMF. As a result, the coupled recurrent regularizing process of RMRM converges the parameters to the region that represents the specialty of user choices and, simultaneously, enhances its credibility.

8.3.2 Learning Regularized C-HMF Model

Given observations $\{Y_{ij} | (i, j) \in \mathbf{O}\}$, we can obtain the probabilistic model according to the graphical representation of C-HMF, as shown in the left part of Figure 8.4:

$$P(\mathbf{U}_i^C) = \text{PoGE}(\mathbf{U}_i^C | \{\boldsymbol{\mu}_n^C, \varphi_n\}_{n \in \mathbf{T}_i}) \quad (8.11)$$

$$P(\mathbf{V}_j^C) = \mathcal{N}(\mathbf{V}_j^C | \boldsymbol{\mu}_j^C, \text{diag}[\mathbf{b}]) \quad (8.12)$$

$$P(Y_{ij} | \mathbf{U}_i^C, \mathbf{V}_j^C) = \mathcal{N}(Y_{ij} | \mathbf{U}_i^{C\top} \mathbf{V}_j^C, t\omega_{ij}^{-1}) \quad (8.13)$$

where $\boldsymbol{\mu}_n^C$ and $\boldsymbol{\mu}_j^C$ are the user and the item factor vectors induced from, $\boldsymbol{\mu}_n^S$ and $\boldsymbol{\mu}_j^S$, i.e. the counterparts in S-HMF. The details of using PoGE to construct the prior on \mathbf{U}_i^C will be discussed later in this section. $\text{diag}[\mathbf{b}]$ stands for diagonal variance matrices of Gaussian priors. $t\omega_{ij}^{-1}$ as a whole denotes the variance of likelihood, where ω_{ij} is a confidence score obtained by heteroscedastic modeling w.r.t. credibility of feedback and t is a scale parameter to be learned.

Heteroscedastic Modeling on Credibility

The key component of C-HMF is to model the credibility of feedback on the items that a user has chosen. Intuitively, users with higher reputations tend to give more credible feedback. Therefore, we can employ the reputation model presented in Section 8.2.1 to access the reputation of each user.

Reputation Modeling: As demonstrated in Section 8.2.1, each review of a particular item is associated with a helpfulness score. Typically, a five-level score

set, e.g., $\mathbf{h}=\{\text{Not Helpful}, \text{Somewhat Helpful}, \text{Helpful}, \text{Very Helpful}, \text{Most Helpful}\}$, is often applied to measure the helpfulness of a review. Here, we extend the beta reputation model (cf. Section 8.2.1) to assess the reputation of each user. Intuitively, if a user gives a lot of reviews that mostly receive high helpfulness scores, then this user tends to be a high-reputation user. As the helpfulness scores for user reviews are not binary feedback, i.e., positive or negative, as presented in Section 8.2.1, they cannot directly serve as evidence. However, five-level helpfulness scores are very suitable to be represented as a typical fuzzy set [104]. First, we can assign values $\mathbf{h}=0,1,2,3,4$ to the corresponding five-level helpfulness scores. Then, the membership functions of helpful (+) and unhelpful (-) can be given as follows:

$$\begin{cases} \mu_+(h) = \frac{h + \alpha}{h_{MAX} + \alpha} \\ \mu_-(h) = 1 - \mu_+(h) \end{cases} \quad (8.14)$$

where h_{MAX} is the maximum score in \mathbf{h} , e.g., $h_{MAX} = 4$ in the five-level score set above, and $\alpha \geq 0$ is a smooth parameter that bounds the degree of membership in $[\alpha/(h_{MAX} + \alpha), 1]$, e.g., the Not Helpful score has the smallest helpfulness $\mu_+(0)$ for a review. Then, we represent the evidence \mathbf{e}_i for user i through all their helpfulness scores \mathbf{h}_i as follows:

$$\mathbf{e}_i \stackrel{\text{def}}{=} \{\langle \mu_+(h_n), \mu_-(h_n) \rangle | h_n \in \mathbf{h}_i\} \quad (8.15)$$

As a result, we can still use Eq. 8.5 to denote the probability of evidence \mathbf{e}_i , where we have r positive feedback where $r = \sum \mu_+(h_n)$ and s negative feedback where $s = \sum \mu_-(h_n)$. Then, we define the reputation score of a user based on Eq. 8.6):

Definition 8.2: (Reputation Score) Given the helpfulness scores \mathbf{h}_i of a user i , the reputation score of this user is defined by:

$$\varphi = \mathcal{R}(\mathbf{e}_i | \mathbf{h}_i) \stackrel{\text{def}}{=} \frac{r + \alpha}{r + s + \alpha + \beta} \quad (8.16)$$

In practice, we can set $\beta > \alpha$ *a priori*. That is, we assign a relatively low score, $\phi_i = \alpha/(\alpha + \beta) < 0.5$, to a new user without any observed helpful ratings, because

spam users often create a new account when conducting an attack to avoid being tracked by the system. Obviously, ϕ_i arrives at the upper bound only if a user receives a lot of high helpfulness scores for their reviews. This implies that high-reputation users are also experienced users. On the contrary, ϕ_i becomes lower if a user always gives false reviews.

Credibility Scoring: We assign the feedback credibility for an item choice in terms of two scores: the reputation of a user (a global score), and the helpfulness of the review (a local score). Thus, we obtain the following:

$$\omega_{ij} \stackrel{\text{def}}{=} \begin{cases} \varphi_i \mu_+(h_{ij}) + \epsilon & (i, j) \text{ is an observed entry} \\ \epsilon & \text{otherwise} \end{cases} \quad (8.17)$$

That is, the observation is associated with a high credibility score only if a high-reputation user gives a helpful review. In particular, we set $\epsilon = 0$ for explicit rating data while $\epsilon =$ is set to a small constant for implicit rating data (cf. Section 8.2.2). Since a higher credibility score means a higher confidence of that item choice, the variance function of a feedback can be given by $f^C(Y_{ij}) = t\omega_{ij}^{-1}$ (recall that lower variance means higher confidence), where t is a scale parameter to learn.

PoGE-Prior

In particular, we use PoGE to construct the prior for each user in order to incorporate the knowledge of a set of experts indexed by \mathbf{T}_i .

$$PoGE(\mathbf{U}_i^C | \{\boldsymbol{\mu}_n^C, \varphi_n\}_{n \in \mathbf{T}_i}) = \prod_{n \in \mathbf{T}_i} \mathcal{N}(\mathbf{U}_i^C | \boldsymbol{\mu}_n^C, \text{diag}[\hat{w}_n^{-1} \boldsymbol{\alpha}]) \quad (8.18)$$

where w_n is a weight parameter. In general, PoE (Product of Experts) [77] has an intractable form. Fortunately, the product of Gaussian densities has a closed form, that is, a new Gaussian density [231]. Therefore, we can obtain the following Gaussian distribution from Eq. 8.18:

$$PoGE(\mathbf{U}_i^C | \{\boldsymbol{\mu}_n^C, \varphi_n\}_{n \in \mathbf{T}_i}) = \mathcal{N}(\mathbf{U}_i^C | \hat{\boldsymbol{\mu}}_i^C, \text{diag}[\hat{w}_n^{-1} \boldsymbol{\alpha}]) \quad (8.19)$$

where $\hat{\boldsymbol{\mu}}_i^C = \frac{\sum_{n \in \mathbf{T}_i} w_n \boldsymbol{\mu}_n^C}{\hat{w}_i}$ with $\hat{w}_i = \sum_{n \in \mathbf{T}_i} w_n$. Obviously, the mean parameter $\boldsymbol{\mu}_i^C$ of the PoGE distribution is a weighted average of the user factor vectors of all related experts.

In this chapter, we construct the related expert set as follows:

$$\mathbf{T}_i = i \cup \mathcal{N}_i^K \cup \mathcal{N}^R \cup \mu_0 \quad (8.20)$$

In Eq. 8.20, i stands for the target user itself. \mathcal{N}_i^K is the top-K neighbors of user i ; the neighbors could be a set of users with an explicit relationship with i , e.g., trusters [142] or followers [239]; they can also be constructed from the data [116] if no explicit relation is available. \mathcal{N}^R denotes the top-R high-reputation experts in the system. Moreover, μ_0 is an optional expert with a zero-mean Gaussian prior to avoid overfitting. As illustrated in Eq. 8.11, we use the reputation score φ_n (cf. Eq. 8.16) of a user as the weight w_n of an expert in PoGE (cf. Eq. 8.18). By taking the log-form of PoGE over this expert set \mathbf{T}_i , we easily obtain the following summation form:

$$\begin{aligned} & \log \text{PoGE}(\mathbf{U}_i^C | \{\boldsymbol{\mu}_n^C, \varphi\}_{n \in \mathbf{T}_i}) \quad (8.21) \\ &= \log \mathcal{N}(\mathbf{U}_i^C | \boldsymbol{\mu}_i^C, \text{diag}[\varphi_n^{-1} \boldsymbol{\alpha}]) + \sum_{k \in \mathcal{N}_i^K} \log \mathcal{N}(\mathbf{U}_i^C | \boldsymbol{\mu}_i^C, \text{diag}[\varphi_n^{-1} \boldsymbol{\alpha}]) \\ &+ \sum_{r \in \mathcal{N}^R} \log \mathcal{N}(\mathbf{U}_i^C | \boldsymbol{\mu}_r^C, \text{diag}[\varphi_r^{-1} \boldsymbol{\alpha}]) + \log \mathcal{N}(\mathbf{U}_i^C | \mathbf{0}, \text{diag}[\varphi_0^{-1} \boldsymbol{\alpha}]) \\ &= \underbrace{\varphi_i \|\boldsymbol{\alpha}^{-1} \cdot (\mathbf{U}_i^C - \boldsymbol{\mu}_i^C)\|_2^2}_{\text{self-based regularization}} + \underbrace{\sum_{k \in \mathcal{N}_i^K} \varphi_k \|\boldsymbol{\alpha}^{-1} \cdot (\mathbf{U}_i^C - \boldsymbol{\mu}_r^C)\|_2^2}_{\text{neighbor-based regularization}} \\ &+ \underbrace{\sum_{r \in \mathcal{N}^R} \varphi_r \|\boldsymbol{\alpha}^{-1} \cdot (\mathbf{U}_i^C - \boldsymbol{\mu}_r^C)\|_2^2}_{\text{expert-based regularization}} + \underbrace{\varphi_0 \|\boldsymbol{\alpha}^{-1} \cdot \mathbf{U}_i^C\|_2^2}_{\text{complexity-regularization}} + \underbrace{T(\cdot)}_{\text{self-based regularization}} \end{aligned}$$

From the above equation, we find that \mathbf{U}_i^C is respectively regularized by four types of experts as specified in \mathbf{T}_i . In the first term, $\boldsymbol{\mu}_i^C$ is the user factor vector of the target user itself so it serves for self-based regularization. Since the majority of users are tail users with limited data, it is useful to borrow information from their neighbors.

As a result, the user factor vectors $\{\boldsymbol{\mu}_k^C\}$ from i 's neighbors \mathcal{N}_i^K are employed for *neighbor-based regularization*. Moreover, we involve a set of high-reputation experts \mathcal{N}^R in the system to conduct *expert-based regularization* because effective *self-based regularization* and *neighbor-based regularization* are often not available, e.g., a fully cold-start user who has no data available and no neighbors or a spam user who only links other spam users as his neighbors. The last regularization term is simply the most frequently used L_2 -norm regularizer when α is set 1, which penalizes the complexity to prevent overfitting. Due to the equivalence between Eq. 8.18 and Eq. 8.19, Eq. 8.21 can be reformed to Eq. 8.22:

$$\begin{aligned} & \log PoGE(\mathbf{U}_i^C | \{\boldsymbol{\mu}_n^C, \varphi\}_{n \in \mathbf{T}_i}) \\ &= \log \mathcal{N}(\mathbf{U}_i^C | \hat{\boldsymbol{\mu}}_i^C, \text{diag}[\varphi_i^{-1} \boldsymbol{\alpha}]) = \hat{\varphi}_i \|\boldsymbol{\alpha}^{-1} \cdot (\mathbf{U}_i^C - \hat{\boldsymbol{\mu}}_i^C)\|_2^2 + T(\cdot) \end{aligned} \quad (8.22)$$

where $\hat{\boldsymbol{\mu}}_i^C = \frac{\sum_{n \in \mathbf{T}_i} \varphi_n \boldsymbol{\mu}_n^C}{\hat{\varphi}_i}$ with $\hat{\varphi}_i = \sum_{n \in \mathbf{T}_i} \varphi_n$. From the perspective of Eq. 8.21, φ_n controls the penalty of loss for fitting $\boldsymbol{\mu}_n^C$, i.e. a higher reputation expert has a larger regularization effect. From the perspective of Eq. 8.22, the empirical prior mean $\hat{\boldsymbol{\mu}}_i^C$ is a weighted average user factor vector over \mathbf{T}_i , so $\hat{\boldsymbol{\mu}}_i^C$ receives more contributions from higher reputation experts with a larger φ_n . Note that φ_0 is not a reputation score but a common regularization parameter as λ in Eq. 8.2, and it can be determined by usual regularization parameter selection methods, such as cross-validation.

Parameter Learning

We can obtain the marginal log-likelihood by integrating $\mathbf{U}^C, \mathbf{V}^C$ from the joint distribution:

$$\log P(\mathbf{Y}) = \log \int P(\mathbf{Y}, \mathbf{U}^C, \mathbf{V}^C) d\mathbf{U}^C d\mathbf{V}^C \quad (8.23)$$

where $P(\mathbf{Y}, \mathbf{U}^C, \mathbf{V}^C) = \prod_{ij \in \mathbf{O}} P(Y_{ij} | \mathbf{U}_i^C, \mathbf{V}_j^C) \prod_i P(\mathbf{U}_i^C) \prod_j P(\mathbf{V}_j^C)$.

However, the computation of Eq. 8.23 is generally intractable. To enable it to run

efficiently on large-scale data and the precise learning parameters for our model, we use the variational Bayesian (VB) method, which provides a good balance between efficiency and accuracy in learning latent features [109, 133, 196]. Now, if we let $Q(\mathbf{U}^C, \mathbf{V}^C)$ be the variational distribution, we can then obtain the lower bound by applying Jensen's inequality [196].

$$\log P(\mathbf{Y}) \geq \int Q(\mathbf{U}^C, \mathbf{V}^C) \log \frac{P(\mathbf{Y}, \mathbf{U}^C, \mathbf{V}^C)}{Q(\mathbf{U}^C, \mathbf{V}^C)} d\mathbf{U}^C d\mathbf{V}^C = \mathcal{L}(Q) \quad (8.24)$$

The lower bound $\mathcal{L}(Q)$ can be rewritten using the expectation conditional on $Q(\mathbf{U}^C, \mathbf{V}^C)$ from Eq. 8.24, and it becomes tight only when $Q(\mathbf{U}^C, \mathbf{V}^C) = P(\mathbf{U}^C, \mathbf{V}^C | \mathbf{Y})$.

$$\mathcal{L}(Q) = \mathbb{E}_Q[\log P(\mathbf{U}^C, \mathbf{V}^C | \mathbf{Y}) + \log P(\mathbf{U}^C) + \log P(\mathbf{V}^C)] + H[Q(\mathbf{U}^C, \mathbf{V}^C)] \quad (8.25)$$

Generally, it usually assumes $Q(\mathbf{U}^C, \mathbf{V}^C)$ has a factorial form [109, 133, 196]:

$$Q(\mathbf{U}^C, \mathbf{V}^C) = Q(\mathbf{U}^C)Q(\mathbf{V}^C) = \prod_i Q(\mathbf{U}_i^C) \prod_j Q(\mathbf{V}_j^C) \quad (8.26)$$

Here, $Q(\mathbf{U}_i^C)$ and $Q(\mathbf{V}_j^C)$ are variational Gaussian distributions with diagonal variance matrices:

$$Q(\mathbf{U}_i^C) = \mathcal{N}(\mathbf{U}_i^C | \mathbf{u}_i^C, \text{diag}[\boldsymbol{\lambda}_i^C]) \quad Q(\mathbf{V}_j^C) = \mathcal{N}(\mathbf{V}_j^C | \mathbf{v}_j^C, \text{diag}[\boldsymbol{\gamma}_j^C]) \quad (8.27)$$

Then, we can write Eq. 8.25 as the following form by using the Eq. 8.11, 8.12, 8.13

and 8.26:

$$\begin{aligned}
\mathcal{L}(Q) &= \sum_{ij \in \mathbf{O}} \mathbb{E}_{Q(\mathbf{U}_i^C)Q(\mathbf{V}_j^C)} [\log P(Y_{ij} | \mathbf{U}_i^C, \mathbf{V}_j^C)] & (8.28) \\
&+ \sum_i \mathbb{E}_{Q(\mathbf{U}_i^C)} \log P(\mathbf{U}_i^C) + H[Q(\mathbf{U}_i^C)] + \sum_j \mathbb{E}_{Q(\mathbf{V}_j^C)} \log P(\mathbf{V}_j^C) + H[Q(\mathbf{V}_j^C)] \\
&= \sum_{ij \in \mathbf{O}} \mathbb{E}_{Q(\mathbf{U}_i^C)Q(\mathbf{V}_j^C)} [\log \mathcal{N}(Y_{ij} | \mathbf{U}_i^{C\top} \mathbf{V}_j^C, t\omega_{ij}^{-1})] \\
&+ \sum_i \mathbb{E}_{Q(\mathbf{U}_i^C)} \log \mathcal{N}(\mathbf{U}_i^C | \boldsymbol{\mu}_i^C, \text{diag}[\hat{\omega}_i^{-1} \mathbf{a}]) + H[\mathcal{N}(\mathbf{U}_i^C | \mathbf{u}_i^C, \text{diag}[\boldsymbol{\lambda}_i^C])] \\
&+ \sum_j \mathbb{E}_{Q(\mathbf{V}_j^C)} \log \mathcal{N}(\mathbf{V}_j^C | \boldsymbol{\mu}_j^C, \text{diag}[\mathbf{b}]) + H[\mathcal{N}(\mathbf{V}_j^C | \mathbf{v}_j^C, \text{diag}[\boldsymbol{\gamma}_j^C])] \\
&= \frac{1}{2} \left(\sum_{ij \in \mathbf{O}} \log 2\pi t^{-1} \omega_{ij} - t^{-1} \omega_{ij} [(\mathbf{Y}_{i,j} - \mathbf{u}_i^{C\top} \mathbf{v}_j^C)^2 \right. \\
&\quad \left. + (\mathbf{u}_i^C)^{\top} \boldsymbol{\gamma}_j^C + \boldsymbol{\lambda}_i^{C\top} (\mathbf{v}_j^C)^2 + \boldsymbol{\gamma}_j^C \right) \\
&- \frac{1}{2} \left(\sum_i \log \|2\pi \hat{\varphi}_i \mathbf{a}^{-1}\|_2 + [(\mathbf{u}_i^C - \boldsymbol{\mu}_i^C)^2 + \boldsymbol{\lambda}_i^C]^{\top} \hat{\varphi}_i \mathbf{a}^{-1} - \log \|2\pi e \boldsymbol{\lambda}_i^C\|_2 \right) \\
&- \frac{1}{2} \left(\sum_j \log \|2\pi \mathbf{b}^{-1}\|_2 + [(\mathbf{v}_j^C - \boldsymbol{\mu}_j^C)^2 + \boldsymbol{\gamma}_j^C]^{\top} \mathbf{b}^{-1} - \log \|2\pi e \boldsymbol{\gamma}_j^C\|_2 \right)
\end{aligned}$$

Let us denote $\{\bar{\mathbf{U}}^C, \boldsymbol{\Lambda}^C, \bar{\mathbf{V}}^C, \boldsymbol{\Gamma}^C\}$ as the variational parameters and $\{\mathbf{a}, \mathbf{b}, t\}$ as the model parameters where $\bar{\mathbf{U}}^C = [\mathbf{u}_i^C]_{1 \leq i \leq N}$ stands for a matrix consisting of mean vectors and $\boldsymbol{\Lambda}^C = [\boldsymbol{\lambda}_i^C]_{1 \leq i \leq N}$ denotes a matrix consisting of the variance vectors of $Q(\mathbf{U}^C)$, and $\bar{\mathbf{V}}^C, \boldsymbol{\Gamma}^C$ are defined similarly w.r.t. $Q(\mathbf{V}^C)$. To maximize $\mathcal{L}(Q)$, we can use coordinate ascend, i.e. iteratively optimizing $\mathcal{L}(Q)$ by searching for a solution for one parameter at a time and fixing the others. Table 8.2 summarizes the updating scheme for each parameter.

The Tricks for Complexity Reduction

The data matrix \mathbf{Y} and its corresponding weight matrix $\mathbf{W} = t^{-1}\omega$ are very sparse as they pertain to explicit rating data, where non-zero entries in these two

Table 8.2 : Parameter updating scheme for C-HMF

In the following equations, we denote $W_{ij} = t^{-1}\omega_{ij}$

Update parameters $\{\mathbf{u}_i^C, \boldsymbol{\lambda}_i^C\}$ of distribution $Q(\mathbf{U}_i^C)$ in parallel, for each i :

$$\mathbf{u}_i^C \leftarrow \boldsymbol{\Psi}_i \left[\bar{\mathbf{V}}^C \text{diag}(\mathbf{W}_{i,:}) \mathbf{Y}_{i,:}^\top + \hat{\varphi}_i \mathbf{a} \cdot^{-1} \cdot^* \boldsymbol{\mu}_i^C \right] \quad (8.29)$$

where $\boldsymbol{\Psi}_i^{-1} = \text{diag}(\hat{\varphi}_i \mathbf{a} \cdot^{-1}) + \bar{\mathbf{V}}^C \text{diag}(\mathbf{W}_{i,:}) \bar{\mathbf{V}}^{C\top} + \text{diag}(\boldsymbol{\Gamma}^C \mathbf{W}_{i,:}^\top)$

$$\boldsymbol{\lambda}_i^C \leftarrow (\bar{\mathbf{V}}^C \cdot^2 + \boldsymbol{\Gamma}^C) \mathbf{W}_{i,:}^\top + \hat{\varphi}_i \cdot^{-1} \mathbf{a} \quad (8.30)$$

Update parameters $\{\mathbf{v}_j^C, \boldsymbol{\gamma}_j^C\}$ of the distribution $Q(\mathbf{V}_j^C)$ in parallel, for each j :

$$\mathbf{v}_j^C \leftarrow \boldsymbol{\Psi}_j \left(\bar{\mathbf{U}}^C \text{diag}(\mathbf{W}_{:,j}) \mathbf{Y}_{:,j} + \mathbf{b} \cdot^{-1} \cdot^* \boldsymbol{\mu}_j^C \right) \quad (8.31)$$

where $\boldsymbol{\Psi}_j^{-1} = \text{diag}(\mathbf{b} \cdot^{-1}) + \bar{\mathbf{U}}^C \text{diag}(\mathbf{W}_{:,j}) \mathbf{U}^{C\top} + \text{diag}(\boldsymbol{\Lambda}^C \mathbf{W}_{:,j})$

$$\boldsymbol{\gamma}_j^C \leftarrow (\bar{\mathbf{U}}^C \cdot^2 + \boldsymbol{\Lambda}^C) \mathbf{W}_{:,j}^\top + \mathbf{b} \quad (8.32)$$

Update model parameters $\{\mathbf{a}, \mathbf{b}, t\}$:

$$\mathbf{a} \leftarrow \frac{\sum_i \hat{\varphi}_i [(\mathbf{u}_i^C - \boldsymbol{\mu}_i^C) \cdot^2 + \boldsymbol{\lambda}_i^C]}{N} \quad (8.33)$$

$$\mathbf{b} \leftarrow \frac{\sum_j [(\mathbf{v}_j^C - \boldsymbol{\mu}_j^C) \cdot^2 + \boldsymbol{\gamma}_j^C]}{M} \quad (8.34)$$

$$t \leftarrow \frac{\sum_{ij \in \mathbf{O}} \omega_{ij} [(\mathbf{Y}_{i,j} - \mathbf{u}_i^{C\top} \mathbf{v}_j^C) \cdot^2 + (\mathbf{u}_i^C \cdot^2)^\top \boldsymbol{\gamma}_j^C + (\mathbf{v}_j^C + \boldsymbol{\lambda}_i^C)^\top \boldsymbol{\lambda}_i^C]}{|\mathbf{O}|} \quad (8.35)$$

matrices are associated with observed ratings. Due to the factorial variational distribution $Q(\mathbf{U}^C, \mathbf{V}^C)$, the parameter updating scheme of Table 8.2 is naturally parallelizable. The updating scheme in Table 8.2 can be implemented in the same way as that used by Kim and Choi [109] who designed a scalable parameter updating scheme for variational Bayesian MF. Accordingly, the time complexity is $O(3K[\sum_i |\mathbf{W}_{i,:}^{\geq 0}| + \sum_j |\mathbf{W}_{:,j}^{\geq 0}|]) = O(6K|\mathbf{O}|)$ as illustrated in [109], where $|\mathbf{W}_{i,:}^{\geq 0}|$ equals the number of observed ratings for user i , $|\mathbf{W}_{:,j}^{\geq 0}|$ equals the number of observed ratings for item j , and $|\mathbf{O}|$ is the total number of observed ratings. In practice, the length of the latent factor vector, K , is small, and in our experiments, it yields good results for $K \leq 10$. Normally, the data density, $s = |\mathbf{O}|/NM$, of most explicit rating data sets is very small, i.e. large sparsity, in the real world, usually $s \leq 0.01\%$ (e.g., the RED dataset). Therefore, this updating scheme is executed very efficiently.

In the case of implicit rating data, the blank entries in data matrix \mathbf{Y} are also modeled as implicit feedback [82,91]. Accordingly, in this case, the weight matrix \mathbf{W} is a full matrix, i.e. $|\mathbf{O}| = |\mathbf{W}^{\geq 0}|$, having the space complexity $O(NM)$ (cf. Eq. 8.10 and Eq. 8.17). Normally, it is impractical to load such a full matrix \mathbf{W} into memory. From the analysis above, the time for running this updating scheme on implicit rating data is $1/s$ (i.e., often more than 10,000) times slower than running it on explicit rating data. To improve the running performance on the implicit rating data, we can apply the following trick to reduce the complexity. If we let $\tilde{\mathbf{W}}_{i,:} = \mathbf{W}_{i,:} - c$ and $c = t^{-1}\epsilon$, we write each row of \mathbf{W} as $\mathbf{W}_{i,:} = \tilde{\mathbf{W}}_{i,:} + c$. According to Eq. 8.17, it is easy to see that $\mathbf{W}_{i,:}$ only has non-zero entries on observed ratings. Now, let us take updating for $\{\mathbf{u}_i^C, \lambda_i^C\}$ as an example. In Eq. 8.29, $\bar{\mathbf{V}}^C \text{diag}(\mathbf{W}_{i,:}) \bar{\mathbf{V}}^{C\top}$ can be rewritten as $\bar{\mathbf{V}}^C \text{diag}(\tilde{\mathbf{W}}_{i,:}) \bar{\mathbf{V}}^{C\top} + c \bar{\mathbf{V}}^C \bar{\mathbf{V}}^{C\top}$; obviously, the term $c \bar{\mathbf{V}}^C \bar{\mathbf{V}}^{C\top}$ is not dependent on the user index i , so it can be pre-computed in time at most $O(K^2M)$ and less than $O(KM)$ using parallel multiplication. Similarly, $\bar{\mathbf{V}}^C \text{diag}(\mathbf{W}_{i,:}) \bar{\mathbf{Y}}_{i,:}^\top$

can be written as $\bar{\mathbf{V}}^C \text{diag}(\tilde{\mathbf{W}}_{i,:}) \mathbf{Y}_{i,:}^\top + c \bar{\mathbf{V}}^C \mathbf{Y}_{i,:}^\top$, where the term $c \bar{\mathbf{V}}^C \mathbf{Y}_{i,:}^\top$ can be computed in time less than $O(KM)$ due to the sparse $\mathbf{Y}_{i,:}$. Moreover, $\mathbf{\Gamma}^C \mathbf{W}_{i,:})^\top$ can be written as $\mathbf{\Gamma}^C \tilde{\mathbf{W}}_{i,:}^\top)^\top + c \mathbf{\Gamma}^C \mathbf{1}$, where the term $c \mathbf{\Gamma}^C \mathbf{1}$ can be computed in less time than $O(KM)$ because $\mathbf{\Gamma}^C \mathbf{1}$ is equivalent to summing $\mathbf{\Gamma}^C$ by rows. Using the same trick, the additional time in parallel computing $\{\lambda_i^C\}$ is also $O(KM)$. Therefore, the overall additional time cost is $O(4KM)$ when learning $\{\mathbf{u}_i^C, \lambda_i^C\}$ in this parallel fashion. When applying this trick to updating $\{\mathbf{v}_j^C, \gamma_j^C\}$, the overall additional time cost is $O(4KN)$. Moreover, we can compute t by summing over i in a parallel way, where the additional time cost is also $O(KM)$ since $\omega_{i,:} = t \mathbf{W}_{i,:}$. As a result, the overall additional time cost for implicit rating data is $O(K[M + N])$, so the whole time complexity is $O(6K|\tilde{W}^{\geq 0}|) + O(4KM) + O(4KN) + O(K[M + N]) = O(6K|\tilde{W}^{\geq 0}| + 5K[M + N]) < O(6K[|\tilde{W}^{\geq 0}| + M + N])$. Normally, $M + N \ll |\tilde{W}^{\geq 0}|$, so $O(6K[|\tilde{W}^{\geq 0}| + M + N])$ is within the same order as $O(6K|\tilde{W}^{\geq 0}|)$.

