FreshGraph: A spam-aware recommender system for cold start problem

Di Zhang, Qian Zhang, Guanquan Zhang, Jie Lu
Decision Systems and e-Service Intelligence Laboratory,
Centre for Artificial Intelligence
University of Technology Sydney, Australia
Di.Zhang-7@student.uts.edu.au, Qian.Zhang-1@uts.edu.au
Guangquan.Zhang@uts.edu.au, Jie.Lu@uts.edu.au

***Abstract*—Recommender systems provide personalized recommendation to help users levitating from information overload. Collaborative filtering based recommendation methods are playing a dominant role in the industry because of its versatility and simplicity. However, its performance suffers from sparse data, and being less effective in cold-start problem settings. In real world scenario, when users are recommended with items, it is very easy to overwhelm the target users with impersonalized information, which drives away valuable audience. In this paper, we propose a two-steps spam aware recommendation framework to effectively recommend new items to target users. By utilizing heterogeneous information graph structure, we first use item-user Meta-Path similarity measure for user candidate selection. Then we use entropy encoding measurement to identify false positive from candidate list to prevent possible spam from happening. The proposed method leverages the semantic information that persists inside the graph structure, which not only considers item content features, but also take user activeness into account for more effective audience targeting. The proposed method produces an explainable top-K user list for the new item, while K is a trailed number to each given item individually. Meanwhile, the proposed method is also adaptive to data change overtime, while capable of processing requests in a real-time fashion.**

***Index Terms*—recommender systems, heterogeneous information graph, collaborative filtering, abnormal detection**

I. Introduction

Recommender system is an indispensable technology in this big data era [1]. It evolves around our everyday life on multiple fronts. From daily curated news feed, to online shopping portal, to music play list that we listen, and movies we watch. Recommender systems help us to find personalized products or services from the ever-increasing information overloads and make life more efficient and focused.

The boosting of recommender systems is in 2006, when a 100 million data-sets were released by Netflix [2] and Netflix Prize follows in 2009. The main technology collaborative filtering is greatly prompted and still remains one of the most important forefront in the field of recommender systems in both research and industry. Collaborative Filter's popularity are not only because of its effectiveness. Its simplicity in data engineering and low dependency in domain knowledge greatly lowered the barrier in real-world adoption. However, collaborative filtering is known for its drawbacks in cold start problems when there is no or little historical data available for the new item.

Many studies have shown promising results that the information presented in knowledge graph can effectively improve the shortcoming [3] [4]. Heterogeneous information graph can effectively enhance the performance of recommender systems as user-item relationships can be learned and leveraged from the heterogeneous network [5] [6]. Node2vec is an effective feature learning approach using network sampling strategy, such as breadth-first sampling and depth-first sampling [7]. Entity2Vec demonstrates how such technique can be adopted in recommendation system domain beyond link completion [8].

Along with the evolution of the recommendation techniques, the way of how we deliver our recommendation to the end user is also changing. Instead of passively waiting users to see recommendations on hosting site, more and more business is actively pushing personalized content to users. SMS, push notification, or news letter, just to name a few, are the new norm for business keeping customer connected. Such proactive approach makes cold start problem extra dangerous, as, if not handled well, it can upset customer with unrelated information, causing user turning off notification or marking newsletter as spam. That not only discourage the users' loyalty but also damages brand image. As a result, most business would rather under-reaching than spamming users with irrelevant recommendations.

Further more, item and user are individually unique. A real world example, big budget, studio productions mostly are far more popular than indie films. As shown in Fig. 1 which uses Movielens 100K data set [1]. We can see, items popularity and user activeness are vastly different from each other according to the histogram. Traditional Top-K recommendation often have limited reach for the popular item, while running high risk of spamming with niche products.

In this paper we propose FreshGraph, a novel a two-steps spam aware recommendation framework that targets the cold-start side effects: First, we use heterogeneous information graph for candidate selection. By extending the PathSim similarity measure [9], we rank our user candidate list based on the user-item Meta-Path similarity score. The similarity calculation incorporate both item feature similarity and user

[1]https://grouplens.org/datasets/movielens/

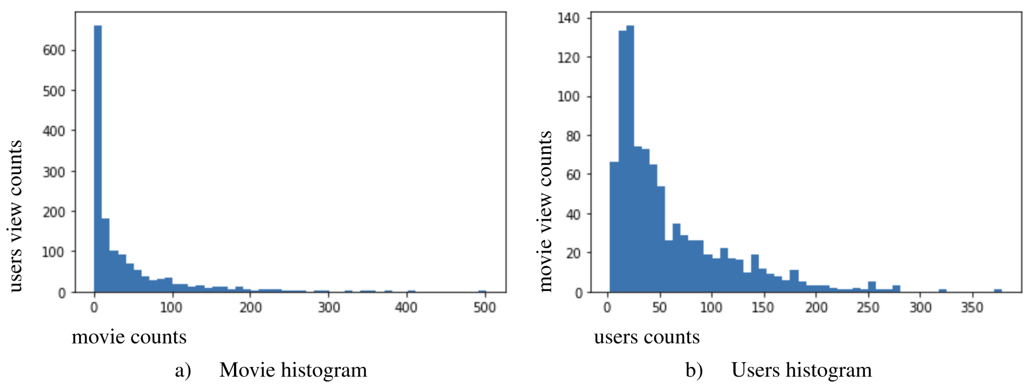

Fig. 1. The histogram to show the user activeness and item popularity.

