UTS UNIVERSITY OF TECHNOLOGY SYDNEY

Advanced Multi-Graph Architectures for Natural Language Understanding

by Bowen Xing

Thesis submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy

under the supervision of Prof. Ivor W. Tsang

University of Technology Sydney
Faculty of Engineering and Information Technology

November 2023

CERTIFICATE OF ORIGINAL AUTHORSHIP

I, Bowen Xing, declare that this thesis is submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the Faculty of Engineering and Information Technology at the University of Technology Sydney.

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

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

This research is supported by the Australian Government Research Training Program.

Signature: Production Note:
Signature removed prior to publication.

Date: 15 Aug 2023

Acknowledgments

I would like to express my deepest gratitude to my parents, whose unwavering support, encouragement, and love have been my pillars throughout this journey. Their belief in me has been a constant source of inspiration. I would like to thank Prof Ivor W. Tsang, my supervisor, for his invaluable guidance, expertise, and mentorship. His insightful feedback, patience, support, and encouragement have played a pivotal role in shaping this research and my academic growth. I am truly fortunate to have had the opportunity to work under his guidance. And I appreciate my colleagues and friends. We spent three happy years together.

Abstract

Natural language processing (NLP) is a challenge while important subfield of artificial intelligence (AI). And natural language understanding (NLU) is a crucial component of NLP. In recent years, graph structures and graph neural networks are widely utilized in NLU. In single task scenarios, graph structures can represent the sentence or document in graphs (e.g. syntax graph of a sentence). And sometimes only one graph cannot represent sufficient information, therefore multiple graph structures are proposed to sufficiently represent the relations and dependencies. In multi-task scenarios, the interactions among the multiple tasks can be represented in a multi-task graph. And different kinds of graph neural networks have been proposed to achieve information aggregation on the graphs. In this thesis, the multi-graph structures for single tasks, the multi-task graph structures for multi-task learning, and the graph neural networks working on them are collectively referred to as multi-graph architectures.

In this thesis, we design multi-graph architectures based on interconnected graphs to represent diverse linguistic dimensions and multi-task interactions, including syntactic, semantic, and contextual information. Each graph encapsulates specific linguistic features, fostering a more comprehensive understanding of language nuances. The fusion of these graphs enables the model to capture intricate relationships and dependencies among words, concepts, sentence and different tasks, contributing to a more robust and context-aware NLU system.

This thesis explores the design principles, implementation details, and experimental results of multi-graph architectures applied to various NLU tasks, such as aspect sentiment classification, joint dialog sentiment classification and act recognition, and joint multiple intent detection and slot filling. Comparative analyses against state-of-the-art models demonstrate the efficacy of the proposed multi-graph architectures in handling ambiguous language constructs and improving overall NLU performance in both single-task and multi-task scenarios.

Publications

Conference Paper

1. Haotian Wu*,**Bowen Xing*** and Ivor W. Tsang. MTKDN: Multi-Task Knowledge Disentanglement Network for Recommendation. In Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM 2023).

2. **Bowen Xing** and Ivor W. Tsang. Co-evolving Graph Reasoning Network for Emotion-Cause Pair Extraction. In Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD 2023).

3. **Bowen Xing** and Ivor W. Tsang. Co-guiding Net: Achieving Mutual Guidances between Multiple Intent Detection and Slot Filling via Heterogeneous Semantics-Label Graphs. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP 2022), pages 159-169.

4. **Bowen Xing** and Ivor W. Tsang. Group is better than individual: Exploiting Label Topologies and Label Relations for Joint Multiple Intent Detection and Slot Filling. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP 2022), pages 3964-3975.

5. **Bowen Xing** and Ivor W. Tsang. Neural Subgraph Explorer: Reducing Noisy Information via Target-oriented Syntax Graph Pruning. In Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI 2022), pages 4425–4431.

6. **Bowen Xing** and Ivor W. Tsang. DARER: Dual-task Temporal Relational Recurrent Reasoning Network for Joint Dialog Sentiment Classification and Act Recognition. In Findings of the Association for Computational Linguistics: ACL 2022, pages 3611–3621.

Journal Paper

1. **Bowen Xing** and Ivor W. Tsang. Co-guiding for Multi-intent Language Understanding. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2023.

2. **Bowen Xing** and Ivor W. Tsang. Relational Temporal Graph Reasoning for Dual-task Dialogue Language Understanding. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2023.

3. **Bowen Xing** and Ivor W. Tsang. Understand me, if you refer to Aspect Knowledge: Knowledge-aware Gated Recurrent Memory Network. IEEE Transactions on Emerging Topics in Computational Intelligence, 2022, 6(5):1092–1102.

4. **Bowen Xing** and Ivor W. Tsang. Out of Context: A New Clue for Context Modeling of Aspect-based Sentiment Analysis. Journal of Artificial Intelligence Research (JAIR), 2022, 74: 627-659.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Natural language processing (NLP) is a subfield of artificial intelligence (AI) that focuses on the interaction between computers and human language. The goal of NLP is to enable machines to understand, interpret, and generate human language in a way that is both meaningful and contextually relevant. Natural language understanding (NLU), a subset of NLP, refines the focus to the comprehension and interpretation of human language, aiming to imbue machines with a deeper understanding of the meaning and context within textual or spoken communication. Beyond mere syntactic analysis, NLU endeavors to extract semantic meaning, discern intent, and navigate the subtleties of language. NLU includes various tasks like sentiment analysis, named entity recognition, relation extraction, dialog understanding, spoken language understanding, question answering and so on. In NLU, single-task scenarios denote that there is only one stream of predictions corresponding to a single task for the input. And multi-task scenarios denote that there are multiple streams of labels corresponding multiple tasks for the input.

In recent years, graph structures (e.g., syntax graphs, relation graphs, semantics graphs) have been widely used in both of single-task and multi-task scenarios of NLU. And different kinds of graph neural networks (e.g., graph convolutional networks, graph attention networks, relational graph convolutional networks) are adopted to achieve information aggregation on the graph structures. Generally, graph structures can benefit NLP in the following aspects. First, representing linguistic structures. Graphs provide a flexible way to represent the syntactic and semantic structure of language, like parse trees or semantic nets. This allows capturing complex linguistic relationships. Second, knowledge representation. Knowledge graphs can store real-world entities and relationships which provides useful background knowledge for NLP models. This helps with tasks like entity linking, disambiguation, and question answering. Third, incorporating contextual information. Graphs can capture contextual information

like document structure or the flow of a conversation. This provides useful context beyond just the current sentence for NLP models. Forth, multi-hop reasoning. Graph algorithms like message passing over nodes can allow models to make multi-step inferences through reasoning over linguistic structures and world knowledge. This helps for complex reasoning tasks. Fifth, leveraging neural networks. Graph neural networks can operate directly on graph structured data to learn effective language representations and make predictions. Knowledge sharing: Linguistic or world knowledge encoded in graphs can be shared across tasks and applications instead of having to relearn. Overall, graphs provide a flexible way to represent and leverage diverse linguistic information for more robust and accurate NLP models. The combination of graphs and neural networks is a promising approach for advancing NLP.

Here we introduce the definition of *Multi-graph Architectures* in this thesis. It includes two parts: (1) multi-graph structures; (2) graph neural networks working on multi-graph structures to achieve information aggregation. In single-task scenarios, sometimes there are different kins of graph structures are designed by different works to capture the specific information. And in some tasks, the information corresponding to different graph structures are complementary to each other. Therefore, sketching the multiple graphs into a unified graph and allowing them to interact can potentially capture more sufficient information and provide more beneficial clues for the tasks. However, few previous works focus on this direction. In multi-task scenarios, we observe that the sequence corresponding to different tasks can be represented into several task-specific groups of nodes, among which there are rich relations. However, previous works either do not adopt the graph structures or simply adopt the fully-connected graph, which is composed of all nodes from multiple tasks, ignoring modeling the rich relationships among the task-specific nodes.

In this thesis, the research question we focus on is how to design advanced multi-graph structures for both single task problems and multi-task problems in the field of natural language understanding. Besides, we also investigate the research question of how to design advanced graph neural networks to achieve information aggregation on the designed multi-graph structures. In this thesis, we present five works, as shown in Fig 1.1. KaGRMN and NSE tackle the task of aspect sentiment classification, which is a single-task scenario. DARER tackles the task of joint dialog sentiment classification and act recognition, which is a multi-task scenario. Co-guiding Net and ReLa-Net tackle the task of joint multiple intent detection and slot filling. Specifically, the content in this thesis are summarized as follows:

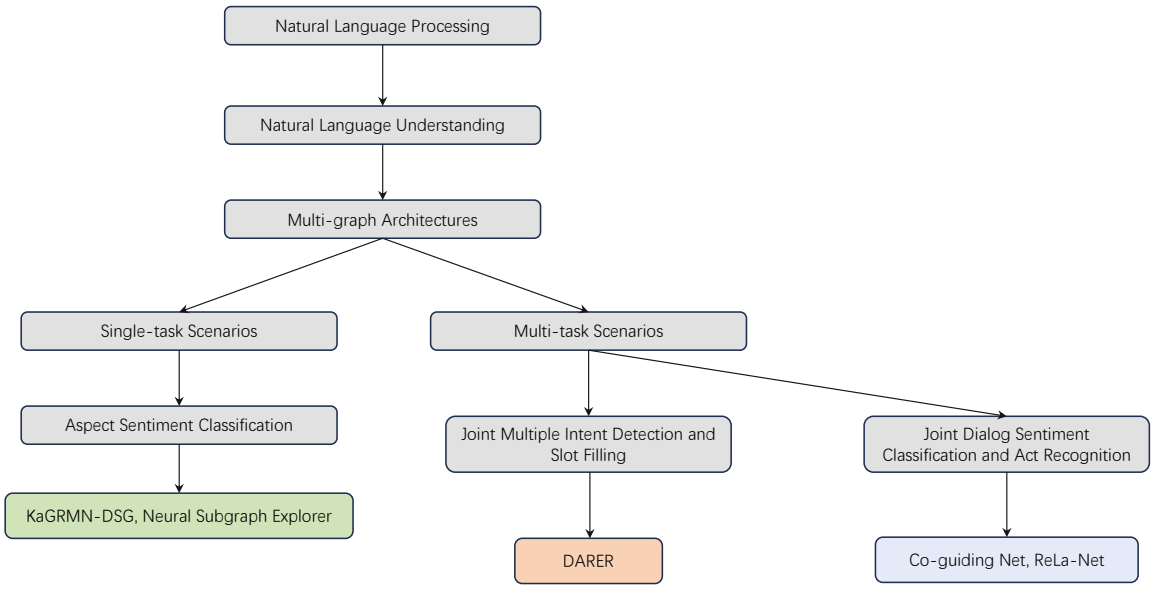


Figure 1.1: Roadmap of this thesis.

The content in chapter 3 corresponds to the KaGRMN-DSG work published on IEEE TETCI 2022. This work proposes to leverage the external aspect knowledge to enhance the aspect semantics, which can facilitate the aspect sentiment classification. Besides, the DSG module aims to merge the two kinds of syntactic information to comprehensively understand the context semantics.

The content in chapter 4 corresponds to the Neural Subgraph Explorer work published on IJCAI 2022. This work proposes to prune the syntax graph regarding the specific aspect, aiming to retain the aspect-related semantics while eliminate the aspect-unrelated semantics.

The content in chapter 5 corresponds to the DARER and DARER^2 works published on ACL 2022 and IEEE TPAMI 2023. This work proposes a novel framework leverage temporal information, label information, and semantics to work together to let DSC and DAR gradually promote each other. DARER utilizes RGCNs for speaker-aware temporal modeling and dual-task reasoning, while DARER^2 adopts our proposed ReTeFormers.

The content in chapter 6 corresponds to the Co-guiding Net published on EMNLP 2022. This work makes the first attempt to achieve the mutual guidances between the multiple intent detection and the slot filling. To implement this idea, we propose a two-stage framework and the Co-guiding Net, which includes our proposed heterogeneous graph attention network applying on the designed semantics-label graphs.

The content in chapter 7 corresponds to the ReLa-Net published on EMNLP

2022. This work discovers and exploits the dual-task label topologies and relations between the multiple intent detection and slot filling tasks. Moreover, we also propose the label-aware inter-dependent decoding mechanism to further exploit the label correlations for decoding.

Chapter 2

Literature Review

2.1 Graph Architectures for NLU

Graph structures and graph neural networks have been widely adopted in various NLU tasks. For example, in relation extraction task, [100] utilized a multi-view graph structure to capture the possible relationships among tokens and selected important words for relation extraction. In aspect sentiment classification task, [79] proposed a star-shaped aspect-centered relational graph structure and proposed a relational graph attention network to capture the aspect-related syntactic information. In multi-intent spoken language understanding task, GL-GIN [57] and AGIF [58] adopted the dual-task graph structure and employed the graph attention networks to capture the interaction between the multiple intent detection task and slot filling task. In commonsense reasoning task, KagNet [40] adopted a schema graph derived from the commonsense question to enhance the prediction of the answer. Despite of the success that has been achieved in NLU field, little attention has been paid to multi-graph architectures, which is the core of this thesis.

2.2 Aspect Sentiment Classification

In early studies [28, 46], sentiment classifiers were built by traditional machine learning algorithms which demanded labor-intensive feature engineering. Most recently proposed ASC models are based on neural networks which can automatically learn representations. Conventionally, neural ASC model contains an aspect encoder, a context encoder and an aspect-to-context attention mechanism [5, 12, 45, 71, 83].

Different kinds of networks are adopted as the encoder. As for LSTM, [69] employed two separated LSTMs to encode the aspect-left and -right word sequences and then combined the two last hidden sates for classification;

[19] proposed to leverage external document sentiment analysis corpus in a multi-task framework to enhance the context modeling of LSTM. Besides, convolutional neural networks (CNN) and Memory Networks (MNs) are also exploited as encoders. [22] introduced parameterized filters and parameterized gates into CNN to integrate aspect information for context encoding. [101] designed a gated CNN layer to extract the aspect-specific features from the context hidden states. Based on standard MNs, [81] proposed the target-sensitive memory networks to focus on the impact of aspect semantics on the classification.

The attention mechanism is utilized to extract aspect-related sentiment features via assigning a weight to each context word regarding its relevance to the aspect. [45] proposed an aspect-to-context attention and a context-to-aspect attention to study the interactions between the aspect and context. [71] proposed an algorithm to automatically mine useful supervised information for the attention mechanism through the training process.

However, attention mechanisms may hardly capture the important words which is far from the aspect in the input context. As the development of graph neural networks [35,62,78,80], recent works utilize graph convolution network (GCN) [35,68,70,105] and graph attention network (GAT) [23, 78, 79] to model the syntax graph for shortening the distance between the aspect and its sentiment trigger words and leveraging the syntactical information. [105] employed LSTM as encoder and exploit GCN to capture local syntactic information via encoding the syntax graph produced by off-the-shelf dependency parsers. [79] proposed Relational MHA, which can capture the global dependency between the aspect and each context word via operating on the star-shaped aspect-oriented syntax graph.

To enhance the context modeling, [19] and [6] trained their models on both document-level sentiment classification and ASC tasks in the multi-task framework with a shared encoder. [90] proposed an aspect-aware LSTM which introduces aspect information into LSTM cells to generate better context hidden states in which more aspect-related information is retained and aspect-irrelevant information is discarded. As BERT has proven its power of language modeling on heterogeneous NLP tasks, more recently proposed work [70, 79] adopted BERT as the context encoder to obtain high-quality hidden states.

However, prior models neglect to leverage aspect knowledge, resulting in inadequate aspect semantics. And the syntactic information they captured is insufficient. In this thesis, we propose KaGRMN-DSG (chapter 3) to solve these two challenges. There are two main differences from our model and previous works. The first one is leveraging

aspect knowledge, which is achieved by a novel KaGR-MN. The other one is combining both of local syntactic information and global relational information, which is achieved by a DSG-Net.

Besides, since existing models apply GNNs on the whole syntax graph, they suffer from two problems: *noisy information aggregation* and *loss of distant correlations*. In this thesis, we propose a novel model termed Neural Subgraph Explorer (chapter 4) to solve these two problems.

2.3 Joint Dialog Sentiment Classification and Act Recognition

Dialog Sentiment Classification [15, 18, 29, 65, 112, 113] and Dialog Act Recognition [26, 59, 61, 64] are both utterance-level classification tasks. Recently, it has been found that these two tasks are correlative, and they can work together to indicate the speaker's more comprehensive intentions [33]. With the development of well-annotated corpora, [3, 39], in which both the act label and sentiment label of each utterance are provided, several models have been proposed to tackle the joint dialog sentiment classification and act recognition task.

[3] proposed a multi-task framework based on a shared encoder that implicitly models the dual-task correlations. [33] integrated the identifications of dialog acts, predictors and sentiments into a unified model. To explicitly model the mutual interactions between the two tasks, [53] proposed a stacked co-interactive relation layer and [37] proposed a context-aware dynamic convolution network to capture the crucial local context. More recently, [55] proposed Co-GAT, which applies graph attentions on a fully-connected undirected graph consisting of two groups of nodes corresponding to the two tasks, respectively.

In this thesis, we propose a novel model termed DARER (chapter 5), which is different from previous works on three aspects. First, we model the inner- and inter-speaker temporal dependencies for dialog understanding. Second, we model the cross- and self-task temporal dependencies for dual-task reasoning; Third, we achieve prediction-level interactions in which the estimated label distributions act as important and explicit clues other than semantics.

2.4 Joint Multiple Intent Detection and Slot Filling

The correlations between intent detection and slot filling have been widely recognized. To leverage them, a group of models [11, 16, 17, 36, 43, 48, 54, 56, 88, 106, 109] are proposed to tackle the joint task of intent detection and slot filling in a multi-task manner. However, the intent detection modules in the above models can only handle the utterances expressing a single intent, which may not be practical in real-world scenarios, where there are usually multi-intent utterances.

To this end, [32] proposed a multi-intent SLU model, and [13] proposed the first model to jointly model the tasks of multiple intent detection and slot filling via a slot-gate mechanism. Furthermore, as graph neural networks have been widely utilized in various tasks [2, 66, 79, 92, 94, 97], they have been leveraged to model the correlations between intent and slot. [58] proposed an adaptive graph-interactive framework to introduce the fine-grained multiple intent information into slot filling achieved by GATs. More recently, [57] proposed another GAT-based model, which includes a non-autoregressive slot decoder conducting parallel decoding for slot filling and achieves the state-of-the-art performance.

In this thesis, we propose two novel models, Co-guiding Net (chapter 6) and ReLa-Net (chapter 7), to tackles the joint task of multiple intent detection and slot filling.

Existing methods only model the one-way guidance from multiple intent detection to slot filling. Besides, they adopt homogeneous graphs and vanilla GATs to achieve the interactions between the predicted intents and slot semantics. In contrast, Co-guiding Net (1) achieve the mutual guidances between the two tasks; (2) propose the heterogeneous semantics-label graphs to represent the dependencies among the semantics and predicted labels; (3) we propose the Heterogeneous Graph Attention Network to model the semantics-label interactions on the heterogeneous semantics-label graphs.

Previous models ignore the correlations among the two tasks' labels, treating their embeddings as separated parameters to be learned. Our ReLa-Net is significantly different from previous ones. This is the first work to discover, capture and leverage the topologies and relations among the labels for joint multiple intent detection and slot filling.

Chapter 3

KaGRMN-DSG

3.1 Introduction

Aspect-level sentiment classification (ASC) [63] is a fine-grained task of sentiment classification or emotion recognition [4, 52, 89, 107]. ASC aims to infer the fine-grained sentiment of a given aspect mentioned in a review. Generally, an aspect is a noun phrase included in a review sentence. For example, in a review "It took so long to get the check, while the dinner is great.", there are two aspects (*check* and *dinner*) of opposite sentiments. ASC has received increasing attention and interest from both academia and the industry due to its wide applications in real-life scenarios such as dialog systems [1], online reviews [51] and social networks [10]. Prior works have noticed the importance of aspect-context interaction. Different kinds of attention mechanisms [12, 45, 71] are proposed to extract aspect-relevant semantics from the hidden states of context words. And more recently, syntactic information is widely leveraged to facilitate the interactions between the aspect and its related words that are distant in context sequence. Graph Convolutional Networks (GCN) [70, 105] and Graph Attention Networks (GAT) [25, 79] are adopted to encode the syntax graphs predicted by off-the-shelf dependency parsers. [68, 105] employed GCNs to capture the local syntactic information. [79] proposed relational multi-head attention (Relational MHA) to capture the global relational information between aspect and each context word.

However, little attention has been spent on aspect representation and its conveyed semantics. Aspect representation and its semantics not only guide the aspect-context interaction but also provide important clues for ASC. Despite its importance, in previous works [5,79,105], aspect representation is simply derived by pooling the hidden states of aspect words. In Sec. 3.3.8 we empirically study the aspect representation generated by BERT [9], and two cases are shown in Table 3.8. We can find that

BERT cannot capture the exact meanings and property information of *Mountain Lion OS* and *iTune*, although it is one of the strongest language models. Merely relying on pre-trained large language models cannot obtain sufficiently effective and informative aspect representation, making it hard for machines to address ASC. In contrast, humans can easily handle ASC and we conjecture the key to master this task is to leverage the adequate aspect knowledge they often refer to as the clue. Thinking of and leveraging the aspect knowledge are instinctive reactions of humans when they read an aspect in a review. For example, there is a review "Just a not bad restaurant, because the cheese and chips are both very soft." With the knowledge of 'cheese' and 'chips', humans are aware that the former should be soft and the latter should be click (not soft). Hence it is easy for humans to infer the positive sentiment of 'cheese' and the negative sentiment of 'chips'. However, in contrast, in ASC models there is no such mechanism, and aspect knowledge has not been explored or leveraged. Inheriting this deficiency, the aspect representation and semantics derived by prior models may lose important aspect information, which hinders aspect sentiment reasoning and make ASC challenging for machines.

On the other hand, both GCN and Relational MHA are useful for modeling distinct syntax graphs, but they have respective shortages: GCN is hard to capture the global relations between aspect and its non-adjacent context words on the syntax graph; Relational MHA fails to capture the local syntactic information among context words because they are isolated from each other on the star-shaped aspect-oriented syntax graph. However, prior works only consider one of them, resulting in insufficient syntactic information.

To tackle the aforementioned two challenges, we suggest that (1) aspect knowledge should be explicitly leveraged in ASC models; (2) both kinds of syntactic information should be combined to capture sufficient syntactic information. We observe that there is plenty of entity descriptions in popular and easily accessible knowledge bases, such as DBpedia[1] and Wikipedia[2]. From their statistics, there are about 6.6 Million and 50 Million entities in current DBpedia and Wikipedia datasets. These descriptions can sufficiently represent the entities' meanings and conveying a wealth of entities' knowledge. In ASC, aspects are always entities, making it more convenient to retrieve their descriptions.

In this work, we propose a **K**nowledge-**a**ware **G**ated **R**ecurrent **M**emory **N**etwork with **D**ual **S**yntax **G**raph Modeling (**KaGRMN-DSG**) model as our solution to the

[1]https://wiki.dbpedia.org/
[2]https://www.wikipedia.org/

two challenges. Specifically, its novelty lies in three core modules. The **first** one is Knowledge-aware Gated Recurrent Memory Network (KaGR-MN) which recurrently integrates the aspect knowledge into aspect representation and then context memories. An aspect-to-description attention mechanism is devised to dynamically summarize the needed aspect knowledge from the aspect description regarding the current semantic state. An adaptive knowledge integrating gate is designed to adaptively integrate the summarized knowledge into aspect representation. Then a self multi-head attention is employed to contextualize the integrated knowledge and update the context memory bank. The **second** one is Dual Syntax Graph Network (DSG-Net), which marries the proposed Position-aware GCN and Relational MHA, then learns the dual syntactic interaction to comprehensively capture sufficient syntactic information. The **third** one is the knowledge integrating gate (KI Gate) which re-enhances the final representation with further needed knowledge.

We highlight our contributions as follows:

(1) Based on plenty of informative entity descriptions from easily accessible knowledge bases, we end-to-end embed and leverage the aspect knowledge to address ASC.

(2) We propose a novel KaGR-MN, which combines the advantages of LSTM, Transformer, and Memory Networks. It recurrently embeds and integrates beneficial aspect knowledge into aspect representation and all context memories.

(3) We propose a dual syntax graph network, in which the local syntactic information and global relational information are combined to comprehensively capture sufficient syntactic information.

(4) We conduct extensive experiments on three benchmark datasets. Results show that our model achieves new state-of-the-art performances, significantly outperforming previous best results. Ablation study and further analysis validate the effectiveness of our model.

3.2 Model

Overview The architecture of our KaGRMN-DSG model is illustrated in Fig. 3.1. To extract the beneficial clues for aspects, knowledge-aware gated recurrent memory network and knowledge integration gate incorporate summarized aspect knowledge to enrich aspect representation and all context memories. To capture sufficient syntactic information, dual syntax graph network combines local syntactic information and global relational information, then learns their mutual interaction. To comprehensively abstract high-level clues, aspect-to-context attention mechanism aggregates

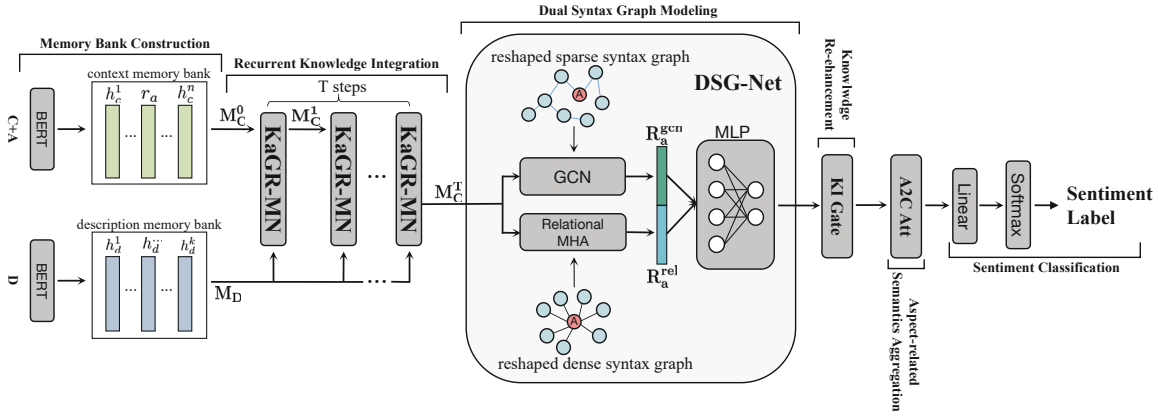


Figure 3.1: The architecture of KaGRMN-DSG. The internal architecture of KaGR-MN cell is shown in Fig.3.2

aspect-related semantics from all hidden states into the final representation. And we believe that these modules can effective cooperate to further improve aspect sentiment reasoning.

Description Retrieval We use aspect (**A**) to query DBpedia first and then Wikipedia to get its description (**D**). If multiple descriptions are returned (polysemy), the one with the highest semantic similarity to context (**C**) is selected as **D**. The semantic similarity of a description candidate and review context is calculated as:

$$avg(C) = \frac{1}{N_C}\sum_{i=1}^{N_C} e(c_i) \tag{3.1}$$

$$avg(D') = \frac{1}{N_D}\sum_{i=1}^{N_D} e(d_i) \tag{3.2}$$

$$sim(C, D') = cos\big(\alpha * avg(C) + (1-\alpha) * e(dl), avg(D')\big) \tag{3.3}$$

where N_C and N'_D denotes the number of words in the context and description candidate respectively, $e(w)$ denotes the word embedding[3] of word w, dl denotes domain label (e.g. the dl of Lap14 dataset is 'laptop'). Here we intuitively set α as 0.5 because both of the context semantics $avg(C)$ and domain information $e(dl)$ are important in selecting the correct description candidate. The reason why we use domain label here is that sometimes there may be not enough words conveying domain-specific semantics for distinguishing the needed description. Besides, the retrieval is enhanced with some rules, such as soft matching with lemmatization and stop word filtering. Finally, about 70% aspects in the datasets can be equipped with retrieved descriptions.