By applying this trick, updating the equations depends on the sparse weight matrix $\tilde{\mathbf{W}}$ instead of the full matrix \mathbf{W} , so the space complexity to store $\tilde{\mathbf{W}}$ is the same as the sparse weight matrix in the case of explicit rating data. Therefore, it can be concluded that the time and space complexities of the learning parameters on the implicit rating data is a little higher than those on the explicit rating data, but still in the same order.

8.3.3 Learning Regularized S-HMF Model

When the parameter set, $\{\bar{\mathbf{U}}^C, \mathbf{\Lambda}^C, \bar{\mathbf{V}}^C, \mathbf{\Gamma}^C\}$, is learned from C-HMF, we obtain the distribution of \mathbf{U}_i^C and \mathbf{V}_j^C approximated by the variational distributions $Q(\mathbf{U}_i^C)$ and $Q(\mathbf{V}_j^C)$. Therefore, we can sample $\boldsymbol{\mu}_i^S \sim Q(\mathbf{U}_i^C)$ and $\boldsymbol{\mu}_j^S \sim Q(\mathbf{V}_j^C)$ as the means of empirical prior distributions for S-HMF. To avoid unnecessary sampling noise, the expectations, $\boldsymbol{\mu}_i^S = \mathbb{E}[Q(\mathbf{U}_i^C)] = \mathbf{u}_i^C$ and $\boldsymbol{\mu}_j^S = \mathbb{E}[Q(\mathbf{V}_j^C)] = \mathbf{v}_j^C$ are often used as the means of empirical prior distributions. As a result, we can write the probabilistic

model of S-HMF, shown in the right part of RMRM in Figure 8.4 as follows:

$$P(\mathbf{U}_i^S) = PoGE(\mathbf{U}_i^S | \{\boldsymbol{\mu}_n^S, \varphi_n\}_{n \in \mathbf{T}_i}) \quad (8.36)$$

$$P(\mathbf{V}_j^S) = \mathcal{N}(\mathbf{V}_j^S | \boldsymbol{\mu}_j^S, \text{diag}[\mathbf{d}]) \quad (8.37)$$

$$P(Y_{ij} | \mathbf{U}_i^S, \mathbf{V}_j^S) = \mathcal{N}(Y_{ij} | \mathbf{U}_i^{S\top} \mathbf{V}_j^S, r\eta_{ij}^{-1}) \quad (8.38)$$

where $\text{diag}[\mathbf{d}]$ are diagonal covariance matrices. Additionally, $r\eta_{ij}^{-1}$ denotes the variance of likelihood, where η_{ij} is a novelty score given by the variance model and r is a scale parameter to be learned.

Heteroscedastic Modeling on Specialty

As discussed previously, popular items tend to be widely known by users and have more interaction, so both the choices of and the feedback for these items may largely be influenced by others, whereas tail items tend to be chosen more independently, thus the choices of these items can better reflect the personal preferences of users [217]. As a result, we model the specialty of user choices on the basis of the popularity of items.

Specialty Modeling: If we denote the probability of choosing item j as θ_j , then we have the multinomial distribution over all the items, where $\Gamma(\cdot)$ is the gamma function and $|\mathbf{O}_j|$ denotes the number of observed choices of item j :

$$Mult(\{|\mathbf{O}_j|\} | \boldsymbol{\theta}) = \frac{\Gamma(\sum_j |\mathbf{O}_j| + 1)}{\prod_j \Gamma(|\mathbf{O}_j| + 1)} \prod_j \theta_j^{|\mathbf{O}_j|} \quad (8.39)$$

Moreover, we place a symmetric Dirichlet-prior, $Dir(\boldsymbol{\theta} | \alpha)$ on $\boldsymbol{\theta}$, where the hyper-parameter α can be interpreted as the number of pre-given, pseudo-choices of each item. Then, we can obtain the posterior for all observations:

$$Dir(\{|\mathbf{O}_j| + \alpha\}) \propto \int Mult(\{|\mathbf{O}_j|\} | \boldsymbol{\theta}) Dir(\boldsymbol{\theta} | \alpha) d\boldsymbol{\theta} \quad (8.40)$$

The expectation of this posterior on choosing item j is:

$$\mathbb{E}_{|\mathbf{O}_j|}[Dir(\{|\mathbf{O}_j| + \alpha\})] = \frac{|\mathbf{O}_j| + \alpha}{\sum_j (|\mathbf{O}_j| + \alpha)} = \bar{p}(j|\alpha) \quad (8.41)$$

where $\bar{p}(j|\alpha)$ is the smoothed version of the probability of choosing item j to avoid zero probability, a.k.a. Laplace smoothing of the new items or the items with uncounted choice in a given dataset. In information theory, self-information is a measure of the information content associated with an event in a probability space. Here, a choice is such an event. As analyzed previously, choices on tail items can reflect users' special preferences, i.e., these choices contain more information content. As a result, we give the following definition of specialty of choice in terms of self-information:

Definition 8.3: (Specialty of Choice) Given all observed choices, the specialty of a choice on an item j is measured by self-information:

$$\phi_j = -\log \bar{p}(j|\alpha) \quad (8.42)$$

Specialty Score: We assign the credibility of feedback on a choice in terms of two scores: the reputation of a user (a global score), and the helpfulness of the review (a local score), thus we obtain:

$$\eta_{ij} \stackrel{\text{def}}{=} \begin{cases} \psi_j + \epsilon & (i, j) \text{ indexes an observation} \\ \epsilon & \text{otherwise} \end{cases} \quad (8.43)$$

That is, the observation is associated with a high credibility score only if a high-reputation user gives a helpful review. In particular, we set $\epsilon = 0$ for explicit rating data while ϵ is set to a small constant for implicit rating data (cf. Section 8.2.2).

Since a higher credibility score means a higher level of confidence in that choice, the variance function of a piece of feedback can be given by $f^S(Y_{i,j}) = r\eta_{ij}^{-1}$ (note that lower variance means higher confidence), where t is a scale parameter to be learned.

Parameter Learning

Similar to the derivation of VB on C-HMF, we can easily obtain the lower bound of marginal log-likelihood of S-HMF:

$$\log P(\mathbf{Y}) \geq \int Q(\mathbf{U}^S, \mathbf{V}^S) \log \frac{P(\mathbf{Y}, \mathbf{U}^S, \mathbf{V}^S)}{Q(\mathbf{U}^S, \mathbf{V}^S)} d\mathbf{U}^S d\mathbf{V}^S = \mathcal{L}(Q) \quad (8.44)$$

where $Q(\mathbf{U}^S, \mathbf{V}^S) = \prod_i Q(\mathbf{U}_i^S) \prod_j Q(\mathbf{V}_j^S) = \mathcal{N}(\mathbf{U}_i^S | \mathbf{u}_i^S, \text{diag}[\boldsymbol{\lambda}_i]) \mathcal{N}(\mathbf{V}_j^S | \mathbf{v}_j^S, \text{diag}[\boldsymbol{\gamma}_j])$ is a factorized variational Gaussian distribution. The parameter updating scheme is given in Table 8.3; here, the variational parameters $\{\bar{\mathbf{U}}^S, \boldsymbol{\Lambda}^S, \bar{\mathbf{V}}^S, \boldsymbol{\Gamma}^S\}$ and the model parameters $\{\mathbf{c}, \mathbf{d}, r\}$ are updated in turn to maximize $\mathcal{L}(Q)$, where $\bar{\mathbf{U}}^S = [\mathbf{u}_i^S]_{1 \leq i \leq N}$ denotes a matrix consisting of mean vectors and $\boldsymbol{\Lambda}^S = [\boldsymbol{\lambda}_i^S]_{1 \leq i \leq N}$ denotes a matrix consisting of variance vectors w.r.t. $Q(\mathbf{U}^S)$, and where $\bar{\mathbf{V}}^S, \boldsymbol{\Gamma}^S$ are defined similarly w.r.t. $Q(\mathbf{V}^S)$. With the same trick as that applied to C-HMF, we can efficiently implement this parameter updating scheme on the implicit rating data.

After the parameters of S-HMF are learned, we can either sample $\boldsymbol{\mu}_i^S \sim Q(\mathbf{U}_i^S)$ and $\boldsymbol{\mu}_j^S \sim Q(\mathbf{V}_j^C)$ or use the expectations of the variational Gaussian distribution, $\boldsymbol{\mu}_i^S = \mathbb{E}[Q(\mathbf{U}_i^S)] = \mathbf{u}_i^S$, to construct the PoGE-based empirical priors for the coupled model, C-HMF (cf. Eq. 8.18).

8.4 Algorithm and Prediction

So far, we have presented the details of RMRM and the parameter learning schemes w.r.t. C-HMF and S-HMF, respectively. Algorithm 2 summarizes the whole learning process with recurrent regularization in terms of coupled empirical priors.

In Algorithm 1, we run k-step variational updating for both C-HMF (cf. Line 6) and S-HMF (cf. Line 9). In practice, this works well with a small k (less than 10). This type of updating strategy can be viewed as k-step, mean-field, contrastive divergence [Welling and Hinton 2002], which has proved its effectiveness theoretically.

Table 8.3 : Parameter updating scheme for S-HMF

In the following equations, we denote $W_{ij} = r^{-1}\eta_{ij}$

Update parameters $\{\mathbf{u}_i^S, \boldsymbol{\lambda}_i^S\}$ of distribution $Q(\mathbf{U}_i^S)$ in parallel, for each i :

$$\mathbf{u}_i^S \leftarrow \Psi_i \left[\bar{\mathbf{V}}^S \text{diag}(\mathbf{W}_{i,:}) \mathbf{Y}_{i,:}^\top + \hat{\varphi}_i \mathbf{a} \cdot^{-1} \cdot^* \boldsymbol{\mu}_i^S \right] \quad (8.45)$$

where $\Psi_i^{-1} = \text{diag}(\hat{\varphi}_i \mathbf{a} \cdot^{-1}) + \bar{\mathbf{V}}^S \text{diag}(\mathbf{W}_{i,:}) \bar{\mathbf{V}}^{S\top} + \text{diag}(\boldsymbol{\Gamma}^S \mathbf{W}_{i,:}^\top)$

$$\boldsymbol{\lambda}_i^S \leftarrow (\bar{\mathbf{V}}^S \cdot^2 + \boldsymbol{\Gamma}^S) \mathbf{W}_{i,:}^\top + \hat{\varphi}_i \cdot^{-1} \mathbf{a} \quad (8.46)$$

Update parameters $\{\mathbf{v}_j^S, \boldsymbol{\gamma}_j^S\}$ of the distribution $Q(\mathbf{V}_j^S)$ in parallel, for each j :

$$\mathbf{v}_j^S \leftarrow \Psi_j \left(\bar{\mathbf{U}}^S \text{diag}(\mathbf{W}_{:,j}) \mathbf{Y}_{:,j} + \mathbf{b} \cdot^{-1} \cdot^* \boldsymbol{\mu}_j^S \right) \quad (8.47)$$

where $\Psi_j^{-1} = \text{diag}(\mathbf{b} \cdot^{-1}) + \bar{\mathbf{U}}^S \text{diag}(\mathbf{W}_{:,j}) \mathbf{U}^{S\top} + \text{diag}(\boldsymbol{\Lambda}^S \mathbf{W}_{:,j})$

$$\boldsymbol{\gamma}_j^S \leftarrow (\bar{\mathbf{U}}^S \cdot^2 + \boldsymbol{\Lambda}^S) \mathbf{W}_{:,j}^\top + \mathbf{b} \quad (8.48)$$

Update model parameters $\{\mathbf{a}, \mathbf{b}, t\}$:

$$\mathbf{c} \leftarrow \frac{\sum_i \hat{\varphi}_i [(\mathbf{u}_i^S - \boldsymbol{\mu}_i^S) \cdot^2 + \boldsymbol{\lambda}_i^S]}{N} \quad (8.49)$$

$$\mathbf{d} \leftarrow \frac{\sum_j [(\mathbf{v}_j^S - \boldsymbol{\mu}_j^S) \cdot^2 + \boldsymbol{\gamma}_j^S]}{M} \quad (8.50)$$

$$r \leftarrow \frac{\sum_{ij \in \mathbf{O}} \omega_{ij} [(\mathbf{Y}_{i,j} - \mathbf{u}_i^{S\top} \mathbf{v}_j^S) \cdot^2 + (\mathbf{u}_i^S \cdot^2)^\top \boldsymbol{\gamma}_j^S + (\mathbf{v}_j^S + \boldsymbol{\lambda}_i^S)^\top \boldsymbol{\lambda}_i^S]}{|\mathbf{O}|} \quad (8.51)$$

Algorithm 2 Parameter Learning for RMRM

- 1: Pre-compute credibility score ω_{ij} for each entry using Eq. 8.17;
 - 2: Pre-compute specialty score η_{ij} for each entry using Eq. 8.43;
 - 3: $it \leftarrow 0$
 - 4: **while** $it \leq MAX_ITERATION$ **do**
 - Learning C-HMF*
 - 5: Construct empirical priors via Eq. 8.11 and 8.12 using $\{\boldsymbol{\mu}_i^S, \boldsymbol{\mu}_j^S\}$ from S-HMF
 - 6: Run k -step parameter updating as Table 8.2
 - 7: **for** each fully cold-start user c **do**
 - 8: $\mathbf{U}_c^C = \mathbb{E} \left[PoGE \left(\{\mathbf{U}_t^C, \varphi_t\}_{t \in \mathcal{N}_i^K \cup \mathcal{N}^R} \right) \right]$
 - 9: **end for**
 - Learning S-HMF*
 - 10: Construct empirical priors via Eq. 8.36 and 8.37 using $\{\boldsymbol{\mu}_i^C, \boldsymbol{\mu}_j^C\}$ from C-HMF
 - 11: Run k -step parameter updating as Table 8.3
 - 12: **for** each fully cold-start user c **do**
 - 13: $\mathbf{U}_c^S = \mathbb{E} \left[PoGE \left(\{\mathbf{U}_t^S, \varphi_t\}_{t \in \mathcal{N}_i^K \cup \mathcal{N}^R} \right) \right]$
 - 14: **end for**
 - 15: **end while**
-

Moreover, we use PoGE to approximate the distribution of user factors of those who are fully cold-start users without any feedback (cf. Lines 7 and 10). Since there is no data available for a fully cold-start user to update his/her user factors, we post-update them using the updated user factors from their mostly related trusters when the sub-iterations of C-HMF and S-HMF are finished.

Prediction: After the parameters of RMRM are learned, we obtain the regularized estimates $\{\mathbf{U}^C, \mathbf{V}^C\}$ for C-HMF and $\{\mathbf{U}^S, \mathbf{V}^S\}$ for S-HMF. We can predict the

missing entries of the user-item matrix using the MF reconstruction form using these estimates. According to the variational approximation, we have $\mathbf{U}_i^C \sim Q(\mathbf{u}_i^C, \boldsymbol{\lambda}_i^C)$ and $\mathbf{V}_j^C \sim Q(\mathbf{v}_j^C, \boldsymbol{\gamma}_j^C)$; the means of \mathbf{U}_i^C and \mathbf{V}_j^C are just \mathbf{u}_i^C and \mathbf{v}_j^C which can be obtained from Eq. 8.29 and Eq. 8.31. Therefore, we can reconstruct the value of entry (i, j) as follows:

$$\hat{Y}_{ij} = \mathbb{E} [P(\mathbf{U}_i^{C\top} \mathbf{V}_j^C)] \approx \mathbb{E}[Q(\mathbf{U}_i^C)Q(\mathbf{V}_j^C)] = \mathbf{u}_i^{C\top} \mathbf{v}_j^C \quad (8.52)$$

Similarly, we can reconstruct the value of entry (i, j) using \mathbf{U}_i^S and \mathbf{V}_j^S :

$$\hat{Y}_{ij} = \mathbb{E} [P(\mathbf{U}_i^{S\top} \mathbf{V}_j^S)] \approx \mathbb{E}[Q(\mathbf{U}_i^S)Q(\mathbf{V}_j^S)] = \mathbf{u}_i^{S\top} \mathbf{v}_j^S \quad (8.53)$$

Whether to choose the prediction result from Eq. 8.52 or Eq. 8.53 is dependent on specific data sets. In general, the prediction results from Eq. 8.53 place more emphasis on the personal taste for specific choices, so-called RMRM-S, whereas the prediction result from Eq. 8.52 may achieve better performance in a system with a large amount of spam feedback, so-called RMRM-C. In practice, we choose one of these dependent on a real-world environment.

8.5 Discussion

So far, we have presented RMRM and the corresponding learning algorithm. In fact, the idea and the methods adopted by RMRM have direct connections with other methods. Hence, we discuss these connections in this section.

8.5.1 Multi-objective Optimization

RMRM consists of two main components, where C-HMF models user choices by emphasizing credibility and S-HMF models user choices by emphasizing specialty. Each component leads to an objective for optimization, so RMRM deals with a multi-objective optimization problem (MOP) [44]. However, the conventional MOP

often has two independent objectives, thus it needs to obtain solutions using higher-level information, whereas the two objectives of RMRM are coupled by the empirical priors induced from each other. In fact, the two objectives of RMRM are constructed from the same data, and we use a recurrent algorithm to learn the parameters that are regularized by the empirical priors induced from each other objective model. Therefore, RMRM is a variant case of MOP.

In general, the optimal solution of MOP is not unique, and it often uses a genetic algorithm to search the solution space [Deb 2014]. The recurrent learning algorithm of RMRM induces new empirical priors in each iteration, and S-HMF and C-HMF are reset using the new priors, which leads to new objectives for optimization, cf. Eq. 8.24 and Eq. 8.44. Hence, an iteration of RMRM corresponds to a generation of a genetic algorithm to search for the optimal solution. Taking Eq. 8.44 as an example, the new objective may find better estimates of the parameters, provided that we have learned better priors $P(\mathbf{U}_i^S)$ and $P(\mathbf{V}_j^S)$, leading to better $P(\mathbf{Y}|\mathbf{U}^S, \mathbf{V}^S)$. As a result, the marginal likelihood $P(\mathbf{Y})$ is improved (cf. Eq. 8.44). Moreover, the new empirical priors from the peer model can help to find a better optimal solution in the next iteration. In comparison, the objective function of a single-objective model, such as MF, does not change with iterations so they more easily become stuck in local minima.

8.5.2 Social Regularization from PoGE Perspective

In recent years, one prevalent approach of RSs has been to incorporate social relationships for regularization [94, 141]. This method is built on the basic idea that users' preferences are mostly influenced by others with the strongest social relationships, typically, their trusters. In general, the social regularization on a user i often leads to the following two forms of the regularization term, and we denote

them as SR1 [141] and SR2 [94, 141], respectively.

$$\begin{aligned}
 SR1 : \quad & \lambda_1 \left\| \mathbf{U}_i - \frac{\sum_{t \in \mathbf{T}_i} s_{it} \mathbf{U}_t}{\sum_{t \in \mathbf{T}_i} s_{it}} \right\|_2^2 \\
 SR2 : \quad & \lambda_2 \sum_{t \in \mathbf{T}_i} s_{it} \|\mathbf{U}_i - \mathbf{U}_t\|_2^2
 \end{aligned}$$

where \mathbf{T}_i denotes i 's truster set, s_{it} is the strength or similarity between i and t [141], or where we can simply use $s_{it} = 1$ to denote an observed link [94].

It is interesting to find that, in fact, both SR1 and SR2 are identical from the PoGE perspective, as both of them actually correspond to the same PoGE-prior. Now, let us set up the PoGE-prior for user i as follows:

$$PoGE(\mathbf{U}_i | \{\mathbf{U}_t, s_{it}\}_{t \in \mathbf{T}_i}) = \prod_{t \in \mathbf{T}_i} \mathcal{N}(\mathbf{U}_i | \mathbf{U}_t, \lambda_2^{-1} s_{it}^{-1} \mathbf{1}) \quad (8.54)$$

Obviously, we can obtain SR2 by taking the negative log-form of Eq. 8.54. According to Eq. 8.19, we can obtain the following equivalent form from Eq. 8.54:

$$PoGE(\mathbf{U}_i | \{\mathbf{U}_t, s_{it}\}_{t \in \mathbf{T}_i}) = \mathcal{N}\left(\mathbf{U}_i \left| \frac{\sum_{t \in \mathbf{T}_i} s_{it} \mathbf{U}_t}{\sum_{t \in \mathbf{T}_i} s_{it}}, \lambda_1^{-1} \mathbf{1} \right.\right) \quad (8.55)$$

where $\lambda_1 = \lambda_2 \sum_{t \in \mathbf{T}_i} s_{it}$. By taking the negative log-form of Eq. 8.55, we immediately obtain SR1. Therefore, SR1 and SR2 are actually derived from the same PoGE-prior, so we prove the identity between them. In fact, by using SR1 and SR2, the evaluation results are very close [141]. The small difference is probably caused by the settings of the regularization parameters λ_1 and λ_2 and by the random initialization of the parameters.

8.6 Experiments

We conduct empirical evaluations using two real-world datasets that cover the cases of, respectively, explicit rating data and implicit rating data. We compare

RMRM with a set of state-of-the-art methods gauged by various metrics. The overall results prove that our approach significantly outperforms all the compared methods.

8.6.1 A Comparison of the State-of-the-Art Methods

In the following experiments, a group of state-of-the-art methods introduced in Section 3.5 with the following settings; some are used for explicit rating data and others for implicit rating data.

- *PMF*: This method learns the factors of users and items from a rating matrix without taking additional information into account.
- *Trust- k NN*: This method takes the top- k , high-reputational trusters of a user as the neighbors, and then predicts the user’s rating of an item by averaging the available neighbors’ ratings for that item [102].
- *SoRec*: This method jointly models the trust-link matrix and a user-item rating matrix, which shares user factors to propagate the interaction between two matrices.
- *SoReg*: This method utilizes the trust relationships to construct the regularizer to learn user factors.
- *SocialMF*: This method is very similar to SoReg. The main difference lies in the setting of similarities for trusters.
- *MF-IR*: This is a zero-mean WRMF model to deal with implicit rating data.
- *SoRec-IR*, *SoReg-IR*, *SocialMF-IR*: The original versions of SoRec, SoReg, and SocialMF were designed for learning preferences from explicit rating data. To enable them to deal with implicit rating data, we extend them using weight modeling, as in MF-IR.

- *C-HMF*: One of the main components of RMRM is to enhance credibility-based modeling. Moreover, we use zero-mean regularization since the single model does not have the empirical priors learned through S-HMF.
- *S-HMF*: One of the main components of RMRM is to enhance specialty-based modeling. Moreover, we use zero-mean regularization since the single model does not have the empirical priors learned through C-HMF.
- *RMRM*: RMRM is the main model proposed in this chapter. Since C-HMF and S-HMF can model both explicit rating data and implicit rating data in a unified way, RMRM naturally has an advantage. In particular, we use RMRM-C to denote the prediction results generated using Eq. 8.52, and RMRM-S to denote the prediction result generated using Eq. 8.53.

8.6.2 Explicit Rating Data Evaluation

Data Preparation

We construct a truncated dataset from the RED dataset [151], as mentioned in the introduction by filtering both users and items with fewer than three ratings. This is because no data will be available for training if a user or an item only accounts for one or two ratings that are held out for testing. In addition, we need at least two testing items for a user in order to evaluate the accuracy of the ranking for these items. The statistics of this evaluation dataset are illustrated in Table 8.4.

Figure 8.6 demonstrates the long-tail distributions for the number of ratings w.r.t. items and users. We find that a large number of both items and users in the tail have very few ratings. Therefore, this dataset is suitable to evaluate the performance of recommendations for users and items in the tail of distributions. The hyperparameters of the compared methods are tuned by cross-validation. Here, we find that the length of the latent factor vector can produce good results with this

Table 8.4 : Statistics of the Epinions Dataset

# users: 39,902	# items: 63,027
# trust links: 43,8965	# trusters / # users: 11
max # of trusters: 1,713	# users with zero truster: 14,202
# ratings: 734,441	density: 0.029%
# ratings / # users: 18	# ratings / # items: 11
max # ratings of user: 1,809	max # ratings of item: 2,112

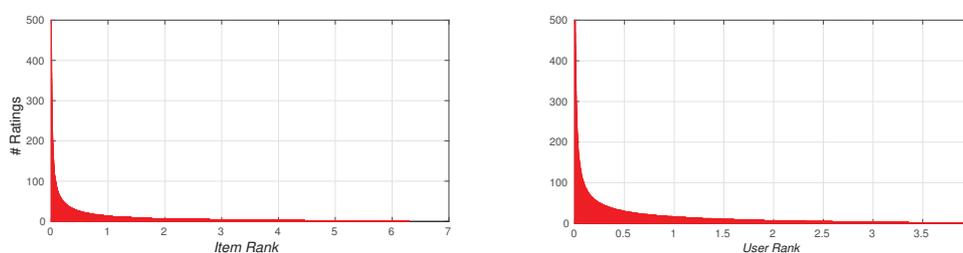


Figure 8.6 : Long-tail distributions for the number of ratings of items and users (truncated from 0 to 500).

dataset by setting $K=5$.

Figure 8.7 illustrates the distributions of the number of helpful scores w.r.t. items and users in this evaluation dataset. We find that they have similar long-tail distributions with those in Figure 6. This is a natural phenomenon because helpful scores are based on reviews, and more reviews tend to receive more helpful scores. Thus, these helpful scores are used for the reputation model. In this experiment, we set $\alpha = 1$ and $\beta = 3$ in Eq. 8.16 to compute the reputation scores. That is, the initial reputation score is 0.25 for new users.

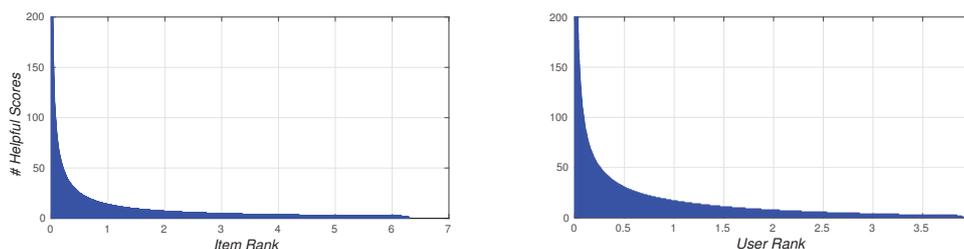


Figure 8.7 : The distributions for the number of helpful scores w.r.t. items and users (truncated from 0 to 200).

Prediction of Long-tail Distributed Items

Improving the prediction performance of long-tail items would obviously bring more business profit to a company by precisely targeting a specific group of users. To evaluate the prediction performance of long-tail items, we randomly hold out 20% of the data from the evaluation dataset as the ground truths for testing, denoted as $\mathbf{T}_{s_{20}}$. Then, as shown below, we split $\mathbf{T}_{s_{20}}$ into four parts according to the popularity of the items so that we can compare the performance of different methods using both short-head items and long-tail items.

- Most Popular: The items in the headmost 5% of the distribution, as shown in the left-hand image of Figure 6.
- Less Popular: The items in the 5-20% interval of the distribution.
- Shallow Tail: The items in the 20-50% interval of the distribution.
- Deep Tail: The items in the endmost 50% of the distribution.

We evaluate the MAE of all comparative methods of these four parts of the distribution. Note that the data becomes extremely sparse in the deep tail, and, as a result, Trust- k NN is barely effective, as all of the neighbors tend never to rate the testing items. In such a case, we simply predict the ratings of an item as $\text{Mean} + \epsilon$,

where Mean denotes the mean rating for all items, and ϵ is a small random value, following standard Gaussian distribution.

Figure 8.8 reports the results of the comparison of all methods for the four parts of the distribution. We find the performance of Trust- k NN decreases when the data become sparser since the tail items are rarely rated, which results in the random prediction mentioned above. Obviously, such neighborhood-based methods have a limitation when conducting recommendations in the long tail. We find that PMF outperforms Trust- k NN, as it does not need to search the neighborhood; instead, similarity is implicitly represented by latent factors. However, PMF suffers from the three aforementioned typical issues in long-tail recommendations. As illustrated by the four cases shown in Figure 8, we find that PMF achieves a relatively higher accuracy in the cases of Most Popular and Less Popular, but that the performance becomes worse when the available ratings for items become fewer, especially in the case of Deep Tail. In comparison, C-HMF improves the ability to alleviate shilling attacks, and it enables user preference for the long-tail items that are to be targeted in terms of heteroscedasticity modeling. As a result, C-HMF significantly outperforms PMF.