individual activeness. We then adopt abnormal detection technique into our second step for accuracy improvement. By detecting compression cost to find false positive signals from the ranked candidate list in step one.

Abnormal detection for dynamic network [10] is a well researched topic with a number of important business applications. However as far as we know, there isn't any research using abnormal detection for spamming prevention in recommender systems.

This paper unifies top-K recommendation and spam detection into a holistic approach using information heterogeneous graph. Its contributions are as follows:

- The framework effectively targets cold start problem by considering both item contents and users activeness using heterogeneous information graph.
- The framework adopts abnormal detection into its recommendation for accuracy improvement with spamming check included.
- The result produced through this framework is explainable and traceable based on graph structure. It enables extracting human-readable rules for later business decision making.
- The framework is simple to implement which is self evolving overtime. Model allows general approach for various problems
- Graph based computation approach is also adaptable for real time processing.

The rest of this paper is constructed as follows: In Section II, we discuss the related research that is adopted in recommender system. In Section III, we explain our framework and related algorithms. Lastly, in Section IV, we conclude our findings and future work.

II. Related Work

Our research is conceptually based on previous findings in both recommender system and graph based data mining field. Such as, collaborative filtering approaches, node embedding, as well as network based feature learning and abnormal detection in graphs.

A. Collaborative Filtering

Collaborative Filtering is one of the most popular techniques used in recommender system. The underline intuition of Collaborative Filter is simple. If user X and user$Y$ like the same product A, then it is likely X and Y will share the same interest on a different item B.

Amazon started using CF for its recommendation since 2003 'The Netflix Prize' [2] did a great push on researching of Collaborative Filtering Topic. Many progress and development have been made since base on it. Collaborative Filtering can be categorized into three categories: memory-based, model-based and hybrid. Each of the categories has its own characters, that is designed to tackle different problems of CF. But they also come with their own limitations respectively. We review each one of them in the following part.

a) Memory-base collaborative filtering: Memory based collaborative filtering uses user rating data to compute the similarity between users or items, then make recommendations. Typical examples of this approach are neighborhood-based CF and item-based/user-based top-N recommendations. Memory based CF is know for it's limitation in sparsity and scaling problems. Techniques such as singular value decomposition, latent semantic indexing are adopted to alleviate the sparsity problem. Those techniques shown effective improvement on the performance by filtering out unusable user-item representations and reducing dimensional space. However the computation cost could not be linearly scaled when data size grow exponentially.

b) Model-based collaborative filtering: Machine learning model or data mining techniques are normally used in model based collaborative filtering. There are many model-based CF algorithms. Bayesian networks, latent semantic models clustering models, just to make a few. The general approach is using machine learning pattern recognition ability to map complex pattern from large sparse data matrix into dense low-dimensional representation, thus helps easing the scalability problem [11]. In practice, the challenge for model-based collaborative filtering would come down to the complexity of model building and its on going model maintenance cost.

c) Hybrid Collaborative Filtering : Content-based collaborative filtering, network-based collaborative filtering, and knowledge graph-based collaborative filtering, can all fall into the hybrid collaborative filtering category. [12] [13] For example, Chen et al. uses homogeneous attributes of tweets and the heterogeneous nature of its network to solve the ranking problem at Tweeter [14]. Agarwal et al. from LinkedIn uses the taxonomy activity of its content to make time sensitive recommendation on its user feed [15]. One big advantage of using hybrid CF approach is it can improve the cold-start problem. However commonly, most of the research are more focused on the feature similarity while lacking the consideration of user activeness.

To be more precise, in real world scenario, a user A who watched varieties of movies, is more likely to be a superior candidate in terms of opens to new movies, when comparing to user B who only watches one movie C. However, when it comes to user candidates selection, commonly NNK based calculation are mostly focus on item features or its derived latent features. As a result, if a new movie D has a lot of feature overlapping to movie C, user B would have a higher ranking than user A, even though user A are more valuable.

B. Graph-based Recommender Systems

Heterogeneous information graph, also known as heterogeneous information network, or knowledge graph, is a graph data model in which contains multiple types of nodes and edges. Heterogeneous information graph is being well studied in the data mining area, for item feature extraction as well as semantic information presentation [16] [17] [18]. Recently, inspired by Word2Vec [19] Skip-Gram and negative sampling algorithm, similar technique is used into the information graph space, such Node2Vec [7].