[3]We use Glove word embedding [49].

3.2.1 Memory Bank Construction

In this work, we adopt BERT to encode the description and context to produce their hidden states. For description ($\mathbf{D}$), the formal input is $\langle$`[CLS]`; $\mathbf{D}$; `[SEP]`$\rangle$, where $\langle;\rangle$ denotes concatenation operation. The description is encoded in the single-sentence manner then a series of its hidden states is generated: $\mathbf{H_D} = \{h_d^i \in \mathbb{R}^{d_e}\}_{i=1}^{N_D}$, which is taken as the description memory bank $\mathbf{M_D}$.

As for context ($\mathbf{C}$), we model the context-aspect pair in the sentence-pair manner to generate aspect-aware hidden states [90]. The formal input is $\langle$`[CLS]`; $\mathbf{C}$; `[SEP]`; $\mathbf{A}$; `[SEP]`$\rangle$. In this way, we obtain the hidden state of `[CLS]`: $\mathbf{h_{cls}}$ and a series of aspect-aware context hidden states: $\mathbf{H_C} = \{h_c^i \in \mathbb{R}^{d_e}\}_{i=1}^{N_C}$. As BERT has a strong capability of sentence-pair modeling, $\mathbf{h_{cls}}$ contains not only the information from both of the aspect and the context but also their dependencies. Thus we take $\mathbf{h_{cls}}$ as the initial contextualized aspect representation $\mathbf{r_a^0}$. Then we use $\mathbf{r_a^0}$ to replace the hidden states of aspect words ($\mathbf{H_A} = \{h_a^i \in \mathbb{R}^{d_e}\}_{i=1}^{N_A}$) in $\mathbf{H_C}$, obtaining the initial context memory bank $\mathbf{M_C^0} = [h_c^1, h_c^2, ..., \mathbf{r_a^0}, ..., h_c^N]$, where $N = N_C - N_A + 1$.

$\mathbf{M_D}$ and $\mathbf{M_C^0}$ are two strands of input of KaGR-MN cell. Along time steps, $\mathbf{M_C}$ is recurrently updated while $\mathbf{M_D}$ remains identical.

3.2.2 Knowledge-aware Gated Recurrent Memory Network

As the series of context hidden states and description hidden states have been obtained, now the challenge is *how to incorporate as much beneficial aspect knowledge as possible without losing the original semantics obtained from BERT?*.

The first thing is to conserve the original semantics in the context memories obtained from BERT. To this end, we employ Memory Networks (MNs) as the backbone to store context memories, because that MNs can accurately remember original facts [85]. Secondly, we are supposed to make the integrated knowledge beneficial. In other words, we should provide each sample the aspect knowledge it needs. Hence we propose an aspect-to-description attention (A2D Att) mechanism to summarize the needed aspect knowledge from the description memory bank. Thirdly, we should integrate the beneficial aspect knowledge into the aspect representation. Then we propose an adaptive knowledge integration gate, which borrows the idea of gating mechanisms in LSTM [21]. Gate mechanism has proven its strong ability of information integration in many tasks [7, 90]. However, only integrate knowledge into aspect representation is insufficient, not exploring the full value of aspect knowledge. It is intuitive that the aspect knowledge should be incorporated into all context memories. Besides,

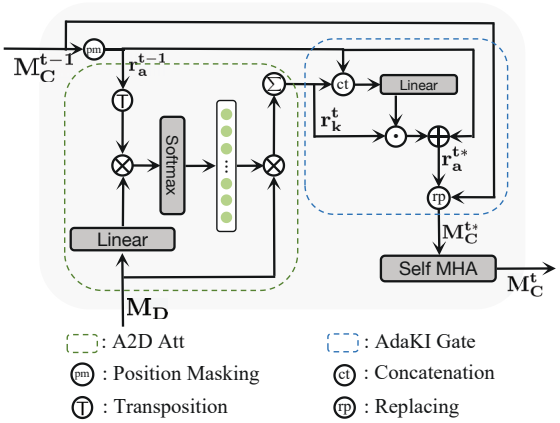


Figure 3.2: The architecture of KaGR-MN cell.

an appropriate mechanism should be devised to update the context memory bank. To achieve these two goals, inspired by Transformer [77], we utilize self multi-head attention to update the context memories, and in the meanwhile, the aspect knowledge in the aspect representation can be spread to all context memories. Finally, all the above mechanisms form the Knowledge-aware Gated Recurrent Memory Network (KaGR-MN), which combines the advantages of MNs, LSTM, and Transformer.

The architecture of KaGR-MN cell is illustrated in Fig. 3.2. In the following texts, we depict the details of KaGR-MN.

3.2.2.1 Dynamic Knowledge Summarizing

Intuitively, on the one hand, the context-aspect pair of each sample may demand individual aspect knowledge, even if they have the same aspect. On the other hand, at each time step, KaGR-MN should integrate specifically needed aspect knowledge according to the current cell state. Therefore, the aspect knowledge summarizing should be dynamic. To achieve this, we design an aspect-to-description attention (A2D Att) mechanism to dynamically summarize the specifically needed aspect knowledge from the description memory bank $\mathbf{M_D}$ at each time step. The architecture of A2D Att is shown in Fig. 3.2. At each time step (t), the aspect representation of previous time step $\mathbf{r_a^{t-1}}$ serves as the cell state and is used to query $\mathbf{M_D}$. Then an attention weight α is assigned to each h_d regarding its importance to $\mathbf{r_a^{t-1}}$:

$$\begin{aligned}\alpha_i =& \texttt{SoftMax}(\mathcal{F}(h_d^i, r_a^{t-1})) \\ =& \frac{\exp(\mathcal{F}(h_d^i, r_a^{t-1}))}{\sum_{k=1}^{N_D} \exp(\mathcal{F}(h_d^k, r_a^{t-1}))}\end{aligned} \tag{3.4}$$

where $\mathcal{F}(h_d^i, r_a^{t-1}))$ is a score function defined as:

$$\mathcal{F}(h_d^i, r_a^{t-1})) = (\mathbf{W_d}\, h_d^i + \mathbf{b_d})\, (\mathbf{r_a^{t-1}})^\mathsf{T}, \tag{3.5}$$

where $\mathbf{W_d}$ and $\mathbf{b_d}$ are weight matrix and bias respectively, and T denotes transposition. Then we can obtain the summarized knowledge representation as: $\mathbf{r_k^t} = \sum_{i=1}^{N_D} \alpha_i h_d^i$.

3.2.2.2 Adaptive Knowledge Integration

As the specifically needed knowledge has been summarized, it should be integrated into the aspect representation regarding the current cell state. As the gate mechanisms [7, 21, 60] have proven their ability of controlling information flow, here we design an **Ada**ptive **K**nowledge **I**ntegration (AdaKI) Gate to integrate $\mathbf{r_k^t}$ into $\mathbf{r_a^{t-1}}$. Its architecture is shown in Fig. 3.2. AdaKI Gate can be formulated as:

$$\mathbf{r_a^{t*}} = \mathbf{r_a^{t-1}} + \mathbf{r_k^t} \odot (\mathbf{W_k}[\mathbf{r_a^{t-1}}, \mathbf{r_k^t}]), \tag{3.6}$$

where $\odot$ denotes Hadamard product, $[,]$ denotes concatenation and $\mathbf{W_k}$ is weight matrix. The core of AdaKI Gate is to produce a gate vector using $\mathbf{r_k^t}$ and $\mathbf{r_a^{t-1}}$. This gate vector achieves the fine-grained control on each dimension of $\mathbf{r_k^t}$.

There are two merits of this fine-grained control. First, AdaKI Gate can determine what knowledge and how much knowledge from $\mathbf{r_k^t}$ should be integrated into $\mathbf{r_a^{t-1}}$. Second, it can map the integrated knowledge into the same semantic space of $\mathbf{r_a^{t-1}}$ and $\mathbf{M_C^{t-1}}$. This adaption helps maintain the semantics consistency of $\mathbf{r_a^{t*}}$ and $\mathbf{M_C^{t-1}}$, which is beneficial to later knowledge contextualizing. In Sec. 3.3.6, we investigate the effect of different knowledge gates used here. After $\mathbf{r_a^{t*}}$ is obtained, it replaces $\mathbf{r_a^{t-1}}$ in $\mathbf{M_C^{t-1}}$, forming $\mathbf{M_C^{t*}}$.

3.2.2.3 Knowledge Contextualizing and Context Memory Bank Updating

Although the needed beneficial knowledge has been integrated into $\mathbf{r_a^{t*}}$, the other context memories in $\mathbf{M_C^{t*}}$ remain the same as the ones in $\mathbf{M_C^{t-1}}$. Intuitively, all context memories should benefit from aspect knowledge to facilitate aspect-related information aggregation. To achieve this, we propose a knowledge contextualizing mechanism to broadcast the newly-integrated knowledge in $\mathbf{r_a^{t*}}$ to all context memories in $\mathbf{M_C^{t*}}$. Here we borrow the idea of self-attention [41, 77], which can effectively relate the different tokens in a sentence and capture the intra-sentence dependencies.

In this work, we adopt the self multi-head attention (Self MHA) formulation in [77]. We first map $\mathbf{M_C^{t*}}$ to queries ($\mathbf{Q}$), keys ($\mathbf{K}$) and values ($\mathbf{V}$) matrices by individual

linear projections, where $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{N \times d_s}$. And this process repeats H_n^s times, where H_n^s is the number of heads and $H_n^s \times d_s = d_e$. The scaled dot-product attention is used to produce the output of each head, then all of the H_n^s outputs are concatenated to form the updated context memory bank $\mathbf{M_C^t}$:

$$\mathbf{M_C^t} = \|_{h=1}^{H_n^s} \texttt{SoftMax}\left(\frac{\mathbf{Q}\mathbf{K}^\mathsf{T}}{\sqrt{d_s}}\right).\mathbf{V} \tag{3.7}$$

This simple but effective knowledge contextualizing mechanism updates $\mathbf{M_C^{t*}}$ and $\mathbf{r_a^{t*}}$ by letting context memories (including $\mathbf{r_a^{t*}}$) exchange useful information with each other, which is beneficial to capture aspect-related information. Along time steps, $\mathbf{r_a}$ and $\mathbf{M_C}$ would contain more and more reasonable and beneficial semantics for ASC.

3.2.3 Dual Syntax Graph Network

As proven in prior works, GCN can capture local syntactic information and Relational MHA can capture the global relation between the aspect and each context word via operating on the star-shaped aspect-oriented syntax graph (as shown in Fig. 3.1). However, as we have discussed in Sec. 3.1, they have respective shortages. They can only capture one of the two kinds of syntactic information and lose the other. Previous models only employ one of them, leading to insufficient syntactical information.

To this end, we propose DSG-Net (as shown in Fig. 3.1) which marries the proposed Position-aware GCN and Relational MHA and learns their interaction, capturing sufficient syntactic information.

3.2.3.1 Local Syntactic Information Modeling

Graph Construction Based on the original syntax graph G[4], we first add a new aspect node A and merge all edges between nodes of aspect words and non-aspect context words to A. Then we delete all of the original nodes of aspect words and their edges. The obtained graph is similar to G, and only several context word nodes are connected to A. Thus we term it sparse graph G_s (shown in Fig.3.1).

Position-aware GCN In this work, we augment the standard GCN with a position weight $w_p^i = 1 - \frac{|i-\tau|}{N+1}$, in which τ denotes the position of aspect, i denotes the i^{th} context word. As the Self MHA in KaGR-MN does not consider the order of context memories, some positional and ordering information may be lost. w_p^i can supplement this information, which helps capture local syntactic information. Besides, it indicates

[4]obtained by spaCy toolkit: https://spacy.io/

the position of A and highlights the potential aspect-related words which are generally closer to A. In l^{th}-layer, the local neighborhood information is aggregated as:

$$h_i^l = \sum_{j \in \mathcal{N}_i^s} \mathbf{W_g^s}(w_p^j \; h_j^{l-1})/(d_i + 1) + \mathbf{b_g^s}, \tag{3.8}$$

in which $\mathcal{N}_i^s$ is the first-order neighbors of node i (including i) in G_s, d_i is the degree of node i, $\mathbf{W_g^s}$ and $\mathbf{b_g^s}$ are weight matrix and bias.

3.2.3.2 Global Relational Information Modeling

We obtain the star-shaped aspect-oriented syntax graph following [79]. In this syntax graph, every context word directly connects to the aspect node A, so we term it dense graph G_d. Then we employ the Relational MHA to model the global relational dependency between aspect and each context word. The node representation is:

$$\begin{aligned} h_i &= \sum_{m=1}^{H_n^d} \left(\sum_{j \in \mathcal{N}_i^d} \beta_{ij}^m \mathbf{W_m^1} h_j \right) / H_n^d, \\ \beta_{ij}^m &= \texttt{SoftMax}(g_{ij}^m), \\ g_{ij}^m &= \texttt{ReLU}\left(r_{ij}\mathbf{W_m^2} + \mathbf{b_m^1}\right)\mathbf{W_m^3} + \mathbf{b_m^2}, \end{aligned} \tag{3.9}$$

where H_n^d denotes the head number, r_{ij} is the embedding of the relation between nodes i and j, $\mathbf{W_m^{1,2,3}}$ and $\mathbf{b_m^{1,2}}$ are weight matrices and biases.

3.2.3.3 Dual Syntactic Information Fusion

Now the Position-aware GCN has captured the important local syntactic information and the Relational MHA has captured the important global relational information. To integrate them together and let them compensate for each other, we concatenate the aspect node representations respectively derived by Position-aware GCN and Relational MHA, then we employ a multi-layer perception (MLP), which can automatically abstract the integrated representation [47, 53], to generate the unified node representation sequence, which include the unified aspect representation $\widetilde{\mathbf{R_a}}$.

3.2.4 Knowledge Re-enhancement

After graph modeling, sufficient syntactic information has been integrated into $\widetilde{\mathbf{R_a}}$. On the one hand, some new clues may be captured by DSG-Net and retained in $\widetilde{\mathbf{R_a}}$. Thus $\widetilde{\mathbf{R_a}}$ may further need more aspect knowledge to collaborate with these new clues

to support ASC. On the other hand, as the syntax graph may be imperfectly generated by the parser, some wrong connections and relations may be introduced. In this case, re-integrating some knowledge can help alleviate the influence of the imperfect syntax graph. To this end, we design a knowledge integrating gate (KI Gate) to re-enhance $\widetilde{\mathbf{R_a}}$ with further needed knowledge contained in $\mathbf{r_k^T}$. The function of KI gate is given as:

$$\mathbf{R_a} = \widetilde{\mathbf{R_a}} + \mathbf{r_k^T} * \mathbf{W_k^r}[\widetilde{\mathbf{R_a}}, \mathbf{r_k^T}], \tag{3.10}$$

where $\mathbf{W_k^r}$ is weight matrix. Here $\widetilde{\mathbf{R_a}}$ and $\mathbf{r_k^T}$ produces a gate scalar rather than a gate vector. There is no subsequent contextualizing module thus $\mathbf{r_k^T}$ can be directly integrated into $\widetilde{\mathbf{R_a}}$ without fine-tuning for adaption. In Sec. 3.3.6, we investigate the effect of different knowledge gates used here.

3.2.5 Aspect-related Semantics Aggregation

Here we employ an Aspect-to-Context Attention (A2C Att) mechanism to aggregate the aspect-related semantics retained in all hidden states into a final representation $\mathbf{R_f}$. Similar to A2D Att, A2C Att can be formulated as:

$$\beta_i = \texttt{SoftMax}(\mathcal{F}(h_c^i, \mathbf{R_a})), \tag{3.11}$$

$$\mathcal{F}(h_c^i, R_a) = (\mathbf{W_{ac}}\, h_c^i + \mathbf{b_{ac}})\, (\mathbf{R_a})^{\mathsf{T}}, \tag{3.12}$$

$$\mathbf{R_f} = \sum_{i=1}^{N} \alpha_i h_c^i, \tag{3.13}$$

where $\mathbf{W_{ac}}$ and $\mathbf{b_{ac}}$ are weight and bias.

3.2.6 Sentiment Classification

We concatenate $\mathbf{R_f}$ with $\mathbf{h_{cls}}$ and then fed the final vector into a linear layer, which is followed by a `SoftMax` classifier for prediction:

$$\mathbf{P} = \texttt{SoftMax}(\mathbf{W_p}[\mathbf{h_{cls}}, \mathbf{R_f}] + \mathbf{b_p}), \tag{3.14}$$

where $\mathbf{P}$ is the predicted sentiment distribution, $\mathbf{W_p}$ and $\mathbf{b_p}$ are weight matrix and bias. The cross-entropy loss function is adopted for model training.

There are two reasons why we introduce $\mathbf{h_{cls}}$ here. First, this can add a skip connection to BERT, shortening its loss back-propagation path to facilitate training. The second is for robustness. Possibly the syntax graphs are imperfect and the integrated knowledge contains noise. Hence $\mathbf{h_{cls}}$ serves as a reference and makes the whole model more robust.

3.3 Experiments

3.3.1 Datasets

We conduct experiments on three popular datasets for the ASC task: Lap14 and Res14 datasets are from SemEval 2014 task 4 [51], and Res15 dataset is from SemEval 2015 task 12 [50]. The statistics of all datasets are presented in Table 3.1.

Table 3.1: Dataset statistics of the three datasets.

Dataset	Positive		Neutral		Negative	
	Train	Test	Train	Test	Train	Test
Lap14	994	341	464	169	870	128
Res14	2164	728	637	196	807	196
Res15	912	326	36	34	256	182

3.3.2 Experiment Setup

We adopt the BERT-base uncased version [9]. We train our model using Adam optimizer [34] with default configuration. The hyper-parameters are listed in Table 3.2. Accuracy (Acc) and Macro-F1 (F1) are adopted as evaluation metrics. As there is no official validation set, following previous works [70, 105, 108], we run our model three times with random initialization and report the average results on test sets, as shown in Table 3.3. And to compare with the works reporting best results, we also report the best results on test sets, as shown in Table 3.4. All computations are done on an NVIDIA Quadro RTX 6000 GPU.

3.3.3 Compared Baselines

According to what kinds of external information are utilized, we divide the baselines into several group:
1) No external information is used:

- IAN [45] separately encodes the aspect and context, then model their interactions using an interactive attention mechanism.

2) External corpus is used:

- PRET+MULT [19] first pre-trains the model on document-level task, then trains the model on both document-level sentiment classification and ASC in the multi-task learning framework.

Table 3.2: Setting of hyper-parameters.

Hyper-params	Dataset		
	Lap14	Res14	Res15
learning rate	1×10^{-5}	5×10^{-5}	3×10^{-5}
batch size	32	32	32
dropout rate	0.3	0.3	0.3
d_e	768	768	768
d_s	256	256	128
H_n^s	3	3	6
H_n^d	2	4	6
$\mathbf{T}$	4	4	2
GCN layer number	2	2	2

- TransCap [6] utilizes a devised aspect-based capsule network to transfer knowledge from document-level task to aspect-level task.

3) Syntax Graph is used:

- ASGCN [105] employs a GCN to encode the syntax graph for capturing local syntactic information.
- BiGCN [108] convolutes over hierarchical syntactic and lexical graphs to encode not only original syntactic information but also the corpus level word co-occurrence information.

4) BERT encoder is used:

- BERT-SPC [9] takes the same input as our model and use h_{cls} for sentiment classification.
- AEN-BERT [67] adopts BERT encoder and uses the attentional encoder network to model the interactions between the aspect and context.

5) Both of syntax graph and BERT encoder are used:

- R-GAT+BERT [79] use the relational graph attention network to aggregate the global relational information from all context word into the aspect node representation.
- DGEDT-BERT [70] employs a dual-transformer network to model the interactions between the flat textual knowledge and dependency graph empowered knowledge.
- A-KVMN+BERT [73] uses a key-value memory network to leverage not only word-word relations but also their dependency types.

- BERT+T-GCN [72] leverages the dependency types in T-GCN and use an attentive layer ensemble to learn the comprehensive representation from different T-GCN layers.
- SAGAT [25] utilizes graph attention network and BERT to fully obtain both syntax and semantic information.
- KGCapsAN-BERT [104] utilizes multi-prior knowledge to guide the capsule attention process and use a GCN-based syntactic layer to integrate the syntactic knowledge.

And we label all models with what kinds of external information they leverage, as shown in Table 3.3 and Table 3.4.

3.3.4 Main Results

Table 3.3: Performances comparisons of average results with random initialization. $\mathcal{K}, \mathcal{B}, \mathcal{T}$ and $\mathcal{G}$ denote the model leverages aspect $\mathcal{K}$nowledge, $\mathcal{B}$ERT, extra $\mathcal{T}$raining corpus and syntax $\mathcal{G}$raph, respectively. Best results are in **bold** and previous SOTA results are underlined. * denotes that we produce the results using their original source codes. † indicates KaGRMN-DSG significantly outperforms baselines under t-test ($p < 0.01$).

External Information			Model	Lap14		Res14		Res15	
				Acc	F1	Acc	F1	Acc	F1
–	–	–	IAN [45]	72.05	67.38	79.26	70.09	78.54	52.65
–	$\mathcal{T}$	–	PRET+MULT [19]	71.15	67.46	79.11	69.73	81.30	68.74
–	$\mathcal{T}$	–	TransCap [6]	73.51	69.81	79.55	71.41	-	-
–	–	$\mathcal{G}$	ASGCN [105]	75.55	71.05	80.77	72.02	79.89	61.89
–	–	$\mathcal{G}$	BiGCN [108]	74.59	71.84	81.97	73.48	81.16	64.79
–	$\mathcal{B}$	–	BERT-SPC* [9]	78.47	73.67	84.94	78.00	83.40	65.00
–	$\mathcal{B}$	–	AEN-BERT [67]	79.93	76.31	83.12	73.76	-	-
–	$\mathcal{B}$	$\mathcal{G}$	R-GAT+BERT* [79]	79.31	75.40	86.10	80.04	83.95	69.47
–	$\mathcal{B}$	$\mathcal{G}$	DGEDT-BERT [70]	79.8	75.6	86.3	80.0	84.0	71.0
–	$\mathcal{B}$	$\mathcal{G}$	A-KVMN+BERT* [73]	79.20	75.76	85.89	78.29	83.89	67.88
–	$\mathcal{B}$	$\mathcal{G}$	BERT+T-GCN* [72]	80.56	76.95	85.95	79.40	84.81	71.09
$\mathcal{K}$	$\mathcal{B}$	$\mathcal{G}$	KaGRMN-DSG (Ours)	**81.87**†	**78.43**†	**87.35**†	**81.21**†	**86.59**†	**74.46**†
			Our Improvements	**1.62%**	**1.92%**	**1.22%**	**1.46%**	**2.10%**	**4.74%**

The performance comparison of all models on average scores is shown in Table 3.3, and the comparison on best scores is shown in Table 3.4. We can observe that: Syntax graphs, external training corpus, and BERT can all improve ASC. Especially, simple BERT-SPC significantly outperforms all models that do not adopt BERT, even if some of them leverage syntax graph and external training corpus. This shows the power of pre-trained language models on ASC. And combining BERT and syntactic

Table 3.4: Performances comparisons of best results. $\mathcal{K}, \mathcal{B}, \mathcal{T}$ and $\mathcal{G}$ denote the model leverages aspect $\mathcal{K}$nowledge, $\mathcal{B}$ERT, extra $\mathcal{T}$raining corpus and syntax $\mathcal{G}$raph, respectively. Best results are in **bold** and previous SOTA results are underlined. * denotes that we produce the results using their original source codes.

External Information			Model	Lap14		Res14		Res15	
				Acc	F1	Acc	F1	Acc	F1
–	$\mathcal{B}$	–	BERT-SPC* [9]	78.84	73.95	85.80	78.48	83.76	68.33
–	$\mathcal{B}$	$\mathcal{G}$	SAGAT [25]	80.37	76.94	85.08	77.94	-	-
–	$\mathcal{B}$	$\mathcal{G}$	KGCapsAN-BERT [104]	79.47	76.61	85.36	79.00	-	-
–	$\mathcal{B}$	$\mathcal{G}$	R-GAT+BERT* [79]	79.46	75.75	86.61	80.78	84.13	71.12
–	$\mathcal{B}$	$\mathcal{G}$	A-KVMN+BERT [73]	79.78	76.14	85.98	77.94	84.14	68.49
–	$\mathcal{B}$	$\mathcal{G}$	BERT+T-GCN [72]	80.88	77.03	86.16	79.95	85.26	71.69
$\mathcal{K}$	$\mathcal{B}$	$\mathcal{G}$	KaGRMN-DSG (Ours)	**82.13**	**79.42**	**87.68**	**81.98**	**87.08**	**75.34**
			Our Improvements	**1.55%**	**3.10%**	**1.24%**	**1.49%**	**2.13%**	**5.09%**

information can further improve results as sufficient semantics captured by BERT and the syntactic information conveyed by syntax graphs can cooperate to assist ASC. However, all baselines do not leverage aspect knowledge and only consider either local syntactic information or global relational information. As a result, their derived aspect representation lack some important clues of aspect and their captured syntactic information is insufficient, leading to their inferior performance compared to our KaGRMN-DSG model.

We obtain consistent improvements over baselines in terms of Acc and F1 on all datasets, achieving new state-of-the-art results. On average results, our KaGRMN-DSG overpasses previous best results by 1.92%, 1.46%, and 4.74% in terms of Macro-F1 on Lap14, Res14, and Res15 datasets respectively. On best results, KaGRMN-DSG overpasses previous best results by 3.10%, 1.49%, and 5.09% in terms of Macro-F1 on Lap14, Res14, and Res15 datasets respectively. The improvements are contributed by the superiorities of KaGR-MN, which effectively leverage beneficial aspect knowledge, and DSG-Net, which combines GCN and Relational MHA to capture sufficient syntactic information.

3.3.5 Ablation Study

We empirically analyze KaGRMN-DSG and prove the necessity of every component by conducting an ablation study, whose results are shown in Table 3.5. In this section we answer the following research questions (RQs):

Effect of Aspect Knowledge. To study the pure impact of aspect knowledge, we devise two variants: M_1 and M_2. In M_1, the original description is replaced with the aspect itself. In this case, there is no aspect knowledge available for KaGR-MN

Table 3.5: Results of ablation study.