The remaining models involve trust relationships as the secondary information aspect, which addresses data insufficiency in the tail of distribution. Comparing SoRec with PMF, we find that the involved truster relationships are helpful to improve the accuracy of long-tail items. However, both the rating matrix and the trust matrix convey heterogeneous information, but SoRec cannot find a best trade-off point for all users. To overcome this deficiency, SoReg and SocialMF incorporate the context of trusters to regularize user factor learning. The results prove that SoReg is more effective than SoRec. In particular, RMRM-S is selected in this experiment since we would like to more aggressively emphasize users' special preferences over tail items. From the results, we easily find that RMRM-S achieves

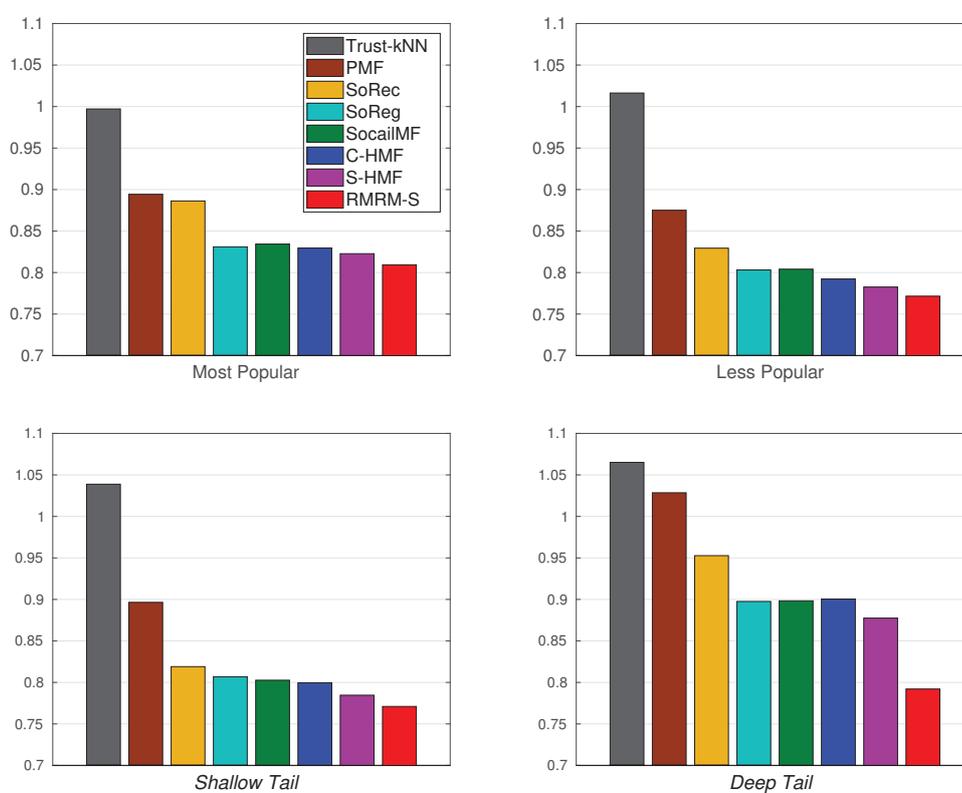


Figure 8.8 : MAEs of rating prediction for the long-tail item distribution.

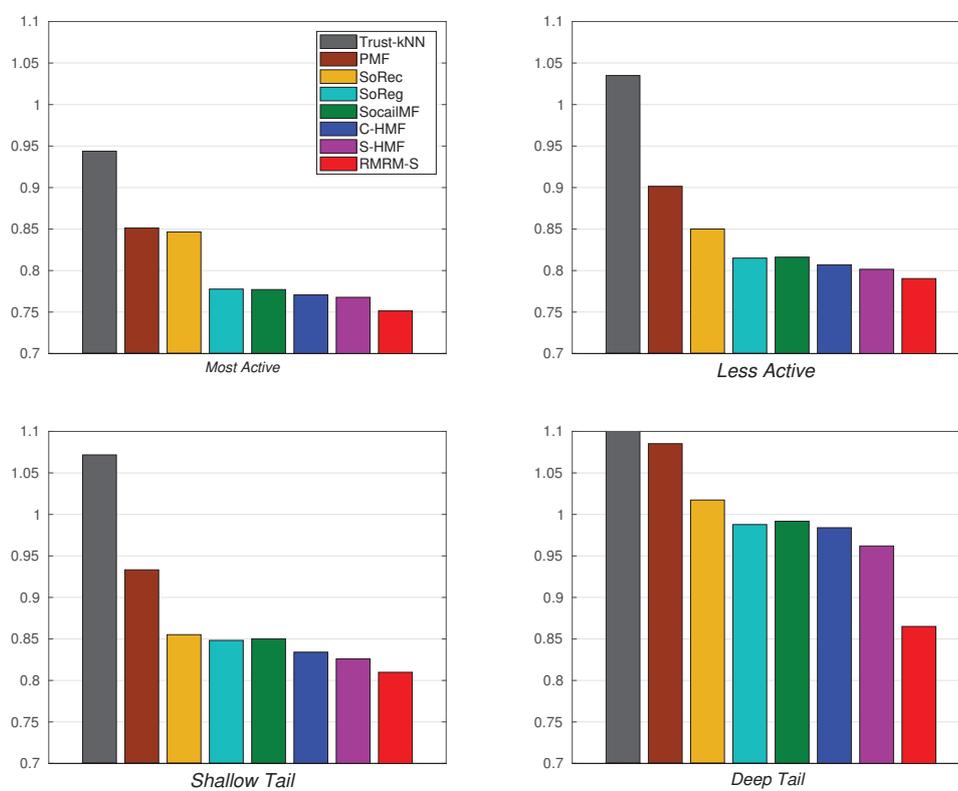


Figure 8.9 : MAEs of rating prediction for the long-tail user distribution.

the best performance for all four cases. Note that the performance of Deep Tail is even better than Most Popular, which demonstrates that our model is able to better learn users' preferences from the long tail. Moreover, the deviations of the MAEs in the four cases are small. Such stable performance over the whole distribution may be attributed to the fusion of reliability and novelty, brought about by the coupled recurrent regularization. Therefore, we can conclude that RMRM-S is the most accurate model for recommending long-tail items.

Prediction on Long-tail Distributed Users

Accurate recommendations for long-tail users can significantly improve users' experiences and users' retention rates. In the next experiment, we conduct an evaluation on the testing set $\mathbf{T}_{S_{20}}$. As in the previous experiment, we split $\mathbf{T}_{S_{20}}$ into four parts according to the activity of the users to compare the performance of both the short-head and long-tail users.

- Most Active: The users in the headmost 5% of the distribution, as shown in the right-hand image of Figure 6.
- Less Active: The users in the 5-20% interval of the distribution.
- Shallow Tail: The users in the 20-50% interval of the distribution.
- Deep Tail: The endmost 50% users of the distribution of the distribution.

Figure 8.9 shows the comparative results of all of the methods for the long-tail user distribution. We observe similar results to those in the previous experiment. Actually, conducting accurate predictions for deep-tail users is more difficult than for deep-tail items because almost all deep-tail users have both few ratings and few trust relationships. For those models that do not use trust relationships, S-HMF achieves the best performance since the heteroscedasticity modeling of user choices enables it to learn users' personal preferences better.

In particular, we found that more than one-third of the users have no links, as illustrated in Table 8.4. Consequently, SoRec cannot obtain secondary information for these users due to the lack of links in the trust matrix. Similarly, no truster is available to conduct regularization for SoReg and SocialMF. As a result, these methods cannot learn user factors when there is no trust link available for a cold-start user. To overcome this deficiency, RMRM-S incorporates top-N high-reputation experts into the system. Hence, RMRM-S can still conduct regularization, even when no direct trusters are available. Since RMRM-S takes the advantages of C-HMF, S-HMF, and SoReg, it results in a significant improvement in recommendations for long-tail users.

Impact of the Number of Involved Trusters

The previous experiments show that borrowing knowledge from trusters can be very helpful to address the challenges of recommendations for long-tail items. In RMRM, the truster set consists of two parts: the trusters that a user actively follows, and the experts with the highest reputation in the system, cf. Eq. (24). We next illustrate the impact of the number of user trusters and the number of system trusters, respectively. In this experiment, we use the same testing set as in the previous experiments.

MAEs for Different Numbers of Involved User Trusters: We fix five high-reputation system experts and vary the number of top-K trusters, where $K \in \{5, 10, 20, 50, All\}$, to compare the performances. Figure 8.10a displays the results when the number of trusters changes. We find that increasing the number of trusters improves the performance of tail users. This is because they account for very little data, so there is a need to incorporate more trusters for regularization. In comparison, we find that the performance of head users is not improved, and even becomes worse when K increases. This can be attributed to the fact that head users account

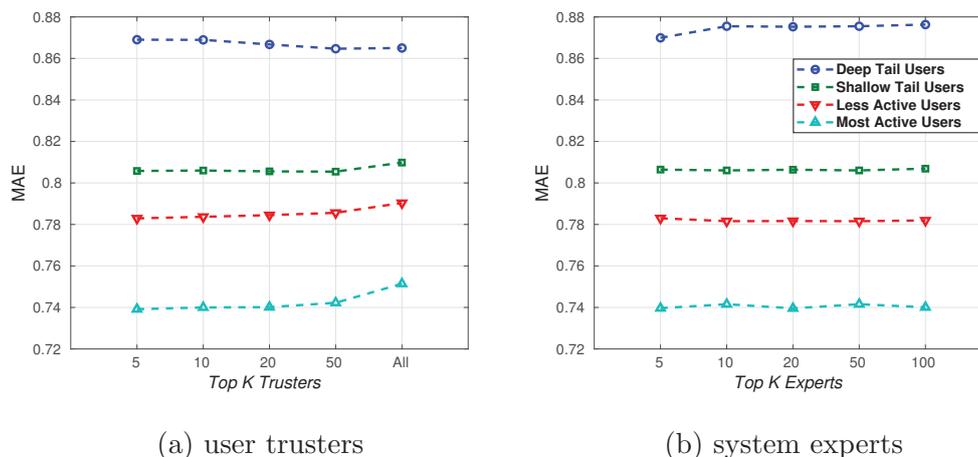


Figure 8.10 : MAEs varying the numbers of involved (a) user trusters, and (b) system experts.

for sufficient data, which enables RSs to learn their preferences without borrowing information from others. Moreover, we find that involving too many trusters does not improve performance. It can thus be interpreted that the priors from too many trusters over-regularize the user preferences learning.

MAEs for Different Number of Involved System Experts: We fix the top five trusters of each user, and vary the number of top-K high-reputation system experts, where $K \in \{5, 10, 20, 50, 100\}$. From Figure 8.10b, we find that the performance is very close under different K, becoming a little worse when K reaches 100. That is, it involves too many experts, which may over-regularize the user factors learning. As a result, we only need to involve a small group of system experts in practice.

Shilling Attack Simulation

In this experiment, we attempted to test the robustness of each model in a shilling attack environment. To simulate such an environment, we created 1,000 virtual spam users to conduct the attack, and we respectively selected 100 items

from the head (0%-20%) and the tail (20%-100%) as the attack targets. In this experiment, we conducted nuke attack in the case of the average attack model [24]. More specifically, we first randomly selected the 50 most popular items from the head of distribution to serve as the filler item set [24]. Before conducting the attack, we assigned each item in the filter item set with the mean rating of that item for each spam user. As a result, we built a fake profile for these spam users who have average preferences that are similar to most users. Then, we simulated the nuke attack on each target item by injecting fifty minimum ratings, i.e. 1, from fifty out of 1,000 spam users by random selection. Thus, we constructed a user-item rating matrix with fake ratings and spam users.

We retrained all the comparison models on this attacked rating matrix and then made predictions. Figure 8.11 illustrates the prediction results for the head items (left) and the tail items (right). Obviously, the MAEs of the head items are lower overall than those of the tail items, which reveals the fact that the tail items are more easily biased by fake ratings due to the few ratings they receive. PMF achieves poor performance because it is completely based on the ratings for each user without incorporating any other information, whereas SoReg and SocialMF are more robust to shilling attack due to the regularization from trusters or experts. In comparison, SoRec does not achieve comparable performance with SoReg and SocialMF, which illustrates that the impact from the fake rating matrix overwhelms that from the trust-link matrix, especially when the trust-link matrix is very sparse. RMRM-S achieves better performance than that of the single S-HMF model because S-HMF in RMRM-S is regularized by the empirical priors from C-HMF. Finally, we find that C-HMF and RMRM-C achieve much better performance than other models. In particular, the results of C-HMF and RMRM-C do not become worse as do the other models in the case of tail items, which proves that the heteroscedastic modeling for credibility is a very effective way to defend against shilling attack.

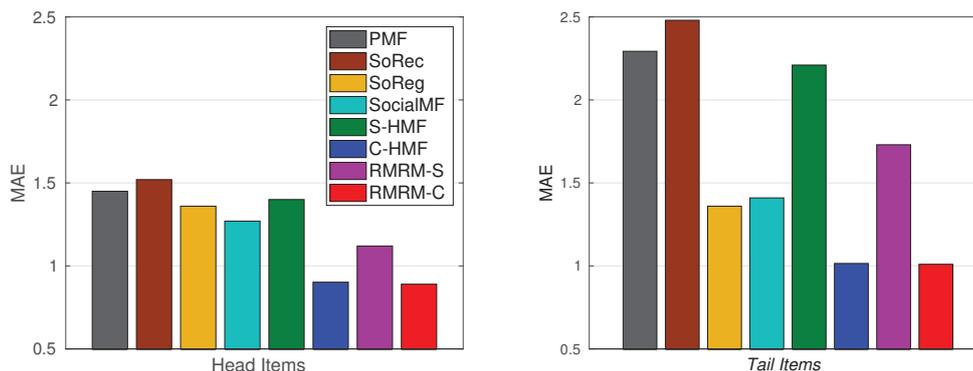


Figure 8.11 : MAEs for head items and tail items with shilling attack.

8.6.3 Implicit Rating Data Evaluation

Data Preparation

With the popularity of mobile phones, millions of apps have been published online, covering all aspects of daily life, including food, shopping, sports, games, and so on. Popular apps (head items) are known by most users, so recommending unpopular apps (tail items) to users is a more meaningful task. Here, we use a publicly available data set of apps for Android from Amazon [147] to evaluate all the compared methods. Since the installation history is always available in the app store, for our experiment, we take the installation record as the implicit rating (with the observed installation of an item as 1). From the raw data, we remove users who have less than three installations and items that have less than four installations. The statistics of this evaluation dataset are illustrated in Table 8.5.

Figure 8.12 shows the distributions of the number of installations w.r.t. users and items of this evaluation set. We see that the number of installations w.r.t. both items and users have obvious long-tail distributions. The hyperparameters of all the compared methods have been tuned by cross-validation. We find that the length of

Table 8.5 : Statistics of Apps for Android Dataset

# users: 234,347	# Apps: 24,141
# installations: 1,274,896	density: 0.023%
# installations / # users: 5.44	# installations / # items: 52.81
max # installations of user: 565	max # installations of item: 11,801

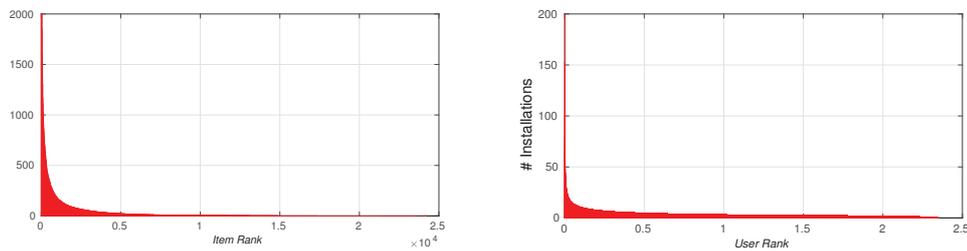


Figure 8.12 : Long-tail distributions over the number installations w.r.t. users and items (truncated).

the latent factor vector can produce good results with this dataset by setting $K=50$. Moreover, we set $\alpha = 1$, $\beta = 1$ for the reputation score, defined by Eq. 8.16.

This dataset does not provide explicit relationships between the users. Intuitively, the number of choices of common apps between two users can be used to measure their similarity. Moreover, the choices on tail items better reflect user preferences. Therefore, we find the top- K neighbors of user i by ranking the weighted sum: $\sum_{v \in \mathbf{O}(i,j)} \{\psi_v | j \neq i\}$ for all users except i , where ψ_v is defined by Eq. 8.42 and $\mathbf{O}(i, j)$ is the set of common apps between i and j in the training set.

Evaluation of Tail Items Recommendation

For a real-world RS, generating an accurate list of attractive items for each user is more meaningful than accurately predicting ratings because, of course, the final goal of RSs is to find items desired by different users. In this experiment, we randomly

hold out 20% of the observations from each item as the testing set, denoted as $\mathbf{T}_{s_{20}}$, and use the remainder for the training set. As in the previous experiments, we split $\mathbf{T}_{s_{20}}$ into four item groups according to the number of installations, namely *Most Popular*, *Less Popular*, *Shallow Tail*, and *Deep Tail*.

Table 8.6 reports the mean AP@10, AP@20, nDCG@10, and nDCG@20 for testing the items in each group. In the case of Most Popular, the results from all models are relatively close; this is due to the sufficient data of the head items. Overall, RMRM models achieve better performance than the other models, which proves that RMRM can better capture user preferences for tail items. In particular, RMRM-S and S-HMF achieve better performance than the other models in the cases of Most Popular and Less Popular, whereas RMRM-C and C-HMF outperform the others in the cases of Shallow Tail and Deep Tail. This reflects the fact that the apps in the tail are known by very few people, so that their installation and corresponding feedback mainly come from two types of users: (a) users who really have interest in these apps (i.e., valuable feedback), and (b) Internet marketers (i.e., valueless feedback). Accordingly, the designs of RMRM-C and C-HMF emphasize the feedback from the former and de-emphasize the feedback from the latter. Furthermore, RMRM-C incorporates the empirical priors from S-HMF for regularization, thus it achieves the best performance for recommending items in the tail.

Figure 8.13 depicts the recall@20-50 curves for all compared models for recommending tail items (Shallow Tail and Deep Tail). Similar to the performance shown in Table 8.6, RMRM-based methods outperform the other approaches. Therefore, we find that the curve of RMRM-C is above all the other models with obvious margins, which, again, proves that RMRM-C can better capture users' special preferences and provide more robust protection against shilling attack.

Table 8.6 : Mean AP@5, AP@10, nDCG@10, and nDCG@20 of item recommendations

<i>Most Popular</i>					<i>Less Popular</i>			
Method	AP		nDCG		AP		nDCG	
	@10	@20	@10	@20	@10	@20	@10	@20
MF-IR	0.0135	0.0144	0.016	0.0195	0.0112	0.0121	0.0187	0.0222
SoRec-IR	0.0135	0.014	0.0162	0.0182	0.0111	0.0119	0.018	0.0216
SoReg-IR	0.0133	0.0141	0.0163	0.0191	0.0119	0.0128	0.0191	0.0231
SocialMF-IR	0.0144	0.015	0.0177	0.02	0.0119	0.0128	0.0195	0.0232
S-HMF	0.0149	0.0156	0.0184	0.0211	0.0123	0.0131	0.0199	0.0237
C-HMF	0.0126	0.0131	0.0157	0.0174	0.0107	0.0115	0.0175	0.0208
RMRM-S	0.0153	0.0159	0.0187	0.0212	0.0125	0.0132	0.0202	0.0239
RMRM-C	0.0131	0.0136	0.0161	0.018	0.0109	0.0116	0.0182	0.0213
<i>Shallow Tail</i>					<i>Deep Tail</i>			
Method	AP		nDCG		AP		nDCG	
	@10	@20	@10	@20	@10	@20	@10	@20
MF-IR	0.0108	0.0096	0.0305	0.0331	0.012	0.0077	0.0362	0.0342
SoRec-IR	0.0108	0.0095	0.0302	0.0328	0.012	0.0076	0.0364	0.0338
SoReg-IR	0.0116	0.0103	0.0322	0.0349	0.0134	0.0086	0.0408	0.0383
SocialMF-IR	0.0111	0.0099	0.0315	0.0342	0.0128	0.0082	0.0396	0.0369
S-HMF	0.0129	0.011	0.0355	0.0373	0.016	0.0101	0.0475	0.0428
C-HMF	0.0171	0.014	0.0438	0.0438	0.0238	0.0151	0.0654	0.0578
RMRM-S	0.013	0.011	0.0356	0.0374	0.0165	0.0105	0.0485	0.0448
RMRM-C	0.0175	0.0142	0.0453	0.0445	0.024	0.0154	0.0659	0.0592

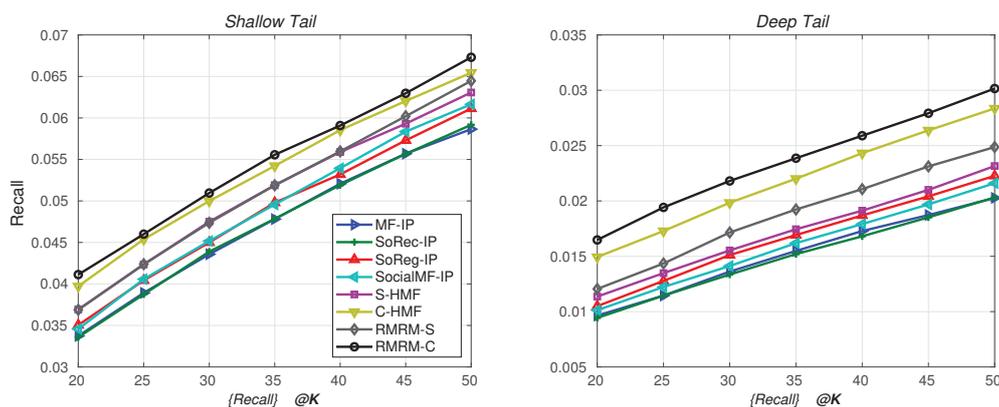


Figure 8.13 : Recall@20-50 of tail-item recommendations for users.

Evaluation of Tail-users' Recommendations

For a company, finding an accurate list of potential users to deliver the information about their apps can reduce large promotion costs. In this experiment, as before, we randomly hold out 20% of the observations from the users as the testing set, denoted as $\mathbf{T}_{s_{20}}$, and use the remainder as the training set. In the same way as the previous experiments, we split $\mathbf{T}_{s_{20}}$ into four user groups, i.e., Most Active, Less Active, Shallow Tail, and Deep Tail, according to the number of apps that a user has installed. Table 8.7 reports the mean AP@10, AP@20, nDCG@10, and nDCG@20 when testing the users in each group. It is easily observed that RMRM-based models are superior to the other models, and that RMRM-S achieves the best performance in the case of Most Active while RMRM-C shows its advantage in the other cases. Both heteroscedasticity modeling on credibility and regularization with coupled empirical priors enable RMRM-C to capture the preferences of tail users more precisely, thus RMRM-C more effectively recommends attractive apps to tail users.

Figure 8.14 shows the recall@20-50 curves of all of the compared models when recommending items for tail users (Shallow Tail and Deep Tail). Similar to the

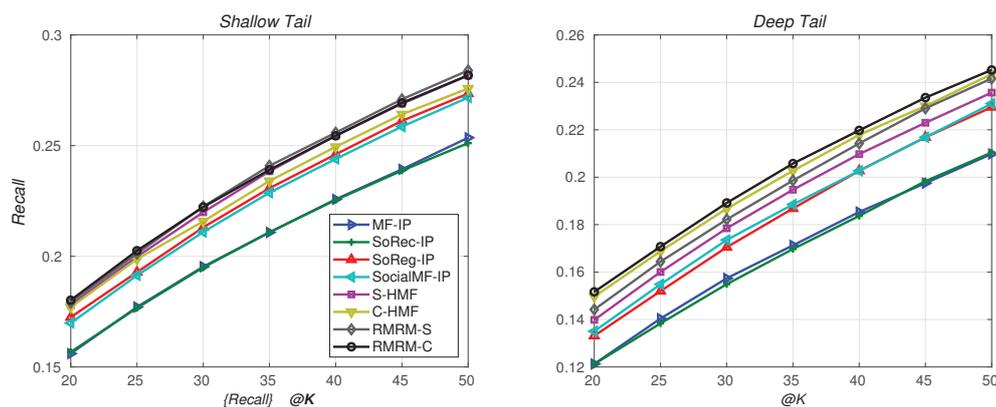


Figure 8.14 : Recall@20-50 of tail-user recommendations for users.

prediction performance for tail users, as shown in Table 8.7, the recall curves of the RMRM approach are above those of other models, which proves that the features learned from RMRM can more reliably represent the traits of items and the personal preferences of users.

8.7 Summary of Contributions

In this chapter, we model a multi-objective RS to address the challenges of recommendations for long-tail items and users. In particular, the non-IID technique is focused on modeling the coupling relationships between group members for making group choices. The main contributions of this work are summarized as follows:

- We analyze the vulnerabilities of current approaches for tail items and users as they pertain to the challenges of popularity bias, cold start, and shilling attack. As a result, we establish a pair of coupled objectives to jointly emphasize the specialty of choices and assess the *credibility* of feedback.
- We design a recurrent mutual regularization process to couple the two objectives, namely specialty and credibility, which are modeled by S-HMF and C-HMF. To implement the recurrent mutual regularization process for RMRM,

Table 8.7 : Mean AP@5, AP@10, nDCG@10, and nDCG@20 of user recommendations

<i>Most Popular</i>					<i>Less Popular</i>			
Method	AP		nDCG		AP		nDCG	
	@10	@20	@10	@20	@10	@20	@10	@20
MF-IR	0.0516	0.0555	0.0693	0.0834	0.0484	0.0522	0.0646	0.0787
SoRec-IR	0.0515	0.0553	0.0693	0.0832	0.0472	0.051	0.0632	0.0768
SoReg-IR	0.0572	0.0614	0.0762	0.0917	0.0516	0.0555	0.0692	0.0835
SocialMF-IR	0.0567	0.0609	0.0762	0.0916	0.0511	0.0551	0.069	0.0836
S-HMF	0.0608	0.065	0.0805	0.0959	0.0554	0.0594	0.0735	0.0882
C-HMF	0.06	0.064	0.079	0.0936	0.0556	0.0594	0.0734	0.0874
RMRM-S	0.0614	0.0655	0.082	0.097	0.0551	0.0593	0.0737	0.0893
RMRM-C	0.0607	0.0648	0.08	0.095	0.0564	0.0604	0.0744	0.0891
RMRM-C	0.0131	0.0136	0.0161	0.018	0.0109	0.0116	0.0182	0.0213
<i>Shallow Tail</i>					<i>Deep Tail</i>			
Method	AP		nDCG		AP		nDCG	
	@10	@20	@10	@20	@10	@20	@10	@20
MF-IR	0.0414	0.0455	0.0637	0.0793	0.0313	0.0352	0.0631	0.0802
SoRec-IR	0.0419	0.0459	0.0644	0.0798	0.0312	0.0351	0.0631	0.08
SoReg-IR	0.0459	0.0502	0.0711	0.0876	0.0347	0.039	0.0699	0.0881
SocialMF-IR	0.0466	0.0508	0.0713	0.0876	0.0355	0.04	0.0711	0.0899
S-HMF	0.0503	0.0546	0.0764	0.093	0.0389	0.0433	0.0768	0.095
C-HMF	0.052	0.0559	0.0779	0.0943	0.042	0.0466	0.0832	0.1025
RMRM-S	0.05	0.0545	0.0763	0.0932	0.0384	0.043	0.0774	0.0964
RMRM-C	0.0522	0.0563	0.0786	0.0946	0.0421	0.0468	0.0833	0.103

we design a scalable algorithm based on the variational Bayesian method to efficiently learn its parameters.

- We conduct empirical evaluations on two real-world data sets. Based on various evaluation metrics, the overall results prove that our approach significantly outperforms the comparison methods.
- RMRM provides a general framework to couple multiple objectives and to learn the comprehensive latent features. Although we focus mainly on the recommendation for long-tail users and items, the proposed approach could potentially be applied to many other recommendation problems where multiple objectives are modeled.

Chapter 9

Attraction-based Recommender Systems for Capturing and Interpreting User Attraction in Content

9.1 Introduction

With the rise of new media, a large amount of social media posts, news articles, and online videos are produced in the cyberspace. In recent years, more and more platforms of content creation, aggregation and distribution have emerged, which aims to attract users with the tailored feed list of content for more traffic. For example, Toutiao is the largest mobile content platform in China, with 120 million daily active users as of Sep. 2017. A similar situation also happens in academia, electronic preprints, known as e-prints, of scientific papers have become the most prevalent way to deliver new ideas or new applications to the public without a long peer review process. The most notable example is arXiv where more than 1.3M e-prints are openly accessible till Jan. 2018*. Researchers around the world submit thousands of new e-prints to deliver their new work in every day. Ambitious researchers would like to promptly obtain the first-hand information from these new e-prints to inspire them and promote their research. However, most researchers often obtain the second-hand information about some new articles from the blogs that they followed.