In recommender system, embedding is studied in Item2Vec [20] and Entity2Vec [8], where treats each item as corpus, then shuffles items order during the training process. In the end, items are embedded into a multi-dimensional vector space. So that, similarity between items can be easily measured by calculating the distance between nodes.

In the cold start problem, the network structure has been long proven to be effective, as it enables encompassing both collaborative and content information. Community based recommender system [21] are often mention as a common approach for solving the problem. Macedo et al. uses Contextual Learning [22] approach, which exploit the contextual signals from event-based social networks, to solve the short-live and constant cold starting problem for ranking events content.

C. Abnormal Detection Using Information Graph

Abnormal detection is a important topic that has wide range of applications such as, finance analytics, network security, spam detection, etc. For a ever updating graph, we can treat the graph as a time-evolving dynamic network [10]. Using graph structure for abnormal detection have been well studied for varies of approaches. Abnormal can be detected via nodes, edges, as well as sub-graphs.

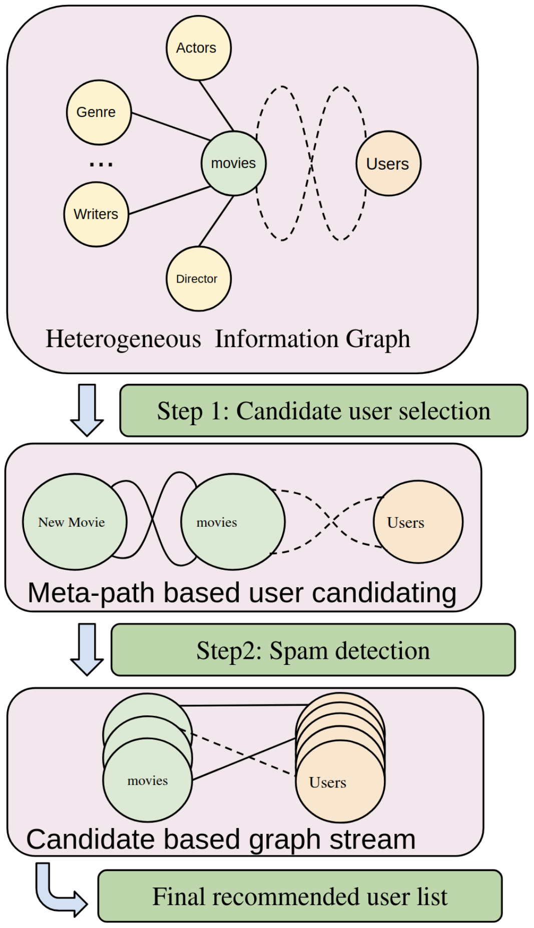
Actors
Genre
...
movies
Users
Writers
Director
Heterogeneous Information Graph
Step 1: Candidate user selection
New Movie
movies
Users
Meta-path based user candidating
Step2: Spam detection
movies
Users
Candidate based graph stream
Final recommended user list

Fig. 2. The FreshGraph framework.

Compression based methods are mainly based on the minimum description length principle [23]. By calculating and detecting the change of graph compression cost also know as entropy encoding cost [24]. Some interesting measurements approach is developed, such as GraphScope [5].

Community based methods are commonly focuses on community structure or its definition by tracking the evolution of communities and their associated nodes over time. E.g. a sudden size/density change in sub-graphs, or change of nodes-community belonging probabilities.

There are graph based spam detection [25] [26] research work being done previously. However, we have not yet found any work directly integrating spam detection into recommender systems.

III. A SPAM-AWARE RECOMMENDER SYSTEM

In this section, we present a two-steps spam aware recommendation framework, FreshGraph, as shown in Fig. 2. We firstly introduce notions used in this paper and our problem settings. Then we propose FreshGraph, which resides within a holistic graph structure that adapts to continuous fresh new data updates overtime. Recall that our purpose of recommendation is when a new item comes into the market, we push notifications to users for the new item. FreshGraph uses heterogeneous information graph to learn user-item relationships and item semantic information. Step 1, we use Meta-Path similarities to rank users and generate user candidates. Next in Step 2, we use abnormal detection to filter out false positive from previous candidate list to produce final output and push notifications to target users.

A. Problem Settings and Definitions

Following definitions are used for describe our approach using a holistic heterogeneous graph settings.

Definition 1 (Heterogeneous Information Graph)**.** A information graph is $G = (V, E)$, where V is the set of nodes (or entities) of the graph. E is the set of edges connecting the nodes in V, $E \subset V \times V$.
Two mapping functions: Entity type mapping function ϕ: $V \to A$, and link type mapping function φ: $E \to R$, where A and R denote the sets of predefined entity and link types, and $|A| > 1$ or $|R| > 1$ indicating that there are more than one type.