Variants	Lap14	Res14	Res15
	Acc	Acc	Acc
M_0: KaGRMN-DSG (full model)	**81.87**	**87.35**	**86.59**
M_1: w/o Aspect Knowledge ($\mathcal{D}$ is replaced with $\mathcal{A}$)	80.30 (↓ 1.57)	86.43 (↓ 0.92)	85.24 (↓ 1.35)
M_2: only KaGRMN (w/o DSG-Net,KI Gate,A2C Att)	80.72 (↓ 1.15)	86.55 (↓ 0.80)	85.36 (↓ 1.23)
M_3: w/o DSG-Net	80.56 (↓ 1.31)	86.43 (↓ 0.92)	85.56 (↓ 1.03)
M_4: w/o Relational MHA	80.98 (↓ 0.89)	86.67 (↓ 0.68)	85.79 (↓ 0.8)
M_5: w/o Position-aware GCN	81.03 (↓ 0.84)	86.76 (↓ 0.59)	85.67 (↓ 0.92)
M_6: w/o KI Gate	81.09 (↓ 0.78)	87.11 (↓ 0.24)	85.79 (↓ 0.80)
M_7: w/o A2C Att	81.50 (↓ 0.37)	87.00 (↓ 0.35)	86.41 (↓ 0.18)
M_8: w/o A2D Att	80.93 (↓ 0.94)	86.70 (↓ 0.65)	85.61 (↓ 0.98)
M_9: w/o Self MHA	80.36 (↓ 1.51)	86.46 (↓ 0.89)	85.24 (↓ 1.35)

and its function becomes modeling the interactions between the aspect and context. Surprisingly, even without knowledge, M_2 can obtain promising results. We attribute this to the advanced architecture and effective functions of KaGR-MN, in which aspect and context are separately encoded and their interactions are effectively modeled by KaGR-MN. On the other hand, the performance degradation of M_0 convincingly demonstrates the pure improvements contributed by the aspect knowledge conveyed by aspect descriptions. In M_2, DSG-Net, KI Gate, and A2C Att are all removed, so M_2 has a BERT+KaGR-MN architecture and the final aspect representation is used for prediction. M_2 consistently outperforms baselines, proving that KaGR-MN can derive a good enough aspect representation in which the clues for aspect sentiment reasoning are retained. Along time steps, recurrently leveraging aspect knowledge, KaGR-MN can capture more and more beneficial clues, semantics and dependencies then retain them in aspect representation and context memories. And effectively utilizing beneficial aspect knowledge is the key advantage of our method compared with previous works.

Effect of Syntactic Information. The results gap of M_3 and M_0 shows the improvement DSG-Net achieves by cooperating with the aspect knowledge. These results validate the advantages of combining both kinds of syntactic information to capture sufficient syntactic information. We then study the effects of Position-aware GCN and Relational MHA. We can observe that both M_4 and M_5 perform worse than M_0, proving both the local syntactic information and global relational information should be captured for ASC. In previous works, only either one of them is considered, leading to insufficient syntactic information. In contrast, our model marries them and lets them compensate for each other, sufficiently capturing syntactic clues.

Effect of Knowledge Integration Gate. Without KI Gate, M_6 obtains worse results

Table 3.6: Results of different knowledge gate settings.

Variants	Gate 1	Gate 2	Lap14	Res14	Res15
			Acc	Acc	Acc
M_0	AdaKI	KI	**81.87**	**87.35**	**86.59**
M_{10}	AdaKI	AdaKI	81.50 (↓ 0.37)	87.05 (↓ 0.30)	85.98 (↓ 0.61)
M_{11}	KI	KI	81.09 (↓ 0.78)	86.73 (↓ 0.62)	85.36 (↓ 1.23)
M_{12}	KI	AdaKI	81.03 (↓ 0.84)	86.58 (↓ 0.77)	85.24 (↓ 1.35)

than M_0. This indicates that after DSG-Net, some aspect knowledge is further needed and KI Gate is efficient to re-enhance the final aspect representation with the needed knowledge.

Effect of Aspect-to-Context Attention. In M_7, the final aspect representation is used for prediction. We can find that M_7 has limited performance degradation compared to M_0. This proves that although previous modules can discover and extract clues for ASC, there are still important clues contained in non-aspect hidden states rather than final aspect representation. Hence it is necessary to employ A2C Att to aggregate the aspect-related semantics in all hidden states into the final representation.

Effect of A2D Att and Self MHA in KaGR-MN Cell. The significant performance decrease of M_8 shows that A2D Att is indispensable to dynamically summarize the specifically needed aspect knowledge from $\mathbf{M_D}$. Without Self MHA, the integrated knowledge in aspect representation can not be contextualized and context memories cannot be updated. As a result, M_9 performs much worse than M_0.

3.3.6 Investigation on Knowledge Gates

KaGRMN-DSG has two different knowledge gates (AdaKI and KI) for knowledge integrating. Here we empirically investigate these two knowledge gates by testing their four different settings. The results are shown in Table 3.6. We can find that M_{10} and M_{12} have slight decreases in performances when respectively compared with M_0 and M_{11}. This is because KI Gate can preserve the knowledge in $\mathbf{r_k^T}$ while AdaKI Gate may lose some knowledge when adapting to the semantic space of $\widetilde{\mathbf{R}_\mathbf{a}}$. M_{11} and M_{12} perform much worse than M_0 and M_{10}. This is because the semantic space adaption of AdaKI Gate in KaGR-MN can maintain the semantics consistency of $\mathbf{r_a^{t*}}$ and $\mathbf{M_C^{t-1}}$, which is crucial for subsequent knowledge contextualizing.

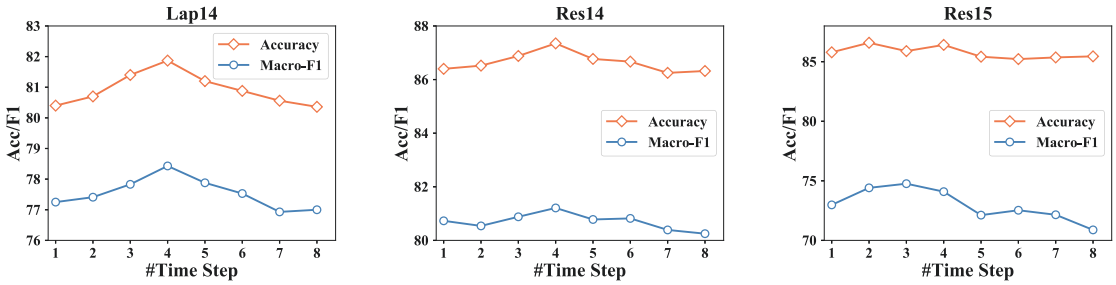


Figure 3.3: Impact of the time step number **T**

Table 3.7: Cases demonstration. [**N**, **P**, **O**] denotes predicted sentiment distribution: [Negative, Positive, Neutral].

Case	[**N**, **P**, **O**]
1. **C**: The [Mountain Lion OS]$^{\mathbf{A}}$ is not hard to figure out if you are familiar with Microsoft Windows. **D**: OS X Mountain Lion is ... Apple Inc.'s desktop and server operating system ...	M_0: [0.0, **0.999**$^{\checkmark}$, 0.001] M_1: [0.01, 0.49$^{\times}$, 0.5]
2. **C**: On start up it asks endless questions just so [iTune]$^{\mathbf{A}}$ can sell you more of their products. **D**: iTunes is a media player, media library, Internet radio broadcaster, mobile device management utility ...	M_0: [**0.57**$^{\checkmark}$, 0.41, 0.02] M_1: [0.03, 0.67$^{\times}$, 0.30]
3. **C**: While the [smoothies]$^{\mathbf{A}}$ are a little big for me, the fresh juices are the best i have ever had! **D**: A smoothie is a drink made from pureed raw fruit and/or vegetables, typically using a blender ...	M_0: [**0.62**$^{\checkmark}$, 0.0, 0.38] M_1: [0.02, 0.97$^{\times}$, 0.01]
4. **C**: All the various Greek and Cypriot dishes are excellent, but the [gyro]$^{\mathbf{A}}$ is the reason to come – if you don't eat one your trip was wasted. **D**: A gyro or gyros is a Greek dish made from meat cooked on a ...	M_0: [0.02, **0.98**$^{\checkmark}$, 0.0] M_1: [0.88$^{\times}$, 0.11, 0.01]

3.3.7 Impact of Time Step Number T

We plot the performance trends of KaGRMN-DSG with increasing **T** on the three datasets, as presented in Fig. 3.3. We can observe that the performances show a trend of increases at first and then decreases. And the best result is obtained when **T** is 2 or 3 for Res15 and 4 for Lap14 and Res14. This shows that appropriately increasing **T** can gradually improve the results, which is consistent with our expectation. This can also prove the effectiveness of the recurrent manner of KaGR-MN. However, too large **T** leads to inferior performances, which is also consistent with our expectation. One possible explanation is that too much knowledge integrated into the aspect representation and context memories will harm their original contextual information. Another is that too many recurrent steps will lead to overfitting on training sets.

Table 3.8: Misunderstanding from BERT presented by semantic cosine similarity (S). v is the average of entity's hidden states. a_i denotes the **A** in case i.

Entity (e)	$S(v_e, v_{a_1})$	Entity (e)	$S(v_e, v_{a_2})$
lion	0.8516	media player	0.4720
mountain	0.7997	radio broadcaster	0.5887
operating system	0.6826	software	0.7051
dangerous animal	0.8272	utility	0.6982

3.3.8 Case Study

We show some cases in Table 3.7. Note that the only difference between KaGRMN-DSG (M_0) and M_1 is that the input **D** in M_1 is replaced with **A**. We can observe that M_0 can accurately predict the correct labels in all cases, while M_1 fails all cases although its overall performance is promising (as shown in Table 3.5)

Without leveraging aspect knowledge, the aspect representation and semantics derived by M_1 are inadequate. As shown in Table 3.8, BERT cannot capture the exact meanings and properties of *Mountain Lion OS* and *iTune*, although it is one of the strongest language models. In Case 1, M_1 regards *Mountain Lion OS* as 'lion' which is 'dangerous'. Then considering 'not hard', M_1 is confused on P and O. In contrast, leveraging aspect knowledge, M_0 captures the exact meaning: an operating system. Then considering the aspect-related semantics ('not hard'), M_0 correctly predicts P. In Case 2, the aspect sentiment expression is a little obscure as there are no explicit sentiment trigger words (e.g. delicious, good, expensive). Even if M_1 captures aspect-related context semantics, it fails due to the lack of property information of *iTune*. Thanks to the integrated aspect knowledge, M_0 is aware that *iTune* is primarily used for media playing rather than selling products, thus correctly predicts N.

Looking into Case 3 and Case 4, we can find that due to the lack of aspect knowledge, M_1 is prone to be affected by some misleading sentiment trigger words: 'best' in case 3 and 'but' in case 4. The reason why M_0 wins M_1 is that M_0 can combine the aspect knowledge and the aspect-related semantics together to capture the correct clues for ASC.

3.3.9 Computation Time Analysis

The comparison of time costing and avg F1 of BERT-SPC, BERT+T-GCN and our KaGRMN-DSG model is shown in Table 3.9. We can find that although our model demands more training time and inference time than BERT-SPC, it overpasses BERT-SPC on avg F1 by a large margin (6.3%). As for BERT+T-GCN, which

Table 3.9: Comparison of training time and inference time (per sample) as well as the avg F1 on the three datasets.

Models	Training Time↓	Inference Time↓	Avg F1↑
BERT-SPC	**0.007309**s	**0.002219**s	73.59%
BERT+T-GCN	0.033835s	0.003350s	76.22%
KaGRMN-DSG	0.015333s	0.004208s	**78.91**%

is the best-performing baseline, although it costs lightly less inference time than our KaGRMN-DSG, it costs much more time for training, and more importantly, its performance is significantly inferior to us. Additionally, since *Local Syntactic Information Modeling* and *Global Relational Information Modeling* both take the output of KaGRMN as input, they can be parallelized theoretically, so the training time and inference time of our KaGRMN-DSG model can be further reduced in practice. In a word, our model may cost more time for training and inference than some baseline models, but it is worthy considering the significant performance improvement.

3.4 Summary

In this paper, we point out the two challenges encountering existing ASC models and we therefore propose a novel KaGRMN-DSG model to end-to-end embed and leverage aspect knowledge, then capture sufficient syntactic information by marrying both kinds of syntactic information. In our model, the integrated beneficial aspect knowledge and sufficient syntactic information can effectively cooperate, yielding new state-of-the-art results.

Chapter 4

Neural Subgraph Explorer

4.1 Introduction

Target (or aspect) sentiment classification (TSC) [69] aims to infer the sentiment polarity of the specific target included in a review context. For example, in the review "The price is reasonable but the service is poor.", there are two targets 'price' and 'service' with opposite sentiment polarities. Generally, TSC task is formulated as a classification task whose input is a given context-target pair.

Earlier dominant neural TSC models are based on attention mechanisms [5, 45], which are designed to capture the correlations between the target and its relevant context words. Although promising progress has been achieved, researchers discovered that attention mechanisms may mistakenly attend to the target's syntactically unrelated words and have difficulty capturing the distant while crucial context words [24, 105]. To this end, the syntax graph of the context is widely leveraged to incorporate the syntactic information via applying graph neural networks (GNNs), e.g. graph convolutional network (GCN) and graph attention network (GAT). With the combination of the BERT [9] which has proven its power in heterogeneous tasks, the recent models of the BERT+Syntax paradigm has achieved state-of-the-art performances [38, 70, 94].

However, despite the remarkable improvements brought by the incorporation of syntactic information, we find existing models suffer from two issues which are illustrated in Fig. 4.1:

- *Noisy information aggregation.* In existing models, the message passing process of GNNs is conducted on the whole syntax graph, and all words transfer information with their neighbors. Consequently, the noisy information contained in target-irrelevant works may be aggregated into the target nodes, which disturbs the prediction. As shown in example (1), 'less expensive' are crucial for inferring

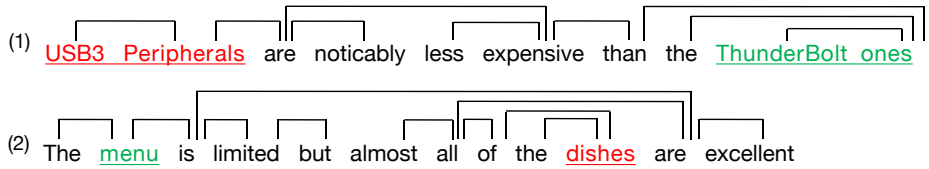


Figure 4.1: Illustration of two examples with syntax dependencies. Example (1) is from Laptop14 dataset while example (2) is from Restaurant14 dataset. Targets are underlined. Red color denotes the sentiment of the target is positive, while color denotes negative.

the positive sentiment of 'USB3 Peripherals' and their distance on the syntax graph is 2. However, the distance between 'less expensive' and another target 'ThunderBolt ones' is also 2, while this review expresses negative sentiment on 'ThunderBolt ones'. In this case, the information of 'less expensive' is aggregated into both targets. For the first target, it is beneficial, while for the later one, its information is noisy, harming the prediction.

- *Loss of distant correlations.* After l-layer GNNs, the information of a node's $\leq l$th-order neighbors can be aggregated into it, while further nodes cannot exchange information with it. Sometimes, the critical words may be distant from the target on the syntax graph. Thus the target loses their crucial information. As shown in example (2), 'excellent' plays the key role in expressing the positive sentiment of 'dishes', while their distance is 4, which means that their correlation cannot be captured by the widely-adopted 2- or 3-layer GNNs [38, 70, 105].

To solve the first issue, target-irrelevant nodes are supposed to be removed from the syntax graph. For the second issue, an alternative solution is to increase the layer number of GNNs. However, this would lead to over-fitting problem and exacerbates the first issue. Another alternative is leveraging the fully-connected self-attention graph [38, 70] to introduce first-order connections. In this case, massive correlative information between the words is introduced. Although the beneficial first-order connections between the target and its related words are introduced, the noisy ones between the target and noisy words are also integrated, which exacerbates the first issue. Thus we should introduce the beneficial first-order connections and eliminate the noisy first-order connections simultaneously.

In this paper, we argue that it is urgent to conduct target-oriented syntax pruning. On the one hand, it can reduce the noisy information via pruning the target-irrelevant nodes. On the other hand, pruning the syntax graph and self-attention graph then merging them can adaptively introduce the beneficial first-order connections between

the target and its related words rather than all first-order correlations, which include the noisy ones. To this end, we propose Neural Subgraph Explorer, whose core is a stacked target-oriented syntax graph pruning layer. We design a multi-hop action score estimator to evaluate the contribution of each node regarding the target. And we leverage the Gumble-Softmax [27] for stable and differentiable discrete action sampling. To obtain the first-order connections between the target and its related words, we apply the non-local self-attention to generate the fully-connected self-attention graph. Then the syntax graph and self-attention graph are pruned and merged into a unified graph, on which the proposed position weighted GCN is applied for message passing. Another advantage of introducing the self-attention graph is that it guarantees the connectivity of the obtained graph, resolving the potential issue that the pruned syntax graph has isolated nodes which harms the message passing. Experimental results on benchmark datasets show that our model achieves new state-of-the-art performance, significantly surpassing existing models.

4.2 Methodology

The architecture of our model is illustrated in Fig. 4.2. We propose the position weighted GCN for graph message passing. And the core of our model is the proposed target-oriented syntax-graph pruning module. Next, we introduce the details.

4.2.1 Contextual and Syntactic Encoding

BERT Encoder. Following the up-to-date models [72,79], we employ BERT encoder to obtain the initial word hidden states. Given the review context word sequence $x_1, x_2, ..., x_{N_c}$ and the target word sequence $a_1, ..., a_{N_t}$, the input of BERT is the target-context pair:

$$\langle [\text{CLS}]; x_1, x_2, ..., x_{N_c}; [\text{SEP}]; a_1, ..., a_{N_t}; [\text{SEP}] \rangle, \tag{4.1}$$

where N_c and N_t are the length of review context and target respectively, and $\langle ; \rangle$ denotes sequence concatenation. Then we obtain the hidden state sequence of the review: $\hat{H}_c = [h^c_1, ..., h^c_{N_c}] \in \mathbb{R}^{N_c \times d}$, including the target word hidden states $\hat{H}_t = [h^t_1, ..., h^t_{N_t}] \in \mathbb{R}^{N_t \times d}$, where d denotes the dimension of the hidden state.

Position Weighted Graph Convolutional Network. Graph convolutional network (GCN) has been widely used to encode the syntax graph to integrate syntactic information into hidden states. In this work, based on standard GCN, we introduce

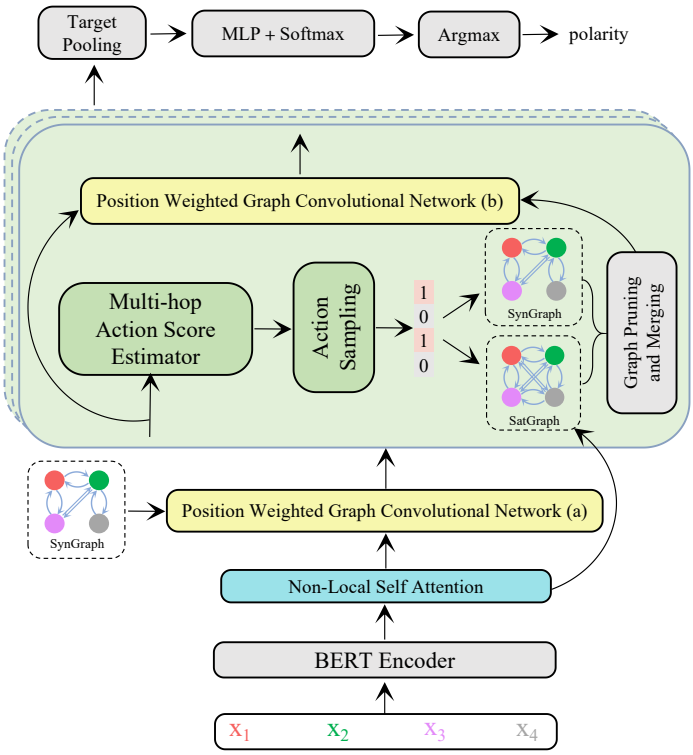


Figure 4.2: The architecture of Neural Subgraph Explorer. SynGraph is obtained from off-the-shelf dependency parser.

a weight for each word to indicate its position relative to the target, forming the position-weighted GCN. By this means, the potential relative words for the target can be highlighted. Specifically, the node updating process can be formulated as:

$$\begin{aligned} h_i^l &= \mathrm{ReLU}\left(\sum_{j=0}^{N} \frac{A_{ij}^{syn}(w_p^j W_g^l h_j^{l-1})}{d_i + 1} + b_g^l\right), \\ \mu_j &= \frac{j-\tau}{N+1}, \\ w_p^j &= 1 - |\mu_j|, \end{aligned} \tag{4.2}$$

where A^{syn} is the adjacent matrix derived from the dependency parsing result and $A_{ij}^0 = 1$ if there is a dependency from node j to i; μ_j is the relative offset between j-th word and the target and w_p^j is the position weight of j-th word; d_i denotes the degree of $i-$th node; W_g^l and b_g^l are parameters. If the target is a phrase, $t-\tau$ is calculated with its left or right boundary index according to which side the word locates.

Now we obtain the syntax-enhanced context hidden states H_c and target hidden states H_t. And by applying mean pooling over H_t, we obtain the initial target representation r_t.

4.2.2 Target-oriented Syntax Graph Pruning

The process of target-oriented syntax graph pruning includes four steps: (1) the multi-hop action score estimator evaluates the value of each word regarding the target, producing the degree that whether the word should be pruned or reserved; (2) the Gumble-Softmax is leveraged for differentiable discrete action sampling and generate the action sequence; (3) the action sequence is used to mask the adjacent matrix of the syntax graph and self-attention graph; (4) the position weighted GCN is adopted for message passing on the obtained graph, updating the hidden states. Next we present the details of each module following the above order.

4.2.2.1 Multi-hop Action Score Estimator

To decide whether a node on the graph should be pruned, each word is supposed to be assigned a score to represent its contribution for expressing the sentiment semantics of the given target. To this end, inspired from [5], we design a multi-hop action score estimator. At each hop, it produces a gate score for each word and a summarized target-centric context vector which is used at next hop. The details are given bellow:

$$\begin{aligned} s_j^{t,t'} &= W_s \left[w_p^j h_j^{t,t'}, \mu_j, C_{t-1}, r_t \right] + b_s, \\ g_j^{t,t'} &= \text{Sigmoid}(s_j^{t,t'}), \\ \alpha_j^t &= \frac{\exp\left(s_j^{t,t'}\right)}{\sum_k^{N_c} \exp\left(s_k^{t,t'}\right)}, \\ I^{t,t'} &= \sum_j^{N_c} \alpha_j^t w_j h_j^{t,t'}, \\ C_t &= \text{GRU}(I^{t,t'}, C_{t-1}), \end{aligned} \tag{4.3}$$

where W_s and b_s are parameters; GRU denotes gated recurrent unit. C_0 is initialized as a zero vector.

Then the output gate $g_j^{t,T'}$ of final step T' can derive

$$p_{j,0}^t = 1 - g^{t,T'},\ p_{j,1}^t = g_j^{t,T'}, \tag{4.4}$$

which represent the possibilities that node j should be pruned or reserved, respectively.

4.2.2.2 Action Sampling

Now we have a problem of discrete action selecting. Although REINFFORCE algorithm [86] are commonly used for this problem, it causes model instability and hard training.

To this end, we leverage the Gumbel-Softmax [27] trick for differentiable action sampling:

$$act_j^t = \frac{\exp\left(\left(\log\left(p_{j,1}^t\right) + \epsilon_1\right)/\pi\right)}{\sum_{a=0}^{1}\exp\left(\left(\log\left(p_{i,a}^t\right) + \epsilon_t\right)/\pi\right)}, \tag{4.5}$$

where ϵ_t is randomly sampled from Gumbel distributionand π is the temperature coefficient which is set 0.1 in this work. $act_j^t = 0$ denotes pruning while $act_j^t = 1$ denotes reserving.

4.2.2.3 Graph Pruning and Merging

Given the set of actions $\{act_j^t\}_{i=1}^{N_c}$ corresponding to the context word, we use it to mask A^{syn}:

$$A_{ij}^{syn,t} = act_j^t \cdot A_{ij}^{syn}, \tag{4.6}$$

where $A^{syn,t}$ is the adjacent matrix of the pruned syntax graph of the t-th layer.

Now in the pruned syntax graph, some target-irrelevant nodes (words) are removed. However, the issue of *loss of distant correlations* is not tackled. Besides, the pruning operation on the whole syntax graph may lead to a potential problem that the pruned syntax graph may include isolated nodes, which hinders the message passing on the obtained graph.

To solve the above two issues, we propose to exploit the self-attention graph (SatGraph), which is a fully-connected semantic graph consisting of all words in the sentence and derived by the self-attention [77]. SatGraph can provide the first-order connections between each two nodes. Therefore, merging the pruned SatGraph with the pruned SynGraph can not only guarantee the connectivity but also can directly connect the target with its related words, promoting the aggregation of target-related information, which is beneficial for prediction.

An intuitive way to obtain the self-attention graph is to retrieve the self-attention matrix at the last layer of BERT [9]. However, considering the self-attentions in $\text{BERT}_{\text{BASE}}$ ($\text{BERT}_{\text{LARGE}}$) are 12-head (16-head), and the word representations are segmented into 12 (16) *local* subspaces, one of the 12 (16) self-attention matrices can only reflect the *local* word correlations in the *local* subspace rather than general global semantic space in which the subsequent modules work. Therefore, due to the segmentation bias between the multi-head *local* subspaces and the general *global* semantic space, the self-attention matrix derived by BERT cannot be used for the self-attention graph.

To this end, we add a non-local self-attention layer between BERT and position weighted GCN, as shown in Fig. 4.2, to obtain the *global* self-attention matrix

$A^{sat} \in \mathbb{R}^{N_c \times N_c}$ representing the correlations between words in the general semantics space which is consistent with subsequent modules. Specifically, A^{sat} is obtained as follows:

$$A^{sat} = \text{Softmax}\left((\hat{H}_c M_q)(\hat{H}_c M_k)^\intercal / \sqrt{d}\right), \tag{4.7}$$

Then the updated hidden states are obtained by:

$$\widetilde{H_c} = A^{sat} \hat{H}_c M_v, \tag{4.8}$$

where $M_q, M_k, M_v \in \mathbb{R}^{d \times d}$ are parameters. And then $\widetilde{H_c}$ is fed to the position weighted GCN before the target-oriented syntax graph pruning layer.

Then we prune SatGraph in the same way as SynGraph and merge them:

$$\begin{aligned} A_{ij}^{sat,t} &= act_j^t \cdot A_{ij}^{sat}, \\ A^t &= (A^{syn,t} + A^{sat})/2. \end{aligned} \tag{4.9}$$

In this process, not only the beneficial first-order connections are introduced, but also the noisy first-order connections are removed duo to pruning operation on SatGraph. Then A^t will be used for messaging passing later.

4.2.2.4 Node Updating

For message passing on the obtained graph, we apply another position weighted GCN (noted as (b) in Fig. 4.2) over A^t.

4.2.2.5 Multi-layer Stacking

In order to let our model gradually improve the graph pruning and learn deep features, we stack the target-oriented syntax graph pruning module in a multi-layer manner.

4.2.3 Prediction and Training

After T layers of target-oriented syntax graph pruning, we obtain the final representations of each context word, which includes N_t target words. We apply mean pooling over all target words to generate the final target representation, getting the final target representation R_t

Then we fed R_t to a multi-layer perception (MLP) and then $softmax$ for classification:

$$P = \text{softmax}\left(W_c^1(W_c^2 R_a + b_c^2) + b_c^1\right). \tag{4.10}$$

Finally, by applying $arg\ max$ on the class vector P, we can get the produced sentiment polarity of the target in the review.

Given D training samples, the training objective is:

$$\ell = -\sum_{i=1}^{D}\sum_{c\in\mathcal{C}} I(y=c)\log(P(y=c)), \tag{4.11}$$

where y is the ground-truth class, I is an indicator function, and $\mathcal{C}$ denotes the sentiment polarity class set.

4.3 Experiment

4.3.1 Experimental Setup

4.3.1.1 Dataset

We conduct experiments on three public benchmark datasets to obtain reliable and authoritative results. Restaurant14 and Laptop14 are from [51], and Restaurant15 is from [50]. We pre-process the datasets following the same way as previous works [72, 79, 105]. The statistics of all datasets are shown in Table 4.1.