The current RSs may suggest users with potentially interested articles according to similar users' history by such techniques as CF. However, CF does not work in

*https://arxiv.org/stats/monthly_submissions

ARTICLE

doi:10.1038/nature24270

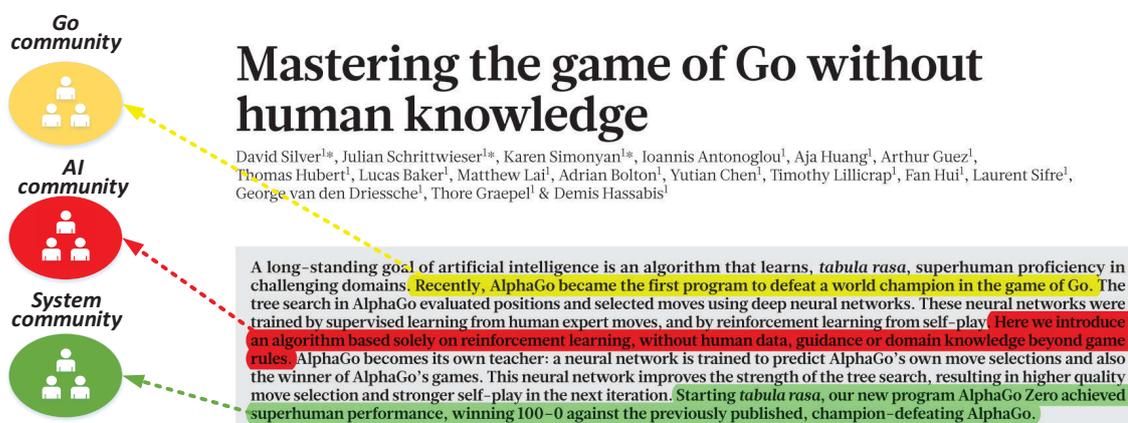


Figure 9.1 : Users with different background may have quite different attractive points on an article [199]

the cold-start cases, namely, on those articles that have not been followed by any other users. To some extent, it actually recommends the second-hand information for a target user since CF can only recommend articles when other users have read them. To overcome this CF deficiency for new articles, content-based filtering (CBF) approach [43] finds articles per the semantic similarity. In fact, users often read an article because they are attracted by a very small point, e.g., the name of a person in a news, an interesting keyword in a paper or some touching words in a song. However, current CBF approaches cannot interpret the most attractive point leading to user selection. Besides interpreting recommendation, we propose to model and infer the attraction on content in this chapter. Then, what is the specialty of attraction modeling? First, the attraction is the highlights that largely lead to a person's selection and decision. For example, we often cannot recite a whole poem but we can always recall some impressive sentences; similarly, we may not remember a whole song but we can hum some touching lyrics. These highlights make a person to be attracted by the poem or the song. Second, the attraction is a subjective

feeling which is often different from person to person. For example, as illustrated in Figure 9.1, readers in Go community may be attracted by the target problem, i.e., Go playing, of this scientific paper while readers in AI community may be attracted by the technical methods. In this case, finding the attractive points is obtaining the strong evidence to interpret why a person likes an article.

Attention mechanism has been shown effective in various tasks such as machine translation [8] and image captioning [221]. Its underlying assumption is that one only focuses on selective parts where as needed. Obviously, attention mechanism shares some common assumption with attraction modeling. Yang et al. [242] proposed hierarchical attention networks for document classification, where two levels of attention mechanisms are respectively applied at the word and sentence level, enabling it to attend to more or less important content when constructing the document representation. Denil et al. [46] use CNN to transform word embeddings in each sentence into an embedding for the entire sentence. At the document level, another CNN is used to transform sentence embeddings into a document embedding vector. These methods try to find salient words or sentences from documents without considering personal factors. However, these attentive words and sentences do not mean the attractive words and sentences to all users since each user may have quite different backgrounds. Therefore, the main difference is that attention mechanism puts focus on salient parts of a content in an objective way ignoring user difference, whereas attraction modeling tries to capture user's subjective focus on content elements from a personal view with considering user features.

In fact, attraction modeling has many potential application areas. For example, users may be unable to read hundreds of papers of a conference. Then, attraction modeling can serve as a personal machine reading assistant for pre-reading articles for each user and undertake personalized filtering. Meanwhile, attraction models can serve as a text extractor to digest sentences which are most attractive to a user.

As a result, users can have a quick selection in terms of scanning the attractive sentences provided by the attraction model only in a small set of filtered articles.

To formally implement attraction modeling, we design a framework to quantitatively score the user’s attractiveness over content elements. Enhancing with this framework, content-based RSs are able to provide interpretation on user behavior and recommendation results. In this chapter, we study two representative attraction models, which respectively model two major types of content on the internet. First, we study the multilevel attraction, i.e. word-level, sentence-level and document-level attraction, over textual content, e.g. posts and news articles. As a result, we design a hierarchical attraction model (HAM) to capture user’s multilevel attraction. Second, we take movies as the representative case to study multimodal attraction modeling since internet movies have accounted for the major traffic in this new media age. We know that the story and the cast members, e.g., actors, directors, and writers, are two most important aspects of a movie to attract the audience. We build a multimodal attraction model (MAM) to comprehensively capture user attraction over both story content and cast content.

9.2 Attraction Model

Attraction model is the key component to detect and interpret users’ preferences. Before integrating the attraction model into RSs, we first present its architecture without associating with some specific recommendation tasks.

For modeling attraction, two basic elements are needed to model: (1) the content element set of a target object and (2) the filter to capture user’s attraction on content. Before introducing the details of model architecture, we define basic notations in attraction modeling. We denote $\mathcal{O} = \{o_1, \dots, o_{|\mathcal{O}|}\}$ as the object set, and $\mathcal{W}_o = \{w_{o,1}, \dots, w_{o,|\mathcal{W}_o|}\}$ is the content element set of an object $o \in \mathcal{O}$. For example, \mathcal{W}_o denotes the word sequence of sentence o . The user set is denoted as

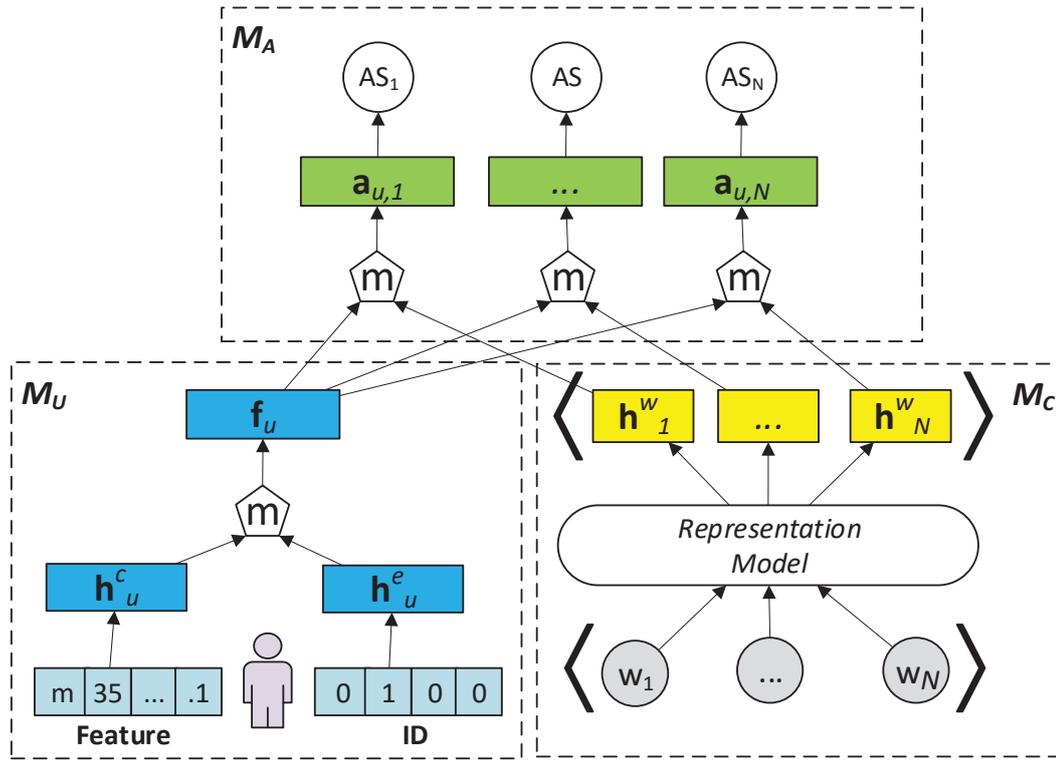


Figure 9.2 : Architecture of Attraction Model

$\mathcal{U} = \{u_1, \dots, u_{|\mathcal{U}|}\}$. Given a user $u \in \mathcal{U}$, we denote $\mathcal{F}_u = \{f_{u,1}, \dots, f_{u,\mathcal{F}_u}\}$ as u 's related features, e.g. personal demographics and/or some contextual features. Given the content element set \mathcal{W}_o of an object o , the goal of attraction model is to score the attractiveness over each $w_{o,i} \in \mathcal{W}_o$.

The architecture of attraction model is illustrated in Figure 9.2, which consists of three main modules: M_U : Attraction Filter module, M_C : Content Representation module and M_A : Attractiveness Score module.

9.2.1 Content Representation Module

As illustrated in Figure 9.2 (M_C), this module focus on modeling the representations of the content elements. Given the content element set \mathcal{W}_o of an object o ,

we feed them into a representation model to obtain corresponding high-level representations $\{\mathbf{h}_{o,1}^W, \dots, \mathbf{h}_{o,N}^W\}$.

For example, we can use a pre-trained word embedding model [171] as the representation model to map the words of a sentence into the corresponding word embeddings. Given an image, we can use CNN to detect the objects in this image and output their feature maps [174].

9.2.2 Attraction Filter Module

As illustrated in Figure 9.2 (M_A), this module focus on modeling the personal filter of user attraction over the content. First, we give the formal definition of *Attraction Filter*.

Definition 9.1: (Attraction Filter) An attraction filter is a vector \mathbf{f}_u , which is used to differentiate user u 's attraction over the representation vector $\mathbf{h}_{o,i}^W$ of each content element $w_{o,i} \in \mathcal{W}_o$.

In general, an attraction filter reflects the user's preferred focus on content, which is relevant to user features, e.g. age, gender, and occupation. We denote all these observed features as \mathcal{F}_u . However, not all user features can be collected, for example, due to the reason of privacy. Therefore, we associated with a latent feature vector, \mathbf{h}_u^E , to represent the user's observed features. \mathbf{h}_u^E is often called user embedding in neural network models. To capture the high-level semantics and coupling relationship between observed features, we often use generate a continuous vector representation for \mathcal{F}_u . More specifically, $\mathbf{h}_u^F = r_{\mathbf{W}}(\mathcal{F}_u)$, where $r_{\mathbf{W}}(\cdot)$ specifies some representation model with the parameters \mathbf{W} . Typically, $r_{\mathbf{W}}(\cdot)$ is modeled by a multilayer perceptron (MLP) [98].

Based on \mathbf{h}_u^F and \mathbf{h}_u^E , we can construct the attraction filter vector \mathbf{f}_u of user u .

Typically, it has the following form:

$$\mathbf{f}_u = f_{\mathbf{W}}(\mathbf{h}_u^F \oplus \mathbf{h}_u^E) \quad (9.1)$$

where \oplus denotes some merge operation over \mathbf{h}_u^F and \mathbf{h}_u^E , e.g. concatenation, plus or element-wise product, $f_{\mathbf{W}}$ is a function with the parameters \mathbf{W} , which can be typically modeled by MLP, for example, $\mathbf{f}_u = a(\mathbf{W}[\mathbf{h}_u^F \oplus \mathbf{h}_u^E])$ defines a two-layer neural network where $a(\cdot)$ is an activation function.

9.2.3 Attractiveness Score Module

To capture user's attraction, we build an attractiveness score module to weight the interaction strength, i.e. attractiveness, between attraction filter \mathbf{f}_u and content element representations $\{\mathbf{h}_{o,1}^W, \dots, \mathbf{h}_{o,N}^W\}$. As illustrated in Figure 9.2 (M_A), the attraction representation $\mathbf{a}_{u,i}$ is modeled by the following function:

$$\mathbf{a}_{u,i} = g_{\mathbf{W}}(\mathbf{f}_u \otimes \mathbf{h}_{o,i}^W) \quad (9.2)$$

where \otimes denotes some operation between \mathbf{f}_u and $\mathbf{h}_{o,i}$, e.g. concatenation or element-wise product, $g_{\mathbf{W}}$ defines a function with the parameters \mathbf{W} , which is typically modeled by MLP, namely, $\mathbf{a}_{u,i} = MLP_{\mathbf{W}}(\mathbf{f}_u \otimes \mathbf{h}_{o,i}^W)$.

So far we obtain the attraction representation $\mathbf{a}_{u,i}$ which reflects user u 's attraction on the content element $w_{o,i}$. Then, we use the following function to quantify the attractiveness score based on $\mathbf{a}_{u,i}$.

$$AS_{u,i} = s_{\mathbf{w}}(\mathbf{a}_{u,i}) \quad (9.3)$$

where $s_{\mathbf{w}}(\cdot)$ is a score function map the attraction representation to a scalar value, i.e. $AS_{u,i} \in \mathbb{R}$. Typically, $s_{\mathbf{w}}(\mathbf{a}_{u,i}) = \mathbf{w}^T \mathbf{a}_{u,i} + b$ is a linear function, where $\mathbf{w} \in \mathbb{R}^A$ and b is the bias.

The normalized attractiveness score can be computed by

$$\bar{A}S_{u,i} = \text{softmax}(AS_{u,i}) = \frac{e^{AS_{u,i}}}{\sum_j e^{AS_{u,j}}} \quad (9.4)$$

However, $AS_{u,i}$ in Eq. 9.3 can be arbitrarily large, which makes softmax to output a weight close to 1 on the maximum $AS_{u,i}$. Furthermore, a large $AS_{u,i}$ easily makes the exponential function overflow in implementation. To resolve this problem, we impose an inverse squared root function to bound the value of $AS_{u,i}$ in this work.

$$\text{isr}^\alpha(x) = \frac{x}{\sqrt{1 + \alpha x^2}} \quad (9.5)$$

We can easily verify that $\text{isr}(x)$ has the range $(-\alpha^{-\frac{1}{2}}, \alpha^{-\frac{1}{2}})$, so the parameter α can be used to control the upper bound and lower bound. A large α makes the upper bound and lower bound close to 0. As a result, the softmax tends to output uniform weights. A small α around 0.001 has the range $(-31.6, 31.6)$ which guarantees the exponential function not to overflow and softmax tends to output a single large weight. Then we have the following adjusted attractiveness score:

$$\bar{AS}_{u,i} = \text{softmax}(\text{isr}^\alpha(AS_{u,i})) \quad (9.6)$$

The attractiveness scores statistically quantify the user’s attraction on the given content elements. These scores enable to find the most attractive content elements and interpret the attractive points that result in user selection. Therefore, we can instantiate this attraction modeling framework to build interpretable content-based RSs.

9.3 Hierarchical Attraction Model on Textual Content

In the previous section, we have presented the framework of attraction model. In this section, we focus on applying the attraction model on textual content. Note that the term “article” is used to refer to textual content, e.g., a post, a news item, or an academic paper.

When a person is reading an article, s/he may be caught by some attractive words in a sentence, e.g., a person’s name or an interesting keyword. Moreover,

a person often likes an article due to a few attractive sentences instead of every sentence. Obviously, this implies a hierarchical attraction modeling on an article, from the word-level attraction to the sentence-level attraction, and to the document-level attraction. According to these analyses, we build a hierarchical attraction model (HAM) to capture personal attraction on textual content. However, the data in the real world often has a long-tail distribution, for example, most users only associate with very few observed choices on articles (i.e., users in the long tail). With insufficient data, we cannot learn user preference and infer the attractive points for them. To relieve this deficiency, we propose to learn personal attraction regularized by group attraction.

9.3.1 Multilevel Attraction

We denote $\mathcal{D} = \{d_1, \dots, d_N\}$ as an article set. For each article $d \in \mathcal{D}$, it consists of a sequence of N_d sentences, $\mathcal{S}_d = \{s_1, \dots, s_{N_d}\}$. For each sentence $s \in \mathcal{S}_d$, it consists of N_s sequential words, $\{w_{s,1}, \dots, w_{s,N_s}\}$. We denote the user set as \mathcal{U} to model their attraction on articles, and \mathcal{G} as all user groups. In particular, each user belongs to a user group $g \in \mathcal{G}$. Then, the users in group g is denoted as $\mathcal{U}_g = \{u_{g,1}, \dots, u_{g,N_g}\}$. Given a user $u \in \mathcal{U}_g$, his user profile about previously liked articles is denoted $\mathcal{D}_u = \{d_{u,1}, \dots, d_{u,N_u}\}$. Given an article liked by $d_{u,i} \in \mathcal{D}_u$, the goal of this work is to capture personal word-level attraction, sentence-level attraction and document-level attraction for each user.

9.3.2 Bidirectional Gated Recurrent Units

GRU [36] are a gating mechanism in RNN. It combines the forget and input gates into a single “update gate” and merges the cell state and hidden state. The resulting model is simpler than standard LSTM [80] models, and has been growing increasingly popular. The architecture of GRU and the corresponding update rule

is given as follows:

$$\begin{aligned} \mathbf{z}_t &= \sigma(\mathbf{W}_z[\mathbf{h}_{t-1}, \mathbf{x}_t]) \\ \mathbf{r}_t &= \sigma(\mathbf{W}_r[\mathbf{h}_{t-1}, \mathbf{x}_t]) \\ \tilde{\mathbf{h}}_t &= \tanh(\mathbf{W}[\mathbf{r}_t * \mathbf{h}_{t-1}, \mathbf{x}_t]) \\ \mathbf{h}_t &= (1 - \mathbf{z}_t) * \mathbf{h}_{t-1} + \mathbf{z}_t * \tilde{\mathbf{h}}_t \end{aligned}$$

where $\sigma(x) = 1/(1 + \exp(-x))$ is sigmoid function, \mathbf{z}_t denotes update gate vector, \mathbf{r}_t denotes reset gate vector, \mathbf{x}_t denotes input vector and \mathbf{h}_t denotes output vector, $\{\mathbf{W}_z, \mathbf{W}_r, \mathbf{W}, \}$ are the weight matrices.

Bidirectional GRU (BiGRU) [36] consists of two GRU-based recurrent neural networks, where the forward \overrightarrow{GRU} outputs a vector, $\overrightarrow{\mathbf{h}}_i^W$, with preserving the preceding subsequence information while the backward \overleftarrow{GRU} outputs a vector, $\overleftarrow{\mathbf{h}}_i^W$, with containing the successive subsequence information.

$$\overrightarrow{\mathbf{h}}_i^W = \overrightarrow{GRU}(\mathbf{w}_i), \quad i \in \{1, \dots, N_s\} \quad (9.7)$$

$$\overleftarrow{\mathbf{h}}_i^W = \overleftarrow{GRU}(\mathbf{w}_i), \quad i \in \{N_s, \dots, 1\} \quad (9.8)$$

Then, $\overrightarrow{\mathbf{h}}_{s,i}^W$ and $\overleftarrow{\mathbf{h}}_{s,i}^W$ can be combined together as a joint representation, e.g. $\mathbf{h}_{s,i}^W = [\overrightarrow{\mathbf{h}}_{s,i}^W, \overleftarrow{\mathbf{h}}_{s,i}^W]$ through the concatenation.

9.3.3 Model Architecture

The architecture of HAM is illustrated in Figure 9.3, where we build a hierarchical model to capture user attraction on different levels of an article, from the words to the sentences, and then to the document.

Different from the current hierarchical document representation [242] without considering subjective factors, our model aims to generate personally attraction-based document representation by considering the difference of attraction between persons. Especially, we design a bottom-up hierarchical attraction model (HAM). It

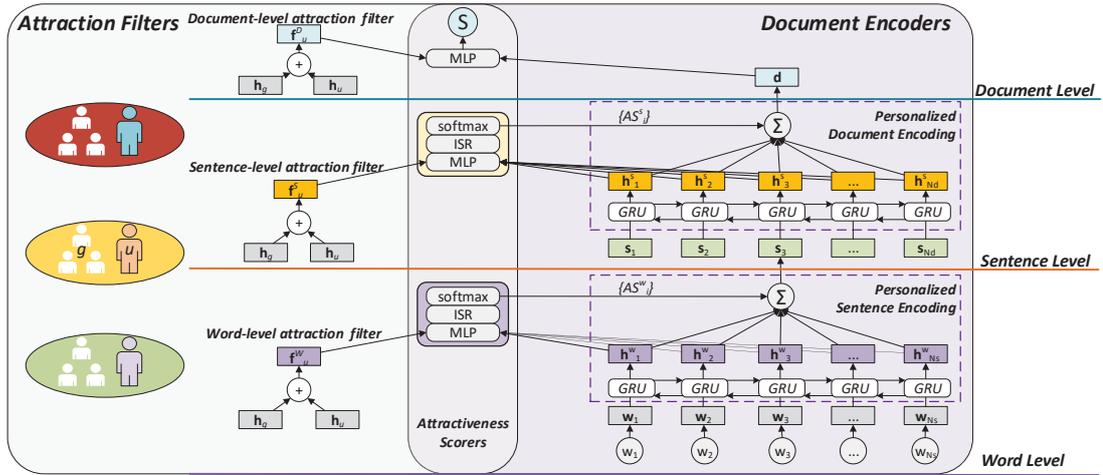


Figure 9.3 : The architecture of hierarchical attraction model

first scores the attractiveness of each word in a sentence and then encodes a sentence based on these attraction-biased word representations. Recursively, HAM encodes a document based on the generated sentence representations weighted by their attractiveness. More specifically, HAM first generates personally attraction-based sentence embeddings by aggregating word embeddings weighted by the word-level attraction filters (as shown in the bottom of Figure 9.3), and then it produces a personally attraction-based document embedding based on the personally attraction-based sentence embeddings (as shown in the middle of Figure 9.3). With the attractiveness scores, this attraction-based encoding process can be employed to analyze personal attraction over the content elements of each level. In the following subsections, we present more technical details about this model.

9.3.4 Multilevel Attraction Filters

To capture user attraction over the content elements at each level, we correspondingly create the following attraction filters, as illustrated in the left part of Figure 9.3.

- **Word-level attraction filter:** Given a sequence of words of a sentence s in an article $d_{u,i}$, we use a word-level attraction module to score personal attraction on each word $w \in s$ in terms of learning a user attraction filter $\mathbf{f}_u^W \in \mathbb{R}^L$ over the word embedding $\mathbf{w} \in \mathbb{R}^L$.
- **Sentence-level attraction filter:** Given a sequence of sentences in an article $d_{u,i}$, we use a sentence-level attraction module to score personal attraction on each sentence $s \in d_{u,i}$ in terms of learning a user attraction filter $\mathbf{f}_u^S \in \mathbb{R}^K$ over the sentence embedding $\mathbf{s} \in \mathbb{R}^K$.
- **Document-level attraction filter:** Given a collection of candidate articles \mathcal{D}_c , we use a document-level attraction module to score user preference on each article $d \in \mathcal{D}_c$ in terms of learning a user attraction filter $\mathbf{f}_u^d \in \mathbb{R}^M$ over the document embedding $\mathbf{d} \in \mathbb{R}^M$.

In consideration of the real-world data distribution, most data often follow power-law or long-tail distribution [85]. That is, the majority of users are often associated with few data and only the minority of users have sufficient data. Without sufficient data, a model may fail to learn users' attraction. To relieve this deficiency, we incorporate group-level features [84] when modeling user attraction filter. In particular, we can associate each user with a predefined group. Normally, there are several ways to group users: (1) groups can be assigned according to some observable user features, e.g., occupations, when we want to study the group preference about that feature; (2) we can generate groups according to social networking, e.g., coauthoring network, when social influence is heavy on the users' choice; (3) we can cluster users into groups according to the similarity on observed data.

Given user u , we treat his/her assigned group g as the observed feature of u , i.e. $\mathcal{F}_u = \{g\}$, as defined in Section 9.2.1. As presented in Section 9.2.2, we have the user embedding \mathbf{h}_u^E , and a group embedding \mathbf{h}_g^F can be associated with group

g. Then, we apply the plus operation to merge \mathbf{h}_g^F and \mathbf{h}_u^E . As a result, we obtain the following:

$$\mathbf{h}_u = \mathbf{h}_g^F + \mathbf{h}_u^E \quad (9.9)$$

where \mathbf{h}_u^E is regularized by L2-norm $\lambda \|\mathbf{h}_u^E\|_2^2$. Therefore, \mathbf{h}_u serves as a group-regularized user embedding. The regularization parameter controls the shrinkage of \mathbf{h}_u^E toward zeros, i.e., it controls how close is \mathbf{h}_u to \mathbf{h}_g^F . A very small λ places tiny regularization on \mathbf{h}_u^E so \mathbf{h}_u can have arbitrarily large values, namely it bypasses the effect of \mathbf{h}_g^F . On the contrary, a very large λ tightly shrinks \mathbf{h}_u^E to zeros, therefore \mathbf{h}_u tends to be fully determined by \mathbf{h}_g^F .

As a result, we create three attraction filters, \mathbf{f}_u^W to model user attraction on words, \mathbf{f}_u^S to model user attraction on sentences, and \mathbf{f}_u^D to model user attraction on documents, as shown in the left part of Figure 9.3.

$$\mathbf{f}_u^W = \text{ReLU}(\mathbf{W}^W \mathbf{h}_u + \mathbf{b}^W) \quad (9.10)$$

$$\mathbf{f}_u^S = \text{ReLU}(\mathbf{W}^S \mathbf{h}_u + \mathbf{b}^S) \quad (9.11)$$

$$\mathbf{f}_u^D = \text{ReLU}(\mathbf{W}^D \mathbf{h}_u + \mathbf{b}^D) \quad (9.12)$$

where $\{\mathbf{W}^W, \mathbf{W}^S, \mathbf{W}^D\}$ are weight matrices of the rectifier neural networks and $\{\mathbf{b}^W, \mathbf{b}^S, \mathbf{b}^D\}$ are the corresponding bias vectors.

9.3.5 Hierarchical Attractive Content Encoder

Encoding Sentence via Attractive Words

Scoring attractiveness over words: Given a sequence of words $\{w_{s,1}, \dots, w_{s,N_s}\}$ of a sentence s in article $d_{u,i}$ liked by user u , we aim to find the most attractive words to u , and then we can encode a personally attraction-based sentence representation based on the word embedding vectors weighted by the attractiveness scores over them.

Given a sentence, we map each word $w_{s,i}$ into a word embedding \mathbf{w}_i as illustrated in the bottom-right part of Figure 9.3. Then, we feed this word embedding sequence $\{\mathbf{w}_1, \dots, \mathbf{w}_{N_s}\}$ into a BiGRU-based RNN. The forward \overrightarrow{GRU} outputs a recurrent word embedding with preserving the preceding subsequence information while the backward \overleftarrow{GRU} outputs a recurrent word embedding with containing the successive subsequence information.

$$\overrightarrow{\mathbf{h}}_i^W = \overrightarrow{GRU}(\mathbf{w}_i), \quad i \in \{1, \dots, N_s\} \quad (9.13)$$

$$\overleftarrow{\mathbf{h}}_i^W = \overleftarrow{GRU}(\mathbf{w}_i), \quad i \in \{N_s, \dots, 1\} \quad (9.14)$$

The joint recurrent word embedding is set as the concatenation of $\overrightarrow{\mathbf{h}}_i^W$ and $\overleftarrow{\mathbf{h}}_i^W$, i.e. $\mathbf{h}_i^W = [\overrightarrow{\mathbf{h}}_i^W, \overleftarrow{\mathbf{h}}_i^W]$.

Then, we feed the word-level attraction filter \mathbf{f}_u^W and each word embedding \mathbf{h}_i^W into the attractiveness score module. In this module, the attraction representation is modeled by a tanh neural network.

$$\mathbf{a}_i^W = \tanh(\mathbf{W}^W[\mathbf{f}_u^W, \mathbf{h}_i^W] + \mathbf{b}^W) \quad (9.15)$$

Accordingly, the normalized attraction weights are give as follows:

$$\bar{A}S_i^W = \text{softmax}(\text{isr}^{\alpha=4}(\mathbf{w}^{W\top} \mathbf{a}_i^W + b^W)) \quad (9.16)$$

where we set $\alpha = 4$ which can achieve good performance in our experiments.

Attraction-based sentence encoding: The sentence representation should preserve the most information from attractive words to user u in a sentence. Since the $\{\bar{A}S_i^W\}$ weights the personal attractiveness, the sentence embedding \mathbf{s} is encoded as the mixture of recurrent word embeddings weighted by their attractiveness:

$$\mathbf{s} = \sum_{i=1}^{N_s} \bar{A}S_i^W \mathbf{h}_i^W \quad (9.17)$$

Encoding Document via Attractive Sentences

The attraction-based sentence embedding for all sentences of article $d_{u,i}$ can be obtained from the above sentence encoding module. Then, we can build a document encoding module on these sentence embedding vectors. As shown in the middle-right part of Figure 9.3, the architecture of personally attraction-based document encoding module is similar to the sentence encoding module.

Scoring attractiveness over sentences: We input the sequence of attraction-based embedding $\{\mathbf{s}_1, \dots, \mathbf{s}_{N_d}\}$ for all sentences into BiGRU networks to generate a recurrent sentence embedding which preserves the information of both preceding sentence sequence and successive sentence sequence.

$$\overrightarrow{\mathbf{h}}_i^S = \overrightarrow{GRU}(\mathbf{s}_i), \quad i \in \{1, \dots, N_d\} \quad (9.18)$$

$$\overleftarrow{\mathbf{h}}_i^S = \overleftarrow{GRU}(\mathbf{s}_i), \quad i \in \{N_d, \dots, 1\} \quad (9.19)$$

$\mathbf{h}_i^S = [\overrightarrow{\mathbf{h}}_i^S, \overleftarrow{\mathbf{h}}_i^S]$ denotes the concatenated recurrent sentence embedding.