For example, we use movie attribute information to enrich user-item ratings in Movielens data set using graph model as shown in Fig. 3. In this paper we use *"actors", "director", "writer", "genre"* etc. as different types of nodes. These nodes are in the same graph of user nodes and item nodes. How to process with different nodes in one graph? Graph schema indicates how different types of entities link with each other. It serves as a template to describe the structure as well as the semantic relationship between object types.

Between two entities x and y, there are different paths connecting the two nodes. As for the case of Movielens, *MovieA* and *MovieB* can be connected via *MovieA-Actor-MovieB* or *MovieA-Director-MovieB* or *MovieA-User-MovieB* path connections. We call those path, which containing multiple entities, Meta-Path.

Definition 2 (Meta-Path)**.** A Meta-Path $\mathcal{P}$ is a path defined on the graph schema $T_G = (A, R)$.
Meta-Path $\mathcal{P}$ is denoted as $A_1 \xrightarrow{\text{r1}} A_2 \xrightarrow{\text{r2}} ... \xrightarrow{\text{rn}} A_n$.

Relationship R is denoted as $r1 \bullet r2 \bullet ...rn$ for different types of relationship between different types of entity nodes, where $\bullet$ denotes composition operator or relations.

For Meta-Path $\mathcal{P}_i$ shares same graph schema, there could also be multiple path p connecting source entity a_i to target a_{i+1}. Each path p inside Meta-Path $\mathcal{P}_i$ is a path instance, $p \in \mathcal{P}_i$. The number of path instances p between a_i and $\boldsymbol{a}_{\text{i+1}}$, is called path count. Reverse Meta-Path $\mathcal{P}^{'}$ is the reversed relation sequence of $\mathcal{P}_i$, if $\mathcal{P}^{'}$ is the reverse path of $\mathcal{P}$ in $T_G = (A, R)$, reverse path is denoted as $\mathcal{P}^{\text{-1}}$.

Meta-Path $\mathcal{P}_1 : A_1 \xrightarrow{r_1} A_2$ is the Meta-Path connects source entity type A_1 and target A_2. Similarly, $\mathcal{P}_2 : A_2 \xrightarrow{r_2} A_3$, $\mathcal{P}_2$ is the Meta-Path between source entity type A_2 and target entity type A_3. Here we call $\mathcal{P}_1$ and $\mathcal{P}_2$ are contactable. Then $\mathcal{P}_1$ and $\mathcal{P}_2$ can be combined as $\mathcal{P}_{1,2} : A_1 \xrightarrow{r_1} A_2 \xrightarrow{r_2} A_3$. For example, $Movie \to Director$ and $Director \to Movie$ can be combined to $Movie \to Director \leftarrow Movie$.

As data changes over time with new records keep coming into the graph, we can snapshot our graph based on time t. by having multiple graph snap shot together, this leads to Graph Stream. As shown in Fig. 5

Definition 3 (Graph Stream)**.** A graph stream $\mathcal{G}$ is a sequence of graphs $G_{(t)}$, i.e., $\mathcal{G} := \{G_{(1)}, G_{(2)}, ..., G_{(t)}, ...\}$. Graph stream grows over time indefinitely. For graph stream segment $\mathcal{G}_t := \{G_{(1)}, G_{(2)}, ..., G_{(t)}\}$ which is a definite graph with length t.

Definition 4 (Stream Graph Partition [5])**.** We group source (i.e. item) nodes into H partitions. The set of nodes in each partition is denoted as $\{P_1, P_2, ..., P_H\}$ as item nodes partition P, $1 \leq H \leq M$. The same happens to the target (i.e. user) nodes which is grouped into L partitions, denoted as $Q = \{Q_1, Q_2, ..., Q_L\}$, $1 \leq L \leq N$.

For statistic measurement PathSim [9] and GraphScope [5] algorithm is used in our framework for similarity and drift measurement.

Definition 5 (PathSim [9])**.** Given a Meta-Path $\mathcal{P}$, PathSim between two entities x and y is:

$$sim(x, y) = \frac{2 \times \{|p_{\text{x...y}} : p_{\text{y...x}}| \in P\}}{\{|p_{\text{x...x}} : p_{\text{x...x}}| \in P\} + \{|p_{\text{y...y}} : p_{\text{y...y}}| \in P\}} \quad (1)$$

where x stands for source node, while y for target node. x, y shares the same entity type A_i. $p_{\text{x...y}}$ stands for Meta-Path between entity x and y.

Unlike the path count, random walk or pairwise random walk measurement, which are biased towards node with highly visible nor highly concentrated object. PathSim score uses the semantics of peer similarity for measurement.

Definition 6 (Graph Stream Segment Encoding Cost [5])**.**

$$C^{(t)} := \log(t) + C_p^{(t)} + C_g^{(t)} \quad (2)$$

$C^{(t)}$ is the encoding cost of $\mathcal{G}_t$.