Dataset	Positive		Neutral		Negative	
	Train	Test	Train	Test	Train	Test
Laptop14	994	341	464	169	870	128
Restaurant14	2164	728	637	196	807	196
Restaurant15	912	326	36	34	256	182

Table 4.1: Dataset statistics of the three datasets.

4.3.1.2 Implementation Details

We adopt the $\text{BERT}_{\text{BASE}}$ uncased version as the BERT encoder and it is fine-tuned in the experiments. We train our Neural Subgraph Hunter using AdamW optimizer. The dependency parser used in our experiments is from spaCy toolkit[1]. In our experiments, the dimension of hidden units are 768. The dropout rate for BERT encoder is 0.1, while the dropout rate for other modules is 0.3. The batch size is 16 and epoch number is 30. The learning rates are 1e-5, 5e-5, 3e-5 for Lap14, Res14 and Res15 datasets respectively. The weight decay rages are 0.05 for Res14 and Res15 datasets while 0.001 for Lap14 datasets. The layer number of the position weighted GCN (a) and (b)are both 2. The hop number of the multi-hop action score estimator is 3. The layer number of target-oriented syntax graph pruning is 2.

[1]https://spacy.io/.

Accuracy (Acc) and Macro-F1 (F1) are used as evaluation metrics. Since there is no official validation set for the datasets, we report the average results over three random runs.

4.3.1.3 Baselines for Comparison

The baselines can be divided into four categories regarding whether BERT and syntax are leveraged:
(A) BERT × Syntax ×: 1. IAN [45] separately encodes the target and context, then models their interactions through an interactive attention mechanism. 2. RAM [5] uses a GRU attention mechanism to recurrently extracts the target-related semantics.
(B) BERT × Syntax ✓: 3. ASGCN [105] utilizes GCN to leverage syntactic information. 4. BiGCN [108] employs GCN to convolute over hierarchical syntactic and lexical graphs.
(C) BERT ✓ Syntax ×: 5. BERT-SPC [9] takes the concatenated context-target pair as input and uses the output hidden state of [CLS] token for classification. 6. AEN-BERT [67] employs multiple attention layers to learn target-context interactions.
(D) BERT ✓ Syntax ✓: 7. ASGCN+BERT [105]. Since the backbone of our Neural Subgraph Explore is BERT+GCN, we augment the ASGCN model with BERT encoder to form a baseline. 8. KGCapsAN-BERT [104] utilizes multi-prior knowledge to guide the capsule attention process and use a GCN-based syntactic layer to integrate the syntactic knowledge. 9. R-GAT+BERT [79] uses the relational graph attention network to aggregate the global relational information from all context words into the target node representation. 10. DGEDT-BERT [70] employs a dual-transformer network to model the interactions between the flat textual knowledge and dependency graph empowered knowledge. 11. A-KVMN+BERT [73] uses a key-value memory network to leverage not only word-word relations but also their dependency types. 12. BERT+T-GCN [72] leverages the dependency types in T-GCN and uses an attentive layer ensemble to learn the comprehensive representation from different T-GCN layers. 13. DualGCN+BERT [38] uses orthogonal and differential regularizers to model the interactions between semantics and syntax.

Note that all of the BERT encoders in baselines are BERT-base uncased version, the same as ours. And for fair comparison, we reproduce the average results of R-GAT+BERT, A-KVMN+BERT, BERT+T-GCN and DualGCN+BERT on three random runs because they report the best results rather than average results in their original paper.

4.3.2 Main Results

Model	Laptop14		Restaurant14		Restaurant15	
	Acc (%)	F1 (%)	Acc (%)	F1 (%)	Acc (%)	F1 (%)
IAN‡ [45]	72.05	67.38	79.26	70.09	78.54	52.65
RAM‡ [5]	74.49	71.35	80.23	70.80	79.30	60.49
ASGCN [105]	75.55	71.05	80.77	72.02	79.89	61.89
BiGCN [108]	74.59	71.84	81.97	73.48	81.16	64.79
BERT-SPC [9]	78.47	73.67	84.94	78.00	83.40	65.00
AEN-BERT [67]	79.93	76.31	83.12	73.76	-	-
ASGCN+BERT† [105]	78.92	74.35	85.87	79.32	83.85	68.73
KGCapsAN-BERT [104]	79.47	76.61	85.36	79.00	-	-
R-GAT+BERT† [79]	79.31	75.40	86.10	80.04	83.95	69.47
DGEDT-BERT [70]	79.8	75.6	86.3	80.0	84.0	71.0
A-KVMN+BERT† [73]	79.20	75.76	85.89	78.29	83.89	67.88
BERT+T-GCN† [72]	80.56	76.95	85.95	79.40	<u>84.81</u>	71.09
DualGCN+BERT† [38]	<u>80.83</u>	<u>77.35</u>	<u>86.64</u>	<u>80.76</u>	84.69	<u>71.58</u>
Neural Subgraph Explorer (ours)	**82.13**	**78.51**	**87.35**	**82.04**	**86.29**	**74.43**

Table 4.2: Performances comparison (in %). † indicates we reproduce the results using the official source code, and ‡ denotes that the results are retrieved from [Zhang *et al.*, 2019]. Our Neural Subgraph Explorer outperforms previous SOTA models and corresponding baselines on all datasets, being statistically significant ($p < 0.05$ under t-test.)

The performances of our Neural Subgraph Explorer and baselines are shown in Table 4.2. We can observe that BERT can significantly boost the performance while integrating syntactic information into BERT-based models can bring further significant improvement. And the best-performing models are all based on BERT+Syntax paradigm.

However, state-of-the-art models neglect the problem that not all words are useful for expressing the sentiment of the specific target, and they just adopt GNNs to conduct message passing on the whole syntax graph. As a result, the noisy information from the target-irrelevant words is aggregated into the target's and its related words' node representation, which affects the prediction of the specific target's sentiment. To overcome this problem, we propose a Neural Subgraph Explorer model which can adaptively and dynamically prune the noisy nodes corresponding to the target-irrelevant words. As shown in Table 4.2, our model achieves new state-of-the-art performance, obtaining consistent improvements over all baselines in terms of both Acc and F1. Specifically, our model surpasses previous best scores by 1.30%, 0.71%, and 1.16% in terms of Acc on Lap14, Res14, and Res15 datasets, respectively. And in terms of F1, our model surpasses previous best scores by 1.16%, 1.28%, and 3.44% on

Lap14, Res14, and Res15 datasets, respectively. All the improvements are attributed to the discarding of noisy information via pruning noisy nodes on the vanilla syntax graph and introducing first-order dependency to facilitate the aggregation of distant while crucial target-related information.

4.3.3 Ablation Study

Variants	Laptop14		Restaurant14		Restaurant15	
	Acc	F1	Acc	F1	Acc	F1
Full Model	**82.13**	**78.51**	**87.35**	**82.04**	**86.29**	**74.43**
NoPrune	79.52	75.70	86.04	79.77	84.50	70.83
	↓2.61	↓2.81	↓1.31	↓2.27	↓1.78	↓3.60
RandPrune	80.67	76.97	86.49	80.44	84.75	72.10
	↓1.46	↓1.54	↓ 0.86	↓1.60	↓1.54	↓2.33
NoMerge	81.61	78.25	87.05	81.64	85.79	73.29
	↓0.52	↓0.26	↓ 0.30	↓0.40	↓0.50	↓1.14

Table 4.3: Results of ablation experiments.

We conduct ablation experiments to look into Neural Subgraph Explorer and investigate how it works well. The experiment results are shown in Table 4.3.

We first study the effect of target-oriented syntax graph pruning via removing the pruning operation. In practice, we achieve this by setting all act_i^t as 1 and this variant is termed NoPrune. From Table 4.3 we can observe that NoPrune obtains much poorer results compared with the full model. The reason is that without noisy node pruning, the main advantage of Neural Subgraph Explorer is lost and the noisy information conveyed in the target-irrelevant nodes harms the prediction.

Then we study the effect of multi-hop action score estimator, which determines whether a word should be pruned. We design a variant named RandPrune, which assigns a random value to each $g_i^{t,T'}$. Therefore, the nodes in the syntax graphs are randomly pruned. From Table 4.3 we can find that RandPrune is significantly inferior to the full model. This is because without the multi-hop action score estimator, RandPrune cannot identify the crucial words that contribute to the expression of the target's sentiment, and the words it prunes may be the critical words and it may reserve the noisy words that harm the prediction.

Finally, we investigate the effect of graph merging via designing a variant termed NoMerge, in which the pruned SynGraph is directly used for message passing. From Table 4.3 we can find that NoMerge has a considerable results drop compared with the full model. This proves the effectiveness of merging the pruned SynGraph and pruned

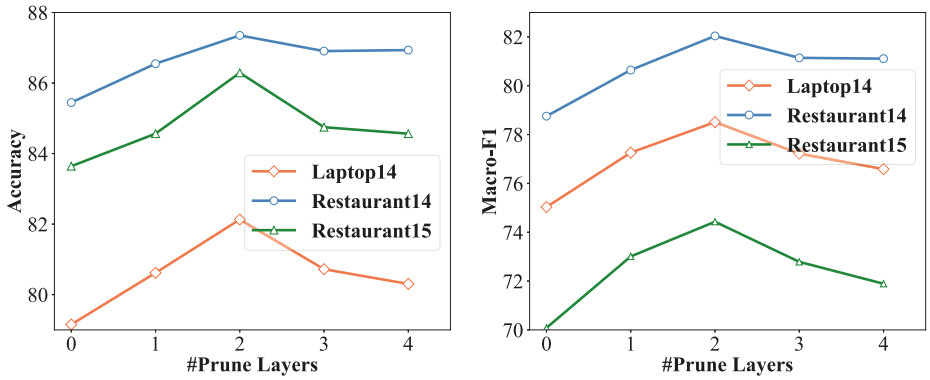


Figure 4.3: The impact of the number of pruning layers.

SatGraph, which can introduce first-order connections to facilitate the information aggregation of distant target-related words and guarantee the connectivity of the obtained graph.

4.3.4 Investigation of Layer Number

4.3.4.1 #Target-oriented Syntax Graph Pruning

To investigate the impact of layer number of target-oriented syntax graph pruning, we vary the value of T from 0 to 4 and illustrate the corresponding Accuracy and Macro-F1 in Fig. 4.3. $T = 0$ denotes there is no target-oriented syntax graph pruning. As a result, the worst performances are obtained. With T increasing from 0 to 2, the performances increase then reach the peaks. However, continuously increasing T results in the drop of results. There are two reasons. The first one is that too large T makes it more difficult to train the model due to a large number of parameters. The other one is that since there is a linear relationship between T and the total GCN layer number of Neural Subgraph Explorer, too large T causes the over-smoothing problem due to too much node aggregation, losing the specific and important features.

4.3.4.2 #Position Weighted GCN

For each position weighted GCN in our model, the layer number T' denotes that after the T'-layer GCN, the information of node i can be aggregated into node j if node i is node j's t'th-order neighbor and $t' < T'$. When $T' = 1$, only a node's directly adjacent nodes can transfer information with it, which is intuitively insufficient for capturing the sentiment features for the target. And as shown in Fig. 4.4, $T' = 1$ results in the worst performances. However, too large T' also has a negative impact on performances because too many GCN layers bring the over-fitting and over-smoothing

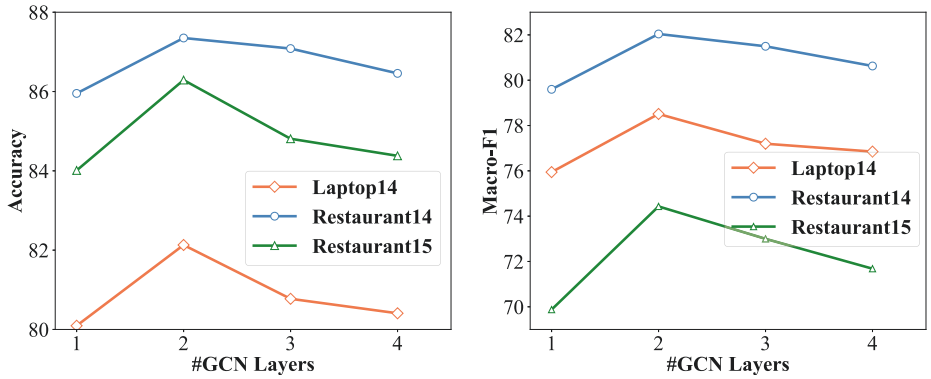


Figure 4.4: The impact of layer number of position weighted GCN.

problem. Finally, as illustrated in Fig. 4.4, the 2-layer position weighted GCN obtains the best results for our Neural Subgraph Explorer.

4.4 Summary

This paper proposes a novel Neural Subgraph Explorer model to tackle the target sentiment classification task. On the one hand, it can discard the noisy information contained in the noisy words regarding the given target through the stacked target-oriented syntax graph pruning module. On the other hand, it introduces first-order connections between the target and the crucial words via merging the pruned self-attention graph with the pruned syntax graph. In this way, more useless information can be removed and more crucial information can be captured. Finally, experiments are conducted on benchmark datasets and the effectiveness of our model has been proven.

Chapter 5

DARER

5.1 Introduction

Dialog language understanding [103] is the fundamental component of the dialogue system. It includes several individual tasks, e.g. dialog sentiment classification, dialog act recognition, slot filling, and (multiple) intent detection. In recent years, as researchers discover the inherent correlations among some specific task-pair, the joint task which tackles two tasks simultaneously has attracted increasing attention. For example, dialog sentiment classification (DSC) and dialog act recognition (DAR) are two challenging tasks in dialog systems [14], while the task of joint DSC and DAR aims to simultaneously predict the sentiment label and act label for each utterance in a dialog [3, 53]. An example is shown in Table 5.1. To predict the sentiment of u_b, besides its *semantics*, its Disagreement act *label* and the Positive sentiment *label* of its *previous* utterance (u_a) can provide useful references, which contribute a lot when humans do this task. This is because the Disagreement act label of u_b denotes it has the opposite opinion with u_a, and thus u_b tends to have a Negative sentiment label, the opposite one with u_a (Positive). Similarly, the opposite sentiment labels of u_b and u_a are helpful to infer the Disagreement act label of u_b. In this paper, we term this process as dual-task reasoning, where there are three key factors: 1) the semantics of u_a and u_b; 2) the temporal relation between u_a and u_b; 3) u_a's and u_b's labels for another task.

In previous works, different models are proposed to model the correlations between DSC and DAR. [3] propose a multi-task model in which the two tasks share a single encoder. [33, 37, 53, 55] try to model the semantics-level interactions of the two tasks. The framework of previous models is shown in Fig. 5.1 (a). For dialog understanding, Co-GAT [55] applies graph attention network (GAT) [78] over an undirected disconnected graph which consists of isolated speaker-specific full-connected

Table 5.1: A dialog snippet from the Mastodon [3] dataset.

Utterances	Act	Sentiment
u_a: I highly recommend it. Really awesome progression and added difficulty	Statement	Positive
u_b: I never have.	Disagreement	Negative

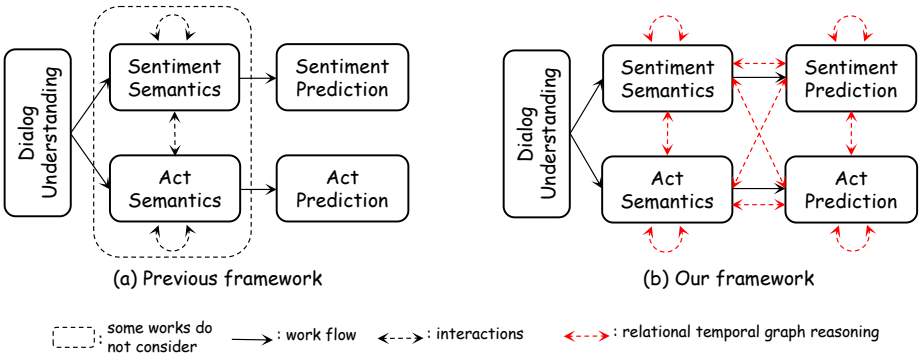


Figure 5.1: Illustration of previous framework and ours.

subgraphs. Therefore, it suffers from the issue that the inter-speaker interactions cannot be modeled, and the temporal relations between utterances are omitted. For dual-task reasoning, on the one hand, previous works only consider the parameter sharing and semantics-level interactions, while the label information is not explicitly integrated into the dual-task interactions. Consequently, the explicit dependencies between the two tasks cannot be captured and previous dual-task reasoning processes are inconsistent with human intuition, which leverages the label information as crucial clues. On the other hand, previous works do not consider the temporal relations between utterances in dual-task reasoning, in which they play a key role.

In this paper, we try to address the above issues by introducing temporal relations and leveraging label information. To introduce temporal relations, we design a **s**peaker-**a**ware **t**emporal **g**raph (SATG) for dialog understanding, and a **d**ual-task **r**easoning **t**emporal **g**raph (DRTG) for dual-task relational reasoning. Intuitively, different speakers' semantic states will change as the dialog goes, and these semantic state transitions trigger different sentiments and acts. SATG is designed to model the speaker-aware semantic states transitions, which provide essential indicative semantics for both tasks. In SATG, there is one group of utterance nodes and two kinds of temporal relations: previous and future. Since the temporal relation is a key factor in dual-task reasoning, DRTG is designed to integrate inner- and inter-task temporal relations, making the dual-task reasoning process more rational and effective. In

SATG, there are two parallel groups of utterance nodes and three kinds of temporal relations: previous, future, and equal.

To leverage label information, we propose a new framework, as shown in Fig. 5.1 (b). Except for semantics-level interactions, it integrates several kinds of prediction-level interactions. First, self-interactions of sentiment predictions and act predictions. In both tasks, there are prediction-level correlations among the utterances in a dialog. In the DSC task, the sentiment state of each speaker tends to be stable until the utterances from others trigger the changes [82]. In the DAR task, there are different patterns (e.g., Questions-Inform and Directives-Commissives) reflecting the interactions between act labels [39]. Second, interactions between the predictions and semantics. Intuitively, the predictions can offer feedback to semantics, which can rethink and then reversely help revise the predictions. Third, prediction-prediction interactions between DSC and DAR, which model the explicit dependencies. However, since our objective is to predict the labels of both tasks, there is no ground-truth label available for prediction-level interactions. To this end, we design a recurrent dual-task reasoning mechanism that leverages the label distributions estimated in the previous step as prediction clues of the current step for producing new predictions. In this way, the label distributions of both tasks are gradually improved along the step. To implement our framework, we propose **D**ual-t**A**sk temporal **R**elational r**E**current **R**easoning Network[1] (DARER) [93], which includes three main components. The *Dialog Understanding* module conducts relation-specific graph transformations (RSGT) over SATG to generate context-, speaker- and temporal-sensitive utterance representations. The *Initial Estimation* module produces the initial label information which is fed to the *Recurrent Dual-task Reasoning* module, in which RSGT operates on DRTG to conduct dual-task relational reasoning. And the RSGTs are achieved by relational graph convolutional networks [62]. Moreover, we design logic-heuristic training objectives to force DSC and DAR to gradually prompt each other in the recurrent dual-task reasoning process.

Then we further propose Relational Temporal Transformer (ReTeFormer) and DARER^2. The main difference between DARER and DARER^2 is that in DARER^2 the original RGCNs applied over SATG and DRTG are replaced with our proposed SAT-ReTeFormer and DTR-ReTeFormer. The core of ReTeFormer is the Relation- and Structure-Aware Disentangled Multi-head Attention, which can achieve fine-grained relational temporal modeling. Generally, DARER^2 has three distinguished advantages over DARER: (1) ReTeFormer integrates dialog structural information, achieving more

[1]The content of DARER was presented as a poster in ACL 2022 conference.

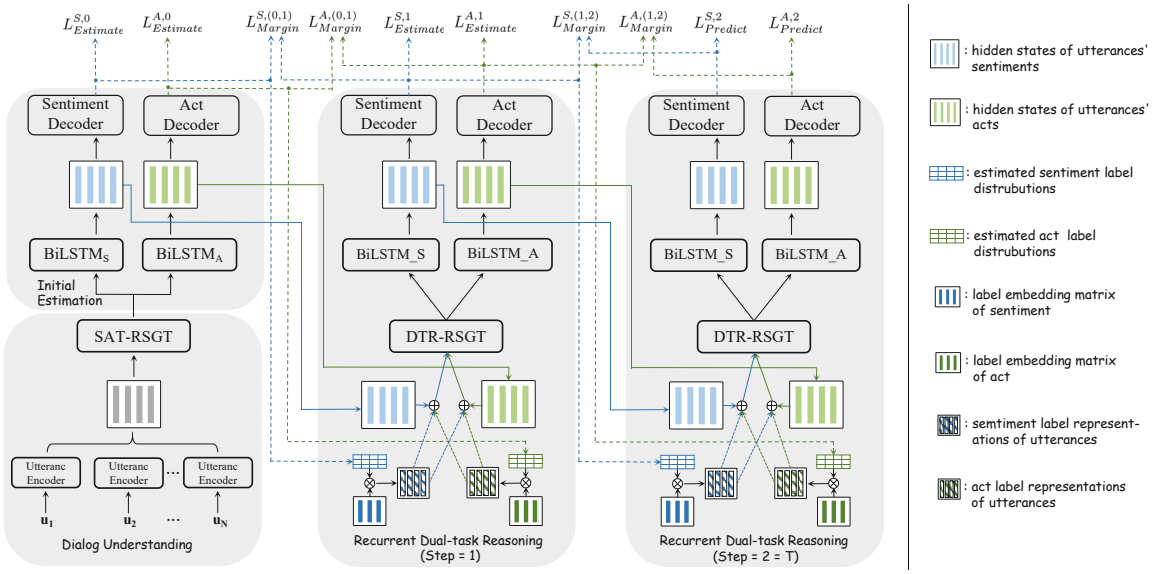


Figure 5.2: The overall network architecture of DARER and DARER^2. In DARER, SAT-RSGT and DTR-RSGT are achieved by RGCNs, while in DARER^2, they are achieved by SAT-ReTeFormer and DTR-ReTeFormer, respectively. Without loss of generality, the step number T in this illustration is set 2.

comprehensive and fine-grained relational temporal graph reasoning; (2) the relational temporal attention mechanism of ReTeFormer can comprehensively and explicitly model the correlations among dual tasks semantics and predictions; (3) the relation specific attention maps derived by ReTeFormer can provide explainable evidence of relational temporal graph reasoning, making the model more reliable.

In summary, this work has three major contributions. (1) We propose DARER, which is based on a new framework that for the first time achieves relational temporal graph reasoning and prediction-level interactions. (2) Our proposed DARER^2 further improves the relational temporal graph reasoning with our proposed ReTeFormer which is based on the Relation- and Structure-Aware Disentangled Multi-head Attention. (3) Experiments prove that DARER and DARER^2 significantly outperform the state-of-the-art models in different scenarios of dual-task dialog language understanding.

5.2 Overall Model Architecture

Given a dialog consisting of N utterances: $\mathcal{D} = (u_1, u_2, ..., u_N)$, our objective is to predict both the dialog sentiment labels $Y^S = y_1^s, ..., y_N^s$ and the dialog act labels $Y^A = y_1^a, ..., y_N^a$ in a single run.

The overall network architecture shared by DARER and DARER^2 is shown in Fig. 5.2. It consists of three modules, whose details are introduced in this section.

5.2.1 Dialog Understanding

5.2.1.1 Utterance Encoding

In previous works, BiLSTM [21, 91] is widely adopted as the utterance encoder to generate the initial utterance representation: $H = (h_0, ..., h_N)$. In this paper, besides BiLSTM, we also study the effect of different pre-trained language model (PTLM) encoders in Sec. 5.5.3.3.

BiLSTM: We apply the BiLSTM over the word embeddings of u_t to capture the inner-sentence dependencies and temporal relationships among the words, producing a series of hidden states $H_{u,i} = (h^0_{u,i}, ..., h^{l_i}_{u,i})$, where l_i is the length of u_i. Then we feed $H_{u,i}$ into a max-pooling layer to get the representation for each u_i.

PTLM: We separately feed each utterance into the PTLM encoder and take the output hidden state of the `[CLS]` token as the utterance representation.

5.2.1.2 Speaker-aware Temporal RSGT

To capture the inter- and intra-speaker semantic interactions and the speaker-aware temporal dependencies between utterances, we conduct Speaker-aware Temporal relation-specific graph transformations (SAT-RSGT). Now we obtain the context-, speaker- and temporal-sensitive utterance representations: $\hat{H} = (\hat{h_0}, ..., \hat{h_N})$.

5.2.2 Initial Estimation

To obtain task-specific utterances representations, we separately apply two BiLSTMs over $\hat{H}$ to obtain the utterance hidden states for sentiments and acts respectively: $H^0_s = \text{BiLSTM}_\text{S}(\hat{H})$, $H^0_a = \text{BiLSTM}_\text{A}(\hat{H})$, where $H^0_s = \{h^0_{s,i}\}^N_{i=1}$ and $H^0_a = \{h^0_{a,i}\}^N_{i=1}$. Then H^0_s and H^0_a are separately fed into Sentiment Decoder and Act Decoder to produce the initial estimated label distributions:

$$
\begin{aligned}
P^0_S &= \{P^0_{S,i}\}^N_{i=1} \; P^0_A = \{P^0_{A,i}\}^N_{i=1}, \\
P^0_{S,i} &= softmax(W^s_d h^0_{a,i} + b^s_d) \\
&= \left[p^0_{s,i}[0], ..., p^0_{s,i}[k], ..., p^0_{s,i}(|\mathcal{C}_s| - 1)\right], \\
P^0_{A,i} &= softmax(W^a_d h^0_{s,i} + b^a_d) \\
&= \left[p^0_{a,i}[0], ..., p^0_{a,i}[k], ..., p^0_{a,i}(|\mathcal{C}_a| - 1)\right],
\end{aligned}
\tag{5.1}
$$

where W_d^* and b_d^* are weight matrices and biases, $\mathcal{C}_s$ and $\mathcal{C}_a$ are sentiment class set and act class set.

5.2.3 Recurrent Dual-task Reasoning

At step t, the recurrent dual-task reasoning module takes two streams of inputs: 1) hidden states $H_s^{t-1} \in \mathbb{R}^{N\times d}$ and $H_a^{t-1} \in \mathbb{R}^{N\times d}$; 2) label distributions $P_S^{t-1} \in \mathbb{R}^{N\times|\mathcal{C}_s|}$ and $P_A^{t-1} \in \mathbb{R}^{N\times|\mathcal{C}_a|}$.

5.2.3.1 Projection of Label Distribution

To achieve the prediction-level interactions, we should represent the label information in vector form to let it participate in calculations. We use P_S^{t-1} and P_A^{t-1} to respectively multiply the sentiment label embedding matrix $M_s^e \in \mathbb{R}^{|\mathcal{C}_s|\times d}$ and the act label embedding matrix $M_a^e \in \mathbb{R}^{|\mathcal{C}_a|\times d}$, obtaining the sentiment label representations $E_S^t = \{e_{s,i}^t\}_{i=1}^N$ and act label representations $E_A^t = \{e_{a,i}^t\}_{i=1}^N$. In particular, for each utterance, its sentiment label representation and act label representation are computed as:

$$\begin{aligned} e_{s,i}^t &= \sum_{k=0}^{|\mathcal{C}_s|-1} p_{s,i}^{t-1}[k] \cdot v_s^k, \\ e_{a,i}^t &= \sum_{k'=0}^{|\mathcal{C}_a|-1} p_{a,i}^{t-1}[k'] \cdot v_a^{k'}, \end{aligned} \tag{5.2}$$

where v_s^k and $v_a^{k'}$ are the label embeddings of sentiment class k and act class k', respectively.