Given the sentence-level attraction filter, \mathbf{f}_u^S , and recurrent sentence embedding sequence $\{\mathbf{h}_1^S, \dots, \mathbf{h}_{N_d}^S\}$. The normalized attraction weights are computed over each sentence attraction representation \mathbf{a}_i^S :

$$\mathbf{a}_i^S = \tanh(\mathbf{W}^S[\mathbf{f}_u^S, \mathbf{h}_i^S] + \mathbf{b}^S) \quad (9.20)$$

$$\bar{A}S_i^S = \text{softmax}(\text{isr}^{\alpha=1}(\mathbf{w}^{S\top} \mathbf{a}_i^S + b^S)) \quad (9.21)$$

Attraction-based document embedding: Similar to sentence encoding with the most attractive words, the document representation should preserve the information of attractive sentences. As a result, we build the attraction-based document embedding are encoded with the recurrent sentence embeddings weighted per their attractiveness:

$$\mathbf{d} = \sum_{i=1}^{N_d} \bar{A}S_i^S \mathbf{h}_i^S \quad (9.22)$$

Document Attractiveness Scoring

Measuring the document-level attractiveness over all documents is similar to attraction scoring over words and sentences. Given the document-level group-regularized user attraction filter, \mathbf{f}_u^D , and the candidate document embeddings $\{\mathbf{d}_{u,1}, \dots, \mathbf{d}_{u,N}\}$, the attractiveness scores on each article d_i can be computed by:

$$\mathbf{a}_i^D = \tanh(\mathbf{W}^D[\mathbf{f}_u^D, \mathbf{h}_i^D] + \mathbf{b}^D) \quad (9.23)$$

$$S_{d_{u,i}} = \mathbf{w}^{D\top} \mathbf{a}_i^D + b^D \quad (9.24)$$

9.3.6 Objectives and Parameter Learning

Loss for Explicit Preference Data

When users provide explicit feedback to differentiate their preferences on content, e.g., like or dislike on an article, we can treat it as a classification problem. Typically, we consider the binary preference label, i.e., like/dislike, on articles, and then the loss function is given as the binary cross entropy:

$$L(y_{u,i}, \hat{y}_{u,i}) = -y_{u,i} \log(\hat{y}_{u,i}) - (1 - y_{u,i}) \log(1 - \hat{y}_{u,i}) \quad (9.25)$$

where $\hat{y}_{u,i} = \sigma(S_{d_{u,i}})$ and $\sigma(\cdot)$ is the logistic function, $y_{u,i}$ denote the true labels, i.e., $y_{u,i} = 1$ for like and $y_{u,i} = 0$ for dislike.

Loss for Implicit Preference Data

In many real-world scenarios, explicit preference data are not available; instead, browsing record, click logs, are much more easily obtained. In such cases, we only have one-class preference data [87], which cannot be applied with a classification loss for optimization. To learn with one-class preference data, it is often treated as ranking problem [178]. More specifically, we construct contrastive pairs to specify the preference order, that is, we have the order $d_{u,i} \succeq d_{u,j}$ over a u liked article

Algorithm 3 The mini-batch learning for classification

- 1: $\mathcal{B} \leftarrow \text{GetMinibatch}(\{\mathcal{M}_u\})$ from all user-article pairs
 - 2: $\{S_{d_{u,i}}\} \leftarrow \text{Compute scores for } d_{u,i} \in \mathcal{B}$ (cf. Eq. 9.24)
 - 3: Compute mean mini-batch loss using Eq. 9.25:
 - 4: $L_{\mathcal{B}} \leftarrow \frac{1}{|\mathcal{B}|} \sum_{\{S_{d_{u,i}}\}} L(y_{u,i}, \hat{y}_{u,i})$
 - 5: Update parameters: $\Theta \leftarrow \Theta - \Gamma_{Adam}(\nabla_{\Theta} L_{\mathcal{B}})$
-

$(d_{u,i} \in \mathbf{D}_u)$ and a randomly selected article $(d_{u,j} \notin \mathbf{D}_u)$. Then, we use the following loss to optimize ranking:

$$L(d_{u,i} \succeq d_{u,j}) = \max(0, m + S_{d_{u,j}} - S_{d_{u,i}}) \quad (9.26)$$

This loss function is a variant of the hinge loss named max-margin loss [121], where m is a parameter to define the maximum margin and we find that $m = 20$ produces good results in our experiments.

Implementation and Training Procedure

We implemented HAM using Keras [37] with the backend of Tensorflow GPU version. We initialize the word embeddings for each article with the pre-trained GloVe vectors [171]. Due to the limited space, we only list a brief scheme of the training procedure in Algorithm 3 for classification and Algorithm 4 for ranking on a mini-batch. For the gradient descent optimizer, we adopt $\Gamma_{Adam}(\cdot)$.

9.4 Multimodal Attraction Model on Movies

In the previous section, we present the hierarchical attraction modeling, which is suitable to study user attraction on textual content, such as post, news, and academic papers. In some other cases, users may be attracted by multiple types of content of an object. For example, the story and the cast members, e.g., actors, directors, and writers, are two most important aspects of a movie to attract the

Algorithm 4 The mini-batch learning for ranking

- 1: $\mathcal{B}_p \leftarrow \text{GetMinibatch}(\{\mathcal{M}_u\})$ from all user-movie pairs
 - 2: $\mathcal{B}_n \leftarrow \text{Sample contractive article } d_{u,j} \text{ for each } d_{u,i} \in \mathcal{B}_p$
 - 3: $\{\langle S_{d_{u,i}}, S_{d_{u,j}} \rangle\} \leftarrow \text{Compute score pairs for each}$
 - 4: $\langle d_{u,i}, d_{u,j} \rangle \in \langle \mathcal{B}_p, \mathcal{B}_n \rangle$ (cf. Eq. 9.24)
 - 5: Compute mean mini-batch loss using Eq. 9.26:
 - 6: $L_{\langle \mathcal{B}_p, \mathcal{B}_n \rangle} \leftarrow \frac{1}{|\langle \mathcal{B}_p, \mathcal{B}_n \rangle|} \sum_{\{\langle S_{d_{u,i}}, S_{d_{u,j}} \rangle\}} L(d_{u,i} \succeq d_{u,j})$
 - 7: Update parameters: $\Theta \leftarrow \Theta - \Gamma_{Adam}(\nabla_{\Theta} L_{\langle \mathcal{B}_p, \mathcal{B}_n \rangle})$
-

audience. As a result, we need to jointly model the user attraction over two content element sets w.r.t. story and cast.

The internet video stream has accounted for the major traffic in this new media age. In this section, we take movies as the representative case to study multimodal attraction modeling. On one hand, when a person reads the story of a movie, s/he may be caught by some attractive words in a sentence. Moreover, only a few sentences of the core plot instead of all sentences may actually attract the user’s attention. Accordingly, we build a hierarchical attraction model on the story (textual content) to capture word-level, sentence-level, and story-level attraction. On the other hand, cast members (categorical content) of a movie are another important factor to attract users so we build another attraction model to weight the attractiveness over each cast member and generate a representation of the whole cast. At the top level, we create a joint representation of story and cast representation to score attractiveness. Due to the complementation of story (textual data) and cast members (categorical data), we build a multimodal attraction model (MAM) to integrate the information from both types of content data to comprehensively capture user attraction.

9.4.1 Multimodal Attraction

We denote the movie set as $\mathcal{M} = \{m_1, \dots, m_N\}$. For each movie $m \in \mathcal{M}$, it consists of a textual story, \mathcal{S}_t , and a set of cast members $\mathcal{C}_m = \{c_1, \dots, c_{N_m}\}$. For each story \mathcal{S}_t , it consists of N_t sentences, $\mathcal{S}_t = \{s_1, \dots, s_{N_t}\}$. For each sentence $s \in \mathcal{S}_t$, it consists of a set of N_s words, $\{w_{s,1}, \dots, w_{s,N_s}\}$. We denote the user set as \mathcal{U} to model their attraction on movies. Given a user $u \in \mathcal{U}$, her user profile about previously liked movies is denoted $\mathcal{M}_u = \{m_{u,1}, \dots, m_{u,N_u}\}$.

Given a movie $m \in \mathcal{M}_u$, one of our tasks is to learn the attractiveness over words for each sentence, and the attractiveness over sentences of the story \mathcal{S}_t from user u 's perspective and generate story-level representation \mathbf{h}^T . Another task is to weight the attractiveness on the cast members \mathcal{C}_m and generate attraction-based cast-level representation \mathbf{h}^C . Then, we can use \mathbf{h}^T and \mathbf{h}^C to score the attractiveness on movie m . When the parameters of attraction model are learned, we can compute personal attractiveness scores over a set of candidate movies \mathcal{M}_C for recommendation and interpret the recommendation with the highlighted sentences of the story and which actors may attract the target user.

9.4.2 Model Architecture

The overview of the architecture of MAM is illustrated in Figure 9.4. This model consists of three main parts: cast content (left), attraction filters (middle) and story content (right). Since movie story is textual content, we build a hierarchical attraction model, similar to the model introduced in the previous section, to score attractiveness over words and sentences for each user. Moreover, we build another attraction model to score attractiveness over cast members. Finally, we compute the personal attractiveness score of a movie on the top of these modules using the cast-level embedding and the story-level embedding.

Different from the HAM presented in the previous section to model a sentence in

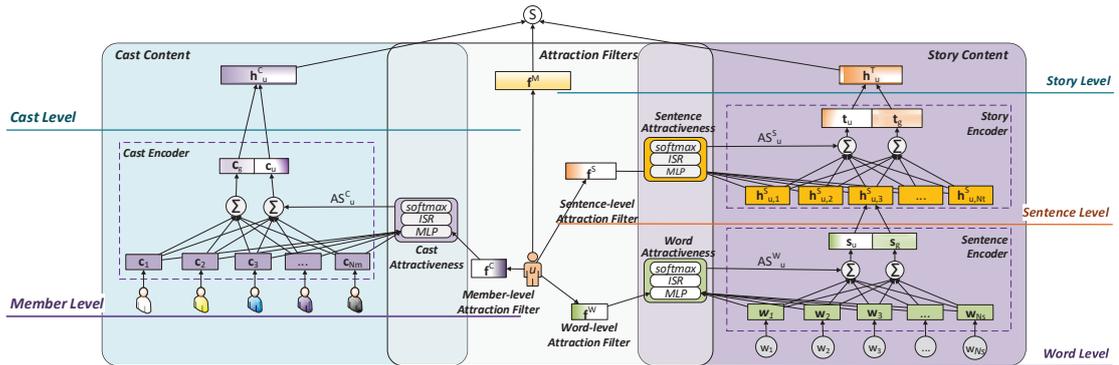


Figure 9.4 : The architecture of hierarchical attraction model over movies with two modalities: Cast (left) and Story (right)

terms of a sequence of words with BiGRU-based RNN, we do not assume the sequential order over words. Instead, we assume the bag-of-words (BoW) model over the words in a sentence without considering their order. Under such an assumption, we can construct similar attraction modeling on both textual content and cast content by treating both of them as a set of unordered content elements. In the following subsections, we give more technical details about these modules.

9.4.3 Story Attraction Module

Sentence Encoder with Word Attraction Filters

Given a set of words $\{w_1, \dots, w_{N_s}\}$ of a sentence s in story \mathcal{S}_t of movie $m_{u,i}$ liked by user u , we aim to score the attractiveness over words from u 's perspective. First, we map each word w_i into a word embedding vector \mathbf{w}_i . Then, we feed these word embedding vectors $\{\mathbf{w}_1, \dots, \mathbf{w}_{N_s}\}$ into the word-level attractiveness score module (illustrated in the overlapped part of user module and story attraction module in Figure 9.4). We use \mathbf{h}_u^E to denote the user embedding of u . In this model, we do not consider additional explicit user features. As a result, we obtain the word-level

user attraction filter:

$$\mathbf{f}_u^W = \text{ReLU}(\mathbf{W}^W \mathbf{h}_u^E + \mathbf{b}^W) \quad (9.27)$$

In MAM, we simply compute the attractiveness score $AS_{u,i}^W$ in terms of the inner product between \mathbf{f}_u^W and each word embedding \mathbf{w}_i

$$AS_{u,i}^W = \mathbf{f}_u^{W\top} \mathbf{w}_i \quad (9.28)$$

Accordingly, the normalized attractiveness score on word w_i is

$$\bar{A}S_{u,i}^W = \text{softmax}(\text{isr}^{\alpha=4}(AS_{u,i}^W)) \quad (9.29)$$

$\alpha = 4$ performs good in our experiments.

Then, we can obtain the attraction-based sentence embedding \mathbf{s}_u by weighted sum over word embedding vectors using $\bar{A}S_{u,i}^W$.

$$\mathbf{s}_u = \sum_{i=1}^{N_s} \bar{A}S_{u,i}^W \mathbf{w}_i \quad (9.30)$$

In HAM, we use BiGRU to model word sequence, the recurrent word embedding preserve preceding and successive information (cf. Eqs. 9.13 and 9.14). In comparison, \mathbf{s}_u in this model only encodes the information from the most attractive words and discards the information from unattractive ones due to the BoW assumption. If a long sentence only contains one attractive word, the word-level attraction filter assigns a large weight on this word to encode the whole sentence, even though this sentence is not attractive as a whole. If such sentence embeddings are passed to the upper level, the sentence-level attraction filter cannot differentiate which sentences are more attractive due to the loss of information from unattractive words in the original sentences. To measure the overall attractiveness of the sentence s at the sentence level, we need to preserve the major information of a sentence apart from the most attractive part encoded by \mathbf{s}_u . Therefore, we use another *neutral-attraction* filter \mathbf{g}^W to extract the major information over all words.

$$\bar{A}S_i^W = \text{softmax}[\text{isr}^{\alpha=32}(\mathbf{g}^{W\top} \mathbf{w}_i + b^W)] \quad (9.31)$$

The filter weights are computed similar to user attractiveness (cf. Eq. 9.29), where the major difference is that the weight vector \mathbf{g} and the bias b^W are user independent and α of isr is set to 32 to keep relatively uniform information for all words. As a result, the *neutral-attraction* sentence embedding \mathbf{s}_g is encoded:

$$\mathbf{s}_g = \sum_{i=1}^{N_s} \bar{A}S_i^W \mathbf{w}_i \quad (9.32)$$

Then, we concatenate the attraction-encoded sentence embedding and the neutral-attraction sentence embedding, $[\mathbf{s}_u, \mathbf{s}_g]$, and input it into the tanh neural network layer to jointly encode \mathbf{s}_u and \mathbf{s}_g into a comprehensive sentence encoding \mathbf{h}_u^S with the parameters \mathbf{W}^S and \mathbf{b}^S

$$\mathbf{h}_u^S = \tanh(\mathbf{W}^S[\mathbf{s}_u, \mathbf{s}_g] + \mathbf{b}^S) \quad (9.33)$$

Story Encoder with Sentence Attraction Filters

Once we obtain the attraction-based sentence embeddings (cf. Eq. 9.22) for all sentences of the story \mathcal{S}_t from the sentence encoder as presented above, we can build a story encoder over these sentence embedding vectors at the sentence level. As shown in the right part of Figure 9.4, the structure of attraction-based story encoder is very similar to the sentence encoder. Therefore, we briefly introduce this story encoder in this subsection.

Given the sentence-level attraction filter, \mathbf{f}_u^S , induced from user embedding \mathbf{h}_u^E , the attractiveness scores over each sentence embedding vector $\mathbf{h}_{u,i}^S \in \{\mathbf{h}_{u,1}^S, \dots, \mathbf{h}_{u,N_t}^S\}$ is given by

$$\mathbf{f}_u^S = \text{ReLU}(\mathbf{W}^S \mathbf{h}_u^E + \mathbf{b}^S) \quad (9.34)$$

$$AS_{u,i}^S = \mathbf{f}_u^{S\top} \mathbf{h}_{u,i}^S \quad (9.35)$$

Accordingly, we can obtain the normalized attractiveness score $\bar{a}_{u,i}^S$ on each sentence embedding and the corresponding attraction-based story embedding \mathbf{t}_u :

$$\bar{A}S_{u,i}^S = \text{softmax}(\text{isr}^{\alpha=2}(a_{u,i}^S)) \quad (9.36)$$

$$\mathbf{t}_u = \sum_{i=1}^{N_t} \bar{A}S_{u,i}^S \mathbf{h}_{u,i}^S \quad (9.37)$$

where $\alpha = 2$ is set through experiments. Apart from encoding the most attractive sentences, we use the neutral-attraction story embedding to preserve other information of the story for the follow-up movie-level attraction scoring.

$$\bar{A}S_i^S = \text{softmax}[\text{isr}^{\alpha=16}(\mathbf{g}^S \mathbf{h}_{u,i}^S + b^S)] \quad (9.38)$$

$$\mathbf{t}_g = \sum_{i=1}^{N_t} \bar{A}S_i^S \mathbf{h}_{u,i}^S \quad (9.39)$$

Then, the comprehensive story embedding \mathbf{h}_u^T is encoded with the parameters \mathbf{W}^T and \mathbf{b}^T :

$$\mathbf{h}_u^T = \tanh(\mathbf{W}^T[\mathbf{t}_u, \mathbf{t}_g] + \mathbf{b}^T) \quad (9.40)$$

9.4.4 Cast Attraction Module

The architecture of cast attraction module is similar to the story attraction module. First, we map the cast members $\{c_1, \dots, c_{N_m}\}$ of the movie $m_{u,i}$ into embedding vectors $\{\mathbf{c}_1, \dots, \mathbf{c}_m\}$. Given the user attraction filter \mathbf{f}_u^C , the attractiveness score over each cast members is:

$$\mathbf{f}_u^C = \text{ReLU}(\mathbf{W}^C \mathbf{h}_u^E + \mathbf{b}^C) \quad (9.41)$$

$$AS_{u,i}^C = \mathbf{f}_u^{C\top} \mathbf{c}_i, \quad i \in \{1, \dots, N_m\} \quad (9.42)$$

Accordingly, the normalized attractiveness score $\bar{a}_{u,i}^C$ and the attraction-based cast embedding \mathbf{c}_u w.r.t. user u are given:

$$\bar{A}S_{u,i}^C = \text{softmax}(\text{isr}^{\alpha=1}(AS_{u,i}^C)) \quad (9.43)$$

$$\mathbf{c}_u = \sum_{i=1}^{N_m} \bar{A}S_{u,i}^C \mathbf{c}_i \quad (9.44)$$

For the follow-up movie-level attraction scoring, we need to preserve the overall neutral-attraction cast information besides the attractive cast member embedding

as done in the story attraction module.

$$\bar{A}S_i^C = \text{softmax}[\text{isr}^{\alpha=16}(\mathbf{g}^C \mathbf{c}_i + b^C)] \quad (9.45)$$

$$\mathbf{c}_g = \sum_{i=1}^{N_m} \bar{A}S_i^C \mathbf{c}_i \quad (9.46)$$

Finally, we obtain the comprehensive cast embedding \mathbf{h}_u^C , which is encoded with the parameters \mathbf{W}^C and \mathbf{b}^C

$$\mathbf{h}_u^C = \tanh(\mathbf{W}^C[\mathbf{c}_u, \mathbf{c}_g] + \mathbf{b}^C) \quad (9.47)$$

9.4.5 Multimodal Movie Attraction Scoring

After we obtain the comprehensive story embedding \mathbf{h}_u^T from story attraction module and the comprehensive cast embedding \mathbf{h}_u^C from cast attraction module. We concatenate them as the joint multimodal movie embedding $[\mathbf{h}_u^T, \mathbf{h}_u^C]$. Then, the attraction scores on the movie m can be computed with the user's movie-level filter \mathbf{f}_u^M over $[\mathbf{h}_u^T, \mathbf{h}_u^C]$:

$$S_{m_u} = \mathbf{f}_u^{M\top}[\mathbf{h}_u^T, \mathbf{h}_u^C] \quad (9.48)$$

9.4.6 Objective and Parameter Learning

Ranking Loss

In this case, it is also typical one-class preference data. Therefore, we treated it as a ranking problem as discussed in Section 9.3.6. Given a user u , we construct a contrastive pair to specify the attractiveness order, that is, we have the order $m_{u,i} \succeq m_{u,j}$ over a movie ($m_{u,i} \in \mathcal{M}_u$) explicitly selected by u and an unselected movie ($m_{u,j} \notin \mathcal{M}_u$). Then, we use the max-margin loss [121] to optimize the ranking order over pairs:

$$L_{m_{u,i} \succeq m_{u,j}} = \max(0, \text{margin} + S_{m_{u,j}} - S_{m_{u,i}}) \quad (9.49)$$

where the parameter *margin* needs to be tuned over data.

Algorithm 5 The learning procedure for a mini-batch

- 1: $\mathcal{B} \leftarrow \text{GetMinibatch}(\{\mathcal{M}_u\})$ from all user-movie pairs
 - 2: $\mathcal{N} \leftarrow \text{Sample contractive movies } m_{u,j} \text{ for each } m_{u,i} \in \mathcal{B}$
 - 3: Compute mini-batch loss using Eq. 9.49:
 - 4: $L_{\mathcal{B}} \leftarrow \frac{1}{|\mathcal{B}|} \sum_{\langle m_{u,i}, m_{u,j} \rangle \in \langle \mathcal{B}, \mathcal{N} \rangle} L_{m_{u,i} \succeq m_{u,j}}$
 - 5: Update parameters: $\Theta \leftarrow \Theta - \Gamma_{Adam}(\nabla_{\Theta} L_{\mathcal{B}})$
-

Training Procedure

This model is also implemented using Keras [37] with Tensorflow as the backend. We initialize the word embeddings with the pre-trained GloVe vectors [171]. The learning algorithm is similar to the procedure given Algorithm 4, we list a very brief scheme of the learning procedure on a mini-batch in Algorithm 5, where $\Gamma_{Adam}(\cdot)$ denotes Adam [111] based gradient descent optimizer.

9.5 Experiments

To evaluate HAM on text content as presented in Section 9.3, we conduct experiments on the two real-world datasets: (1) *DBbook*[†] is a binary-label dataset to indicate if a book is relevant for a user or not, and (2) *CiteULike* dataset [222] is a unary label dataset to indicate the academic papers that users have added to their libraries. Therefore, we train our model on DBbook dataset as a classification problem, while we train our model on CiteULike dataset as a ranking problem.

To evaluate MAM on multimodal content as presented in Section 9.4, the experiments are conducted on the real-world movie watch dataset MovieLens 1M [69]. We demonstrate our model from three aspects: (1) recommendation accuracy; (2) new movie recommendation, and (3) interpretation of attraction on movies.

[†]<http://challenges.2014.eswc-conferences.org/index.php/RecSys>

9.5.1 Comparison Methods

The following state-of-the-art content-based methods are compared for content recommendation. CF methods cannot deal with textual content and recommend new articles as the study focus in this work so they are not included for comparison.

- *CENTROID*: We create user profiles using the centroid [157] of all word embedding vectors from the users' articles. Then, we rank recommendations by the similarity between the user profile and the centroid of word embedding vectors of candidate articles.
- *CTR*: It performs regression for users over the latent topic distribution of each article learned from LDA.
- *CWER*: Similar to CTR, we create the collaborative word embedding regression (CWER) to perform regression for users over the centroid word embedding vector of each article initialized by GloVe embeddings.
- *FM*: Factorization machines take the normalized term frequency for each article as the features.
- *HAM*: This is our model proposed in Section 9.3.
- *HAM-P*: This is a variant of HAM, where we remove the group attraction filters and only consider user attraction filters (cf. Eq. 9.10, 9.11 and 9.12).
- *MAM*: This is the full multimodal attraction model proposed in Section 9.4.
- *MAM-S*: This is the single-modal version MAM that only has the story attraction module.
- *MAM-C*: This is the single-modal version MAM that only has the cast attraction module.

9.5.2 Attraction on Books for Classification

Data Preparation

The books available in the dataset have been mapped to their corresponding DBpedia[‡] URIs. The mapping contains 8,170 DBpedia URIs. We use these mappings to extract the abstract from the DBpedia repository by SPARQL. Since this dataset does not contain group information, we conduct clustering algorithms to split users into 20 groups where the similarity is measured by the number of common books between users. The brief statistics of this dataset is reported in Table 9.1.

Table 9.1 : The statistics of DBbook dataset

# users	6,181	# groups	20
# books	6,733	# labels	72,372
vocabulary	29,554	# labels / # users	11.71
# words/# sentences	9.86	# sentences/# abstracts	5.81

To test the performance of classification in which the positive books are relevant to users, we randomly held out 20% user labels from the dataset as the testing set, and the remainder was served as the training set. In this chapter, one of the most important tasks is to find new content, correspondingly the new books in this dataset, without knowing any other users' feedback. To simulate this case, we randomly selected 10% books and held out all their user labels from the dataset, and the remainder of 90% books and their labels are used for training.

[‡]<http://dbpedia.org>

Table 9.2 : The classification performance on published books

	CENTROID	CWER	FM	HAM-P	HAM
AUC	0.5466	0.5350	0.5944	0.6533	0.6958
ACC	0.6431	0.6815	0.6828	0.6983	0.7151

Relevance Classification on Published Books

Note that CTR is designed for item ranking, which cannot be applied to classification. Therefore, it is not included in this experiment. Table 9.2 reports the classification performance in terms of AUC and ACC on the published book dataset. CENTROID and CWER are built on the document representation induced from the centroid of word embeddings. However, there are many uninformative and even noisy words in an article, which often make the centroid of word embeddings deviate from the core topic of the article. As a result, the classification performance may degrade due to the obscure document representation. FM outperforms CENTROID and CWER because it actually performs CF by factorizing the interaction between users and the features of articles. Therefore, FM achieves better performance when articles are associated with more user labels. HAMs (HAM and HAM-P) outperform other models, which highlights the underlying design, that is, the hierarchical attraction filters can effectively detect the most attractive features, i.e., words and sentences, and reduce the obscure from uninformative words and sentences. In fact, HAMs conduct a personally attraction-based feature selection process in terms of the attraction filters to highlight the most salient features to a user. In particular, we find that the AUC scores of HAMs exceed other models with a clear margin (above 10%), which proves that HAMs can more effectively differentiate relevant books from irrelevant ones by scoring attraction.

Table 9.3 : The comparison of classification performance with/without group attraction filters w.r.t. users ≤ 5 labels and users ≥ 20 labels

	# labels ≤ 5		# labels ≥ 20	
	HAM	HAM-P	HAM	HAM-P
AUC	0.6372	0.6869	0.6922	0.7085
ACC	0.6745	0.7190	0.7186	0.7212

In HAM, we impose group attraction filter to regularize personal attraction filter (cf. 9.9). Theoretically, the group attraction filter acts as the prior which plays a more important role when fewer labels are observed between a user and books. Table 9.3 demonstrates the classification performance w.r.t. two cases, i.e., the insufficient label users (≤ 5 labels) and the adequate label users (≥ 20 labels). From the results, we find that HAM outperforms HAM-P in the case of ≤ 5 labels, whereas HAM does not show obvious superiority over HAM-P in the case of ≥ 20 labels. This result is consistent with the above analysis, which proves that incorporating group attraction filter can effectively alleviate the overspecialization for those users with insufficient data.

Relevance Classification on New Books

Judging the relevance on new books is more challenging due to the absence of user feedback. The performance is reported in Table 9.4. Since CENTROID and CWER are pure content-based models without the needs of feedback, their performance of is improved slightly (cf. Table 9.4 and Table 9.2) due to more training data (90%). In comparison, the performance of FM drops apparently because no interaction between users and the features of a new article can be used for learning with factorization. HAMs learn user attraction filters on textual contents, and

<i>Invitation to the Game</i>		
User I	<i>Attractiveness on sentences</i>	Invitation to the Game is a science fiction book written by Monica Hughes. It has recently been published as The Game. The book is a hard science fiction dystopian novel set in 2154, a time when machines and robots perform most jobs and children go to government schools. Because of this, very few people are employed, with many people living on a social welfare system for support. The unemployed people have nothing to look forward to, except various illicit drugs. Some have formed gangs, some are shown to be agitating for political reform (in chapter 5 there is a reference to leaflets printed up), and many are involved in organized crime of some form or another. The government, possibly the only government in existence at this point, is shown to have complete control over its citizens by restricting the unemployed to designated areas (DAs), and having similar control over the working-class. The working-class people are taught to hate the unemployed citizens, and the unemployed generally want money and employment, in a classic class struggle. The story is told from the perspective of Lisse, a recent graduate of school.
	<i>Attractiveness on words</i>	The book is a hard science fiction dystopian novel set in 2154, a time when machines and robots perform most jobs and children go to government schools.
User II	<i>Attractiveness on sentences</i>	Invitation to the Game is a science-fiction book written by Monica Hughes. It has recently been published as The Game. The book is a hard science fiction dystopian novel set in 2154, a time when machines and robots perform most jobs and children go to government schools. Because of this, very few people are employed, with many people living on a social welfare system for support. The unemployed people have nothing to look forward to, except various illicit drugs. Some have formed gangs, some are shown to be agitating for political reform (in chapter 5 there is a reference to leaflets printed up), and many are involved in organized crime of some form or another. The government, possibly the only government in existence at this point, is shown to have complete control over its citizens by restricting the unemployed to designated areas (DAs), and having similar control over the working-class. The working-class people are taught to hate the unemployed citizens, and the unemployed generally want money and employment, in a classic class struggle. The story is told from the perspective of Lisse, a recent graduate of school.
	<i>Attractiveness on words</i>	Some have formed gangs , some are shown to be agitating for political reform (in chapter 5 there is a reference to leaflets printed up), and many are involved in organized crime of some form or another.

Figure 9.5 : Attractiveness visualization on the abstract of *Invitation to the Game* w.r.t. sentences and words in the most attractive sentences for two users. The larger size and deeper color of font denotes the larger attractiveness weight is assigned.

these user attraction filters can efficiently to score new content. As a result, HAMs outperform other models on the new books.