$\log(t)$ stands for the length of the graph segment. i.e number of graph snapshots, $C_p^{(t)}$ stands for the encoding cost of partitions inside $\mathcal{G}_t$. $C_p^{(t)} = \log M + \log N + \log H + \log L - M \cdot H(p) - N \cdot H(q)$, where M and N are the numbers of item and user nodes. H, L are the numbers of item and user nodes partition.

p and q are multinomial random variable of the probability in user and item partitions. i.e. $p_i \in p$, $p_i = \frac{m_i}{M}$, m_i is the i-th source node partition $1 < i < K$. The same applies to q to the user nodes partitions. $H(X)$ is the entropy of X, defined as

$$H(X) = \sum_{x \in K} p(x) \log p(x) \quad (3)$$

where K is the partitions size.

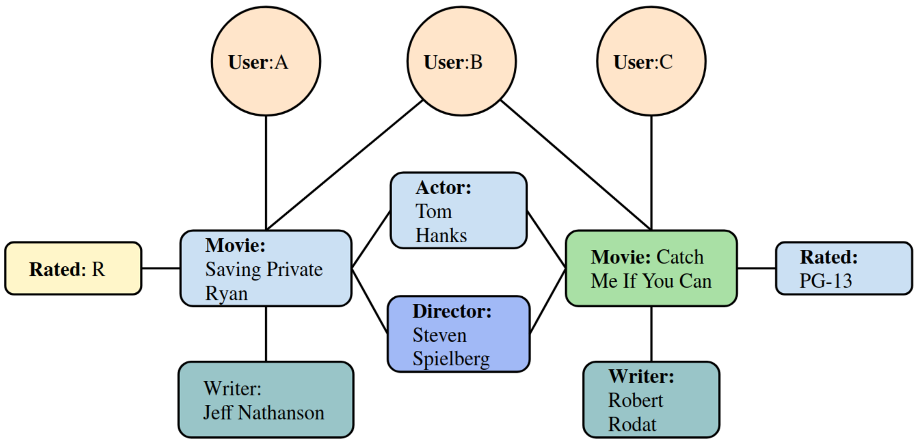

Fig. 3. Enriched Movielens information graph.

$C_g^{(t)}$ stands for the encoding cost of graphs inside $\mathcal{G}_t$:

$$C_g^{(t)} = \sum_{p=1}^{H_t} \sum_{q=1}^{L_t} (\log E_{p,q}^{(t)} + |\mathcal{G}_{p,q}^{(t)}| * H(\mathcal{G}_{p,q}^{(t)}))$$

and similarly $H(\mathcal{G}_{p,q}^{(t)})$, is the entropy term for the sub-graph segment $\mathcal{G}_{p,q}^{(t)}$ as $H(\mathcal{G}_{p,q}^{(t)}) = -(p_{p,q}^{(t)} \log p_{p,q}^{(t)} + (1-p_{p,q}^{(t)}) \log(1-p_{p,q}^{(t)})$, H_t and L_t are the number of partitions at the current time t.

B. Framework Description

Step 1: User candidating: For a new item, we normally don't have much user interaction data for recommender system to learn. Using graph, based on its network structure new item can be added into exiting graph based on its attributes. Meta-Path can be established between new item and existing items. Inductively, new item and exiting users connection can be propagate through the graph network.

1) Ranked Items list using Meta-Path similarity measure: Here, we go through the candidating step using Meta-Path similarity measure which is inspired by PathSim as defined in Definition 5

PathSim measures similarity between two nodes, which shares the same entities type. The intuition for a single Meta-Path measure can be seen as: if source entity and target entity are exactly the same, then the round trip Meta-Paths $\mathcal{P}_1 : A_1 \xrightarrow{r_1} A_2$ and $\mathcal{P}_1^{-1} : A_2 \xrightarrow{r_1^{-1}} A_1$ are always symmetric.

Extending from the single Meta-Path approach, In the same graph network, between the source node and target node, Usually we have multiple Meta-Paths exist. This allow us to re-apply the algorithm to multiple different Meta-Path. For example, *Movie-Actor-Movie, Movie-Director-Movie, Movie-Genre-Movie*.