5.2.3.2 Dual-task Reasoning RSGT

To achieve the self- and mutual-interactions between the semantics and predictions, for each node in DRTG, we superimpose its corresponding utterance's label representations of both tasks on its hidden state:

$$\begin{aligned} \hat{h}_{s,i}^t &= h_{s,i}^{t-1} + e_{s,i}^t + e_{a,i}^t, \\ \hat{h}_{a,i}^t &= h_{a,i}^{t-1} + e_{s,i}^t + e_{a,i}^t. \end{aligned} \tag{5.3}$$

Thus the representation of each node contains the task-specific semantic features and both tasks' label information, which are then incorporated into the relational reasoning process to achieve semantics-level and prediction-level interactions.

The obtained $\hat{\mathbf{H}}_\mathbf{s}^\mathbf{t}$ and $\hat{\mathbf{H}}_\mathbf{a}^\mathbf{t}$ both have N vectors, respectively corresponding to the N sentiment nodes and N act nodes on DRTG. Then we feed them into the Dual-task

Reasoning relation-specific graph transformations (DTR-RSGT) conducted on DRTG. Now we get $\overline{H}_s^t$ and $\overline{H}_a^t$.

5.2.3.3 Label Decoding

For each task, we use a task-specific BiLSTM (TS-LSTM) to generate a new series of task-specific hidden states:

$$\begin{aligned} H_s^t &= \mathrm{BiLSTM_S}(\overline{H}_s^t), \\ H_a^t &= \mathrm{BiLSTM_A}(\overline{H}_a^t). \end{aligned} \tag{5.4}$$

Besides, as $\overline{H}_s^t$ and $\overline{H}_a^t$ both contain the label information of the two tasks, the two TS-LSTMs have another advantage of label-aware sequence reasoning, which has been proven can be achieved by LSTM [110].

Then H_S^t and H_A^t are separately fed to Sentiment Decoder and Act Decoder to produce P_S^t and P_A^t.

5.2.4 Training Objective

Intuitively, there are two important logic rules in our model. First, the produced label distributions should be good enough to provide useful label information for the next step. Otherwise, noisy label information would be introduced, misleading the dual-task reasoning. Second, both tasks are supposed to learn more and more beneficial knowledge from each other in the recurrent dual-task reasoning process. Scilicet the estimated label distributions should be gradually improved along steps. In order to force our model to obey these two rules, we propose a constraint loss $L_{Constraint}$ that includes two terms: $L_{Estimate}$ and L_{Margin}, which correspond to the two rules, respectively.

Estimate Loss $L_{Estimate}$ is the cross-entropy loss forcing model to provide good enough label distributions for the next step. At step t, for DSC task, $\mathcal{L}_{Estimate}^{S,t}$ is defined as:

$$\mathcal{L}_{Estimate}^{S,t} = \sum_{i=1}^{N} \sum_{k=0}^{|\mathcal{C}_s|-1} y_i^s[k] \log\left(p_{s,i}^t[k]\right). \tag{5.5}$$

Margin Loss L_{Margin} works on the label distributions of two adjacent steps, and it promotes the two tasks gradually learning beneficial knowledge from each other via forcing DARER to produce better predictions at step t than step $t-1$. Besides, although the model can receive more information at step t than $t-1$, this information is imperfect because there are some incorrect predictions of the previous step. Therefore,

we use the margin loss to force the model to leverage the *beneficial* information to output better predictions. For DSC task, $\mathcal{L}_{Margin}^{S,(t,t-1)}$ is a margin loss defined as:

$$\mathcal{L}_{Margin}^{S,(t,t-1)} = \sum_{i=1}^{N} \sum_{k=0}^{|\mathcal{C}_s|-1} y_i^s[k] \max(0, p_{s,i}^{t-1}[k] - p_{s,i}^{t}[k]). \tag{5.6}$$

If the correct class's possibility at step t is worse than at step t-1, $mathcalL_{Margin}^{S,(t,t-1)} > 0$. Then the negative gradient further force the model to predict better at step t. Otherwise, the correct class's possibility at step t is better than or equal to the one at step t-1. In this case, $mathcalL_{Margin}^{S,(t,t-1)} = 0$.

Constraint loss $L_{Constraint}$ is the weighted sum of $L_{Estimate}$ and L_{Margin}, with a hyper-parameter γ balancing the two kinds of punishments. For DSC task, $\mathcal{L}_{Constraint}^{S}$ is defined as:

$$\mathcal{L}_{Constraint}^{S} = \sum_{t=0}^{T-1} \mathcal{L}_{Estimate}^{S,t} + \gamma_s * \sum_{t=1}^{T} \mathcal{L}_{Margin}^{S,(t,t-1)}. \tag{5.7}$$

Final Training Objective The total loss for DSC task ($\mathcal{L}^S$) is the sum of $\mathcal{L}_{Constraint}^{S}$ and $\mathcal{L}_{Prediction}^{S}$:

$$\mathcal{L}^S = \mathcal{L}_{Prediction}^{S} + \mathcal{L}_{Constraint}^{S}, \tag{5.8}$$

where $\mathcal{L}_{Prediction}^{S}$ is the cross-entropy loss of the produced label distributions at the final step T:

$$\mathcal{L}_{Prediction}^{S} = \sum_{i=1}^{N} \sum_{k=0}^{|\mathcal{C}_s|-1} y_{s,i} \log\left(p_{s,i}^{T}[k]\right). \tag{5.9}$$

The total loss of DAR task ($\mathcal{L}^A$) can be derivated similarly like eqs. (5.5) to (5.9).

The final training objective of our model is the sum of the total losses of the two tasks:

$$\mathcal{L} = \mathcal{L}^S + \mathcal{L}^A. \tag{5.10}$$

5.3 DARER

Based on the overall model introduced in Section 3, DARER achieves SAT-RSGT and DTR-RSGT via applying RGCNs on the speaker-aware temporal graph (SATG) and dual-task reasoning temporal graph (DRTG), respectively.

Table 5.2: All relation types in SATG (assume there are two speakers). $I_s(i)$ indicates the speaker node i is from. $pos(i,j)$ indicates the relative position of node i and j.

r_{ij}	1	2	3	4	5	6	7	8
$I_s(i)$	1	1	1	1	2	2	2	2
$I_s(j)$	1	1	2	2	1	1	2	2
$pos(i,j)$	$>$	$\leq$	$>$	$\leq$	$>$	$\leq$	$>$	$\leq$

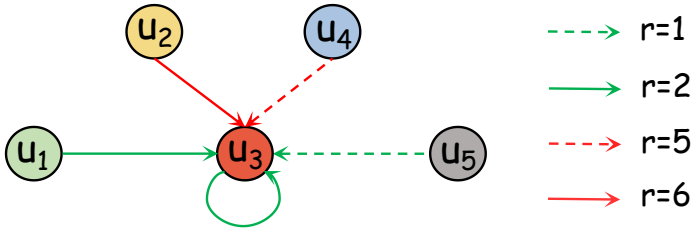


Figure 5.3: An example of SATG. u_1, u_3 and u_5 are from speaker 1 while u_2 and u_4 are from speaker 2. w.l.o.g, only the edges directed into u_3 node are illustrated.

5.3.1 SAT-RSGT

5.3.1.1 Speaker-aware Temporal Graph

We design a SATG to model the information aggregation between utterances in a dialog. Formally, SATG is a complete directed graph denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$. In this paper, the nodes in $\mathcal{G}$ are the utterances in the dialog, i.e., $|\mathcal{V}| = N, \mathcal{V} = (u_1, ..., u_N)$, and the edge $(i, j, r_{ij}) \in \mathcal{E}$ denotes the information aggregation from u_i to u_j under the relation $r_{ij} \in \mathcal{R}$. Table 5.2 lists the definitions of all relation types in $\mathcal{R}$. In particular, there are three kinds of information conveyed by r_{ij}: the speaker of u_i, the speaker of u_j, and the relative position of u_i and u_j. Naturally, the utterances in a dialog are chronologically ordered, so the relative position of two utterances denotes their temporal relation. An example of SATG is shown in Fig. 5.3. Compared with the previous dialog graph structure [53, 55], our SATG has two main advancements. First, as a complete directed graph, SATG can model both the intra- and inter-speaker semantic interactions. Second, incorporating temporal information, SATG can model the transitions of speaker-aware semantic states as the dialog goes on, which benefits both tasks.

5.3.1.2 SAT-RGCN

Inspired from [62], we apply SAT-RGCN over SATG to achieve the information aggregation:

$$\hat{h}_i = W_1 h_i^0 + \sum_{r \in \mathcal{R}} \sum_{j \in \mathcal{N}_i^r} \frac{1}{|N_i^r|} W_1^r h_j^0, \tag{5.11}$$

Table 5.3: All relation types in DRTG. $I_t(i)$ indicates that node i is a sentiment (S) node or act (A) node.

r'_{ij}	1	2	3	4	5	6	7	8	9	10	11	12
$I_t(i)$	S	S	S	S	S	S	A	A	A	A	A	A
$I_t(j)$	S	S	S	A	A	A	S	S	S	A	A	A
$pos(i,j)$	<	=	>	<	=	>	<	=	>	<	=	>

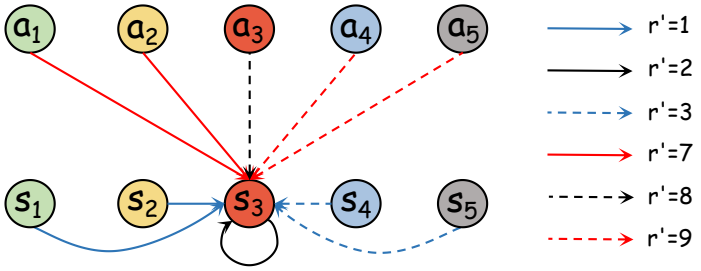


Figure 5.4: An example of DRTG. s_i and a_i respectively denote the node of DAC task and DAR task. w.l.o.g, only the edges directed into s_3 are illustrated.

where W_1 is self-transformation matrix and W_1^r is relation-specific matrix.

5.3.2 DTR-RSGT

5.3.2.1 Dual-task Reasoning Temporal Graph

Inspired by [94–96, 98, 99], we design a DRTG to provide an advanced platform for dual-task relational reasoning. It is also a complete directed graph that consists of $2N$ dual nodes: N sentiment nodes and N act nodes. The definitions of all relation types in $\mathcal{R}'$ are listed in Table 5.3. Intuitively, when predicting the label of a node, the information of its dual node plays a key role, so we emphasize the temporal relation of '=' rather than merge it with '<' like SATG. Specifically, the relation r'_{ij} conveys three kinds of information: the task of n_i, the task of n_j and the temporal relation between n_i and n_j. An example of DRTG is shown in Fig. 5.3. Compared with the previous dual-task graph structure [53, 55], our DRTG has two major advancements. First, the temporal relations in DRTG can make the DTR-RSGT capture the temporal information, which is essential for dual-task reasoning, while this cannot be achieved by the co-attention [53] or graph attention network [55] operating on their non-temporal graphs. Second, in DRTG, the information aggregated into a node is decomposed by different relations that correspond to individual contributions, rather than only depending on the semantic similarity measured by the attention mechanisms.

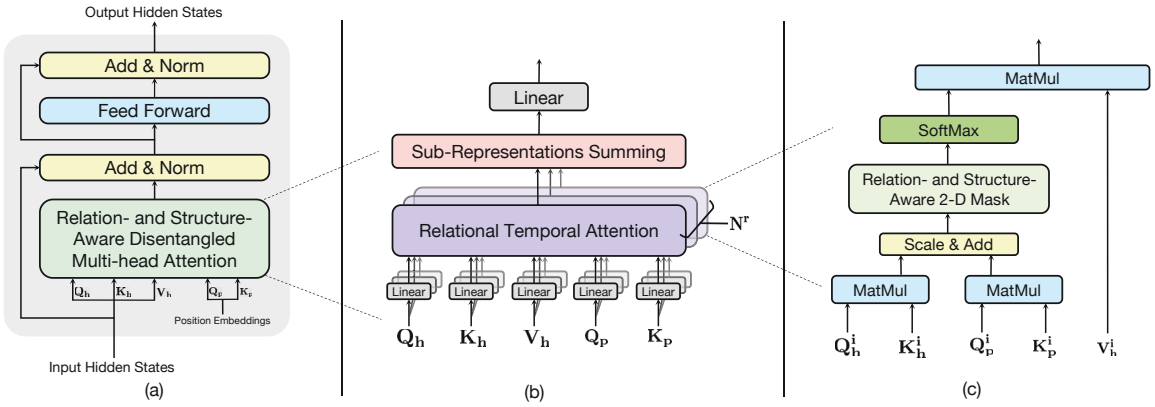


Figure 5.5: (a) Illustration of ReTeFormer. (b) Illustration of Relation- and Structure-Aware Disentangled Multi-head Attention. (c) Illustration of Relational Temporal Attention corresponding to the i-th relation. $\mathbf{Q_h}, \mathbf{K_h}$ and $\mathbf{V_h}$ denote the query matrix, key matrix and value matrix of input hidden states. $\mathbf{Q_p}$ and $\mathbf{K_p}$ denote the query matrix and key matrix of the absolute position embeddings. $\mathbf{N^r}$ denotes the number of relations.

5.3.2.2 DTR-RSGT

We apply DTR-RGCN to DRTG to achieve information aggregation. Specifically, the node updating process of DTR-RGCN can be formulated as:

$$\overline{h}_i^t = W_2 \hat{h}_i^t + \sum_{r \in \mathcal{R}'} \sum_{j \in \mathcal{N}_i^{r'}} \frac{1}{\left|N_i^{r'}\right|} W_2^r \hat{h}_j^t, \tag{5.12}$$

where W_2 is self-transformation matrix and W_2^r is relation-specific matrix.

5.4 DARER^2

Based on the overall model introduced in Section 3, DARER^2 achieves SAT-RSGT and DTR-RSGT via applying our proposed ReTeFormers on the speaker-aware temporal graph (SATG) and dual-task reasoning temporal graph (DRTG), respectively. Next, we first introduce the details of ReTeFormer, then the SAT-RSGT and DTR-RSGT of DARER^2.

5.4.1 Relational Temporal Transformer

The architecture of ReTeFormer is shown in Fig. 5.5. The core of ReTeFormer is the Relation- and Structure- Aware Disentangled Multi-head Attention, which can handle the Relation Modeling and Temporal Modeling simultaneously.

Relational Modeling The relational graph can be disentangled into different views, each of which corresponds to a specific relation and has its own adjacency matrix. In ReTeFormer, each head of Relational Temporal Attention corresponds to a specific relation and has its own parameterization, so as to achieve the relation-specific information aggregation. To make sure that the information aggregation of each relation is along the relation-specific structure, we design the Relation- and Structure-Aware 2-D Mask which uses the relation-specific adjacency matrix to mask the correlation matrix. Since the final node representation receives information along multiple relations, we design the Dynamic 1-D Mask and Merge module to extract and sum each node's sub-representations obtained from multi-head Relational Temporal Attentions.

Temporal Modeling A dialog can be regarded as a temporal sequence of utterances. ReTeFormer utilizes position embedding to achieve temporal modeling. The position embedding is one of the foundations of Transformer [77], which adds the position embedding to the input representation. However, recently it has been proven that adding together the position embeddings and word embeddings at input harms the attention and further limit the model's expressiveness because this operation brings mixed correlations between the two heterogeneous information resources (semantics and position) and unnecessary randomness in the attention [31]. To this end, Ke et al. (2021) [31] propose to model word contextual correlation and positional correlation separately with different parameterizations and then add them together. And our ReTeFormer follows this manner.

Next, we introduce the details of Relation- and Structure- Aware Disentangled Multi-head Attention, which is the core of our ReTeFormer[2].

5.4.1.1 Relation-Specific Scaled Dot-Product Attention

For the head corresponding to relation r, the correlation score $\hat{\alpha}^r_{ij}$ between every two nodes is obtained via relational modeling and temporal modeling:

$$\begin{aligned} \hat{\alpha}^r_{ij} &= \frac{1}{\sqrt{d}}(Q^r_{[h,i]})(K^r_{[h,j]})^T + \frac{1}{\sqrt{d}}(Q^r_{[p,i]})(K^r_{[p,j]})^T, \\ Q^r_{[h,i]} &= h_i W^r_Q, \quad K^r_{[h,j]} = h_j W^r_K, \\ Q^r_{[p,i]} &= p_j U^r_Q, \quad K^r_{[p,j]} = p_j U^r_K, \end{aligned} \tag{5.13}$$

where h_* p_* denote the input hidden state and position embedding, respectively; W^r_Q and W^r_K denote the relation-specific projection matrix for the hidden states; U^r_Q and

[2]In this section, we omit the introduction of the residual connection, the layer normalization, and the feed-forward layers, whose details are the same as vanilla Transformer [77].

U_K^r denote the relation-specific projection matrix for the position embeddings; $\sqrt{d}$ is the scaling term for retaining the magnitude of $\hat{\alpha}_{ij}^r$.

Now we obtain the relation-specific correlation matrix M^r for each relation r, which represents each two nodes' correlation along the specific relation.

5.4.1.2 Relation- and Structure-Aware 2-D Mask

Although we obtain the relation-specific correlation scores of all node pairs, a specific relation has its own adjacent structure which is crucial for information aggregation. And the attention score between two nodes should also be calculated regarding the relation-specific neighbors. To achieve this, we design the relation- and structure-aware 2-D mask to introduce the relation-specific structure into the attention mechanism. Specifically, the relational graph can derive $\mathbf{N^r}$ relation-specific adjacency matrice via disentangling. And for each relation r, its adjacency matrix A^r is used to mask its correlation matrix M^r. Finally, the normalized relation-specific attention score α_{ij}^r is obtained as follows:

$$\begin{aligned} \alpha_{ij}^r &= \text{softmax}\left(f_{mask}^{2D}(\alpha_{ij}^r, A_{ij}^r)\right), \\ f_{mask}^{2D} &= \begin{cases} \hat{\alpha}_{ij}^r & A_{ij}^r = 1 \\ -\infty & A_{ij}^r = 0, \end{cases} \end{aligned} \tag{5.14}$$

where f_{mask}^{2D} denotes the function of the relation- and structure-aware 2-D mask.

5.4.1.3 Output Node Representation

For the attention head corresponding to relation r, the updated sub-representation of node i (or the information that node i should receive along relation r) is:

$$\begin{aligned} \hat{h}_i^r &= \sum_{j \in \mathcal{N}_i^r} \alpha_{ij} V_{[h,j]}^r. \\ V_{[h,j]}^r &= h_j W_V^r \end{aligned} \tag{5.15}$$

A node is always connected to other nodes along different relations. Therefore, the final updated representation node i is the sum of its sub-representations of all attention heads:

$$\hat{h}_i = \sum_{r \in \mathcal{R}} \hat{h}_i^r. \tag{5.16}$$

5.4.2 SAT-ReTeFormer

In DARER[2], the speaker-aware temporal RSGT (Sec. 5.2.1.2) is achieved by the SAT-ReTeFormer rather than the RGCN used in DARER. And the speaker-aware

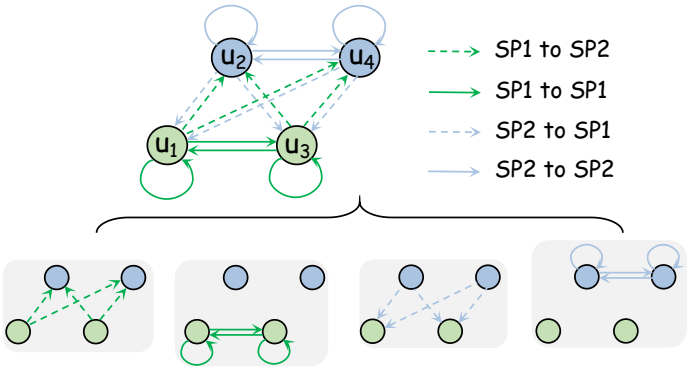


Figure 5.6: An example of the speaker-aware graph for SAT-ReTeFormer and its four disentangled views. Assuming there are four utterances: u_1 and u_3 (in green color) are from the speaker 1 ($SP1$); u_2 and u_4 (in blue color) are from the speaker 2 ($SP2$). Each view has its own adjacency matrix for the corresponding head of relational temporal attention.

graph for SAT-ReTeFormer is shown in Fig. 5.6. The input of SAT-ReTeFormer is the sequence of the initial utterance representations $H = (h_0, ..., h_N)$. In the SAT-ReTeFormer's speaker-aware graph, each node corresponds to an utterance, whose representation h_i corresponds to its position embedding p_i. Compared with RGCN, our SAT-ReTeFormer can explicitly model the correlations among the utterances, integrating both the speaker information and the fine-grained temporal information. After SAT-ReTeFormer, we obtain the context-, speaker- and temporal-sensitive utterance representations: $\hat{H} = (\hat{h_0}, ..., \hat{h_N})$.

5.4.3 DTR-ReTeFormer

In DARER^2, the dual-task reasoning RSGT (Sec. 5.2.3.2) is achieved by DTR-ReTeFormer rather than the RGCN used in DARER. And the dual-task reasoning graph for DTR-ReTeFormer is illustrated in Fig 5.7.

The input of DTR-ReTeFormer is the concatenation of $\hat{\mathbf{H}}^{\mathbf{t}}_{\mathbf{s}}$ and $\hat{\mathbf{H}}^{\mathbf{t}}_{\mathbf{a}}$: $\hat{\mathbf{H}}^{\mathbf{t}}_{\mathbf{s}} \parallel \hat{\mathbf{H}}^{\mathbf{t}}_{\mathbf{a}} = [\hat{h}^t_{s,1}, ..., \hat{h}^t_{s,N}, \hat{h}^t_{a,1}, ..., \hat{h}^t_{a,N}]$. Since $\hat{h}^t_{s,i}$ and $\hat{h}^t_{a,i}$ corresponds to the same utterance (u_i), they have the same position embedding p_i.

Here we demonstrate the details of the semantics- and prediction-level interactions achieved by DTR-ReTeFormer. Assuming that node i is sentiment node and node j is act node, the correlative score between i and j is calculated as follows (bringing Eq. 3

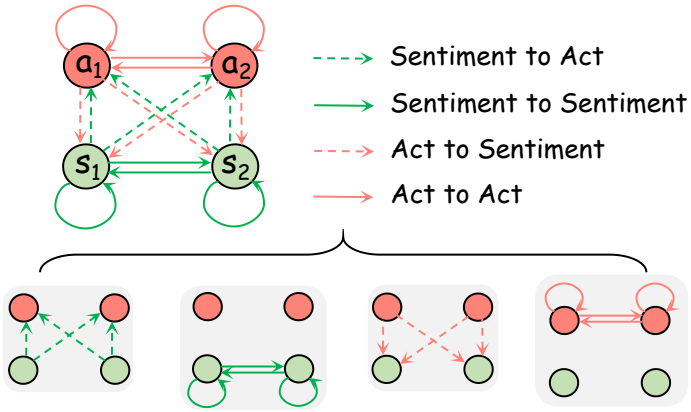


Figure 5.7: An example of the dual-task reasoning graph for SAT-ReTeFormer. W.l.o.g, the dialog includes two utterances. s_1 and s_2 (in green color) denote the sentiment nodes corresponding to the first and second utterances; a_1 and a_2 (in red color) denote the act nodes of the first and second utterances. The graph can be disentangled into four views along the four relations, and each view has its own adjacency matrix used in the corresponding head of relational temporal attention.

into Eq. 13):

$$
\begin{aligned}
\hat{\alpha}_{ij}^{r',t} &= \frac{1}{\sqrt{d}}(\hat{h}_{s,i}^{t} W_Q^r)(\hat{h}_{a,j}^{t} W_K^r)^T + \frac{1}{\sqrt{d}}(p_j U_Q^r)(p_j U_K^r)^T \\
&= \frac{1}{\sqrt{d}}\left((h_{s,i}^{t-1} + e_{s,i}^{t} + e_{a,i}^{t})W_Q^r\right)\left((h_{a,j}^{t-1} + e_{s,j}^{t} + e_{a,j}^{t})W_K^r\right)^T \\
&\quad + \frac{1}{\sqrt{d}}(p_i U_Q^r)(p_j U_K^r)^T \\
&= \frac{1}{\sqrt{d}} h_{s,i}^{t-1} W_Q^r (W_K^r)^T (h_{a,j}^{t-1})^T + \frac{1}{\sqrt{d}} h_{s,i}^{t-1} W_Q^r (W_K^r)^T (e_{s,j}^{t})^T \\
&\quad + \frac{1}{\sqrt{d}} h_{s,i}^{t-1} W_Q^r (W_K^r)^T (e_{a,j}^{t})^T + \frac{1}{\sqrt{d}} e_{s,i}^{t} W_Q^r (W_K^r)^T (h_{a,j}^{t-1})^T \\
&\quad + \frac{1}{\sqrt{d}} e_{s,i}^{t} W_Q^r (W_K^r)^T (e_{s,j}^{t})^T + \frac{1}{\sqrt{d}} e_{s,i}^{t} W_Q^r (W_K^r)^T (e_{a,j}^{t})^T \\
&\quad + \frac{1}{\sqrt{d}} e_{a,i}^{t} W_Q^r (W_K^r)^T (h_{a,j}^{t-1})^T + \frac{1}{\sqrt{d}} e_{a,i}^{t} W_Q^r (W_K^r)^T (e_{s,j}^{t})^T \\
&\quad + \frac{1}{\sqrt{d}} e_{a,i}^{t} W_Q^r (W_K^r)^T (e_{a,j}^{t})^T + \frac{1}{\sqrt{d}} (p_j U_Q^r)(p_j U_K^r)^T .
\end{aligned}
\tag{5.17}
$$

In $\hat{\alpha}_{ij}^{r',t}$, r' denotes the relation of *Act to Sentiment* and t denotes the time step of recurrent dual-task reasoning. We can observe that finally, there are 10 terms in Eq. 17. The 1st term models the semantics-level interaction between node i and node j. The 2nd-9th terms model the prediction-level interactions. Specifically, the 2nd, 3rd, 4th and 7th terms model the semantics-prediction interactions; the 5th, 6th

and 9th terms model the prediction-prediction interactions. The 10th term achieves relation-specific temporal modeling.

Therefore, our proposed DTR-ReTeFormer can explicitly and comprehensively model the semantics- and prediction-level interactions. And relational modeling achieves the self-task and cross-task interactions. Besides, in this process, the relation-specific temporal information is considered, facilitating dual-task reasoning.

5.5 Experiments

5.5.1 Datasets and Metrics

Datasets. We conduct experiments on two publicly available dialogue datasets: Mastodon [3] and Dailydialog [39]. The Mastodon dataset includes 269 dialogues for training and 266 dialogues for testing. And there are 3 sentiment classes and 15 act classes. Since there is no official validation set, we follow the same partition as [55]. Finally, there are 243 dialogues for training, 26 dialogues for validating, and 266 dialogues for testing. As for Dailydialog dataset, we adopt the official train/valid/test/ split from the original dataset [39]: 11,118 dialogues for training, 1,000 for validating, and 1,000 for testing. And there are 7 sentiment classes and 4 act classes.

Evaluation Metrics. Following previous works [3, 53, 55], on Dailydialog dataset, we adopt macro-average Precision (P), Recall (R), and F1 for the two tasks, while on Mastodon dataset, we ignore the neutral sentiment label in DSC task and for DAR task we adopt the average of the F1 scores weighted by the prevalence of each dialogue act.