Table 9.4 : The classification performance on new books

	CENTROID	CWER	FM	HAM-P	HAM
AUC	0.5533	0.5667	0.5399	0.6461	0.6838
ACC	0.6449	0.6981	0.6869	0.7104	0.7287

Interpretability Case Study

The most important value of attraction modeling is to disclose the attractive points for interpreting the cause of user historical selection and the reason for recommendation. In this experiment, we randomly pick two users and a book with the

title *Invitation to the Game* to illustrate user attractiveness on its abstract. Figure 9.5 visualizes the attractiveness, according to the weights (cf. Eq. 9.21) over the sentences, and the weights (cf. Eq. 9.16) over the words in the most attractive sentence. From the results, we can easily find the attraction difference between two users. User I is more attracted by the plots of science and technology (see the highlighted sentence), especially for the keyword *robots*, while User II is more attracted by the plots of gangsters and crimes (see the highlighted words). Obviously, HAM acts as a good interpreter to tell the insight of user selection and why the recommendation is made.

9.5.3 Attraction on Academic Papers for Ranking

CiteULike Dataset

CiteULike is an online platform which allows users to create personal libraries by saving interesting papers. This dataset [222] consists of the unary labels to specify the papers in the users' libraries. In addition, it provides the abstract of each paper. We adopt the same way to create 20 user groups as clustering users in DBbook dataset. Table 9.5 reports the brief statistics.

Table 9.5 : The statistics of CiteULike dataset

# users	5,551	# groups	20
# papers	16,980	# likes	204,987
vocabulary	48,004	# likes / # users	36.93
# words/# sentences	10.01	# sentences/# abstracts	9.64

For testing the ranking performance on published paper recommendation, we randomly held out 20% saved papers in user libraries as the testing set, and the remainder was used for training. Recommendation on new papers in terms of content-

based approach is a key study problem in this paper. To simulate this case, we randomly selected 10% papers and held out all their save record from the dataset, and the remainder of 90% papers and their save record were used for training. For each hold-out test sample in the above two testing sets, we randomly draw ten noisy samples to test whether the testing methods can correctly rank the true sample at a top position out of the noisy samples.

Recommendation on Published Papers

In RSs, ranking recommendation items is a more common task than classifying items as studied above. Table 9.6 reports the recommendation accuracy. CTR ranks the user preference on an article according to the topic distribution. As CWER, the uninformative words may overwhelm the topic distribution of an article in CTR. As a result, CTR achieves similar performance with CWER. Due to the relatively sufficient papers in users' libraries (37 papers on average, cf. Table 9.5), FM has sufficient data to perform factorization. As a result, it outperforms all baselines except HAMs. For Table 9.5, we can find each sentence consists of 10.01 words on average while each abstract consists of 9.64 sentences on average. With these relatively long sentences and articles, HAMs can extract the most attractive words and sentences while filtering out uninformative words and sentences, which is closer to the way of finding attractive papers by a human. As a result, HAMs significantly outperform other comparison methods.

Figure 9.6 (a) illustrates the recall of all comparison methods. We find that the plots of HAMs are above the plots of baselines with a large margin, which proves that HAMs can more effectively capture users' preferences on candidate papers and correctly rank them in terms of scoring the attractiveness over each paper (cf. Eq. 9.24).

Table 9.6 : The ranking performance on published papers

Method	MAP			nDCG		
	@5	@20	@50	@5	@20	@50
CENTROID	0.3852	0.4199	0.4473	0.4999	0.5937	0.6544
CTR	0.5189	0.5627	0.5869	0.6297	0.7181	0.7600
CWER	0.5152	0.5652	0.5902	0.6240	0.7179	0.7600
FM	0.5747	0.6190	0.6411	0.6834	0.7662	0.7996
HAM-P	0.6535	0.6944	0.7126	0.7491	0.8190	0.8449
HAM	0.6639	0.7005	0.7191	0.7577	0.8227	0.8488

Recommendation on New Papers

The new paper recommendation cannot be handled by pure CF methods, so the content-based approach is more desirable for recommending new content in this new media age. The ranking performance of all comparison methods for new papers is reported in Table 9.7 and the recall is illustrated in Figure 9.6 (b). CENTROID underperforms other methods for the reason analyzed in the above subsections. CWER and CTR achieve similar performance, which proves the effectiveness of personally attraction-based content-based matching even without any users' like records. In comparison, FM does not achieve good performance as in the last experiment due to the absence of feedback on new papers. Detecting attractive points is a critical task for the new paper recommendation, which demonstrates the goal and value of attraction modeling. From Table 9.7 and Figure 9.6, we find that HAMs outperform other baselines with more than 10% improvements in terms of all evaluation metrics, i.e., MAP, nDCG and Recall. It is because researchers often decide to save a paper to their libraries due to the glance of some attractive points. This proves the value of our design that HAMs can precisely capture users' multilevel attraction on an

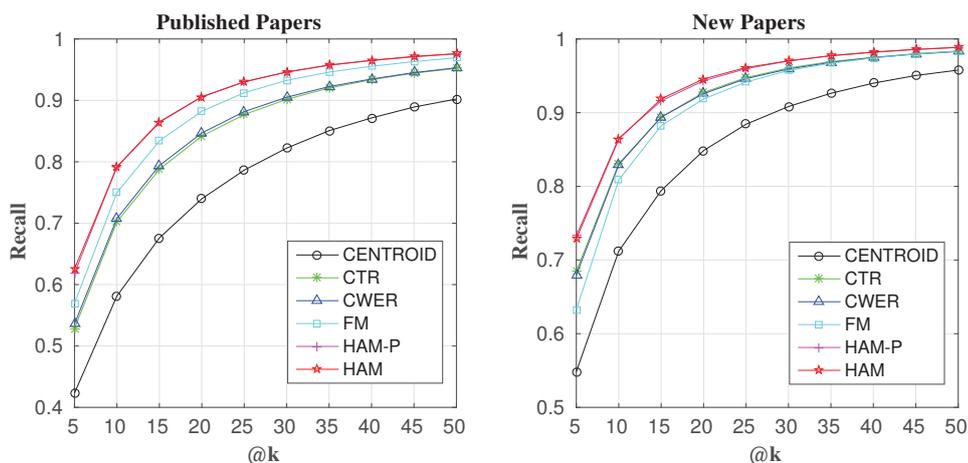


Figure 9.6 : Recall@5-50 on the Published Paper testing set and the New Paper testing set

article over the words and the sentences.

Interpretability Case Study

As illustrated in Introduction, different user groups may have quite different attractive points in an article. We randomly pick a paper to demonstrate these difference. Figure 9.7 lists six sentences of the abstract of the paper titled “*Exploring complex networks*”, and we list the most attractive word by scoring the attractiveness over words in each sentence, only using group attraction filter, cf. Eq. 9.28. From this demonstration, we can easily find the difference of attraction between groups. For example, we find that Group I has more attraction on the topology of the network and the dynamics on the network. In comparison, the attractive words imply that Group V pays more attention to the behavior on the network. Therefore, we can utilize HAM to analyze and interpret the attraction between different groups.

In the second demonstration, we list the top five new papers with the highest attractiveness score (cf. Eq. 9.24) from the testing set and the most attractive word in each sentence. Figure 9.8 lists the results w.r.t. two exemplary users.

Table 9.7 : The ranking performance on new papers

Method	MAP			nDCG		
	@5	@20	@50	@5	@20	@50
CENTROID	0.4216	0.4697	0.4878	0.5242	0.6270	0.6675
CTR	0.5744	0.6249	0.6380	0.6723	0.7552	0.7780
CWER	0.5660	0.6188	0.6324	0.6626	0.7492	0.7728
FM	0.4736	0.5388	0.5539	0.5817	0.6887	0.7153
HAM-P	0.6315	0.6776	0.6893	0.7216	0.7917	0.8129
HAM	0.6352	0.6808	0.6928	0.7264	0.7961	0.8151

From this figure, we may speculate User I is a geneticist and s/he is especially attracted by the topic of genome sequencing. In comparison, User II may be a neuroscientist focus on brain imaging. From the recommended papers for User I, we find three papers have been really saved in user’s library, where we can find the three most attractive words, “gene”, “genome” and “sequencing”. If we take a closer look into the other two articles not in User I’s library, we find that “gene” and “sequencing” are the most attractive words in paper *The effect of sequencing errors on metagenomic gene prediction* and “genome” is the most attractive word in paper *Human genome sequencing using unchained base reads on self-assembling DNA nanoarrays*. Obviously, these two recommended new articles are quite relevant to the papers in User I’s library. Therefore, they are potentially good candidates to attract User I. Similar cases can be observed from User II, “brain”, “neuro”, “cortex” and “imaging” are iconically attractive words in the top five recommended articles with the highest attraction scores.

<i>Exploring complex networks</i>						
1. The most basic issues are structural: how does one characterize the wiring diagram of a food web or the Internet or the metabolic network of the bacterium <i>Escherichia coli</i> ?						
2. Are there any unifying principles underlying their topology?						
3. From the perspective of nonlinear dynamics, we would also like to understand how an enormous network of interacting dynamical systems-be they neurons						
4. Power stations or lasers-will behave collectively						
5. Given their individual dynamics and coupling architecture.						
6. Researchers are only now beginning to unravel the structure and dynamics of complex networks.						
	<i>Sentence 1</i>	<i>Sentence 2</i>	<i>Sentence 3</i>	<i>Sentence 4</i>	<i>Sentence 5</i>	<i>Sentence 6</i>
Group I	network	topology	network	stations	dynamics	networks
Group II	food	underlying	neurons	behave	individual	beginning
Group III	Escherichia	topology	dynamical	collectively	architecture	researchers
Group IV	coli	topology	neurons	collectively	architecture	networks
Group V	characterize	unifying	neurons	behave	individual	complex

Figure 9.7 : Demonstration of the different of group attraction on the abstract of the paper *Exploring complex networks*. We list the most attractive word in each sentence for each group to illustrate their difference.

	Title of Top 5 Attractive Papers	In Library	The Most Attractive Word in Each Sentence
User I	<i>Genome sequencing in microfabricated high-density picolitre reactors</i>	Yes	sequencing, throughput, fibre, sanger, amplification, novo
	<i>The effect of sequencing errors on metagenomic gene prediction</i>	No	gene, metagenomic, gene, gene, specialized, gene, sanger, gene, metagenomic, eukaryotic, outperforms, metagenomic, sequencing, error
	<i>Assembly algorithms for next-generation sequencing data</i>	Yes	sequencing, solid, error, novo, summarizes, bruijn
	<i>ALLPATHS: De novo assembly of whole-genome shotgun microreads</i>	Yes	sequencing, novo, solexa, escherichia, genomes, polymorphism, genome, genome, genomes, edges, genome
	<i>Human genome sequencing using unchained base reads on self-assembling DNA nanoarrays</i>	No	genome, probe, genomes, variants, genome, genome
User II	<i>Automatic "pipeline" analysis of 3-D MRI data for clinical trials: application to multiple sclerosis</i>	No	imaging, image, regulatory, rater, image, scans, algorithms, sclerosis, pipeline, lesion, rater, detect, sclerosis, neurological
	<i>Intrinsic signal changes accompanying sensory stimulation: functional brain mapping with magnetic resonance imaging</i>	Yes	magnetic, brain, signal, relaxation, oxygenation
	<i>Coding of border ownership in monkey visual cortex</i>	No	cortex, neural, cell, displays, square, neurons, contrast, coding, luminance, onset, differences, responses, cues, neurons, displays, neurons, receptive, cortex
	<i>Functional connectivity in the motor cortex of resting human brain using echo-planar MRI</i>	Yes	pixel, cortex, brain, fluctuations, oxygenation
	<i>The precuneus: a review of its functional anatomy and behavioural correlates</i>	Yes	neuroimaging, cortical, lesion, imaging, retrieval, resting, correlates, cortex, vegetative, precuneus, subcortical, connectional, links, activation, imaging, anterior, involved, retrieval

Figure 9.8 : Demonstration of two exemplary users' most attractive papers in the testing set. We list the most attractive word in each sentence of the recommended papers for interpretation.

9.5.4 Multimodal Attraction on Movies

Data Preparation

We collect user watch records from the MovieLens 1M dataset. However, this dataset does not contain any story and cast data. Fortunately, researchers have

Table 9.8 : Statistics of content-enriched MovieLens dataset

#movies	3,900	#users	6,040
#watch record	1,000,209	#cast	9,398
story vocabulary	22,582	#sentences/#story	10.2
#cast/#movie	6.44	#plays/#cast members	2.10

provided a good mapping from MovieLens ID to DBPedia URI [160]. We queried all available story abstract and cast data from DBPedia. The statistics of the data are reported in Table 9.8.

For testing the performance on released movie recommendation, we randomly held out 20% user watch records as the testing set, and the remainder were served as the training set. One of the most important tasks is to recommend new movies without knowing any watch record. To simulate this case, we randomly selected 10% movies and held out all their watch records from the dataset, and the remainder of 90% movies and their watch records were used for training. For each hold-out test sample in above two testing sets, we randomly draw ten noisy samples to test whether the testing methods can rank the true sample at a top position out of noisy samples.

Recommendation for Released Movies

We evaluate the recommendation performance on released movies associated with users' watch records, that is, people have known the story and cast members in these movies. Table 9.9 reports the recommendation accuracy. CTR scores the user preference according to the topic distribution over a movie story. However, there are many uninformative words which may obscure the core topic distribution of the story. CENTORID and CWER are built on the story embedding derived

Table 9.9 : Ranking performance on released movies

Method	MAP@5	MAP@20	MRR@5	MRR@20
CENTROID	0.1738	0.1481	0.0763	0.0958
CTR	0.1226	0.1069	0.0514	0.0692
CWER	0.1666	0.1580	0.0798	0.1089
MAM-C	0.4243	0.3963	0.2118	0.2398
MAM-S	0.3816	0.3451	0.1822	0.2093
MAM	0.4252	0.3997	0.2187	0.2464

from the centroid of word embeddings. Since word embedding is an unnormalized vector, it allows large elements to specify the significance. As a result, CENTORID and CWER outperform CTR but they still suffer the obscure from uninformative words. MAM-S leads CENTORID and CWER with a large margin, the MAP and the MRR are at least 200% higher than baselines. This highlights the design of our model, that is, we place two types of filters in MAM-S, one is to extract the most attractive words and sentences and the other is to filter out noisy words and sentences. MAM-C surprisingly performs well, this discloses the fact that the attractiveness of a movie is heavily related to the attractiveness of its cast. Thanks to the multimodal modules over story and cast to comprehensively capture users' attraction from different aspects, MAM achieves the best performance out of all comparison methods.

Figure 9.9 depicts the recall of all comparison methods. We find that the plots of MAM-C, MAM-S, and MAM are above the plots of baselines with apparent margins, i.e., MAM can more accurately retrieve the attractive movies for each user in top positions. MAM combines the information from both modules, which leads to the best recall.

Table 9.10 : Ranking performance on new movies

Method	MAP@5	MAP@20	MRR@5	MRR@20
CENTROID	0.2381	0.2409	0.1623	0.1900
CTR	0.1056	0.1374	0.0798	0.1089
CWER	0.1971	0.2346	0.1461	0.1801
MAM-C	0.1817	0.1664	0.1132	0.1370
MAM-S	0.3001	0.3059	0.2091	0.2371
MAM	0.2573	0.2671	0.1794	0.2090

Recommendation for New Movies

We apply the above design to recommend the attractive new movie, which cannot be handled by pure CF methods, to demonstrate the goal and value of attraction modeling. Content-based methods are more capable of tackling such cases widely seen in this new media age. The ranking performance is reported in Table 9.10 and the recall is illustrated in Figure 9.9 (b). CTR underperforms other methods for the reason analyzed in the above subsection. CENTROID and CTR achieve similar performance to the first experiment, which proves the effectiveness of content-based matching using word embedding for new movies even without any user watch record. Similarly, MAM-S achieves comparable performance with the above case. However, MAM-C is the special case. We find that the performance drops drastically when compared with Table 9.9. In fact, the reason behind is quite straightforward. We can find most cast only appeared in two movies (cf. Table 9.8). Accordingly, users cannot tell whether they will be attracted by an unknown cast. Figure 9.10 shows two testing samples of a user. The left movie is associated with a high attractiveness score due to the known cast members (in red) in this user’s training set, whereas the

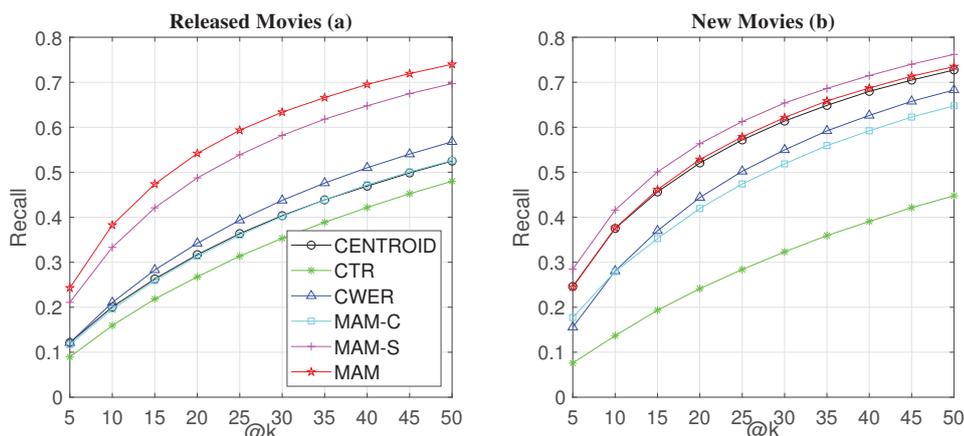


Figure 9.9 : R@5-50 on the Released Movies and the New Movies

<i>Wild America (1997)</i>	<i>Bogus (1996)</i>
William Dear, Scott Bairstow, Jonathan Taylor Thomas, Devon Sawa	Norman Jewison, Gérard Depardieu, Whoopi Goldberg, Alvin Sargent, Haley Joel Osment

Figure 9.10 : Two comparative testing samples of *User 182*: the left movie *Wild America* obtains a high attraction score because of the cast members in red appear in user's watched movies while the cast members of *Bogus* never appear in user's movie list.

cast members in the right movie are absent from user's training set, which results in low attractiveness scores. As a result, MAM-C tends to assign low rank on these movies. This also proves the factor that the attractiveness of a movie is heavily dependent on its cast members. Accordingly, the multimodal model MAM slightly underperforms MAM-S due to the ineffectiveness of MAM-C.

Interpretation and Visualization

The most important value of multimodal attraction modeling is not only for recommendation but for obtaining insight into the underlying causes of user selection by disclosing the multi-aspect attractive points. In this experiment, we pick two case studies to visualize the user attractiveness scores output by MAM. Figure 9.11

<i>Invitation to the Game</i>		
User I	<i>Attractiveness on sentences</i>	Invitation to the Game is a science fiction book written by Monica Hughes. It has recently been published as The Game. The book is a hard science fiction dystopian novel set in 2154, a time when machines and robots perform most jobs and children go to government schools. Because of this, very few people are employed, with many people living on a social welfare system for support. The unemployed people have nothing to look forward to, except various illicit drugs. Some have formed gangs, some are shown to be agitating for political reform (in chapter 5 there is a reference to leaflets printed up), and many are involved in organized crime of some form or another. The government, possibly the only government in existence at this point, is shown to have complete control over its citizens by restricting the unemployed to designated areas (DAs), and having similar control over the working-class. The working-class people are taught to hate the unemployed citizens, and the unemployed generally want money and employment, in a classic class struggle. The story is told from the perspective of Lisse, a recent graduate of school.
	<i>Attractiveness on words</i>	The book is a hard science fiction dystopian novel set in 2154, a time when machines and robots perform most jobs and children go to government schools.
User II	<i>Attractiveness on sentences</i>	Invitation to the Game is a science-fiction book written by Monica Hughes. It has recently been published as The Game. The book is a hard science fiction dystopian novel set in 2154, a time when machines and robots perform most jobs and children go to government schools. Because of this, very few people are employed, with many people living on a social welfare system for support. The unemployed people have nothing to look forward to, except various illicit drugs. Some have formed gangs, some are shown to be agitating for political reform (in chapter 5 there is a reference to leaflets printed up), and many are involved in organized crime of some form or another. The government, possibly the only government in existence at this point, is shown to have complete control over its citizens by restricting the unemployed to designated areas (DAs), and having similar control over the working-class. The working-class people are taught to hate the unemployed citizens, and the unemployed generally want money and employment, in a classic class struggle. The story is told from the perspective of Lisse, a recent graduate of school.
	<i>Attractiveness on words</i>	Some have formed gangs , some are shown to be agitating for political reform (in chapter 5 there is a reference to leaflets printed up), and many are involved in organized crime of some form or another.

Figure 9.11 : Statistical attractiveness on movie *Election (1999)* w.r.t. sentences, words in the most attractive sentences and cast members from the perspectives of User 156 and User 2163. The larger size and deeper color of font denote the larger attractiveness weight is assigned.

illustrates the statistical attractiveness, according to the weights (cf. Eqs. 9.29, 9.36 and 9.43), over the sentences of movie story, words in the most attractive sentence, and the cast members points for User 156 and User 2163. The results show that we can easily find the attraction difference between two users. User 156 is attracted by the first sentence which highlights the genre of this movie, i.e., comedy-drama, while User 2163 is attracted by the last sentence which highlights the award of this movie. Similarly, we find User 156 is attracted by the director Alexander Payne while User 2163 is attracted by the star Reese Witherspoon. Therefore, we can easily use MAM to analyze user selection and tell the insight about the recommendation made.

9.6 Summary of Contributions

In this chapter, we enhance content-based RSs with attraction modeling. In particular, the non-IID technique is focused on modeling the coupling relationships between and content elements. The main contributions of this work are summarized as follows:

- We propose the attraction modeling to build interpretable content preference systems in this new media era. Moreover, we provide a framework of attraction model to capture personal attractiveness over content elements.
- We design a hierarchical attraction model (HAM) to capture multilevel attraction over an article, i.e., word-level attraction, sentence-level attraction, and document-level attraction.
- We design a multimodal attraction model (MAM) to learn the personal attraction over multi-type content. In particular, we take the story content and the cast content of a movie as two modalities to study the attraction modeling.
- Extensive experiments on a real-world dataset are conducted to evaluate the above design. All quantitative results show that our attraction models consistently outperforms state-of-the-art methods. Moreover, we demonstrate a collection of cases by exploiting the statistical attractiveness scores over the textual and cast content to interpret recommendation results.

Part V

Summary and Prospect

Chapter 10

Conclusion

In this thesis, we analyzed the issues and challenges in classic RSs and then presented the necessity of building non-IID RSs that consider various heterogeneities and coupling relationships within and between users, items, and interactions. As a result, we presented the design of six non-IID RSs in terms of machine learning approaches. In this chapter, we conclude the contributions of this thesis.

10.1 Non-IID RS Modeling on Users

In this section, we focus on designing non-IID RSs by modeling the heterogeneity between users and capturing the underlying coupling relationships. We studied the non-IID modeling for two typical RSs with respect to users: a GBRS and SNRS.

10.1.1 GBRS Modeling: Learning Comprehensive Group Preference Representation

We proposed a deep learning approach to overcome the deficiencies in current GBRSs. To comprehensively learn the group characteristics, a DNN was designed to model the heterogeneity between group members and learn a group preference representation from feedback data of all group members. Essentially, our model aims to learn high-level comprehensive embedding to represent group preference, to avoid the vulnerabilities in a shallow representation. The empirical evaluation on a real-world dataset proved that our approach can achieve performance that is far superior to other state-of-the-art models. Because our approach constructs a deep architecture that is able to disentangle group-specific features at a high level, it is

applicable to many other areas that study group behavior with coupled interactions among members.

10.1.2 SNRS Modeling: Learning Comprehensive Group Preference Representation

We proposed a framework for modeling influential contexts in an SNRS and further instantiated a context embedded multi-relation RS, namely *ICE-MRS*, with neural networks. In particular, *ICE-MRS* learns user/item influence-aware embeddings from influential contexts through an influence aggregation unit (IAU). Extensive experiments proved that *ICE-MRS* outperforms other state-of-the-art methods as well as its effectiveness at handling the cold-start problem. In addition, we demonstrated the interpretability of *ICE-MRS* using a real case. In fact, *ICE-MRS* is a general influence-embedding model that can be applied to other domains with heterogeneous networks, such as user group behavior analysis and biological interaction network, as well as RSs.

10.2 Non-IID Recommender Systems Modeling on Items

In this section, we focus on designing non-IID RSs by modeling the heterogeneity between items and capturing the underlying coupling relationships. We studied the non-IID modeling for two typical RSs with respect item: a CDRS and SBRS.

10.2.1 CDRS Modeling: Leveraging Knowledge Across Heterogeneous Domains

We discussed the emerging requirements of cross-domain recommendation and analyzed the limitations of current cross-domain CF methods. As a result, we proposed weighted irregular tensor factorization (WITF) to more effectively model heterogeneity and their couplings between domains. In particular, WITF addresses

an irregular tensor factorization problem wherein each domain can have a specific item set. Furthermore, an effective post learning procedure was designed to fine tune the user and item factors of the target domain. In addition, the WITF framework can deal with explicit preference data (e.g., ratings and implicit preference data, such as clicks, in a unified algorithm).

We evaluated our approach on three typical real-world application scenarios. The evidence from all the results showed that our approach outperforms all other state-of-the-art methods, especially for cold-start cases. This is because WITF can more effectively capture the domain-specific user preference through the triadic relationship between users, items, and domains, whereas traditional models only model the dyadic relation, and thus, they fail to represent the heterogeneities when transferring knowledge between domains.

10.2.2 SBRS Modeling: Modeling Selection with Influential Users and Items

To deal with the deficiencies of recommendation in traditional RSs without considering the couplings of the choices within a session and between the previous sessions and current session, we proposed neural cross-session filtering (NCSF) to build a more effective and efficient personalized SBRS. NCSF is a neural model that jointly models the context of intra-session coupling and inter-session coupling when recommending the next item. The empirical evaluation on the real-world E-commerce dataset proved the comprehensive superiority of this approach over other state-of-the-art methods. Moreover, NCSF is a general architecture, and therefore it can be applied in many other domains besides RS; for example, the topic drift problem in NLP.

10.3 Non-IID RS Modeling on Complex Interaction

In this section, we focus on designing non-IID RSs by modeling the complex interaction between users and items. In particular, we studied the non-IID modeling technique on an MORS and ABRS.

10.3.1 MORS Modeling: Optimizing Credibility and Novelty for Long-tail Recommendation

We addressed the challenges of improving the recommendations of items and users in long-tail distributions, and analyzed the ineffectiveness of current approaches. As a result, we proposed an MORS that targets the optimization on both credibility and specialty. To jointly optimize these two coupled objectives, we proposed the recurrent mutual regularization model (RMRM), which consists of two coupled components, namely C-HMF, which emphasizes the credibility of ratings, and S-HMF, which emphasizes the specialty of choices, wherein the parameters of C-HMF and S-HMF are regularized in terms of the empirical priors induced from each other. The empirical evaluations of two real-world datasets illustrated that RMRM is capable of conducting more reliable predictions than the other compared methods, especially for both items and users in the tail of distributions.

In fact, RMRM provides a general framework for learning latent features that are regularized by multi-objective empirical priors. Therefore, RMRM and its extension could be applied in many areas outside of RSs, such as CV, audio processing, and multimedia clustering, all of which depend largely on the MF technique, and thus could benefit from multi-objective regularization.

10.3.2 ABRS Modeling: Capturing and Interpreting Attraction Points in Content

We proposed attraction modeling to build interpretable content-based RSs. By modeling attraction over content elements, we could more effectively capture user preference and interpret user selection. In particular, a framework of the attraction model was proposed to score personal attractiveness over content elements.

To study user attraction on textual content, we designed a hierarchical attraction model (HAM) to learn users' multilevel attraction over an article. In particular, the HAM can statistically interpret users' historical selection and recommendation results by learning multilevel attractiveness scores over words, sentences, and documents. Moreover, it is eligible to conduct new content recommendation.

To study user attraction on multiple types of content, we design a multimodal attraction model (MAM) to learn user attraction on movie story and cast members. MAM can provide the interpretation of user selection w.r.t. the multi-aspect attractive points. Moreover, it can conduct recommendation for new movies by jointly considering the multimodal attraction.

The experimental results on three real-world datasets proved that attraction modeling can improve the performance in content-based RSs. Moreover, the statistical interpretability of the attraction model is helpful for analyzing user behavior and explaining the recommendation results.

Chapter 11

Open Challenges and Future Directions

At this stage, research on recommendation is still experiencing numerous challenges, some classic problems have not been solved, and multiple open issues must be conquered. It is important to scrutinize the intrinsic complexities and nature of the underlying recommendation problems. To this end, more complex and expressive models must be built to learn the representation of coupling relationships and heterogeneity of recommended users and items.

11.1 New Evaluation Methods and Metrics

Accuracy metrics (cf. Chapter 3 Section 3.4) mode for RSs. However, sometimes improving the accuracy by one or two percentage points does not significantly generate an improved user experience or more commercial benefits [149]. Therefore, some researchers have proposed evaluation methods other than accuracy [54, 66, 75], including diversity, novelty, and serendipity.

Non-IID RSs often need to model heterogeneity and coupling relationships from multiple aspects; therefore, how to comprehensively evaluate a non-IID RS with considering multiple evaluation criteria is a crucial development direction. In fact, it is difficult to improve multiple metrics simultaneously, such as accuracy and diversity [251]. In addition to experimental metrics, real-world business systems are often more concerned with certain business benefit indicators, such as the conversion rate, purchase rate, and purchase amount, which are affected by the recommendation. Therefore, new evaluation methods and metrics are required to assess the

effectiveness and efficiency of next-generation non-IID RSs.