We derive Ranked Items list use equation (4) to calculate item-item similarity score, by aggregate similarity scores

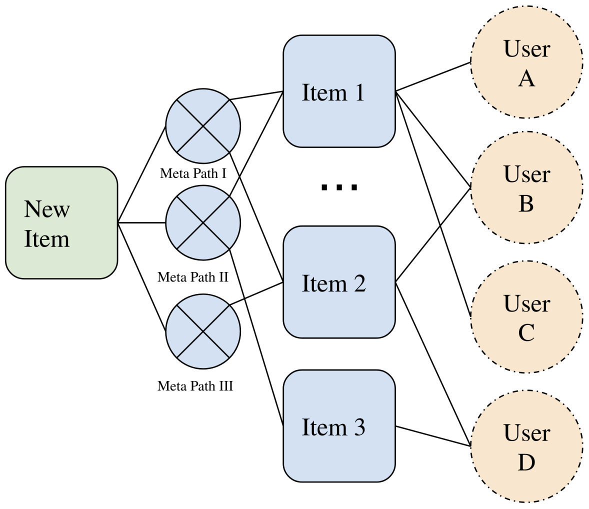

Fig. 4. Item-user similarity propagation

across multiple Meta-Paths between the source and target node.

$$C(\mathcal{P}) = \sum_{i=1}^{i} sim(s_i, t_i) \tag{4}$$

where i denotes the number of different Meta-Path P_i between source node s and target node t, $sim(s_i, t_i)$ stands for the similarity score between s, t on Meta-Path $\mathcal{P}_i$. Hence we can get a list of item list based on the new item, and rank them in a ordered list.

2) Ranked user list extend from item-item similarity measure: Next we use the item-item similarity ranking for user candidating calculation. we extend *Movie-...-Movie* Meta-Path to *Movie-...-User* Meta-Path, for example: *Movie-Actor-Movie*, *Movie-Director-Movie*, *Movie-Genre-Movie* is now

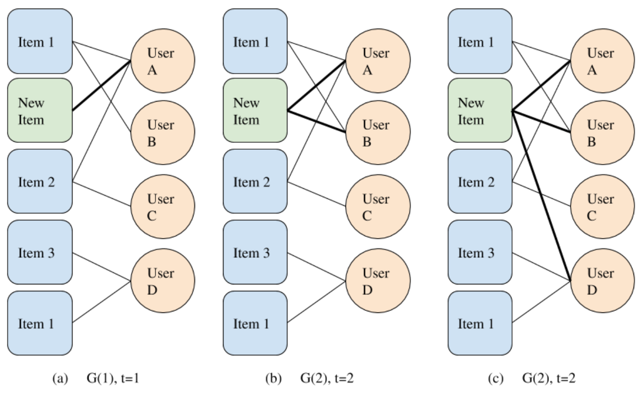

Fig. 5. Pairwise item-user graph and its time-based graph stream

extended to *Movie-Actor-Movie-User*, *Movie-Director-Movie-User*, *Movie-Genre-Movie-User*.

As demonstrated in Fig. 4, we can calculate item-user similarity score as

$$Score(u) = \sum_{j=1}^{j} C_j(\mathcal{P}) \tag{5}$$

where j denotes the number of different items, that have a linked relation to a given user u. The connection can either be direct connected $(d = 1)$, or connected within certain distance $(d = n)$. d here can be considered as a hyper parameter for number of different relationships in between user and item. Thus, Besides learning similarity from movies feature based entities (e.g.: actor, director, rating, genre, etc.), user and item correlation as well as users activeness are reflected. Finally, we can have a ranked user list based on $Score(u)$ value, $RL(i_{new}) = \{u_1, u_2, ..., u_K\}$, where i_{new} stands for the given new item and K is the number of candidate users.

Step 2: Spam detection: After getting ranked user list $RL(i_{new})$ from Step 1 above, we now can simplify existing heterogeneous graph as in Fig. 4 into a pairwise network G_1 shown in Fig. 5 (a). It is based on candidate users list along with their related items.

In this step, we start adding one users and item connection at a time. The sequence is based on the candidate similarity score, from the highest to lowest. Forming graphs in each steps, as demonstrated in Fig. 5. We have a graph stream $\mathcal{G}$ based on the sequence of input. As defined in Definition 4, each of these graphs contains M item nodes and N user nodes plus their relationships. In movie recommendation case, we consider *Movie* as item nodes while *User* as user nodes. Corresponding to the $RL(i_{new})$, the graph stream is of length K.

Based on the information theoretic principal, we detect the drift based on the encoding cost change. Similar approach had already shown effectiveness in abnormal detection research space, such as Graphscope [5] as defined defined in Definition 6. According to GraphScope equation (2). Mathematically, it comes down to calculate and compare the encoding cost between the new graph $G_{(t+1)}$ and graph stream segment $\mathcal{G}_t$. If similar graph can be merged into the target stream segment, we should see a small decrease of encoding cost overall. When $G_{(t+1)}$ can no longer merged into segments $\mathcal{G}_t$ while saving encoding cost, it indicates the detection of a drift.

The intuition is, every time we introduce a new user to target item, a new user-item relation is formed. Hence, we have a new snapshot of graph $G_{(t+1)}$. If the new user is similar to the previous user candidates, then $G_{(t+1)}$ should be able to merge into the same graph stream segment $\mathcal{G}_t$. When drift happens, it means the new user's interests is no longer matching with users in previous merged group. Hence, current and the following users would have a high chance of seeing the new item as spamming. In that case, we discard the rest of candidate from $RL(i_{new})$, and return the new Top-K' users list, where $K' = t$.