5.5.2 Implement Details and Baselines

Both of DARER and DARER^2 are trained with Adam optimizer with the learning rate of $1e^{-3}$ and the batch size is 16. We exploit 300-dimensional Glove vectors for the word embeddings. And the epoch number is 100 for Mastodon and 50 for DailyDialog. Next, we introduce the different settings of other hyper-parameters for DARER and DARER^2

DARER The dimension of hidden states (label embeddings) is 128 for Mastodon and 256 for DailyDialog. The step number T for recurrent dual-task reasoning is set to 3 for Mastodon and 1 for DailyDialog. The coefficient γ_s and γ_a are 3 for Mastodon and $1e^{-4}$ for DailyDialog. To alleviate overfitting, we adopt dropout, and the ratio is 0.2 for Mastodon and 0.3 for DailyDialog.

Table 5.4: Experiment results. * denotes we reproduce the results using official code. † denotes that our DARER and DARER^2 significantly outperforms the previous best model Co-GAT with $p < 0.01$ under t-test and ‡ denotes $p < 0.05$. ↑ denotes the improvement achieved by our model over Co-GAT.

Models	Mastodon						DailyDialog					
	DSC			DAR			DSC			DAR		
	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)
JointDAS [3]	36.1	41.6	37.6	55.6	51.9	53.2	35.4	28.8	31.2	76.2	74.5	75.1
IIIM [33]	38.7	40.1	39.4	56.3	52.2	54.3	38.9	28.5	33.0	76.5	74.9	75.7
DCR-Net [53]	43.2	47.3	45.1	60.3	56.9	58.6	56.0	40.1	45.4	79.1	79.0	79.1
BCDCN [37]	38.2	62.0	45.9	57.3	61.7	59.4	55.2	45.7	48.6	80.0	80.6	80.3
Co-GAT [55]	44.0	53.2	48.1	60.4	60.6	60.5	65.9	45.3	51.0	81.0	78.1	79.4
Co-GAT*	45.40	48.11	46.47	62.55	58.66	60.54	58.04	44.65	48.82	79.14	79.71	79.39
DARER	**56.04**†	**63.33**†	**59.59**†	**65.08**‡	**61.88**†	**63.43**†	**59.96**‡	**49.51**†	**53.42**†	**81.39**†	**80.80**‡	**81.06**†
	↑23.4%	↑31.6%	↑28.2%	↑4.0%	↑5.5%	↑4.8%	↑3.3%	↑10.9%	↑9.4%	↑2.8%	↑1.4%	↑2.1%
DARER^2	**58.53**†	**67.06**†	**62.38**†	**68.26**†	**67.15**†	**67.70**†	**65.58**†	**48.28**†	**54.34**†	**81.41**†	**81.79**‡	**81.60**†
	↑28.9%	↑39.4%	↑34.2%	↑9.1%	↑14.5%	↑11.8%	↑13.0%	↑8.1%	↑11.3%	↑2.9%	↑2.6%	↑2.8%

DARER^2 The dimension of hidden states (label embeddings) is 256 for Mastodon and 300 for DailyDialog. The step number T for recurrent dual-task reasoning is set to 5 for Mastodon and 3 for DailyDialog. For Mastodon dataset, the coefficients γ_s and γ_a are 10 and 1. For DailyDialog, the coefficients γ_s and γ_a are 0.1 and $1e^{-6}$. The dropout ratio is 0.4 for both Mastodon and DailyDialog.

For all experiments, we pick the model performing best on the validation set and then report the average results on the test set based on three runs with different random seeds. All computations are conducted on NVIDIA RTX 6000.

We compare our model with: JointDAS [3], IIIM [33], DCR-Net (Co-Attention) [53], BCDCN [37] and Co-GAT [55].

5.5.3 Main Results

5.5.3.1 Comparison with Baselines

Table 5.4 lists the experiment results on the test sets of the two datasets. We can observe that:

1. Our models significantly outperform all baselines, achieving new state-of-the-art (SOTA). In particular, over Co-GAT, the existing SOTA, DARER achieves an absolute improvement of 13.1% in F1 score on DSC task in Mastodon, a relative improvement of over 28%. And DARER^2 achieves even larger improvements: over 34% improvement in F1 score on DSC task in Mastodon dataset. The satisfying results of DARERs come from (1) our framework integrates not only semantics-level interactions but also

prediction-level interactions, thus capturing explicit dependencies other than implicit dependencies; (2) our SATG represents the speaker-aware semantic states transitions, capturing the important basic semantics benefiting both tasks; (3) our DRTG provides a rational platform on which more effective dual-task relational reasoning is conducted. (4) the advanced architecture of our DARER models allows DSC and DAR to improve each other in the recurrent dual-task reasoning process gradually.
2. DARER and DARER^2 show more prominent superiority on DSC task than DAR task. We surmise the probable reason is that generally, the act label is more complicated to deduce than the sentiment label in dual-task reasoning. For instance, it is easy to infer u_i's Negative label on DSC given u_i's Agreement label on DAR and u_{i-1}'s Negative label on DSC. Reversely, given the label information that u_i and u_{i-1} are both negative on DSC, it is hard to infer the act label of u_i because there are several act labels possibly satisfying this case, e.g., Disagreement, Agreement, Statement.
3. Our models' improvements on DailyDialog are smaller than those on Mastodon. We speculate this is caused by the extremely unbalanced sentiment class distribution in DailyDialog. As shown in Fig. 5.8, in DailyDialog dataset, over 83% utterances do not express sentiment, while the act labels are rich and varied. This hinders DARER from learning valuable correlations between the two tasks.

5.5.3.2 Comparison of DARER and DARER^2

From Table 5.4, we can find that DARER^2 outperforms DARER, further improving the performance. This can be attributed to the fact that the proposed SAT-ReTeFormer and DTR-ReTeFormer in DARER^2 can more effectively model the relational and temporal interactions than the RGCNs adopted in DARER. Especially, in DTR-ReTeFormer, since the input hidden state is superimposed with both tasks' label representation of the corresponding utterance, the relation- and structure-aware disentangled multi-head attention can explicitly and sufficiently model the relation-specific dual-task interactions, including semantics-semantics interactions, semantics-prediction interactions, and prediction-prediction interactions.

5.5.3.3 Effect of Pre-trained Language Model

In this section, we study the effects of three PTLM encoders: BERT [9], RoBERTa [44], and XLNet [102], which replace the BiLSTM utterance encoder in the state-of-the-art model Co-GAT and our DARER models. We adopt the base versions of the PTLMs implemented in PyTorch by [87]. In our experiments, the whole models are trained by AdamW optimizer with the learning rate of $1e^{-5}$ and the batch size is

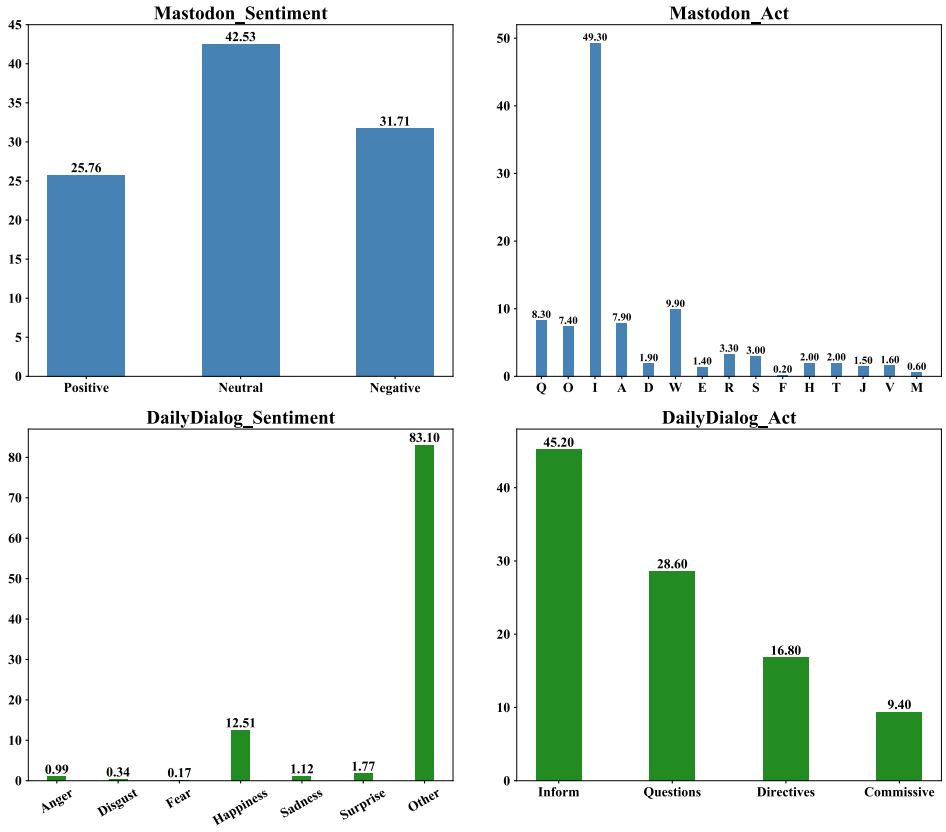


Figure 5.8: Illustration of class distributions on Mastodon and DailyDialog datasets.

16. And the PTLMs are fine-tuned in the training process. Results are listed in Table 5.5. We can find that since single PTLM encoders are powerful in language understanding, they obtain promising results even without any interactions between utterances or the two tasks. Nevertheless, stacking DARER on PTLM encoders further obtains around 4%-10% absolute improvements on F1. This is because our DARER models achieve relational temporal graph reasoning prediction-level interactions, which complement the high-quality semantics grasped by PTLM encoders. In contrast, Co-GAT only models the semantics-level interactions, whose advantages are diluted by the powerful PTLMs. Consequently, based on PTLM encoders, Co-GAT brings much less improvement than our DARER models.

5.5.4 Ablation Study

Our DARER and DARER^2 share the same overall model architectures, which include **label embeddings**, **constraint loss**, **SAT-RSGT**, **DTR-RSGT**, **TS-LSTMs**, **SATG** and **DRTG**. To study the effectiveness of each component, we conduct ablation experiments on DARER and Table 5.6 lists the results.

Mastodon Dataset (Test)		Predicted		
		Neg	Pos	Neu
Ground-Truth	Neg	271	35	66
	Pos	60	192	39
	Neu	153	104	222
F1 score (%)		63.32	61.74	47.01

DailyDialog Dataset (Test)		Predicted						
		No Emotion	Happiness	Surprise	Fear	Disgust	Sadness	Anger
Ground-Truth	No Emotion	5887	341	32	2	3	24	32
	Happiness	396	609	9	0	2	0	3
	Surprise	54	9	52	0	0	0	1
	Fear	9	0	1	7	0	0	0
	Disgust	27	0	0	0	15	1	4
	Sadness	70	0	1	0	0	27	4
	Anger	60	5	3	0	1	1	48
F1 score (%)		91.81	62.27	48.60	53.85	44.11	34.84	45.71

Figure 5.9: Illustration of confusion matrices and F1 score on each class on Mastodon and DailyDialog test sets. Note that on Mastodon test set, following previous works, the F1 score of the Neutral class is not counted for the final F1 score.

From Table 3.5, we have the following observations:

(1) Removing **label embeddings** causes prediction-level interactions not to be achieved. The sharp drops in results prove that our method of leveraging label information to achieve prediction-level interactions effectively improves dual-task reasoning via capturing explicit dependencies.

(2) Without **constraint loss**, the two logic rules can hardly be met, so there is no constraint forcing DSC and DAR to gradually prompt each other, resulting in the dramatic decline of performances. (3) As the core of Dialog Understanding, **SAT-RSGT** captures speaker-aware semantic states transitions, which provides essential basic task-free knowledge for both tasks. Without it, some essential indicative semantics would be lost, then the results decrease.

(4) The worst results of 'w/o DTR-RSGT' prove that **DTR-RSGT** is the core of

Table 5.5: Results comparison based on different PTLM encoders.

Models		Mastodon					
		DSC			DAR		
		P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)
BERT	+ Linear	61.79	61.09	60.60	70.20	67.49	68.82
	+ Co-GAT	66.03	58.13	61.56	70.66	67.62	69.08
	+ DARER	65.98	67.39	**66.42**	73.82	71.67	**72.73**
	+ DARER^2	64.47	71.10	**67.61**	75.34	73.04	**74.17**
RoBERTa	+ Linear	57.83	60.54	57.83	62.49	61.93	62.20
	+ Co-GAT	61.28	57.25	58.26	66.46	64.01	65.21
	+ DARER	61.36	67.27	**63.66**	70.87	68.68	**69.75**
	+ DARER^2	63.78	71.44	**66.49**	73.86	72.87	**73.36**
XLNet	+ Linear	61.42	67.80	63.35	67.31	63.04	65.09
	+ Co-GAT	64.01	65.30	63.71	67.19	64.09	65.60
	+ DARER	68.05	69.47	**68.66**	72.04	69.63	**70.81**
	+ DARER^2	67.20	74.24	**70.42**	72.45	71.47	**71.96**

Table 5.6: Results (in F1 score) of ablation experiments on DARER.

Variants	Mastodon		DailyDialog	
	DSC	DAR	DSC	DAR
DARER	**59.59**	**63.43**	**53.42**	**81.06**
w/o Label Embeddings	56.76	62.15	50.64	79.87
w/o $\mathcal{L}_{constraint}$	56.22	61.99	49.94	79.76
w/o SAT-RSGT	57.37	62.96	50.25	80.52
w/o DTR-RSGT	56.69	61.69	50.11	79.76
w/o TS-LSTMs	56.30	61.49	51.61	80.33
w/o Tpl Rels in SATG	58.23	62.21	50.99	80.70
w/o Tpl Rels in DRTG	57.22	62.15	50.52	80.28

DARER, and it plays a vital role in conducting dual-task relational reasoning over the semantics and label information.

(5) The significant results decrease of 'w/o TS-LSTMs' prove that **TS-LSTMs** also plays an important role in DARER by generating task-specific hidden states for both tasks and have some capability of sequence label-aware reasoning.

(6) Removing of the **temporal relations** (Tpl Rels) in SATG or DRTG causes distinct results decline. This can prove the necessity and effectiveness of introducing temporal relations into dialog understanding and dual-task reasoning.

To further study the necessity of $\mathcal{L}_{constraint}$, we compare DARER^2 and the variant w/o $\mathcal{L}_{constraint}$ on the detailed performances of each single step, as shown in Table 5.7. We can observe that the $\mathcal{L}_{constraint}$ in DARER2 can make the model generate better predictions at each step than w/o $\mathcal{L}_{constraint}$. Besides, thanks to the margin loss, DARER^2 can generate better and better predictions along the time step. However,

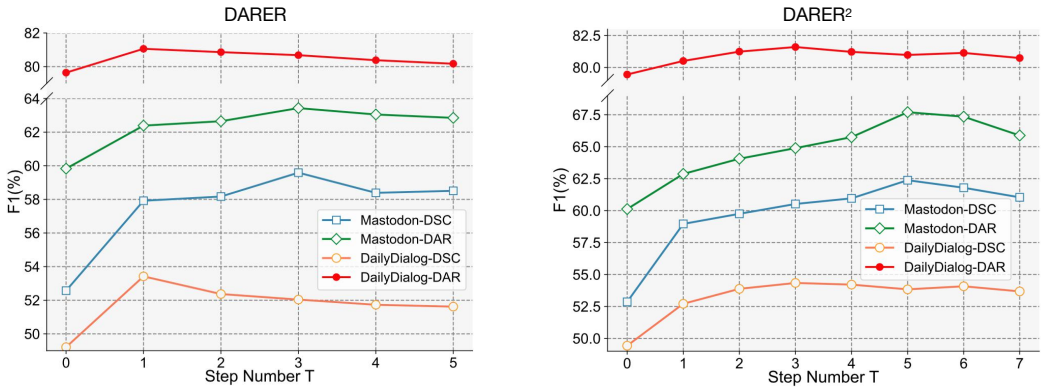


Figure 5.10: Performances of DARER and DARER^2 over different T.

Table 5.7: The comparison of DARER^2 and $\mathcal{L}_{constraint}$ on the performances of each single step.

Model	metric	step					
		0	1	2	3	4	5
DARER^2	F1	57.34	59.41	61.70	62.30	62.38	62.53
	P	59.24	59.94	59.35	58.08	57.48	57.00
	R	55.65	59.29	64.96	67.19	68.36	69.41
w/o $\mathcal{L}_{constraint}$	F1	54.15	58.66	58.89	59.51	58.54	58.75
	P	58.56	62.04	55.33	57.83	55.57	55.46
	R	51.73	62.04	62.95	61.36	61.88	62.45

removing $\mathcal{L}_{constraint}$ leads to fluctuations and significant drops in performance. The reason is that without the estimate loss and margin loss, the two rules cannot be integrated into the training process. Only relying on the cross-entropy loss at the final step cannot effectively improve the predictions of the previous step nor make the model generate better and better predictions along the step.

5.5.5 Superiority of ReTeFormer

In DARER^2 the SAT-RSGT and DTR-RSGT are achieved by our proposed SAT-ReTeFormer and DTR-ReTeFormer respectively, rather than RGCNs. To verify the superiorities of the two ReTeFormers over their RGCN counterparts in DARER, we change the setting of the two ReTeFormers and show the performances in Table 5.8.

We can observe that for both SATG and DTRG, our ReTeFomer shows significant superiority over RGCN. There are two main reasons. First, our proposed ReTeFormer can conduct *fine-grained* relational temporal modeling, while RGCN can only handle the *coarse-grained* relative temporal relations. Intuitively, fine-grained relational temporal modeling can better model the latent structures of the dialog than coarse-grained one, further benefiting dual-task reasoning. Second, the Relation- and Structure-Aware

Table 5.8: Results (in F1 score) of different settings of ReTeFormers. × denotes the corresponding ReTeFormer is replaced with the RGCN counterpart used in DARER.

Variants	-ReTeFormer		Mastodon		DailyDialog	
	SAT	DTR	DSC	DAR	DSC	DAR
DARER^2	✓	✓	**62.38**	**67.70**	**54.34**	**81.60**
M_1	×	✓	61.85	66.57	54.16	81.54
M_2	✓	×	60.25	64.23	53.98	81.22
DARER	×	×	59.59	63.43	53.42	81.06

Disentangled Multi-head Attention in our proposed ReTeFormer can explicitly model the correlations between the nodes. Especially, our DTR-ReTeFormer can explicitly and comprehensively model the self-task and cross-task interactions that are of both semantics- and prediction-level.

5.5.6 Impact of Step Number T

The performances of DARER and DARER^2 over different T are plotted in Fig. 5.10. $T = 0$ denotes the output of the Initial Estimation module is regarded as final predictions. We can find that appropriately increasing T brings results improvements. Particularly, with T increasing from 0 to 1, the results increase sharply. This verifies that the Initial Estimation module can provide useful label information for dual-task reasoning. Furthermore, DARER can learn beneficial mutual knowledge from recurrent dual-task reasoning in which DSC and DAR prompt each other. Generally, when T surpasses a certain point, the performances decline slightly. The possible reason is that after the peak, more dual-task interactions cause too much deep information fusion of the two tasks, leading to the loss of some important task-specific features and overfitting.

5.5.7 Case Study

To better understand how our model works well, we compare the final predictions of Co-GAT and our DARER^2, as shown in Fig. 5.11 (a). We can find that our DARER^2 can correctly predict all labels of both tasks, while there are some errors in Co-GAT's predictions: the act label of u_3 is incorrectly inferred as `Answer`, and the sentiment label of u_4 is incorrectly inferred as `Negative`. We suppose there are two reasons: (1) Co-GAT works on a homogeneous fully-connected dual-task graph, losing the intra- and cross-task dependencies and temporal information among the nodes; (2) Co-GAT

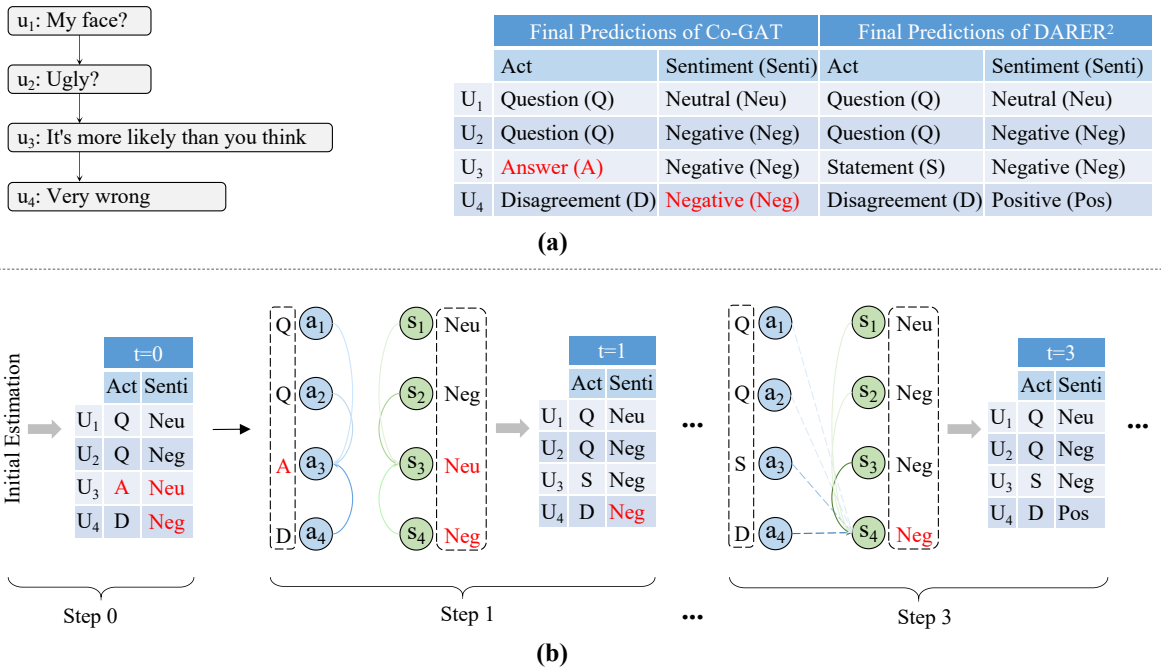


Figure 5.11: Case study. (a) The example dialog and the final predictions of Co-GAT and our DARER^2. The red color denotes error. (b) Illustration of the estimated labels at each time step and the reasoning process. For simplification, we only list the highest probability label rather than the whole label distribution. The dashed box denotes the label estimated at the previous step. a_i and s_i denote the act node and sentiment node of u_i, respectively. The blue solid arrows denote the edges between act nodes. The green solid arrows denote the edges between sentiment nodes. The blue dashed arrows denote the edges from act nodes to sentiment nodes. Deeper color denotes a larger attention weight.

only achieves semantics-level to implicitly models the dual-task dependencies, without incorporating prediction-level interactions.

To show how our DARER^2 conducts the dual-task relational temporal graph reasoning, we illustrate the dual-task reasoning process in Fig. 5.11 (b). At step 0, the initial estimation module produces the initial label distributions. In the first step of dual-task reasoning, some errors in the previously estimated labels are corrected through the intra-task interactions of act recognition task and sentiment classification task. Specifically, in act nodes, the semantics and label information of node a_4 is assigned a large weight and aggregated into node a_3. The `Disagreement` label of a_4 can indicate the `Statement` label of a_3. This is because in the dataset, if an utterance has a `Disagreement` act label, in most cases, its previous utterance has a `Statement` act label, which is also consistent with the real-world scenarios. In sentiment nodes, s_2 is assigned large weight and aggregated into node s_3. Combining the semantics

Table 5.9: Comparison with SOTA on model parameters, training time, GPU memory required, and performance.

Models	Number of Parameters ↓	Training Time per Epoch ↓	GPU ↓ Memory	Avg. F1↑
Co-GAT	3.61M	1.86s	1531MB	53.66%
DARER	2.50M	1.81s	1187MB	61.51%
Improve	**-30.7**%	**-2.7**%	**-22.4**%	**+14.6**%
DARER^2	3.83M	2.16s	1191MB	65.04%
Improve	+6.1%	+16.1%	**-22.2**%	**+21.2**%

'more' of s_3 and the `Negative` label of s_2, the `Negative` label of s_3 can be correctly inferred. Then in the third step of dual-task reasoning, the wrong label of s_4 node is fixed. Specifically, node a_3, a_4 and s_3 are assigned relatively large attention weights for s_4. Regarding the labels of a_3 and a_4, u_4 disagrees u_3, indicating s_3 and s_4 may have opposite sentiments. And further considering the `Negative` label of s_4, our model can produce the correct `Positive` sentiment label for u_4. In this way, our model can gradually generate better labels through recurrent relational temporal graph reasoning.

5.5.8 Computation Efficiency

In practical application, in addition to the performance, the number of parameters, the time cost, and GPU memory required are important factors. Taking Mastodon as the testbed, we compare our DARER models with the up-to-date SOTA (Co-GAT) on these factors, and the results are shown in Table 5.9. Avg. F1 denotes the average of the F1 scores on the two tasks. We can find that although our DARER models surpass SOTA by a large margin, they do not significantly cost more computation resources. Especially, DARER is even more efficient than Co-GAT. As for DARER^2, although it has some more parameters and costs more training time, this is acceptable considering that it can save about 22% GPU memory and improve 21% performance. Therefore, our DARER models are relatively efficient for practical application.

5.5.9 Experiment on joint Multiple Intent Detection and Slot Filling

To verify the generality of our method, we further conduct experiments on the task of joint multiple intent detection and slot filling.

5.5.9.1 Task Definition

The input is an utterance that can be denoted as $U = \{u_i\}_1^n$. Multiple intent detection can be formulated as a multi-label classification task that predicts multiple intents expressed in the input utterance. And slot filling is a sequence labeling task that maps each u_i into a slot label.

5.5.9.2 Model Architecture

We apply our relational temporal graph reasoning to the state-of-the-art model GL-GIN [57], forming DARER and DARER^2 in this joint task scenario. In GL-GIN, the dual-task graph is a semantics-label graph, which is a homogeneous graph including two groups of nodes: predicted intent label nodes and slot semantics nodes. And vanilla GAT is utilized for graph reasoning. In our DARER, the dual-task graph is a relational temporal graph, in which there are (1) intra- and cross-task relations among the two tasks' nodes; (2) coarse-grained temporal relations among the slot semantics nodes. And RGCN is utilized for relational temporal graph reasoning In our DARER^2, the dual-task graph is also a relational temporal graph, including intra- and cross-task relations. And our proposed DTR-ReTeFormer is used for relational temporal graph reasoning. Since DTR-ReTeFormer can achieve the fine-grained temporal modeling, compared with GL-GIN and DARER, DARER^2 can capture the dependencies between B- slot labels and their I- slot labels, and this advantage is proven in Fig. 5.12.

5.5.9.3 Datasets and Metrics

Datasets. Following previous works, the two benchmarks: MixATIS and MixSNIPS [8, 20, 58] are used as testbeds for evaluation. In MixATIS, the split of train/dev/test set is 13162/756/828 (utterances). In MixSNIPS, the split of train/dev/test set is 39776/2198/2199 (utterances).

Evaluation Metrics. Following previous works, multiple intent detection is evaluated by accuracy (Acc); slot filling is evaluated using F1 score; sentence-level semantic frame parsing is evaluated using overall Acc. Overall Acc denotes the ratio of utterances for which both intents and slots are predicted correctly.