11.2 SNRSs

Open Challenges

- How can big social-network data be dealt with?

First, billions of nodes exist in a real-world social network site such as Facebook, which is a great challenge in terms of storage and computation. Second, quite a few different types of connections between nodes exist (e.g., the following relationships between users and co-targeting events). To model all of these heterogeneous relationships is another challenge. Third, the connections between nodes continually change every second, and capturing these constant changes is a great challenge.

- How can the influential nodes for the recommendation be recognized?

Given a target node, only a few connected nodes largely contribute to the final decision instead of all of them. Detecting the most influential nodes is a challenge.

- How can more social information be incorporated without invading privacy?

Incorporating more personal information and relationships may benefit the recommendation quality but protecting users' privacy is still the top priority. Therefore, modeling SNRSs without invasion of privacy is another challenge.

Future Directions

- Representation of social relationships with a network embedding approach. Network embedding has been extensively studied in recent years [40]. It has the potential to be employed in modeling social relations in SNRSs.

- Social dynamics modeling.

To model social dynamics, such as event propagation, connections, and disconnections, the memory mechanism [63, 207, 235] may be helpful.

11.3 GBRs

Open Challenges

- How can group feedback data be collected?

As of today, very few real-world public datasets are available for studying group behavior, which is an obstacle to the development of GBRs. In most studies, the datasets are synthetic from personal feedback, which cannot reflect the ground truth of group decision.

- How recommendations be made to an ad-hoc group?

Unlike a group with fixed members, such as a household, an ad-hoc group, such as a meetup of friends or attendees of a conference, does not have the historical choice record to learn group preferences at the group level.

Future Directions

- Using attention mechanism.

Most aggregation strategies for group-based recommendation concern how to weight the contribution of group members. We may employ the attention mechanism to learn to assign weights to members.

- Context-aware group-based recommendation.

Incorporation of contextual information is another direction to extend GBRs. Given the various contexts, the weights assigned to group members may be quite different.

11.4 CDRSs

Open Challenges

- How much information should be imposed for other domains?

Given multiple domains, a model needs to transfer the appropriate amount of information that should be transferred from each auxiliary domain.

- How should knowledge be transferred between heterogeneous data domains?

In most studies, the data in different domains are of the same type, such as ratings [83,84,137], text [57,208], or images. To model domains with different types of data and transfer knowledge between them is still an open challenge for CDRSs.

Future Directions

- Non-overlap cross-domain transfer.

In most current CDRSs, the overlapped users and/or items across different domains [28] are required to establish the links between domains to leverage the knowledge. However, it is often not possible to find identical users or items from different domains; for example, the user IDs of different sites, such as Facebook and Twitter, are generally different because of privacy. Therefore, to design a CDRS without the need for explicit overlap is a crucial research direction.

- Heterogeneous cross-domain RSs.

Generalized cross-domain tasks including domain adaptation [185], transfer learning [166], multimodal learning [12] and multi-source learning [39] involve multiple domains, which is challenging because the heterogeneities across domains engender significant challenges to modeling. In contrast to the common consensus, information shared across domains, and learning heterogeneous but

complementary information between domains, must be considered in next-generation CDRSs.

11.5 SBRSSs

Open Challenges

- How can multiple intents be detected in a session?

For a long session, it often contains multiple intents, for example, buying food and buying electronics could be two independent intents in the same online transaction. The crucial task is to detect these intents. However, these intents are not easily estimated from the mixed sequence of a session.

- How can the intent of the next item be detected?

In a multi-intent session, the intent may continue to change in every choice. To recommend the next item, the underlying intent must be captured. Therefore, a challenging problem is to detect the intent of the next-item choice.

Future Directions

- Incorporating user and item features.

Most current SBRSSs mainly model the action sequences without considering the extra features of items and users. In real-world applications, much richer types of data are usually available, such as the description of an item and/or the image of an item. Definitely, incorporating this information can more effectively infer users' intents and capture the coupling between items.

- Working with additional contextual information.

Modeling an SBRSS mainly attempts to capture the coupling and transition between items. However, the transition between users' choices is quite relevant to the contextual factors, such as a user's geographical position, the

current weather, or the time of the day. Therefore, incorporating contextual information into SBRs is a promising research direction.

11.6 MORs

Open Challenges

- How can the impacts from multiple objectives be integrated?

Generally, it is difficult to find a solution that is optimal for all objectives. In a different context, the optimal solution may also be quite different.

- How should objectives be personalized for the recommendation?

Different users often pay attention to different objectives; therefore, how to determine a personalized optimal solution for multiple objectives is a great challenge.

Future Directions

- Applying multi-objective optimization methods in the recommendation.

Many existing multi-objective design, analysis, and optimization methods [44, 100], such as multiple-criteria decision analysis and multidisciplinary design optimization, are potentially integrated into MORs.

- Generalized multi-objective recommendation.

In fact, most of the aforementioned non-IID RSs have natural connections with MORs, because all of them attempt to model the coupling relationships over users, items, and domains. For example, we may set up an objective for each group member when conducting group-based recommendation; then, the GBRS can be reduced to an MOP by finding a solution to maximize the satisfaction over all members' objectives. For cross-domain recommendation, we can set up an objective for each domain, and then the MOP is to jointly

optimize all of the objectives over all domains. Given multi-relation networks, the social network, the item network, and the user-item network are generally associated with different objectives for learning; thus, multi-objective optimization is a natural choice for finding the solution to this recommendation problem.

11.7 ABRs

Open Challenges

- How should the attraction models be evaluated?

As previously stated, attraction is a type of subjective feeling for each user. We indirectly validated the effectiveness of attraction modeling by checking whether it can improve recommendation performance. At this stage, it is almost impossible to find thousands of real users who can express their feelings about millions of posts, news stories, and movies. Furthermore, users' attraction may keep changing with surroundings. Therefore, it cannot substantially verify whether the statistical attraction output of our models can genuinely match the true attraction of a user at a certain moment.

Future Directions

- Attraction modeling on more data types.

In Chapter 9, we focused on modeling attraction on textual data (story) and categorical data (cast), which are the most common data in content-based RSs. For other content such as music, images, and videos, it is possible to incorporate some feature learning models, such as CNN, to extract their local representations, and then an attraction model can be modeled over them.

- Attraction modeling for studying user behavior in more applications.

In Chapter 9, we focused on modeling the attraction in content-based RSs.

In fact, other applications such as link prediction and social network analysis may also employ attraction modeling to study user behavior.

- Involving theories from other disciplines to model and explain attraction. Because attraction is a subjective feeling, some advanced approaches involving psychology, neuroscience and brain science, are promising research directions for modeling attraction.
- Context-aware attraction modeling.

Attraction is a subjective feeling that changes according to many context factors, such as time, location, and companions. Therefore, incorporating contextual information will be highly beneficial for capturing users' attraction.

11.8 Unified and Ubiquitous RSs

In this thesis, we presented multiple non-IID RSs from different perspectives. In fact, the ultimate RS of the future will be a unified system, which is the synergy of all of the above mentioned recommendation techniques. In Section 11.6, we presented the multi-objective optimization technique to work with CDRSs, GBRSs, and SNRSs. Similarly, session-based recommendation modeling could be considered in all other non-IID RSs, because all of the choices in these RSs have a sequential order. Furthermore, the cross-domain problem is common in other non-IID RSs, such as SNRSs, GBRSs, and SBRSSs, in which the items may belong to quite different domains. Furthermore, the contextual information, such as the time of day or geographic regions, can be modeled using the cross-domain technique. In an ABRS, all of the recommendation techniques can be employed to capture users' attraction, and vice versa. Obviously, these RSs are interdependent and reinforce each other.

In the future, RSs will be ubiquitous in daily life. We may not explicitly perceive their existence because we would be accustomed to ubiquitous recommendations

throughout our lives, for food, clothing, work, health, and all other parts of daily living. RSs can be conscious of all available environmental information, physical conditions, and lifelong data to provide the most effective suggestions and plans.

Bibliography

- [1] E. Acar, D. M. Dunlavy, and T. G. Kolda, “Link prediction on evolving data using matrix and tensor factorizations,” in *Data Mining Workshops, 2009. ICDMW '09. IEEE International Conference on*, 2009, Conference Proceedings, pp. 262–269.
- [2] E. Acar, D. M. Dunlavy, T. G. Kolda, and M. Morup, “Scalable tensor factorizations with missing data,” in *Proceedings of the 2010 SIAM International Conference on Data Mining*. SIAM, 2010, Conference Proceedings, pp. 701–712.
- [3] G. Adomavicius and A. Tuzhilin, “Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 6, pp. 734–749, 2005.
- [4] G. Adomavicius and Y. Kwon, “New recommendation techniques for multi-criteria rating systems,” *IEEE Intelligent Systems*, vol. 22, no. 3, 2007.
- [5] ———, “Multi-criteria recommender systems,” in *Recommender systems handbook*. Springer, 2015, pp. 847–880.
- [6] D. Agarwal and B.-C. Chen, “Regression-based latent factor models,” *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 19–28, 2009.
- [7] C. Anderson, *The Long Tail: Why the Future of Business Is Selling Less of*

More. Hyperion, 2006.

- [8] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” *arXiv preprint arXiv:1409.0473*, 2014.
- [9] B. Bai, Y. Fan, W. Tan, and J. Zhang, “Dltsr: A deep learning framework for recommendation of long-tail web services,” *IEEE Transactions on Services Computing*, 2017.
- [10] M. Balabanovic and Y. Shoham, “Fab: content-based, collaborative recommendation,” *Commun. ACM*, vol. 40, no. 3, pp. 66–72, 1997.
- [11] L. Baltrunas and X. Amatriain, “Towards time-dependant recommendation based on implicit feedback,” in *Workshop on context-aware recommender systems (CARS09)*, 2009, Conference Proceedings.
- [12] T. Baltrušaitis, C. Ahuja, and L. Morency, “Multimodal machine learning: A survey and taxonomy,” *arXiv preprint arXiv:1705.09406*, 2017.
- [13] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, “A theory of learning from different domains,” *Machine learning*, vol. 79, no. 1-2, pp. 151–175, 2010.
- [14] Y. Bengio, A. Courville, and P. Vincent, “Representation learning: A review and new perspectives,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 8, pp. 1798–1828, 2013.
- [15] Y. Bengio, “Learning deep architectures for ai,” *Found. Trends Mach. Learn.*, vol. 2, no. 1, pp. 1–127, 2009.
- [16] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, “A neural probabilistic language model,” *J. Mach. Learn. Res.*, vol. 3, pp. 1137–1155, Mar. 2003.
[Online]. Available: <http://dl.acm.org/citation.cfm?id=944919.944966>

- [17] S. Berkovsky and J. Freyne, “Group-based recipe recommendations: analysis of data aggregation strategies,” in *Proceedings of the fourth ACM conference on Recommender systems*. 1864732: ACM, 2010, Conference Proceedings, pp. 111–118.
- [18] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006.
- [19] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent dirichlet allocation,” *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.
- [20] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, “Recommender systems survey,” *Knowledge-based systems*, vol. 46, pp. 109–132, 2013.
- [21] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.
- [22] D. Boyd and N. Ellison, “Social network sites: definition, history, and scholarship,” *IEEE Engineering Management Review*, vol. 3, no. 38, pp. 16–31, 2010.
- [23] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender, “Learning to rank using gradient descent,” in *Proceedings of the 22nd international conference on Machine learning*, ACM. 1102363: ACM, 2005, Conference Proceedings, pp. 89–96.
- [24] R. Burke, M. P. OMahony, and N. J. Hurley, *Robust Collaborative Recommendation*. Boston, MA: Springer US, 2015, pp. 961–995. [Online]. Available: http://dx.doi.org/10.1007/978-1-4899-7637-6_28
- [25] P. G. Campos, F. Dez, and I. Cantador, “Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols,” *User*

- Modeling and User-Adapted Interaction*, vol. 24, no. 1, pp. 67–119, 2014. [Online]. Available: <http://dx.doi.org/10.1007/s11257-012-9136-x>
- [26] L. Candillier, F. Meyer, and F. Fessant, *Designing specific weighted similarity measures to improve collaborative filtering systems*. Springer, 2008, pp. 242–255.
- [27] I. Cantador, P. Brusilovsky, and T. Kuflik, “2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011),” in *Proceedings of the 5th ACM conference on Recommender systems*, ser. RecSys 2011. New York, NY, USA: ACM, 2011.
- [28] I. Cantador, I. Fernández-Tobas, S. Berkovsky, and P. Cremonesi, *Cross-Domain Recommender Systems*. Springer US, 2015, book section 27, pp. 919–959. [Online]. Available: http://dx.doi.org/10.1007/978-1-4899-7637-6_27
- [29] D. Cao, X. He, L. Miao, Y. An, C. Yang, and R. Hong, “Attentive group recommendation,” in *SIGIR*, 2018.
- [30] L. Cao, “Coupling learning of complex interactions,” *Information Processing & Management*, vol. 51, no. 2, pp. 167–186, 2015. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0306457314000855>
- [31] —, “Non-iid recommender systems: A review and framework of recommendation paradigm shifting,” *Engineering*, vol. 2, no. 2, pp. 212–224, 2016.
- [32] L. Cao, Y. Ou, and S. Y. Philip, “Coupled behavior analysis with applications,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 8, pp. 1378–1392, 2012.
- [33] J. Chen, H. Zhang, X. He, L. Nie, W. Liu, and T.-S. Chua, “Attentive collaborative filtering: Multimedia recommendation with item-and component-level

- attention,” in *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 2017, pp. 335–344.
- [34] M. Chen, Z. Xu, K. Q. Weinberger, and F. Sha, “Marginalized denoising autoencoders for domain adaptation,” in *Proceedings of the 29th International Conference on Machine Learning*, ser. ICML’12. USA: Omnipress, 2012, pp. 1627–1634. [Online]. Available: <http://dl.acm.org/citation.cfm?id=3042573.3042781>
- [35] S. Chen, J. L. Moore, D. Turnbull, and T. Joachims, “Playlist prediction via metric embedding,” in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD ’12. New York, NY, USA: ACM, 2012, pp. 714–722. [Online]. Available: <http://doi.acm.org/10.1145/2339530.2339643>
- [36] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” *arXiv preprint arXiv:1406.1078*, 2014.
- [37] F. Chollet *et al.*, “Keras,” <https://github.com/fchollet/keras>, 2015.
- [38] N. Cliff, “Orthogonal rotation to congruence,” *Psychometrika*, vol. 31, no. 1, pp. 33–42, 1966. [Online]. Available: <http://dx.doi.org/10.1007/BF02289455>
- [39] K. Crammer, M. Kearns, and J. Wortman, “Learning from multiple sources,” *JMLR*, 2008.
- [40] P. Cui, X. Wang, J. Pei, and W. Zhu, “A survey on network embedding,” *CoRR*, vol. abs/1711.08752, 2017. [Online]. Available: <http://arxiv.org/abs/1711.08752>

- [41] A. M. Dai, C. Olah, and Q. V. Le, “Document embedding with paragraph vectors,” *arXiv preprint arXiv:1507.07998*, 2015.
- [42] J. Davidson, B. Liebald, J. Liu, P. Nandy, T. Van Vleet, U. Gargi, S. Gupta, Y. He, M. Lambert, B. Livingston *et al.*, “The youtube video recommendation system,” in *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 2010, pp. 293–296.
- [43] M. de Gemmis, P. Lops, C. Musto, F. Narducci, and G. Semeraro, “Semantics-aware content-based recommender systems,” in *Recommender Systems Handbook*. Springer, 2015, pp. 119–159.
- [44] K. Deb, *Multi-objective optimization*. Springer, 2014, pp. 403–449.
- [45] S. Deng, L. Huang, G. Xu, X. Wu, and Z. Wu, “On deep learning for trust-aware recommendations in social networks,” *IEEE transactions on neural networks and learning systems*, vol. 28, no. 5, pp. 1164–1177, 2017.
- [46] M. Denil, A. Demiraj, and N. de Freitas, “Extraction of salient sentences from labelled documents,” *arXiv preprint arXiv:1412.6815*, 2014.
- [47] D. M. Dunlavy, T. G. Kolda, and E. Acar, “Temporal link prediction using matrix and tensor factorizations,” *ACM Trans. Knowl. Discov. Data*, vol. 5, no. 2, pp. 1–27, 2011.
- [48] C. Dwyer, “Privacy in the age of google and facebook,” *IEEE Technology and Society Magazine*, vol. 30, no. 3, pp. 58–63, 2011.
- [49] A. M. Elkahky, Y. Song, and X. He, “A multi-view deep learning approach for cross domain user modeling in recommendation systems,” in *Proceedings of the 24th International Conference on World Wide Web*, ser. WWW ’15. Republic and Canton of Geneva, Switzerland: International World Wide Web

- Conferences Steering Committee, 2015, Conference Proceedings, pp. 278–288. [Online]. Available: <https://doi.org/10.1145/2736277.2741667>
- [50] S. Feng, X. Li, Y. Zeng, G. Cong, Y. M. Chee, and Q. Yuan, “Personalized ranking metric embedding for next new poi recommendation,” in *Proceedings of the 24th International Conference on Artificial Intelligence*, ser. IJCAI’15. AAAI Press, 2015, pp. 2069–2075. [Online]. Available: <http://dl.acm.org/citation.cfm?id=2832415.2832536>
- [51] F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens, “Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation,” *IEEE Transactions on knowledge and data engineering*, vol. 19, no. 3, pp. 355–369, 2007.
- [52] N. E. Friedkin and E. C. Johnsen, *Social influence network theory: A sociological examination of small group dynamics*. Cambridge University Press, 2011, vol. 33.
- [53] E. Gabrilovich and S. Markovitch, “Computing semantic relatedness using wikipedia-based explicit semantic analysis,” in *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, ser. IJCAI’07. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2007, pp. 1606–1611. [Online]. Available: <http://dl.acm.org/citation.cfm?id=1625275.1625535>
- [54] M. Ge, C. Delgado-Battenfeld, and D. Jannach, “Beyond accuracy: evaluating recommender systems by coverage and serendipity,” in *Proceedings of the fourth ACM conference on Recommender systems*. 1864761: ACM, 2010, Conference Proceedings, pp. 257–260.
- [55] K. Georgiev and P. Nakov, “A non-iid framework for collaborative filtering with restricted boltzmann machines,” in *Proceedings of the 30th International*

- Conference on Machine Learning*, vol. 28. JMLR: W&CP, 2013, Conference Proceedings.
- [56] L. Getoor and B. Taskar, *Introduction to statistical relational learning*. MIT press, 2007.
- [57] X. Glorot, A. Bordes, and Y. Bengio, “Domain adaptation for large-scale sentiment classification: A deep learning approach,” in *Proceedings of the 28th international conference on machine learning (ICML-11)*, 2011, pp. 513–520.
- [58] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, “Using collaborative filtering to weave an information tapestry,” *Commun. ACM*, vol. 35, no. 12, pp. 61–70, Dec. 1992. [Online]. Available: <http://doi.acm.org/10.1145/138859.138867>
- [59] Y. Gong and Q. Zhang, “Hashtag recommendation using attention-based convolutional neural network.” in *IJCAI*, 2016, pp. 2782–2788.
- [60] J. Gorla, N. Lathia, S. Robertson, and J. Wang, “Probabilistic group recommendation via information matching,” in *Proceedings of the 22nd international conference on World Wide Web*. 2488432: International World Wide Web Conferences Steering Committee, 2013, Conference Proceedings, pp. 495–504.
- [61] G. Goth, “Deep or shallow, nlp is breaking out,” *Commun. ACM*, vol. 59, no. 3, pp. 13–16, 2016.
- [62] J. C. Gower and G. B. Dijkstra, *Procrustes problems*. Oxford University Press Oxford, 2004, vol. 3.
- [63] A. Graves, G. Wayne, M. Reynolds, T. Harley, I. Danihelka, A. Grabska-Barwinska, S. G. Colmenarejo, E. Grefenstette, T. Ramalho, J. Agapiou, A. P.

- Badia, K. M. Hermann, Y. Zwols, G. Ostrovski, A. Cain, H. King, C. Summerfield, P. Blunsom, K. Kavukcuoglu, and D. Hassabis, “Hybrid computing using a neural network with dynamic external memory,” *Nature*, vol. 538, no. 7626, pp. 471–476, 2016.
- [64] P. F. Groenen, P. Giaquinto, and H. L. Kiers, *An Improved Majorization Algorithm for Robust Procrustes Analysis*, ser. Studies in Classification, Data Analysis, and Knowledge Organization. Springer Berlin Heidelberg, 2005, book section 18, pp. 151–158. [Online]. Available: http://dx.doi.org/10.1007/3-540-27373-5_18
- [65] W. Gu, S. Dong, and Z. Zeng, “Increasing recommended effectiveness with markov chains and purchase intervals,” *Neural Computing and Applications*, vol. 25, no. 5, pp. 1153–1162, 2014. [Online]. Available: <http://dx.doi.org/10.1007/s00521-014-1599-8>
- [66] A. Gunawardana and G. Shani, *Evaluating Recommender Systems*. Springer US, 2015, book section 8, pp. 265–308. [Online]. Available: http://dx.doi.org/10.1007/978-1-4899-7637-6_8
- [67] I. Guy, “Social recommender systems,” in *Recommender Systems Handbook*. Springer, 2015, pp. 511–543.
- [68] W. L. Hamilton, R. Ying, and J. Leskovec, “Representation learning on graphs: Methods and applications,” *arXiv preprint arXiv:1709.05584*, 2017.
- [69] F. M. Harper and J. A. Konstan, “The movielens datasets: History and context,” *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 5, no. 4, p. 19, 2016.
- [70] R. A. Harshman, “Parafac2: Mathematical and technical notes,” *UCLA Working Papers in Phonetics*, vol. 22, pp. 30–44, 1972.

- [71] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [72] R. He and J. McAuley, “Vbpr: Visual bayesian personalized ranking from implicit feedback.” in *AAAI*, 2016, pp. 144–150.
- [73] X. He and T.-S. Chua, “Neural factorization machines for sparse predictive analytics,” in *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 2017, pp. 355–364.
- [74] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, “Neural collaborative filtering,” in *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2017, pp. 173–182.
- [75] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, “Evaluating collaborative filtering recommender systems,” *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 5–53, 2004.
- [76] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, “Session-based recommendations with recurrent neural networks,” *arXiv preprint arXiv:1511.06939*, 2015.
- [77] G. E. Hinton, “Training products of experts by minimizing contrastive divergence,” *Neural Computation*, vol. 14, no. 8, pp. 1771–1800, 2002. [Online]. Available: <http://dx.doi.org/10.1162/089976602760128018>
- [78] G. E. Hinton, S. Osindero, and Y.-W. Teh, “A fast learning algorithm for deep belief nets,” *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006. [Online]. Available: <http://dx.doi.org/10.1162/neco.2006.18.7.1527>

- [79] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *Science*, vol. 313, no. 5786, pp. 504–507, 2006.
- [80] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [81] T. Hofmann, “Latent semantic models for collaborative filtering,” *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 89–115, 2004.
- [82] L. Hu, J. Cao, G. Xu, L. Cao, Z. Gu, and W. Cao, “Deep modeling of group preferences for group-based recommendation,” in *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014, Conference Proceedings.
- [83] L. Hu, J. Cao, G. Xu, L. Cao, Z. Gu, and C. Zhu, “Personalized recommendation via cross-domain triadic factorization,” in *Proceedings of the 22nd international conference on World Wide Web*. 2488441: International World Wide Web Conferences Steering Committee, 2013, Conference Proceedings, pp. 595–606.
- [84] L. Hu, J. Cao, G. Xu, J. Wang, Z. Gu, and L. Cao, “Cross-domain collaborative filtering via bilinear multilevel analysis,” in *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*. 2540507: AAAI Press, 2013, Conference Proceedings, pp. 2626–2632.
- [85] L. Hu, L. Cao, J. Cao, Z. Gu, G. Xu, and J. Wang, “Improving the quality of recommendations for users and items in the tail of distribution,” *ACM Trans. Inf. Syst.*, vol. 35, no. 3, pp. 1–37, 2017.
- [86] L. Hu, L. Cao, J. Cao, Z. Gu, G. Xu, and D. Yang, “Learning informative priors from heterogeneous domains to improve recommendation in cold-start user domains,” *ACM Transactions on Information Systems (TOIS)*, vol. 35, no. 2, p. 13, 2016.

- [87] ———, “Learning informative priors from heterogeneous domains to improve recommendation in cold-start user domains,” *ACM Trans. Inf. Syst.*, vol. 35, no. 2, pp. 1–37, 2016.
- [88] L. Hu, L. Cao, S. Wang, G. Xu, J. Cao, and Z. Gu, “Diversifying personalized recommendation with user-session context,” in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 2017, Conference Proceedings, pp. 1858–1864.
- [89] L. Hu, W. Cao, J. Cao, G. Xu, L. Cao, and Z. Gu, “Bayesian heteroskedastic choice modeling on non-identically distributed linkages,” in *Data Mining (ICDM), 2014 IEEE International Conference on.* IEEE, 2014, Conference Proceedings, pp. 851–856.
- [90] X. Hu, X. Meng, and L. Wang, “Svd-based group recommendation approaches: an experimental study of moviepilot,” in *Proceedings of the 2nd Challenge on Context-Aware Movie Recommendation.* 2096117: ACM, 2011, Conference Proceedings, pp. 23–28.
- [91] Y. Hu, Y. Koren, and C. Volinsky, “Collaborative filtering for implicit feedback datasets,” in *Eighth IEEE International Conference on Data Mining*, 2008, Conference Proceedings, pp. 263–272.
- [92] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck, “Learning deep structured semantic models for web search using clickthrough data,” in *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management*, ser. CIKM ’13. New York, NY, USA: ACM, 2013, pp. 2333–2338. [Online]. Available: <http://doi.acm.org/10.1145/2505515.2505665>
- [93] M. Jamali and M. Ester, “Trustwalker: a random walk model for combining

- trust-based and item-based recommendation,” in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1557067: ACM, 2009, Conference Proceedings, pp. 397–406.
- [94] ———, “A matrix factorization technique with trust propagation for recommendation in social networks,” in *Proceedings of the fourth ACM conference on Recommender systems*, ACM. 1864736: ACM, 2010, Conference Proceedings, pp. 135–142.
- [95] A. Jameson and B. Smyth, *Recommendation to Groups*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007, vol. 4321, book section 20, pp. 596–627. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-72079-9_20
- [96] N. Jhanwar, S. Chaudhuri, G. Seetharaman, and B. Zavidovique, “Content based image retrieval using motif cooccurrence matrix,” *Image and Vision Computing*, vol. 22, no. 14, pp. 1211–1220, 2004.
- [97] X. Jia, X. Li, K. Li, V. Gopalakrishnan, G. Xun, and A. Zhang, “Collaborative restricted boltzmann machine for social event recommendation,” in *Advances in Social Networks Analysis and Mining (ASONAM), 2016 IEEE/ACM International Conference on*. IEEE, 2016, pp. 402–405.
- [98] S. Jian, L. Hu, L. Cao, and K. Lu, “Metric-based auto-instructor for learning mixed data representation,” in *AAAI*, 2018, pp. 3318–3325.
- [99] M. Jiang, P. Cui, F. Wang, W. Zhu, and S. Yang, “Scalable recommendation with social contextual information,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 11, pp. 2789–2802, 2014.
- [100] Y. Jin, *Multi-objective machine learning*. Springer Science & Business Media, 2006, vol. 16.

- [101] I. T. Jolliffe, "Principal component analysis and factor analysis," in *Principal component analysis*. Springer, 1986, pp. 115–128.
- [102] A. Jsang, G. Guo, M. Pini, F. Santini, and Y. Xu, *Combining Recommender and Reputation Systems to Produce Better Online Advice*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2013, vol. 8234, book section 12, pp. 126–138. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-41550-0_12
- [103] A. Jsang and R. Ismail, "The beta reputation system," in *Proceedings of the 15th bled electronic commerce conference*, 2002, Conference Proceedings, pp. 41–55.
- [104] A. Jsang, X. Luo, and X. Chen, *Continuous ratings in discrete bayesian reputation systems*. Springer, 2008, pp. 151–166.
- [105] A. Jsang and W. Quattrociocchi, *Advanced Features in Bayesian Reputation Systems*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2009, vol. 5695, book section 11, pp. 105–114. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-03748-1_11
- [106] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver, "Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering," in *Proceedings of the fourth ACM conference on Recommender systems*. 1864727: ACM, 2010, Conference Proceedings, pp. 79–86.
- [107] H. A. L. Kiers, J. M. F. ten Berge, and R. Bro, "Parafac2part i. a direct fitting algorithm for the parafac2 model," *Journal of Chemometrics*, vol. 13, no. 3-4, pp. 275–294, 1999. [Online]. Available: [http://dx.doi.org/10.1002/\(SICI\)1099-128X\(199905/08\)13:3/4<275::AID-CEM543>3.0.CO;2-B](http://dx.doi.org/10.1002/(SICI)1099-128X(199905/08)13:3/4<275::AID-CEM543>3.0.CO;2-B)

- [108] H. A. Kiers, “Weighted least squares fitting using ordinary least squares algorithms,” *Psychometrika*, vol. 62, no. 2, pp. 251–266, 1997.
- [109] Y.-D. Kim and S. Choi, “Scalable variational bayesian matrix factorization,” in *1st Workshop on Large-Scale Recommender Systems*, 2013, Conference Proceedings.
- [110] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush, “Character-aware neural language models,” in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, ser. AAAI’16. AAAI Press, 2016, pp. 2741–2749. [Online]. Available: <http://dl.acm.org/citation.cfm?id=3016100.3016285>
- [111] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [112] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*, 2013.
- [113] R. Kiros, Y. Zhu, R. R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler, “Skip-thought vectors,” in *Advances in neural information processing systems*, 2015, pp. 3294–3302.
- [114] T. Kolda and B. Bader, “Tensor decompositions and applications,” *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.
- [115] M. Kompan and M. Bielikova, “Content-based news recommendation,” in *E-Commerce and Web Technologies: 11th International Conference, EC-Web 2010, Bilbao, Spain, September 1-3, 2010, Proceedings*, F. Buccafurri and G. Semeraro, Eds., vol. 61, Springer Science & Business Media. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, p. 61.
- [116] Y. Koren, “Collaborative filtering with temporal dynamics,” *Commun. ACM*, vol. 53, no. 4, pp. 89–97, 2010.