Finally, by combining candidate and spam detection together, as shown in Algorithm 1, our final recommendation list for the given item i_{new} on top K' users is derived.

Algorithm 1 Spam detection

```
Input:
    User candidate list: $RL(i_{new})$ whose length is $K$
    A stream graph segment derived from ranked candidate list $RL(i_{new})$: $\mathcal{G}^{(t)} = \{G_1, ..., G_t\}$
Output:
    The new top-$K'$ user candidate ranking list: $RL_{K'}(i_{new})$
1:  for $t = 1 : K$:
2:      Compute current encoding cost $C_t$ for $\mathcal{G}^{(t)}$
3:      Compute $C_{t+1}$ as encoding cost of $\mathcal{G}^{(t)} \bigcup \{G_{t+1}\}$
4:      Compute $c$ as encoding cost of $\{G_{t+1}\}$
5:      if $C_{t+1}$ - $C_t < c$:
6:          $\mathcal{G}^{(t)} \leftarrow \{G_{t+1}\}$, merge graph into segment
7:      else:
8:          $K' = t$
9:          break
10: $RL_{K'}(i_{new}) \leftarrow$ the first $K'$ users in $RL(i_{new})$
11: return  $RL_{K'}(i_{new})$
```

C. Complexity Optimization

To calculate a ranked user candidate list for a given item, there are two major deciding factors for its computation cost:

1) the number of nodes that need to be included in the heterogeneous information graph.
2) the number of different Meta-Path required for calculation to select the user candidates.

For large graph, it is prohibitively expensive to iterate through the whole graph for similarity calculation in either Step 1 or Step 2. However thanks to the unique graph data structure, its nodes relatedness can be leveraged through query optimization.

For starters, a sub-graph can be derived based on query terms:

$$G_{sub} = f(G, q, \Theta) \tag{6}$$

where G stands for heterogeneous information graph. q stands for query terms, and Θ is a hyper parameter. Θ defines the max extensions distance from query matching results.

Before the selection of user candidates, we first query users based on given new item attributes. E.g. query users who had watched similar movies, such as "director: Steven Spielberg" and "Genre: war". This part is very similar to database term queries, if $d = 1$. The special part using graph structure is, we can set a loose boundary from the exact matched results. i.e. users who are indirectly linked to above return users, $d = 2$, or even further $d = n$. With graph database, such query are usually very performant, while cuts down target graph size for candidate selection significantly.

In the step 1, because of the contactable characteristics in Meta-Path as we explained in Definition 2. Partially materialize commuting matrices for short length Meta-Paths can be concatenate into longer ones. Purging the cluster according to similarity boundary can further refine the sub-graph that required for calculation. Again, it can significantly cut down in computation complexity. Thus, It enabled such calculation to be carried out online in a stream processing fashion.

Lastly, in Step 2, individual node level encoding computation can be alternatively substituted by grouping them into user and item partitions, as mentioned in Definition 6. Then calculating encoding cost can be computed at partition level instead of iterating through each node. Thus, we replace Eq. 3 with new cross-entropy equation as following:

$$H(p_i, q_i) = \sum_{x \in \{0,1\}} p_i(x) \log q_i(x) \tag{7}$$

$q_i(x)$ is the node x distribution inside partition P_i. For large $M \times N$ pairwise graph, this can significant reduces complex from $M * N$ to $H * L$, where H and L are the numbers of user and item partitions.

IV. Conclusion and Discussion

In this paper, we proposed a novel framework that fully leverages the advantages of heterogeneous information graph. The framework sort through multiple important aspects of a commender system. It stats from data modeling to information storage and on going data evolution, all the way to real-time online processing. Combining both item features and user interaction within the same heterogeneous graph, it allows the system more capable of handling cold start problem. Included false positive detection mechanism helps for improving final results accuracy. consequently, business can reduce or prevent negative commendation effects, such as spamming problems.

Such approach opens up possible enhancement for several directions. First, heterogeneous information graph provides a rich feature sets combinations for recommendation tasks. However feature selection task can still tricky, that requires business domain knowledge. Research areas such as Meta-Path discovery [27] which automatically selects most relevant Meta-Path, could be a exciting direction going forward. Second, for different domain problems, the hyper parameters can be further tuned.

Acknowledgment

This work was partially supported by DOMAIN HOLDINGS AUSTRALIA LIMITED and UTS joint Research Project RES18/1417.

References

[1] J. Lu, D. Wu, M. Mao, W. Wang, and G. Zhang, "Recommender system application developments: a survey," *Decision Support Systems*, vol. 74, pp. 12–32, 2015.