5.5.9.4 Implement Details and Baselines

Following GL-GIN [57], the word and label embeddings are randomly initialized and trained with the model. The dimension of the word/label embedding is 128 on MixATIS and 256 on MixSNIPS. The dimension of the hidden state is 200. We adopt

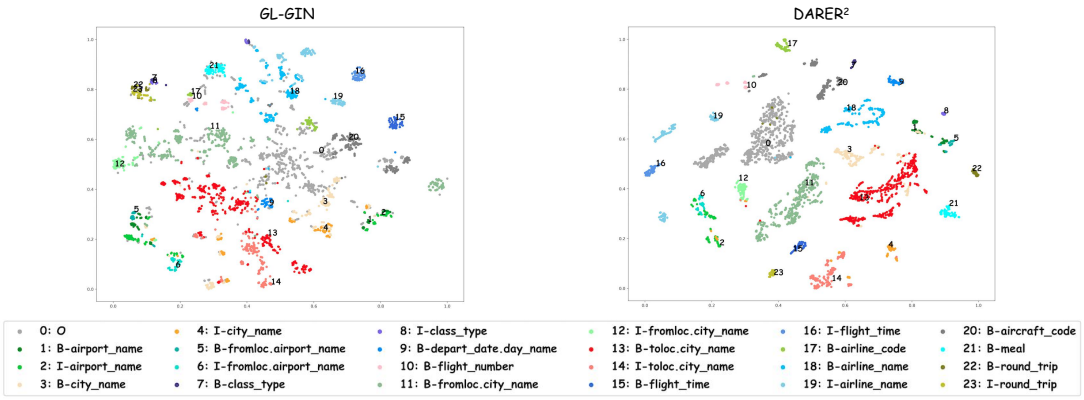


Figure 5.12: Visualizations of slot hidden states generated by GL-GIN and our DARER^2.

Table 5.10: Results comparison. † denotes our model significantly outperforms baselines with $p < 0.01$ under t-test.

Models	MixATIS			MixSNIPS		
	Overall (Acc)	Slot (F1)	Intent (Acc)	Overall (Acc)	Slot (F1)	Intent (Acc)
Attention BiRNN [42]	39.1	86.4	74.6	59.5	89.4	95.4
Slot-Gated [16]	35.5	87.7	63.9	55.4	87.9	94.6
Bi-Model [84]	34.4	83.9	70.3	63.4	90.7	95.6
SF-ID [11]	34.9	87.4	66.2	59.9	90.6	95.0
Stack-Propagation [54]	40.1	87.8	72.1	72.9	94.2	96.0
Joint Multiple ID-SF [13]	36.1	84.6	73.4	62.9	90.6	95.1
AGIF [58]	40.8	86.7	74.4	74.2	94.2	95.1
GL-GIN [57]	43.0	88.2	76.3	73.7	94.0	95.7
DARER	**44.7**†	**88.4**	**76.7**†	**74.7**†	**94.4**†	**96.5**†
DARER^2	**49.0**†	**89.2**†	**77.3**†	**76.3**†	**94.9**†	**96.7**†

Adam optimizer for model training with the default setting. For all experiments, we select the best model on the dev set and report its results on the test set.

We compare our model with Attention BiRNN [42], Slot-Gated [16], Bi-Model [84], SF-ID [11], Stack-Propagation [54], Joint Multiple ID-SF [13], AGIF [58] and GL-GIN [57]

5.5.9.5 Results and Analysis

The results comparison is shown in Table 5.10. We can observe that our DARER models significantly outperform the state-of-the-art model GL-GIN on all datasets. In particular, DARER^2 significantly surpasses GL-GIN on MixATIS dataset in terms of Overall Acc, achieving 14% relative improvement. The superior performances of our DARER models verify the advantages of our proposed relation temporal graph

reasoning. On one hand, relation temporal graph reasoning can effectively model the intra- and cross-task relation-specific interactions. On the other hand, it can model the temporal information among the slot semantics nodes. Especially, the DTR-ReTeFormer in DARER^2 can comprehensively model the fine-grained temporal information, capturing the slot dependencies. To further verify this, we visualize the slot hidden state generated by DARER^2 and GL-GIN, as shown in Fig. 5.12 We can observe that DARER^2's clusters are clearer than GL-GIN's. Besides, the B- slot clusters of DARER^2 and their corresponding I- slot clusters are separated clearly. In contrast, some GL-GIN's generated B- slot clusters and their corresponding I- slot clusters even overlap. The high quality of our DARER^2's generated hidden states can be attributed to three facts. First, GL-GIN uses the vanilla GAT for information aggregation, leading to a disadvantage: for each slot node, the different information of the intent label nodes and other slot nodes are directly fused to it. Differently, the DTR-ReTeFormer in DARER^2 can achieve relation-specific information aggregation, which can better leverage the beneficial information via discriminating the contributions of the two tasks' nodes. Second, the GAT in GL-GIN cannot model the temporal information, losing the dependencies among slot nodes. Since each slot node corresponds to a word in utterance, the group of slot nodes can be regarded as a sequence, where there are temporal dependencies (e.g. I-Singer can only occur behind B-Singer). And our DARER models can achieve the relational temporal modeling, then capture the beneficial slot dependencies.

5.6 Summary

In this paper, we present a new framework, which for the first time achieves relational temporal graph reasoning and integrates prediction-level interactions to leverage estimated label distribution as explicit and important clues other than implicit semantics. We design the SATG and DRTG to facilitate relational temporal graph reasoning of dialog understanding and dual-task reasoning. To achieve our framework, we first propose a novel model named DARER to model the relational interactions between temporal information, label information, and semantics to let two tasks gradually promote each other, which is further forced by the proposed logic-heuristic training objective. Then we propose DARER^2, which further enhances relational temporal graph reasoning by adopting our proposed SAT-ReTeFormer and DTR-ReTeFormer. Experimental results demonstrate the superiority of our DARER models, which surpasses previous models by a large margin in different dual-task dialog language understanding scenarios.

Our work brings two insights for dialog understanding and multi-task reasoning in dialog systems: (1) exploiting the relational temporal information of the dialog for graph reasoning; (2) leveraging estimated label distributions to capture explicit correlations between the multiple tasks. In the future, we will apply our method to other multi-task learning scenarios in dialog systems.

Chapter 6

Co-guiding Net

6.1 Introduction

Spoken language understanding (SLU) [103] is a fundamental task in dialog systems. Its objective is to capture the comprehensive semantics of user utterances, and it typically includes two subtasks: intent detection and slot filling [74]. Intent detection aims to predict the intention of the user utterance and slot filling aims to extract additional information or constraints expressed in the utterance.

Recently, researchers discovered that these two tasks are closely tied, and a bunch of models [11, 16, 36, 43, 54] are proposed to combine the single-intent detection and slot filling in multi-task frameworks to leverage their correlations.

However, in real-world scenarios, a user usually expresses multiple intents in a single utterance. To this end, [32] begin to tackle the multi-intent detection task and [13] make the first attempt to jointly model the multiple intent detection and slot filling in a multi-task framework. [58] propose an AGIF model to adaptively integrate the fine-grained multi-intent prediction information into the autoregressive decoding process of slot filling via graph attention network (GAT) [78]. And [57] further propose a non-autoregressive GAT-based model which enhances the interactions between the predicted multiple intents and the slot hidden states, obtaining state-of-the-art results and significant speedup.

Despite the promising progress that existing multi-intent SLU joint models have achieved, we discover that they suffer from two main issues:

(1) **Ignoring the guidance from slot to intent**. Since previous researchers realized that "slot labels could depend on the intent" [13], existing models leverage the information of the predicted intents to guide slot filling, as shown in Fig. 6.1(a). However, they ignore that slot labels can also guide the multi-intent detection task. Based on our observations, multi-intent detection and slot filling are bidirectionally

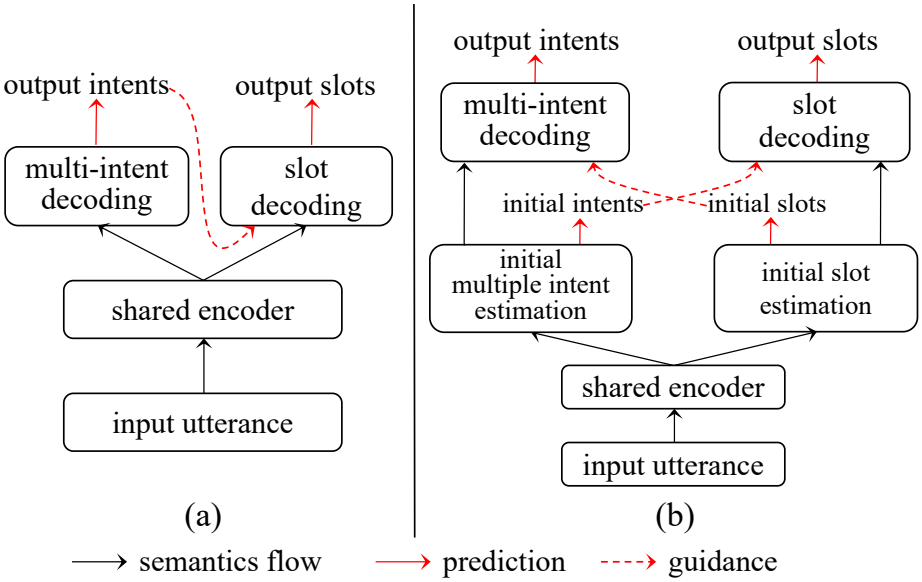


Figure 6.1: (a) Previous framework which only models the unidirectional guidance from multi-intent predictions to slot filling. (b) Our framework which models the mutual guidances between the two tasks.

interrelated and can mutually guide each other. For example, in Fig 6.2, not only the intents can indicate the slots, but also the slots can infer the intents. However, in previous works, the only guidance that the multiple intent detection task can get from the joint model is sharing the basic semantics with the slot filling task. As a result, the lack of guidance from slot to intent limits multiple intent detection, and so the joint task.

(2) **Node and edge ambiguity in the semantics-label graph.** [57, 58] apply GATs over the constructed graphs to model the interactions among the slot semantics nodes and intent label nodes. However, their graphs are homogeneous, in which all nodes and edges are treated as the same type. For a slot semantics node, the information from intent label nodes and other slot semantics nodes play different roles, while the homogeneous graph cannot discriminate their specific contributions, causing ambiguity. Therefore, the heterogeneous graphs should be designed to represent the relations among the semantic nodes and label nodes to facilitate better interactions.

In this paper, we propose a novel model termed Co-guiding Net to tackle the above two issues. For the first issue, Co-guiding Net implements a two-stage framework as shown in Fig. 6.1 (b). The first stage produces the initial estimated labels for the two tasks and the second stage leverages the estimated labels as prior label information to allow the two tasks mutually guide each other. For the second issue, we propose two heterogeneous semantics-label graphs (HSLGs): (1) a slot-to-intent semantics-label

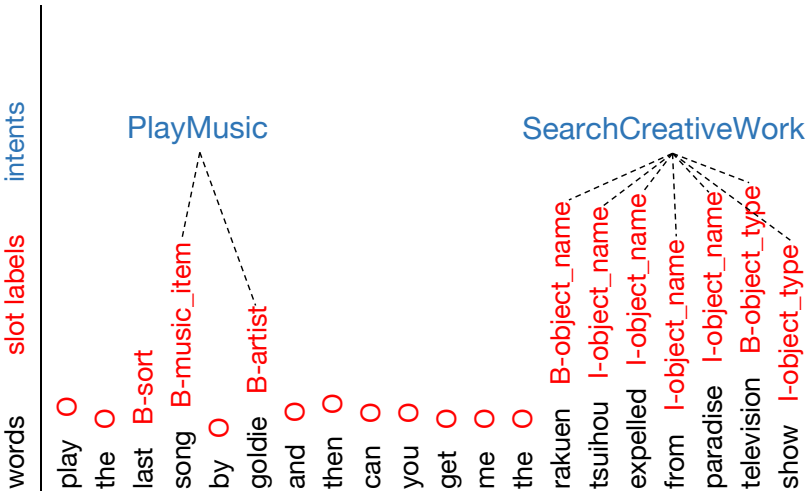


Figure 6.2: Illustration of the bidirectional interrelations between intent (blue) and slot (red) labels. The sample is retrieved from MixSNIPS dataset.

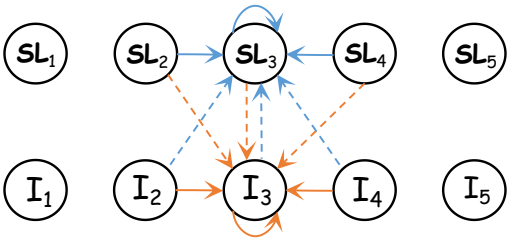
graph (S2I-SLG) that effectively represents the relations among the intent semantics nodes and slot label nodes; (2) an intent-to-slot semantics-label graph (I2S-SLG) that effectively represents the relations among the slot semantics nodes and intent label nodes. Moreover, two heterogeneous graph attention networks (HGATs) are proposed to work on the two proposed graphs for modeling the guidances from slot to intent and intent to slot, respectively. Experiment results show that our Co-guiding Net significantly outperforms previous models, and model analysis further verifies the advantages of our model.

The contributions of our work are three-fold: (1) We propose Co-guiding Net[1], which implements a two-stage framework allowing multiple intent detection and slot filling mutually guide each other. We make the first attempt to achieve the mutual guidances between the two tasks. (2) We propose two heterogeneous semantics-label graphs as appropriate platforms for interactions between semantics nodes and label nodes. And we propose two heterogeneous graph attention networks to model the mutual guidances between the two tasks. (3) Experiment results demonstrate that our model achieves new state-of-the-art performance.

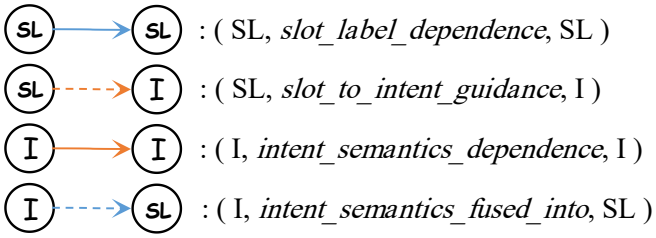
6.2 Co-guiding

Problem Definition Given a input utterance denoted as $U = \{u_i\}_1^n$, multiple intent detection can be formulated as a multi-label classification task that outputs multiple intent labels corresponding to the input utterance. And slot filling is a sequence labeling task that maps each u_i into a slot label.

[1]https://github.com/XingBowen714/Co-guiding



(a) Slot-to-Intent Semantics-Label Graphs (S2I-SLG)



(b) Relation types of S2I-SLG

Figure 6.3: The illustration of S2I-SLG and its relation types. w.l.o.g, only the edges directed into SL_3 and I_3 are shown, and the local window size is 1.

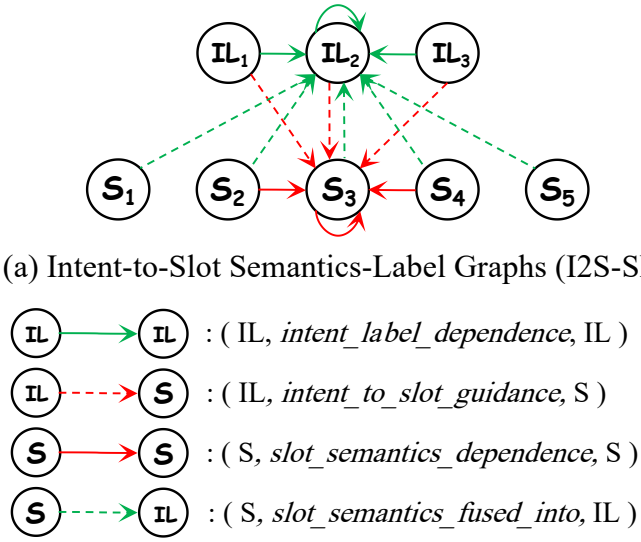
Next, before diving into the details of Co-guiding Net's architecture, we first introduce the construction of the two heterogeneous graphs.

6.2.1 Graph Construction

6.2.1.1 Slot-to-Intent Semantics-Label Graph

To provide an appropriate platform for modeling the guidance from the estimated slot labels to multiple intent detection, we design a slot-to-intent semantics-label graph (S2I-SLG), which represents the relations among the semantics of multiple intent detection and the estimated slot labels. S2I-SLG is a heterogeneous graph and an example is shown in Fig. 6.3 (a). It contains two types of nodes: intent semantics nodes (e.g., I_1, ..., I_5) and slot label (SL) nodes (e.g., SL_1, ..., SL_5). And there are four types of edges in S2I-SLG, as shown in Fig. 6.3 (b). Each edge type corresponds to an individual kind of information aggregation on the graph.

Mathematically, the S2I-SLG can be denoted as $\mathcal{G}_{s2i} = (\mathcal{V}_{s2i}, \mathcal{E}_{s2i}, \mathcal{A}_{s2i}, \mathcal{R}_{s2i})$, in which $\mathcal{V}_{s2i}$ is the set of all nodes, $\mathcal{E}_{s2i}$ is the set of all edges, $\mathcal{A}_{s2i}$ is the set of two node types and $\mathcal{R}_{s2i}$ is the set of four edge types. Each node v_{s2i} and each edge e_{s2i} are associated with their type mapping functions $\tau(v_{s2i}) : \mathcal{V}_{s2i} \rightarrow \mathcal{A}_{s2i}$ and $\phi(e_{s2i}) : \mathcal{E}_{s2i} \rightarrow \mathcal{R}_{s2i}$. For instance, in Fig. 6.3, the SL_2 node belongs to $\mathcal{V}_{s2i}$, while its node type SL belongs to $\mathcal{A}_{s2i}$; the edge from SL_2 to I_3 belongs to $\mathcal{E}_{s2i}$, while its edge type *slot_to_intent_guidance* belongs to $\mathcal{R}_{s2i}$. Besides, edges in S2I-SLG are



(a) Intent-to-Slot Semantics-Label Graphs (I2S-SLG)

(b) Relation types of I2S-SLG

Figure 6.4: The illustration of I2G-SLG and its relation types. w.l.o.g, only the edges directed into IL_3 and S_3 are shown, and the local window size is 1.

based on local connections. For example, node I_i is connected to $\{\mathrm{I}_{i-w}, ..., \mathrm{I}_{i+w}\}$ and $\{\mathrm{SL}_{i-w}, ..., \mathrm{SL}_{i+w}\}$, where w is a hyper-parameter of the local window size.

6.2.1.2 Intent-to-Slot Semantics-Label Graph

To present a platform for accommodating the guidance from the estimated intent labels to slot filling, we design an intent-to-slot semantics-label graph (I2S-SLG) that represents the relations among the slot semantics nodes and the intent label nodes. I2S-SLG is also a heterogeneous graph and an example is shown in Fig. 6.4 (a). It contains two types of nodes: slot semantics nodes (e.g., S_1, ..., S_5) and **i**ntent **l**abel (IL) nodes (e.g., IL_1, ..., IL_5). And Fig. 6.4 (b) shows the four edge types. Each edge type corresponds to an individual kind of information aggregation on the graph.

Mathematically, the I2S-SLG can be denoted as $\mathcal{G}_{i2s} = (\mathcal{V}_{i2s}, \mathcal{E}_{i2s}, \mathcal{A}_{i2s}, \mathcal{R}_{i2s})$. Each node v_{i2s} and each edge e_{i2s} are associated with their type mapping functions $\tau(v_{i2s})$ and $\phi(e_{i2s})$. The connections in I2S-SLG are a little different from S2I-SLG. Since intents are sentence-level, each IL node is globally connected with all nodes. For S_i node, it is connected to $\{\mathrm{S}_{i-w}, ..., \mathrm{S}_{i+w}\}$ and $\{\mathrm{IL}_1, ..., \mathrm{IL}_m\}$, where w is the local window size and m is the number of estimated intents.

6.2.2 Model Architecture

In this section, we introduce the details of our Co-guiding Net, whose architecture is shown in Fig.6.5.

6.2.2.1 Shared Self-Attentive Encoder

Following [57,58], we adopt a shared self-attentive encoder to produce the initial hidden states containing the basic semantics. It includes a BiLSTM and a self-attention module. BiLSTM captures the temporal dependencies:

$$h_i = \text{BiLSTM}\left(x_i, h_{i-1}, h_{i+1}\right), \tag{6.1}$$

where x_i is the word vector of u_i. Now we obtain the context-sensitive hidden states $\hat{\boldsymbol{H}} = \{\hat{h_i}\}_1^n$.

Self-attention captures the global dependencies:

$$\boldsymbol{H'} = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d_k}}\right)\boldsymbol{V}, \tag{6.2}$$

where $\boldsymbol{H'}$ is the global contextual hidden states output by self-attention; $\boldsymbol{Q}, \boldsymbol{K}$ and $\boldsymbol{V}$ are matrices obtained by applying different linear projections on the input utterance word vector matrix.

Then we concatenate the output of BiLSTM and self-attention to form the output of the shared self-attentive encoder: $\boldsymbol{H} = \hat{\boldsymbol{H}} \| \boldsymbol{H'}$, where $\boldsymbol{H} = \{h_i\}_1^n$ and $\|$ denotes concatenation operation.

6.2.2.2 Initial Estimation

Multiple Intent Detection To obtain the task-specific features for multiple intent detection, we apply a BiLSTM layer over $\boldsymbol{H}$:

$$h_i^{[I,0]} = \text{BiLSTM}_\text{I}\left(h_i, h_{i-1}^{[I,0]}, h_{i+1}^{[I,0]}\right). \tag{6.3}$$

Following [57, 58], we conduct token-level multi-intent detection. Each $h_i^{[I,0]}$ is fed into the intent decoder. Specifically, the intent label distributions of the i-th word are obtained by:

$$y_i^{[I,0]} = \text{sigmoid}\left(\boldsymbol{W}_I^1\left(\sigma(\boldsymbol{W}_I^2\boldsymbol{h}_i^{[I,0]} + \boldsymbol{b}_I^2)\right) + \boldsymbol{b}_I^1\right), \tag{6.4}$$

where σ denotes the non-linear activation function; W_* and b_* are model parameters.

Then the estimated sentence-level intent labels $\{\text{IL}_1, ..., \text{IL}_m\}$ are obtained by the token-level intent voting [57].

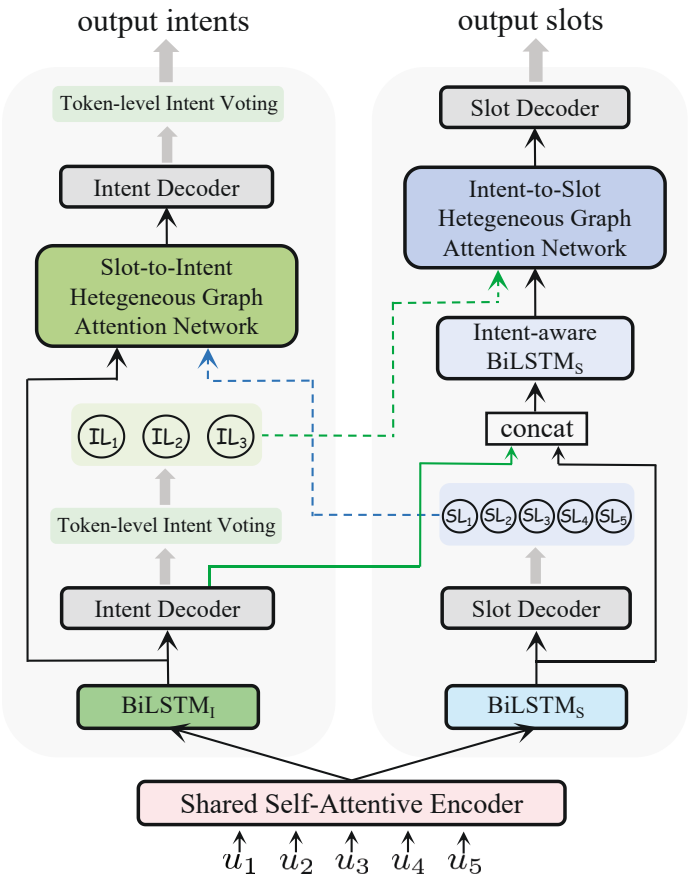


Figure 6.5: The architecture of Co-guiding Net. Each HGAT is triggered by its own task's semantics and the counterpart's predicted labels. The green and blue dashed arrow lines denote the projected label representations from the predicted intents and slots, respectively. The green solid arrow line denotes the intent distribution generated by the Intent Decoder at the first stage.

Slot Filling [57] propose a non-autoregressive paradigm for slot filling decoding, which achieves significant speedup. In this paper, we also conduct parallel slot filling decoding.

We first apply a BiLSTM over $\boldsymbol{H}$ to obtain the task-specific features for slot filling:

$$h_i^{[S,0]} = \text{BiLSTM}_\text{S}(h_i, h_{i-1}^{[S,0]}, h_{i+1}^{[S,0]}). \tag{6.5}$$

Then use a softmax classifier to generate the slot label distribution for each word:

$$y_i^{[S,0]} = \text{softmax}\left(\boldsymbol{W}_S^1\left(\sigma(\boldsymbol{W}_S^2\boldsymbol{h}_i^{[S,0]}+\boldsymbol{b}_S^2)\right)+\boldsymbol{b}_S^1\right). \tag{6.6}$$

And the estimated slot label for each word is obtained by $\text{SL}_i = \arg\max(y_i^{[S,0]})$.

6.2.2.3 Heterogeneous Graph Attention Network

State-of-the-art models [57, 58] use a homogeneous graph to connect the semantic nodes of slot filling and the intent label nodes. And GAT [78] is adopted to achieve information aggregation. In Sec. 6.1, we propose that this manner cannot effectively learn the interactions between one task's semantics and the estimated labels of the other task. To tackle this issue, we propose two heterogeneous graphs (S2I-SLG and I2S-SLG) to effectively represent the relations among the semantic nodes and label nodes. To model the interactions between semantics and labels on the proposed graphs, we propose a Heterogeneous Graph Attention Network (HGAT). When aggregating the information into a node, HGAT can discriminate the specific information from different types of nodes along different relations. And two HGATs (S2I-HGAT and I2S-HGAT) are applied on S2I-SLG and I2S-SLG, respectively. Specifically, S2I-HGAT can be formulated as follows:

$$
\begin{aligned}
h_i^{l+1} &= \Big\Vert_{k=1}^{K} \sigma\left(\sum_{j\in\mathcal{N}_{s2i}^i} W_{s2i}^{[r,k,1]}\alpha_{ij}^{[r,k]}h_j^l\right), r=\phi\left(e_{s2i}^{[j,i]}\right), \\
\alpha_{ij}^{[r,k]} &= \frac{\exp\left(\left(W_{s2i}^{[r,k,2]}h_i^l\right)\left(W_{s2i}^{[r,k,3]}h_j^l\right)^{\mathsf{T}}/\sqrt{d}\right)}{\sum\limits_{u\in\mathcal{N}_{s2i}^{r,i}}\exp\left(\left(W_{s2i}^{[r,k,2]}h_i^l\right)\left(W_{s2i}^{[r,k,3]}h_u^l\right)^{\mathsf{T}}/\sqrt{d}\right)},
\end{aligned}
\tag{6.7}
$$

where K denotes the total head number; $\mathcal{N}_{s2i}^i$ denotes the set of incoming neighbors of node i on S2I-SLG; $W_{s2i}^{[r,k,*]}$ are weight matrices of edge type r on the k-th head; $e_{s2i}^{[j,i]}$ denotes the edge from node j to node i on S2I-SLG; $\mathcal{N}_{s2i}^{r,i}$ denotes the nodes connected to node i with r-type edges on S2I-SLG; d is the dimension of node hidden state.

I2S-HGAT can be derived like Eq. 6.7.

6.2.2.4 Intent Decoding with Slot Guidance

In the first stage, we obtain the initial intent features $H^{[I,0]} = \{h_i^{I,0}\}_i^n$ and the initial estimated slot labels sequence $\{\mathrm{SL}_1, ..., \mathrm{SL}_n\}$. Now we project the slot labels into vector form using the slot label embedding matrix, obtaining $E_{sl} = \{e_{sl}^1, ..., e_{sl}^n\}$.