- [117] Y. Koren, R. Bell, and C. Volinsky, “Matrix factorization techniques for recommender systems,” *Computer*, vol. 42, no. 8, pp. 30–37, 2009.
- [118] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
- [119] A. Lacerda, “Multi-objective ranked bandits for recommender systems,” *Neurocomputing*, vol. 246, pp. 12–24, 2017.
- [120] T. K. Landauer, *Latent semantic analysis*. Wiley Online Library, 2006.
- [121] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang, “A tutorial on energy-based learning,” *Predicting Structured Data*, vol. 1, p. 0, 2006.
- [122] H.-H. Lee and W.-G. Teng, “Incorporating multi-criteria ratings in recommendation systems,” in *Information Reuse and Integration, 2007. IRI 2007. IEEE International Conference on*. IEEE, 2007, pp. 273–278.
- [123] C. Lei, D. Liu, W. Li, Z.-J. Zha, and H. Li, “Comparative deep learning of hybrid representations for image recommendations,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2545–2553.
- [124] K. Lerman, “Social networks and social information filtering on digg,” *arXiv preprint cs/0612046*, 2006.
- [125] M. Levy and K. Bosteels, “Music recommendation and the long tail,” in *1st Workshop On Music Recommendation And Discovery (WOMRAD)*. ACM RecSys, 2010, Conference Proceedings.
- [126] B. Li, “Cross-domain collaborative filtering: A brief survey,” in *Proceedings of the 2011 IEEE 23rd International Conference on Tools with Artificial Intelli-*

- gence*. 2084563: IEEE Computer Society, 2011, Conference Proceedings, pp. 1085–1086.
- [127] B. Li, Q. Yang, and X. Xue, “Can movies and books collaborate? cross-domain collaborative filtering for sparsity reduction,” in *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI-09)*, 2009, Conference Proceedings, pp. 2052–2057.
- [128] ———, “Transfer learning for collaborative filtering via a rating-matrix generative model,” in *Proceedings of the 26th Annual International Conference on Machine Learning*. 1553454: ACM, 2009, Conference Proceedings, pp. 617–624.
- [129] L. Li, W. Chu, J. Langford, and R. E. Schapire, “A contextual-bandit approach to personalized news article recommendation,” in *Proceedings of the 19th international conference on World wide web*. ACM, 2010, pp. 661–670.
- [130] Q. Li, C. Wang, and G. Geng, “Improving personalized services in mobile commerce by a novel multicriteria rating approach,” in *Proceedings of the 17th International Conference on World Wide Web*, ser. WWW ’08. New York, NY, USA: ACM, 2008, pp. 1235–1236. [Online]. Available: <http://doi.acm.org/10.1145/1367497.1367743>
- [131] S. Li, J. Kawale, and Y. Fu, “Deep collaborative filtering via marginalized denoising auto-encoder,” in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 2015, pp. 811–820.
- [132] J. Lian, F. Zhang, X. Xie, and G. Sun, “Cccfnet: a content-boosted collaborative filtering neural network for cross domain recommender systems,” in

- Proceedings of the 26th International Conference on World Wide Web Companion*. International World Wide Web Conferences Steering Committee, 2017, pp. 817–818.
- [133] Y. J. Lim and Y. W. Teh, “Variational bayesian approach to movie rating prediction,” in *Proceedings of KDD Cup and Workshop*, vol. 7. ACM, 2007, Conference Proceedings, pp. 15–27.
- [134] J. Liu, P. Dolan, and E. R. Pedersen, “Personalized news recommendation based on click behavior,” in *Proceedings of the 15th international conference on Intelligent user interfaces*. ACM, 2010, pp. 31–40.
- [135] N. N. Liu and Q. Yang, “Eigenrank: a ranking-oriented approach to collaborative filtering,” in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. 1390351: ACM, 2008, Conference Proceedings, pp. 83–90.
- [136] M. Long, J. Wang, G. Ding, W. Cheng, X. Zhang, and W. Wang, “Dual transfer learning,” in *Proceedings of the 12th SIAM International Conference on Data Mining*, 2012, Conference Proceedings, pp. 540–551. [Online]. Available: <http://epubs.siam.org/doi/abs/10.1137/1.9781611972825.47>
- [137] B. Loni, Y. Shi, M. Larson, and A. Hanjalic, *Cross-Domain Collaborative Filtering with Factorization Machines*, ser. Lecture Notes in Computer Science. Springer International Publishing, 2014, vol. 8416, book section 72, pp. 656–661. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-06028-6_72
- [138] P. Lops, M. De Gemmis, and G. Semeraro, “Content-based recommender systems: State of the art and trends,” in *Recommender systems handbook*. Springer, 2011, pp. 73–105.

- [139] P. Loyola, C. Liu, and Y. Hirate, “Modeling user session and intent with an attention-based encoder-decoder architecture,” in *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 2017, pp. 147–151.
- [140] Z. Lu, Z. Dou, J. Lian, X. Xie, and Q. Yang, “Content-based collaborative filtering for news topic recommendation.” in *AAAI*, 2015, pp. 217–223.
- [141] H. Ma, C. Liu, I. King, and M. Lyu, “Probabilistic factor models for web site recommendation,” in *SIGIR*, vol. 11, 2011, Conference Proceedings, pp. 265–274.
- [142] H. Ma, H. Yang, M. R. Lyu, and I. King, “Sorec: social recommendation using probabilistic matrix factorization,” in *Proceeding of the 17th ACM conference on Information and knowledge management*, ACM. 1458205: ACM, 2008, Conference Proceedings, pp. 931–940.
- [143] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King, “Recommender systems with social regularization,” in *Proceedings of the fourth ACM international conference on Web search and data mining*, ACM. 1935877: ACM, 2011, Conference Proceedings, pp. 287–296.
- [144] N. Manouselis and C. Costopoulou, “Analysis and classification of multi-criteria recommender systems,” *World Wide Web*, vol. 10, no. 4, pp. 415–441, 2007.
- [145] J. Masthoff, *Group Recommender Systems: Combining Individual Models*. Springer US, 2011, book section 21, pp. 677–702. [Online]. Available: http://dx.doi.org/10.1007/978-0-387-85820-3_21
- [146] —, “Group recommender systems: aggregation, satisfaction and group attributes,” in *recommender systems handbook*. Springer, 2015, pp. 743–776.

- [147] J. McAuley, R. Pandey, and J. Leskovec, “Inferring networks of substitutable and complementary products,” in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2783381: ACM, 2015, Conference Proceedings, pp. 785–794.
- [148] P. McCullagh, “Generalized linear models,” *European Journal of Operational Research*, vol. 16, no. 3, pp. 285–292, 1984.
- [149] S. M. McNee, J. Riedl, and J. A. Konstan, “Being accurate is not enough: how accuracy metrics have hurt recommender systems,” in *CHI '06 Extended Abstracts on Human Factors in Computing Systems*. 1125659: ACM, 2006, Conference Proceedings, pp. 1097–1101.
- [150] P. Melville and V. Sindhwani, “Recommender systems,” in *Encyclopedia of machine learning*. Springer, 2011, pp. 829–838.
- [151] S. Meyffret, E. Guillot, L. Mдини, and F. Laforest, “Red: a rich opinions dataset for recommender systems,” LIRIS, Report, 2012 2012. [Online]. Available: <https://hal.archives-ouvertes.fr/hal-01010246>
- [152] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” *arXiv preprint arXiv:1301.3781*, 2013.
- [153] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in *Advances in neural information processing systems*. 2999959: Curran Associates Inc., 2013, Conference Proceedings, pp. 3111–3119.
- [154] M.-A. Mittermayer and G. Knolmayer, *Text mining systems for market response to news: A survey*. Institut für Wirtschaftsinformatik der Universität Bern, 2006.

- [155] A. Mnih and K. Kavukcuoglu, “Learning word embeddings efficiently with noise-contrastive estimation,” in *Advances in neural information processing systems*, 2013, Conference Proceedings, pp. 2265–2273.
- [156] A. Mnih and Y. W. Teh, “A fast and simple algorithm for training neural probabilistic language models,” *arXiv preprint arXiv:1206.6426*, 2012.
- [157] C. Musto, G. Semeraro, M. de Gemmis, and P. Lops, “Learning word embeddings from wikipedia for content-based recommender systems,” in *European Conference on Information Retrieval*. Springer, 2016, pp. 729–734.
- [158] M. Mrup, “Applications of tensor (multiway array) factorizations and decompositions in data mining,” *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 1, no. 1, pp. 24–40, 2011. [Online]. Available: <http://dx.doi.org/10.1002/widm.1>
- [159] P. Niyogi, F. Girosi, and T. Poggio, “Incorporating prior information in machine learning by creating virtual examples,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2196–2209, 1998.
- [160] T. D. Noia, V. C. Ostuni, P. Tomeo, and E. D. Sciascio, “Sprank: Semantic path-based ranking for top-n recommendations using linked open data,” *ACM Trans. Intell. Syst. Technol.*, vol. 8, no. 1, pp. 9:1–9:34, Sep. 2016. [Online]. Available: <http://doi.acm.org/10.1145/2899005>
- [161] C. Olah, “Understanding lstm networks,” Aug. 2015. [Online]. Available: <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>
- [162] A. v. d. Oord, S. Dieleman, and B. Schrauwen, “Deep content-based music recommendation,” in *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS’13.

- USA: Curran Associates Inc., 2013, pp. 2643–2651. [Online]. Available: <http://dl.acm.org/citation.cfm?id=2999792.2999907>
- [163] Y. Ouyang, W. Liu, W. Rong, and Z. Xiong, “Autoencoder-based collaborative filtering,” in *International Conference on Neural Information Processing*. Springer, 2014, pp. 284–291.
- [164] M. OConnor, D. Cosley, J. A. Konstan, and J. Riedl, “Polylens: a recommender system for groups of users,” in *ECSCW 2001*. Springer, 2002, Conference Proceedings, pp. 199–218.
- [165] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang, “One-class collaborative filtering,” in *Eighth IEEE International Conference on Data Mining*. IEEE, 2008, Conference Proceedings, pp. 502–511.
- [166] S. Pan and Q. Yang, “A survey on transfer learning,” *IEEE TKDE*, vol. 22, no. 10, pp. 1345–1359, 2010.
- [167] W. Pan, E. Xiang, N. Liu, and Q. Yang, “Transfer learning in collaborative filtering for sparsity reduction,” in *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, 2010, Conference Proceedings.
- [168] S. E. Park, S. Lee, and S. g. Lee, “Session-based collaborative filtering for predicting the next song,” in *2011 First ACIS/JNU International Conference on Computers, Networks, Systems and Industrial Engineering*, May 2011, Conference Proceedings, pp. 353–358.
- [169] Y.-J. Park and A. Tuzhilin, “The long tail of recommender systems and how to leverage it,” in *Proceedings of the 2008 ACM conference on Recommender systems*. 1454012: ACM, 2008, Conference Proceedings, pp. 11–18.

- [170] M. Pazzani and D. Billsus, “Learning and revising user profiles: The identification of interesting web sites,” *Machine learning*, vol. 27, no. 3, pp. 313–331, 1997.
- [171] J. Pennington, R. Socher, and C. Manning, “Glove: Global vectors for word representation,” in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.
- [172] I. Porteous, A. Asuncion, and M. Welling, “Bayesian matrix factorization with side information and dirichlet process mixtures,” in *AAAI*, ser. 2010, 2010, Book. [Online]. Available: <https://www.aaai.org/ocs/index.php/AAAI/AAAI10/paper/view/1871>
- [173] M. Quadrana, A. Karatzoglou, B. Hidasi, and P. Cremonesi, “Personalizing session-based recommendations with hierarchical recurrent neural networks,” in *Proceedings of the Eleventh ACM Conference on Recommender Systems*, ser. RecSys ’17. New York, NY, USA: ACM, 2017, pp. 130–137. [Online]. Available: <http://doi.acm.org/10.1145/3109859.3109896>
- [174] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” in *Advances in neural information processing systems*, 2015, pp. 91–99.
- [175] S. Rendle, “Factorization machines,” in *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, 2010, Conference Proceedings, pp. 995–1000.
- [176] —, “Factorization machines with libfm,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 3, no. 3, p. 57, 2012.
- [177] —, “Scaling factorization machines to relational data,” in *Proceedings of the VLDB Endowment*, vol. 6, no. 5. VLDB Endowment, 2013, pp. 337–348.

- [178] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, “Bpr: Bayesian personalized ranking from implicit feedback,” in *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*, AUAI Press. 1795167: AUAI Press, 2009, Conference Proceedings, pp. 452–461.
- [179] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme, “Factorizing personalized markov chains for next-basket recommendation,” in *Proceedings of the 19th international conference on World wide web*. 1772773: ACM, 2010, Conference Proceedings, pp. 811–820.
- [180] S. Rendle, L. B. Marinho, A. Nanopoulos, and L. Schmidt-Thieme, “Learning optimal ranking with tensor factorization for tag recommendation,” in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1557100: ACM, 2009, Conference Proceedings, pp. 727–736.
- [181] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, “GroupLens: an open architecture for collaborative filtering of netnews,” in *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, ACM. 192905: ACM, 1994, Conference Proceedings, pp. 175–186.
- [182] M. T. Ribeiro, N. Ziviani, E. S. D. Moura, I. Hata, A. Lacerda, and A. Veloso, “Multiobjective pareto-efficient approaches for recommender systems,” *ACM Trans. Intell. Syst. Technol.*, vol. 5, no. 4, pp. 53:1–53:20, Dec. 2014. [Online]. Available: <http://doi.acm.org/10.1145/2629350>
- [183] F. Ricci, L. Rokach, and B. Shapira, *Recommender Systems Handbook*, 2nd ed. Springer Publishing Company, Incorporated, 2015.
- [184] ———, “Recommender systems: introduction and challenges,” in *Recommender systems handbook*. Springer, 2015, pp. 1–34.

- [185] A. Rozantsev, M. Salzmann, and P. Fua, “Beyond sharing weights for deep domain adaptation,” *IEEE TPAMI*, 2018.
- [186] A. Said, “Evaluating the accuracy and utility of recommender systems,” Thesis, Universitätsbibliothek der Technischen Universität Berlin, 2013.
- [187] A. Said, S. Berkovsky, and E. W. D. Luca, “Group recommendation in context,” in *Proceedings of the 2nd Challenge on Context-Aware Movie Recommendation*. 2096113: ACM, 2011, Conference Proceedings, pp. 2–4.
- [188] R. Salakhutdinov and A. Mnih, “Bayesian probabilistic matrix factorization using markov chain monte carlo,” in *Proceedings of the 25th international conference on Machine learning*. 1390267: ACM, 2008, Conference Proceedings, pp. 880–887.
- [189] R. Salakhutdinov, A. Mnih, and G. Hinton, “Restricted boltzmann machines for collaborative filtering,” in *Proceedings of the 24th international conference on Machine learning*, ACM. 1273596: ACM, 2007, Conference Proceedings, pp. 791–798.
- [190] R. Salakhutdinov, J. B. Tenenbaum, and A. Torralba, “Learning with hierarchical-deep models,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 8, pp. 1958–1971, 2013.
- [191] G. Salton, *The SMART Retrieval System—Experiments in Automatic Document Processing*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1971.
- [192] G. Salton, A. Wong, and C. S. Yang, “A vector space model for automatic indexing,” *Commun. ACM*, vol. 18, no. 11, pp. 613–620, Nov. 1975. [Online]. Available: <http://doi.acm.org/10.1145/361219.361220>
- [193] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Item-based collaborative filtering recommendation algorithms,” in *Proceedings of the 10th international*

- conference on World Wide Web*, ACM. 372071: ACM, 2001, Conference Proceedings, pp. 285–295.
- [194] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, *Collaborative Filtering Recommender Systems*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007, vol. 4321, book section 9, pp. 291–324. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-72079-9_9
- [195] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock, “Methods and metrics for cold-start recommendations,” in *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*. 564421: ACM, 2002, Conference Proceedings, pp. 253–260.
- [196] H. Shan and A. Banerjee, “Generalized probabilistic matrix factorizations for collaborative filtering,” in *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, 2010, Conference Proceedings, pp. 1025–1030.
- [197] G. Shani, D. Heckerman, and R. I. Brafman, “An mdp-based recommender system,” *Journal of Machine Learning Research*, vol. 6, no. Sep, pp. 1265–1295, 2005.
- [198] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, “Mastering the game of go with deep neural networks and tree search,” *nature*, vol. 529, no. 7587, pp. 484–489, 2016. [Online]. Available: <http://dx.doi.org/10.1038/nature16961>
- [199] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, “Mastering

- the game of go without human knowledge,” *Nature*, vol. 550, p. 354, 2017. [Online]. Available: <http://dx.doi.org/10.1038/nature24270>
- [200] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton *et al.*, “Mastering the game of go without human knowledge,” *Nature*, vol. 550, no. 7676, p. 354, 2017. [Online]. Available: <http://dx.doi.org/10.1038/nature24270>
- [201] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.
- [202] A. P. Singh and G. J. Gordon, “Relational learning via collective matrix factorization,” in *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM. 1401969: ACM, 2008, Conference Proceedings, pp. 650–658.
- [203] A. Singh and G. Gordon, *A Unified View of Matrix Factorization Models*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2008, vol. 5212, book section 24, pp. 358–373. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-87481-2_24
- [204] N. Srebro and T. Jaakkola, “Weighted low-rank approximations,” in *Proceedings of the Twentieth International Conference on Machine Learning*, vol. 20, 2003, Conference Proceedings, p. 720.
- [205] N. Srebro, J. Rennie, and T. Jaakkola, “Maximum-margin matrix factorization,” in *Advances in neural information processing systems*, vol. 17, 2005, Conference Proceedings, pp. 1329–1336.
- [206] X. Su and T. M. Khoshgoftaar, “A survey of collaborative filtering techniques,” *Advances in artificial intelligence*, vol. 2009, p. 4, 2009.

- [207] S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus, “End-to-end memory networks,” in *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds., 2015, pp. 2440–2448. [Online]. Available: <http://papers.nips.cc/paper/5846-end-to-end-memory-networks>
- [208] J. Tang, S. Wu, J. Sun, and H. Su, “Cross-domain collaboration recommendation,” in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2339730: ACM, 2012, Conference Proceedings, pp. 1285–1293.
- [209] J. Tang, H. Gao, H. Liu, and A. Das Sarma, “etrust: Understanding trust evolution in an online world,” in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2012, pp. 253–261.
- [210] J. Tang, H. Gao, H. Liu, and A. D. Sarma, “etrust: understanding trust evolution in an online world,” in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM. 2339574: ACM, 2012, Conference Proceedings, pp. 253–261.
- [211] J. Tang, X. Hu, and H. Liu, “Social recommendation: a review,” *Social Network Analysis and Mining*, vol. 3, no. 4, pp. 1113–1133, 2013.
- [212] M. A. Tayebi, M. Jamali, M. Ester, U. Glässer, and R. Frank, “Crimewalker: a recommendation model for suspect investigation,” in *Proceedings of the fifth ACM conference on Recommender systems*. ACM, 2011, pp. 173–180.
- [213] T. Tieleman, “Training restricted boltzmann machines using approximations to the likelihood gradient,” in *Proceedings of the 25th international conference*

- on Machine learning*. 1390290: ACM, 2008, Conference Proceedings, pp. 1064–1071.
- [214] K. Train, *Discrete choice methods with simulation*. Cambridge university press, 2003.
- [215] B. Twardowski, “Modelling contextual information in session-aware recommender systems with neural networks,” in *RecSys, 2016, Conference Proceedings*, pp. 273–276.
- [216] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, “Adversarial discriminative domain adaptation,” in *Computer Vision and Pattern Recognition (CVPR)*, vol. 1, no. 2, 2017, p. 4.
- [217] S. Vargas and P. Castells, “Rank and relevance in novelty and diversity metrics for recommender systems,” in *Proceedings of the fifth ACM conference on Recommender systems*. ACM, 2011, Conference Proceedings, pp. 109–116.
- [218] V. Vasuki, N. Natarajan, Z. Lu, and I. S. Dhillon, “Affiliation recommendation using auxiliary networks,” in *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 2010, pp. 103–110.
- [219] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, “Extracting and composing robust features with denoising autoencoders,” in *Proceedings of the 25th international conference on Machine learning*. 1390294: ACM, 2008, Conference Proceedings, pp. 1096–1103.
- [220] T. D. Q. Vinh, T.-A. N. Pham, G. Cong, and X.-L. Li, “Attention-based group recommendation,” *arXiv preprint arXiv:1804.04327*, 2018.
- [221] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, “Show and tell: A neural image caption generator,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3156–3164.

- [222] C. Wang and D. M. Blei, “Collaborative topic modeling for recommending scientific articles,” in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2020480: ACM, 2011, Conference Proceedings, pp. 448–456.
- [223] H. Wang, N. Wang, and D.-Y. Yeung, “Collaborative deep learning for recommender systems,” in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '15. New York, NY, USA: ACM, 2015, pp. 1235–1244. [Online]. Available: <http://doi.acm.org/10.1145/2783258.2783273>
- [224] S. Wang, M. Gong, H. Li, and J. Yang, “Multi-objective optimization for long tail recommendation,” *Knowledge-Based Systems*, vol. 104, pp. 145–155, 2016.
- [225] S. Wang, L. Hu, L. Cao, X. Huang, D. Lian, and W. Liu, “Attention-based transactional context embedding for next-item recommendation,” in *Proceedings of Thirty-Second AAAI Conference on Artificial Intelligence*, 2018, Conference Proceedings.
- [226] S. Wang, S. Huang, T.-Y. Liu, J. Ma, Z. Chen, and J. Veijalainen, “Ranking-oriented collaborative filtering: A listwise approach,” *ACM Trans. Inf. Syst.*, vol. 35, no. 2, pp. 10:1–10:28, Sep. 2016. [Online]. Available: <http://doi.acm.org/10.1145/2960408>
- [227] S. Wang, Y. Wang, J. Tang, K. Shu, S. Ranganath, and H. Liu, “What your images reveal: Exploiting visual contents for point-of-interest recommendation,” in *Proceedings of the 26th International Conference on World Wide Web*, ser. WWW '17. Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee, 2017, pp. 391–400. [Online]. Available: <https://doi.org/10.1145/3038912.3052638>

- [228] X. Wang, X. He, L. Nie, and T.-S. Chua, “Item silk road: Recommending items from information domains to social users,” in *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 2017, pp. 185–194.
- [229] X. Wang, L. Yu, K. Ren, G. Tao, W. Zhang, Y. Yu, and J. Wang, “Dynamic attention deep model for article recommendation by learning human editors’ demonstration,” in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD ’17. New York, NY, USA: ACM, 2017, pp. 2051–2059. [Online]. Available: <http://doi.acm.org/10.1145/3097983.3098096>
- [230] M. Welling and G. Hinton, *A new learning algorithm for mean field Boltzmann machines*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2002, vol. 2415, book section 57, pp. 351–357. [Online]. Available: http://dx.doi.org/10.1007/3-540-46084-5_57
- [231] C. K. Williams and F. V. Agakov, “Products of gaussians and probabilistic minor component analysis,” *Neural Computation*, vol. 14, no. 5, pp. 1169–1182, 2002.
- [232] X. Wu, Q. Liu, E. Chen, L. He, J. Lv, C. Cao, and G. Hu, “Personalized next-song recommendation in online karaokes,” in *Proceedings of the 7th ACM Conference on Recommender Systems*, ser. RecSys ’13. New York, NY, USA: ACM, 2013, pp. 137–140. [Online]. Available: <http://doi.acm.org/10.1145/2507157.2507215>
- [233] Y. Wu, C. DuBois, A. X. Zheng, and M. Ester, “Collaborative denoising auto-encoders for top-n recommender systems,” in *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. ACM, 2016, pp. 153–162.

- [234] L. Xiao, Z. Min, Z. Yongfeng, G. Zhaoquan, L. Yiqun, and M. Shaoping, “Fairness-aware group recommendation with pareto-efficiency,” in *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 2017, pp. 107–115.
- [235] C. Xiong, S. Merity, and R. Socher, “Dynamic memory networks for visual and textual question answering,” in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ser. ICML’16. JMLR.org, 2016, pp. 2397–2406. [Online]. Available: <http://dl.acm.org/citation.cfm?id=3045390.3045643>
- [236] L. Xiong, X. Chen, T. Huang, J. Schneider, and J. Carbonell, “Temporal collaborative filtering with bayesian probabilistic tensor factorization,” in *Proceedings of SIAM Data Mining*, vol. 2010, 2010, Conference Proceedings.
- [237] R. Xu and D. Wunsch, “Survey of clustering algorithms,” *IEEE Transactions on neural networks*, vol. 16, no. 3, pp. 645–678, 2005.
- [238] Z. Xu, C. Chen, T. Lukasiewicz, Y. Miao, and X. Meng, “Tag-aware personalized recommendation using a deep-semantic similarity model with negative sampling,” in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, ser. CIKM ’16. New York, NY, USA: ACM, 2016, pp. 1921–1924. [Online]. Available: <http://doi.acm.org/10.1145/2983323.2983874>
- [239] S.-H. Yang, B. Long, A. Smola, N. Sadagopan, Z. Zheng, and H. Zha, “Like like alike: joint friendship and interest propagation in social networks,” in *Proceedings of the 20th international conference on World wide web*. 1963481: ACM, 2011, Conference Proceedings, pp. 537–546.
- [240] X. Yang, H. Steck, Y. Guo, and Y. Liu, “On top-k recommendation using

- social networks,” in *Proceedings of the sixth ACM conference on Recommender systems*. ACM, 2012, pp. 67–74.
- [241] X. Yang, H. Steck, and Y. Liu, “Circle-based recommendation in online social networks,” in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2012, pp. 1267–1275.
- [242] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, “Hierarchical attention networks for document classification,” in *Proceedings of NAACL-HLT*, 2016, pp. 1480–1489.
- [243] H. Yin, B. Cui, J. Li, J. Yao, and C. Chen, “Challenging the long tail recommendation,” *Proc. VLDB Endow.*, vol. 5, no. 9, pp. 896–907, 2012.
- [244] J. Yin, Z. Zheng, and L. Cao, “Uspan: an efficient algorithm for mining high utility sequential patterns,” in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2012, pp. 660–668.
- [245] Q. You, J. Luo, H. Jin, and J. Yang, “Robust image sentiment analysis using progressively trained and domain transferred deep networks.” in *AAAI*, 2015, pp. 381–388.
- [246] Z. Yu, X. Zhou, Y. Hao, and J. Gu, “Tv program recommendation for multiple viewers based on user profile merging,” *User Modeling and User-Adapted Interaction*, vol. 16, no. 1, pp. 63–82, 2006. [Online]. Available: <http://dx.doi.org/10.1007/s11257-006-9005-6>
- [247] S. Zhang, L. Yao, and A. Sun, “Deep learning based recommender system: A survey and new perspectives,” *arXiv preprint arXiv:1707.07435*, 2017.
- [248] Y. Zhang, Y. Zhuang, J. Wu, and L. Zhang, “Applying probabilistic latent semantic analysis to multi-criteria recommender system,” *AI*

- Commun.*, vol. 22, no. 2, pp. 97–107, Apr. 2009. [Online]. Available: <http://dl.acm.org/citation.cfm?id=1574514.1574517>
- [249] Y. Zheng, C. Liu, B. Tang, and H. Zhou, “Neural autoregressive collaborative filtering for implicit feedback,” in *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, ser. DLRS 2016. New York, NY, USA: ACM, 2016, pp. 2–6. [Online]. Available: <http://doi.acm.org/10.1145/2988450.2988453>
- [250] Y. Zheng, B. Tang, W. Ding, and H. Zhou, “A neural autoregressive approach to collaborative filtering,” in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ser. ICML’16. JMLR.org, 2016, pp. 764–773. [Online]. Available: <http://dl.acm.org/citation.cfm?id=3045390.3045472>
- [251] T. Zhou, Z. Kuscsik, J.-G. Liu, M. Medo, J. R. Wakeling, and Y.-C. Zhang, “Solving the apparent diversity-accuracy dilemma of recommender systems,” in *Proceedings of the National Academy of Sciences*, vol. 107, 2010, Conference Proceedings, pp. 4511–4515.
- [252] T. Zhou, J. Ren, M. Medo, and Y.-C. Zhang, “Bipartite network projection and personal recommendation,” *Physical Review E*, vol. 76, no. 4, p. 046115, 2007.