[2] J. Bennett and S. Lanning, "The netflix prize," in *Proceedings of KDD Cup and Workshop*, 2007, p. 35.

[3] B. Hu, C. Shi, W. X. Zhao, and P. S. Yu, "Leveraging meta-path based context for top-n recommendation with a neural co-attention model," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 1531–1540.

[4] M. Mao, J. Lu, G. Zhang, and J. Zhang, "Multirelational social recommendations via multigraph ranking," *IEEE Transactions on Cybernetics*, vol. 47, no. 12, pp. 4049–4061, 2016.

[5] J. Sun, C. Faloutsos, C. Faloutsos, S. Papadimitriou, and P. S. Yu, "Graphscope: parameter-free mining of large time-evolving graphs," in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2007, pp. 687–696.

[6] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation learning on graphs: Methods and applications," *arXiv preprint arXiv:1709.05584*, 2017.

[7] A. Grover and J. Leskovec, "Node2vec: scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016, pp. 855–864.

[8] E. Palumbo, G. Rizzo, and R. Troncy, "Entity2rec: Learning user-item relatedness from knowledge graphs for top-n item recommendation," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 2017, pp. 32–36.

[9] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, "Pathsim: meta path-based top-k similarity search in heterogeneous information networks," *Proceedings of the VLDB Endowment*, vol. 4, no. 11, pp. 992–1003, 2011.

[10] S. Ranshous, S. Shen, D. Koutra, S. Harenberg, C. Faloutsos, and N. F. Samatova, "Anomaly detection in dynamic networks: a survey," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 7, no. 3, pp. 223–247, 2015.

[11] Q. Zhang, J. Lu, D. Wu, and G. Zhang, "A cross-domain recommender system with kernel-induced knowledge transfer for overlapping entities," *IEEE Transactions on Neural Networks and Learning Systems*, 2018.

[12] D. Wu, J. Lu, and G. Zhang, "A fuzzy tree matching-based personalized e-learning recommender system," *IEEE transactions on fuzzy systems*, vol. 23, no. 6, pp. 2412–2426, 2015.

[13] W. Wang, G. Zhang, and J. Lu, "Member contribution-based group recommender system," *Decision Support Systems*, vol. 87, pp. 80–93, 2016.

[14] J. Chen, R. Nairn, L. Nelson, M. Bernstein, and E. Chi, "Short and tweet: experiments on recommending content from information streams," in *The 28th International Conference on Human Factors in Computing Systems*. New York, USA: ACM Press, 2010, p. 1185.

[15] D. Agarwal, A. Singh, L. Zhang, B.-C. Chen, R. Gupta, J. Hartman, Q. He, A. Iyer, S. Kolar, Y. Ma, and P. Shivaswamy, "Activity ranking in linkedin feed," in *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, USA: ACM Press, 2014, pp. 1603–1612.

[16] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *Proceedings of the 24th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 2015, pp. 1067–1077.

[17] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems*, 2017, pp. 1024–1034.

[18] H. Zhao, Q. Yao, J. Li, Y. Song, and D. L. Lee, "Meta-graph based recommendation fusion over heterogeneous information networks," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2017, pp. 635–644.

[19] 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*, 2013, pp. 3111–3119.

[20] O. Barkan and N. Koenigstein, "Item2vec: Neural item embedding for collaborative filtering," in *IEEE 26th International Workshop on Machine Learning for Signal Processing*. IEEE, 2016, pp. 1–6.

[21] S. Sahebi and W. W. Cohen, "Community-based recommendations: a solution to the cold start problem," *Proceedings of the 11th ACM Conference on Recommender Systems Workshop on Recommender Systems and the Social Web*, pp. 40–44, 2011.

[22] A. Q. Macedo, L. B. Marinho, and R. L. Santos, "Context-aware event recommendation in event-based social networks," in *The 9th ACM Conference on Recommender Systems*. New York, USA: ACM Press, 2015, pp. 123–130.

[23] P. Grunwald, "A tutorial introduction to the minimum description length principle," *arXiv preprint math/0406077*, 2004.

[24] J. Rissanen, "A universal prior for integers and estimation by minimum description length," *The Annals of Statistics*, pp. 416–431, 1983.

[25] G. Wang, S. Xie, B. Liu, and S. Y. Philip, "Review graph based online store review spammer detection," in *2011 IEEE 11th International Conference on Data Mining*. IEEE, 2011, pp. 1242–1247.

[26] J. Abernethy, O. Chapelle, and C. Castillo, "Graph regularization methods for web spam detection," *Machine Learning*, vol. 81, no. 2, pp. 207–225, 2010.

[27] C. Meng, R. Cheng, S. Maniu, P. Senellart, and W. Zhang, "Discovering meta-paths in large heterogeneous information networks," in *Proceedings of the 24th International Conference on World Wide Web*, 2015, pp. 754–764.