Then we feed $H^{[I,0]}$ and E_{sl} into S2I-HGAT to model their interactions, allowing the estimated slot label information to guide the intent decoding:

$$
H^{[I,L]} = \text{S2I-HGAT}\left([H^{[I,0]}, E_{sl}], \mathcal{G}_{s2i}, \theta_I\right), \tag{6.8}
$$

where $[H^{[I,0]}, E_{sl}]$ denotes the input node representation; θ_I denotes S2I-HGAT's parameters. L denotes the total layer number.

Finally, $H^{[I,L]}$ is fed to intent decoder, producing the intent label distributions for the utterance words: $Y^{[I,1]} = \{y_i^{[I,1]}, ..., y_n^{[I,1]}\}$. And the final output sentence-level intents are obtained via applying token-level intent voting over $Y^{[I,1]}$.

6.2.2.5 Slot Decoding with Intent Guidance

Intent-aware BiLSTM Since the B-I-O tags of slot labels have temporal dependencies, we use an intent-aware BiLSTM to model the temporal dependencies among slot hidden states with the guidance of estimated intents:

$$\tilde{h}_i^{[S,0]} = \text{BiLSTM}(y_i^{[I,0]} \| h_i^{[S,0]}, \tilde{h}_{i-1}^{[S,0]}, \tilde{h}_{i+1}^{[S,0]}). \tag{6.9}$$

I2S-HGAT We first project the estimated intent labels $\{\text{IL}_j\}_1^m$ into vectors using the intent label embedding matrix, obtaining $E_{il} = \{e_{il}^1, ..., e_{il}^m\}$. Then we feed $\tilde{H}^S$ and E_{il} into I2S-HGAT to model their interactions, allowing the estimated intent label information to guide the slot decoding:

$$H^{[S,L]} = \text{I2S-HGAT}\left([\tilde{H}^S, E_{il}], \mathcal{G}_{i2s}, \theta_S\right), \tag{6.10}$$

where $[\tilde{H}^{[S]}, E_{il}]$ denotes the input node representation; θ_S denotes I2S-HGAT's parameters.

Finally, $H^{[S,L]}$ is fed to slot decoder, producing the slot label distributions for each word: $Y^{[S,1]} = \{y_i^{[S,1]}, ..., y_n^{[S,1]}\}$. And the final output slot labels are obtained by applying arg max over $Y^{[S,1]}$.

6.2.2.6 Training Objective

Loss Function The loss function for multiple intent detection is:

$$\begin{aligned} \text{CE}(\hat{y}, y) &= \hat{y}\log(y) + (1-\hat{y})\log(1-y), \\ \mathcal{L}_I &= \sum_{t=0}^{1}\sum_{i=1}^{n}\sum_{j=1}^{N_I} \text{CE}\left(\hat{y}_i^I[j], y_i^{[I,t]}[j]\right). \end{aligned} \tag{6.11}$$

And the loss function for slot filling is:

$$\mathcal{L}_S = \sum_{t=0}^{1}\sum_{i=1}^{n}\sum_{j=1}^{N_S} \hat{y}_i^S[j]\log\left(y_i^{[S,t]}[j]\right), \tag{6.12}$$

where N_I and N_S denote the total numbers of intent labels and slot labels; $\hat{y}_i^I$ and $\hat{y}_i^S$ denote the ground-truth intent labels and slot labels.

Margin Penalty The core of our model is to let the two tasks mutually guide each other. Intuitively, the predictions in the second stage should be better than those in the first stage. To force our model obey this rule, we design a margin penalty ($\mathcal{L}^{mp}$) for each task, whose aim is to improve the probabilities of the correct labels. Specifically, the formulations of $\mathcal{L}_I^{mp}$ and $\mathcal{L}_S^{mp}$ are:

$$\begin{aligned}\mathcal{L}_I^{mp} &= \sum_{i=1}^{n}\sum_{j=1}^{N_I} \hat{y}_i^I[j] \ \max\left(0, y_i^{[I,0]}[j] - y_i^{[I,1]}[j]\right), \\ \mathcal{L}_S^{mp} &= \sum_{i=1}^{n}\sum_{j=1}^{N_S} \hat{y}_i^S[j] \max\left(0, y_i^{[S,0]}[j] - y_i^{[S,1]}[j]\right).\end{aligned} \tag{6.13}$$

Model Training The training objective $\mathcal{L}$ is the weighted sum of loss functions and margin regularizations of the two tasks:

$$\mathcal{L} = \gamma\left(\mathcal{L}_I + \beta_I \mathcal{L}_I^{mp}\right) + (1-\gamma)\left(\mathcal{L}_S + \beta_S \mathcal{L}_S^{mp}\right), \tag{6.14}$$

where γ is the coefficient balancing the two tasks; β_I and β_S are the coefficients of the margin regularization for the two tasks.

6.3 Experiments

6.3.1 Datasets and Metrics

Following previous works, MixATIS and MixSNIPS [8, 20, 58] are taken as testbeds. MixATIS includes 13,162 utterances for training, 756 ones for validation and 828 ones for testing. MixSNIPS includes 39,776 utterances for training, 2,198 ones for validation and 2,199 ones for testing.

As for evaluation metrics, following previous works, we adopt accuracy (Acc) for multiple intent detection, F1 score for slot filling, and overall accuracy for the sentence-level semantic frame parsing. Overall accuracy denotes the ratio of sentences whose intents and slots are all correctly predicted.

6.3.2 Implementation Details

Following previous works, the word and label embeddings are trained from scratch[2]. The dimensions of word embedding, label embedding, and hidden state are 256 on MixATIS, while on MixSNIPS they are 256, 128, and 256. The layer number of all

[2]Due to space limitation, the experiments using pre-trained language model as the encoder are presented in Appendix.

Models	MixATIS			MixSNIPS		
	Overall(Acc)	Slot (F1)	Intent(Acc)	Overall(Acc)	Slot(F1)	Intent(Acc)
Attention BiRNN [42]	39.1	86.4	74.6	59.5	89.4	95.4
Slot-Gated [16]	35.5	87.7	63.9	55.4	87.9	94.6
Bi-Model [84]	34.4	83.9	70.3	63.4	90.7	95.6
SF-ID [11]	34.9	87.4	66.2	59.9	90.6	95.0
Stack-Propagation [54]	40.1	87.8	72.1	72.9	94.2	96.0
Joint Multiple ID-SF [13]	36.1	84.6	73.4	62.9	90.6	95.1
AGIF [58]	40.8	86.7	74.4	74.2	94.2	95.1
GL-GIN [57]	43.0	88.2	76.3	73.7	94.0	95.7
Co-guiding Net (ours)	**51.3**†	**89.8**†	**79.1**†	**77.5**†	**95.1**†	**97.7**†

Table 6.1: Results comparison. † denotes our model significantly outperforms baselines with $p < 0.01$ under t-test.

GNNs is 2. Adam [34] is used to train our model with a learning rate of $1e^{-3}$ and a weight decay of $1e^{-6}$. As for the coefficients Eq.6.14, γ is 0.9 on MixATIS and 0.8 on MixSNIPS; on both datasets, β_I is $1e^{-6}$ and β_S is $1e^{0}$. The model performing best on the dev set is selected then we report its results on the test set. All experiments are conducted on RTX 6000.

6.3.3 Main Results

The performance comparison of Co-guiding Net and baselines are shown in Table 6.1, from which we have the following observations:

(1) Co-guiding Net gains significant and consistent improvements on all tasks and datasets. Specifically, on MixATIS dataset, it overpasses the previous state-of-the-art model GL-GIN by 19.3%, 1.8%, and 3.7% on sentence-level semantic frame parsing, slot filling, and multiple intent detection, respectively; on MixSNIPS dataset, it overpasses GL-GIN by 5.2%, 1.2% and 2.1% on sentence-level semantic frame parsing, slot filling and multiple intent detection, respectively. This is because our model achieves the mutual guidances between multiple intent detection and slot filling, allowing the two tasks to provide crucial clues for each other. Besides, our designed HSLGs and HGATs can effectively model the interactions among the semantics nodes and label nodes, extracting the indicative clues from initial predictions.

(2) Co-guiding Net achieves a larger improvement on multiple intent detection than slot filling. The reason is that except for the guidance from multiple intent detection to slot filling, our model also achieves the guidance from slot filling to multiple intent detection, while previous models all ignore this. Besides, previous methods model the semantics-label interactions by homogeneous graph and GAT, limiting the performance.

Models	MixATIS			MixSNIPS		
	Overall(Acc)	Slot (F1)	Intent(Acc)	Overall(Acc)	Slot(F1)	Intent(Acc)
Co-guiding Net	**51.3**	**89.8**	**79.1**	**77.5**	**95.1**	**97.7**
w/o S2I-guidance	47.7 (↓3.6)	88.8 (↓1.0)	77.1 (↓2.0)	76.6 (↓0.9)	94.7 (↓0.4)	96.9 (↓0.8)
w/o I2S-guidance	47.7 (↓3.6)	88.7 (↓1.1)	77.5 (↓1.6)	76.5 (↓1.0)	94.9 (↓0.2)	97.5 (↓0.2)
w/o relations	46.0 (↓5.3)	88.3 (↓1.5)	77.8 (↓1.3)	76.3 (↓1.2)	94.7 (↓0.5)	97.2 (↓0.4)
+ Local Slot-aware GAT	51.1 (↓0.2)	89.4 (↓0.4)	79.0 (↓0.1)	75.9 (↓1.6)	94.7 (↓0.4)	96.4 (↓1.4)

Table 6.2: Results of ablation experiments.

Differently, our model uses the heterogeneous semantics-label graphs to represent different relations among the semantic nodes and the label nodes, then applies the proposed HGATs over the graphs to achieve the interactions. Consequently, their performances (especially on multiple intent detection) are significantly inferior to our model.

(3) The improvements in overall accuracy are much sharper. We suppose the reason is that the achieved mutual guidances make the two tasks deeply coupled and allow them to stimulate each other using their initial predictions. For each task, its final outputs are guided by its and another task's initial predictions. By this means, the correct predictions of the two tasks can be better aligned. As a result, more test samples get correct sentence-level semantic frame parsing results, and then overall accuracy is boosted.

6.3.4 Model Analysis

We conduct a set of ablation experiments to verify the advantages of our work from different perspectives, and the results are shown in Table 6.2.

6.3.4.1 Effect of Slot-to-Intent Guidance

One of the core contributions of our work is achieving the mutual guidances between multiple intent detection and slot filling, while previous works only leverage the one-way message from intent to slot. Therefore, compared with previous works, one of the advantages of our work is modeling the slot-to-intent guidance. To verify this, we design a variant termed *w/o S2I-guidance* and its result is shown in Table 6.2. We can observe that Intent Acc drops by 2.0% on MixATIS and 0.8% on MixSNIPS. Moreover, Overall Acc drops more significantly: 3.6% on MixATIS and 0.9% on MixSNIPS. This proves that the guidance from slot to intent can effectively benefit multiple intent

detection, and achieving the mutual guidances between the two tasks can significantly improve Overall Acc.

Besides, although both of *w/o S2I-guidance* and GL-GIN only leverage the one-way message from intent to slot, *w/o S2I-guidance* outperforms GL-GIN by large margins. We attribute this to our proposed heterogeneous semantics-label graphs and heterogeneous graph attention networks, whose advantages are verified in Sec. 6.3.4.3.

6.3.4.2 Effect of Intent-to-Slot Guidance

To verify the effectiveness of intent-to-slot guidance, we design a variant termed *w/o I2S-guidance* and its result is shown in Table 6.2. We can find that the intent-to-slot guidance has a significant impact on performance. Specifically, *w/o I2S-guidance* cause nearly the same extent of performance drop on Overall Acc, proving that both of the intent-to-slot guidance and slot-to-intent guidance are indispensable and achieving the mutual guidances can significantly boost the performance.

6.3.4.3 Effect of HSLGs and HGATs

In this paper, we design two HSLGs: (i.e., S2I-SLG, I2S-SLG) and two HGATs (i.e., S2I-HGAT, I2S-HGAT). To verify their effectiveness, we design a variant termed *w/o relations* by removing the relations on the two HSLGs. In this case, S2I-SLG/I2S-SLG collapses to a homogeneous graph, and S2I-HGAT/I2S-HGAT collapses to a general GAT based on multi-head attentions. From Table 6.2, we can observe that *w/o relations* obtains dramatic drops on all metrics on both datasets. The apparent performance gap between *w/o relations* and Co-guiding Net verifies that (1) our proposed HSLGs can effectively represent the different relations among the semantics nodes and label nodes, providing appropriate platforms for modeling the mutual guidances between the two tasks; (2) our proposed HGATs can sufficiently and effectively model interactions between the semantics and indicative label information via achieving the relation-specific attentive information aggregation on the HSLGs.

Besides, although *w/o relations* obviously underperforms Co-guiding Net, it still significantly outperforms all baselines. We attribute this to the fact that our model achieves the mutual guidances between the two tasks, which allows them to promote each other via cross-task correlations.

6.3.4.4 Effect of I2S-HGAT for Capturing Local Slot Dependencies

[57] propose a Local Slot-aware GAT module to alleviate the uncoordinated slot problem (e.g., *B-singer* followed by *I-song*) [88] caused by the non-autoregressive

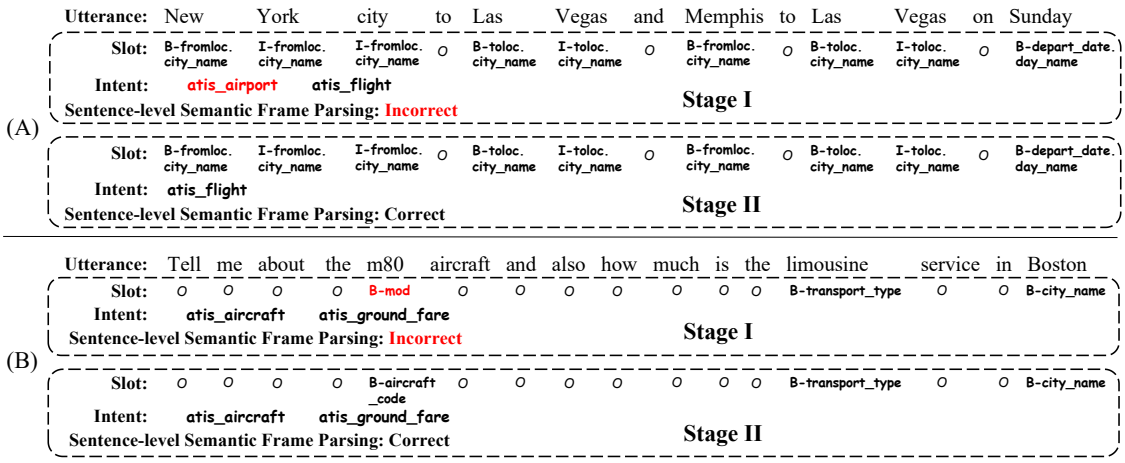


Figure 6.6: Case study of slot-to-intent guidance (A) and intent-to-slot guidance (B). Red color denotes error.

fashion of slot filling. And the ablation study in [57] proves that this module effectively improves the slot filling performance by modeling the local dependencies among slot hidden states. In their model (GL-GIN), the local dependencies are modeled in both of the local slot-aware GAT and subsequent global intent-slot GAT. We suppose the reason why GL-GIN needs the local Slot-aware GAT is that the global intent-slot GAT in GL-GIN cannot effectively capture the local slot dependencies. GL-GIN's global slot-intent graph is homogeneous, and the GAT working on it treats the slot semantics nods and the intent label nodes equally without discrimination. Therefore, each slot hidden state receives indiscriminate information from both of its local slot hidden states and all intent labels, making it confusing to capture the local slot dependencies. In contrast, we believe our I2S-HLG and I2S-HGAT can effectively capture the slot local dependencies along the specific *slot_semantics_dependencies* relation, which is modeled together with other relations. Therefore, our Co-guiding Net does not include another module to capture the slot local dependencies.

To verify this, we design a variant termed *+Local Slot-aware GAT*, which is implemented by augmenting Co-guiding Net with the Local Slot-aware GAT [57] located after the Intent-aware BiLSTM$_s$ (the same position with GL-GIN). And its result is shown in Table 6.2. We can observe that not only the Local Slot-aware GAT does not bring improvement, it even causes performance drops. This proves that our I2S-HGAT can effectively capture the local slot dependencies.

6.3.5 Case Study

To demonstrate how our model allows the two tasks to guide each other, we present two cases in Fig. 6.6.

Slot-to-Intent Guidance From Fig. 6.6 (A), we can observe that in the first stage, all slots are correctly predicted, while multiple intent detection obtains a redundant intent `atis_airport`. In the second stage, our proposed S2I-HGAT operates on S2I-HLG. It aggregates and analyzes the slot label information from the slot predictions of the first stage, extracting the indicative information that most slot labels are about `city_name` while no information about *airport* is mentioned. Then this beneficial guidance information is passed into intent semantics nodes whose representations are then fed to the intent decoder for prediction. In this way, the guidance from slot filling helps multiple intent detection predict correctly.

Intent-to-Slot Guidance In the example shown in Fig. 6.6 (B), in the first stage, correct intents are predicted, while there is an error in the predicted slots. In the second stage, our proposed I2S-HGAT operates on I2S-HLG. It comprehensively analyzes the indicative information of *airecraft* from both of slot semantics node *aircraft* and intent label node `atis_aircraft`. Then this beneficial guidance information is passed into the slot semantics of *m80*, whose slot is therefore correctly inferred.

6.4 Summary

In this paper, we propose a novel Co-guiding Net based on a two-stage framework that allows the two tasks to guide each other in the second stage using the predicted labels at the first stage. To represent the relations among the semantics node and label nodes, we propose two heterogeneous semantics-label graphs, and two heterogeneous graph attention networks are proposed to model the mutual guidances between intents and slots. Experiment results on benchmark datasets show that our model significantly outperforms previous models.

Chapter 7

ReLa-Net

7.1 Introduction

Intent detection and slot filling [74] are two fundamental tasks in spoken language understanding (SLU) [103] which is a core component in the dialog systems. In recent years, a bunch of joint models [11, 16, 36, 42, 43, 54, 56, 106] have been proposed to tackle single intent detection and slot filling at once via modeling their correlations.

In real-world scenarios, a single utterance usually expresses multiple intents. To handle this, multi-intent SLU [32] is explored and [13] first propose to jointly model the multiple intent detection and slot filling in a multi-task framework. [58] and [57] further utilize graph attention networks (GATs) [78] which model the interactions between the embeddings of predicted intent labels and the semantic hidden states of slot filling task to leverage the dual-task correlations in an explicit way. And significantly improvements have been achieved.

However, we argue that previous works are essentially limited by ignoring the rich topological structures and relations in the joint label space of the two tasks because they treat the embeddings of intent labels and slot labels as individual parameter vectors to be learned. Based on our observation, there are two kinds of potential topological structures in the joint label space. (1) The global statistical dependencies among the labels based on their co-occurrence patterns, which we discover are widely-existing phenomenons. Fig. 7.1 shows an example of label co-occurance. (2) The hierarchies in the slot labels. Although there is no officially predefined slot label hierarchy in the datasets, we discover and define two kinds of hierarchies. An example is shown in Fig. 7.2. Firstly, it is intuitive that there is an intrinsic dependency between a B- slot label and its I- slot label: in the slot sequence of an utterance, an I- slot label must occur after its B- slot label, and an I- slot cannot occur solely. Therefore, in the label space, there should be a hierarchy between a B- slot and its I-

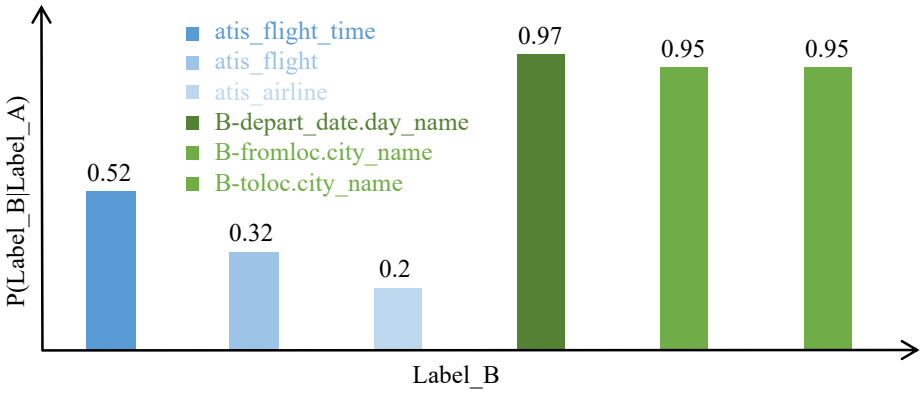


Figure 7.1: Illustration of top-3 high-probability occurring intent (in blue) and slot (in green) labels when `B-depart_date.date_relative` (Label_A) occurs in the training samples of MixATIS dataset.

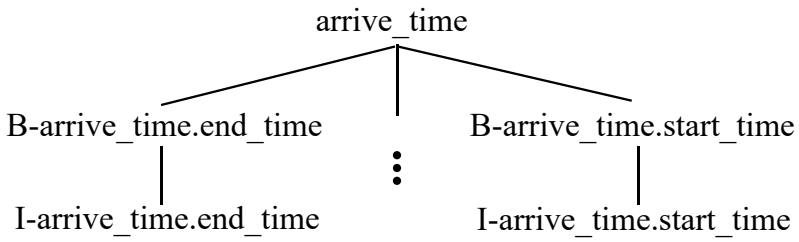


Figure 7.2: Illustration of the hierarchy between the pseudo label `arrive_time` and B- slot labels, and the hierarchy between the B- slot label and its I- label.

slot, while existing methods regard all slot labels as independent ones. Secondly, we discover that some B- slot labels share the same prefix, which indicates their shared semantics. Therefore, the prefix can be regarded as a pseudo slot label connecting these slot labels. We believe the above statistical dependencies and hierarchies can help to capture the inner- and inter-task label correlations which benefit the joint task reasoning. Thus in this paper, we focus on exploiting the label topologies and relations for tackling the joint task.

Besides, previous models decode the two tasks' hidden states independently without leveraging the dual-task correlations. This causes the misalignment of the two tasks' correct predictions. As a result, the performance (Overall Accuracy) on sentence-level semantic frame parsing is much worse than the two tasks. Therefore, we argue that the label embeddings conveying the dual-task correlative information should be leveraged in the decoders to guide the decoding process.

To overcome the above challenges, in this paper, we first construct a heterogeneous label graph (HLG) including both the global statistical dependencies and slot label hierarchies, based on which we define a bunch of edge types to represent the topological

structures and rich relations among the labels. Then we propose a novel model termed **R**ecurrent h**e**terogeneous **L**abel m**a**tching **Net**work (ReLa-Net) to capture the beneficial label correlative information from HLG and sufficient leverage them for tackling the joint task. To capture the label correlative information from HLG, we design a Heterogeneous Label Graph Transformations (HLGT) module, which conducts relation-specific information aggregation among the label nodes. We design a recurrent dual-task interacting module to leverage label correlations for semantics-label interactions. At each step, it takes both tasks' semantic hidden states and label knowledge generated at the previous step as input. Then sequence-reasoning BiLSTM [21] and GATs [78] are utilized for interactions. As for decoding, we design a novel label-aware inter-dependent decoding mechanism that takes both the hidden states and label embeddings as input and measures their correlation scores in the joint label embedding space. By this means, the joint label embedding space serves as a bridge to connect the two tasks' decoding processes which then can be guided by the dual-task inter-dependencies conveyed in the learned label embeddings. We evaluate our ReLa-Net[1] on benchmark datasets and the results show that ReLa-Net achieves new state-of-the-art performance. Further analysis proves that our method can capture nontrivial correlations among the two tasks' labels and effectively leverage them to tackle the joint task.

7.2 Problem Definition

Given a utterance $\mathcal{U}$, the task of joint multiple intent detection and slot filling aims to output a intent label set $O^I = \{o_1^I, ..., o_m^I\}$ and a slot label sequence $O^S = \{o_1^S, ..., o_n^S\}$, where n is the length of $\mathcal{U}$ and m is the number of intents expressed in $\mathcal{U}$.

7.3 Heterogeneous Label Graph

Mathematically, the HLG can be denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A}, \mathcal{R})$, where $\mathcal{V}$ is the set of nodes, $\mathcal{E}$ is the set of edges, $\mathcal{A}$ is the set of node types and $\mathcal{R}$ is the set of relations or edge types. As shown in Fig. 7.3, there are three types of nodes: intent nodes, slot nodes, and pseudo nodes, which correspond to the intent labels, slot labels, and pseudo slot labels.

In HLG, there are two kinds of topologies, which correspond to statistical and hierarchical dependencies, respectively. To represent and capture these dependencies,

[1]https://github.com/XingBowen714/ReLa-Net

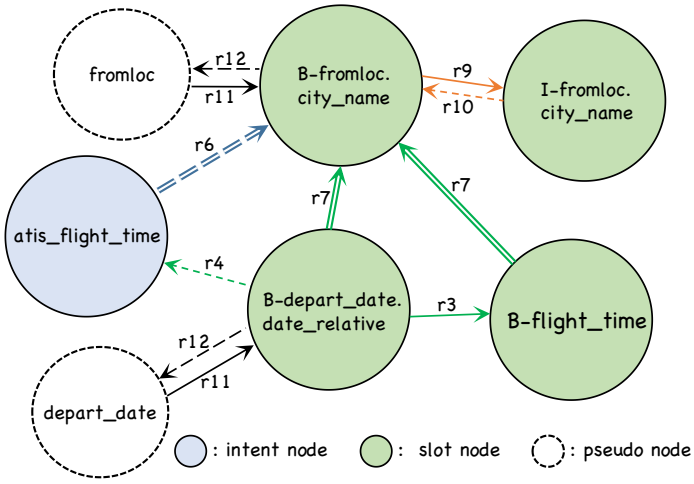


Figure 7.3: Illustration of a snippet from the whole HLG.

we define 12 relations on HLG: $r_1 \sim r_8$ for statistical dependencies and $r_9 \sim r_{12}$ for hierarchical dependencies, whose definitions are shown in Table 7.1. And $|\mathcal{V}| = N^I + N^S + N^P$, in which N^I, N^S and N^P denote the numbers of intent nodes, slot nodes and pseudo nodes.

$r_1 \sim r_8$ are based on the conditional probability between two labels, and two thresholds (λ_1, λ_2) are used to determine whether there is statistical dependency or strong statistical dependency from node i to node j regarding $P(j|i)$. A high value of $P(j|i)$ denotes that when label i occurs, there always exists label j in the sample. Namely, label j can be potentially deduced when label i is predicted. In this case, an edge is established from node i to node j with the corresponding relation. Note that in each sample, most slot labels are `O` which denotes the word has no meaningful slot. Considering the slot label `O` cannot provide valuable clues, although it has high conditional probabilities (always 1.0) with other labels. Therefore, the slot node `O` is set as an isolated node in HLG.

$r_1 \sim r_8$ are based on the hierarchies we discovered in slot labels. A B- slot node can connect and interact with many nodes, including intent nodes, other B- slot nodes, its parent pseudo node, and its I- slot node. Differently, an I- slot node only connects with its B- slot nodes because there is a natural affiliation relation between a pair of B- and I- slot labels: an I- slot label can only appear after its B-slot label. As for the pseudo labels we defined, there is a parent-child relationship between the parent pseudo labels and their child B- slot labels. In HLG, a pseudo label works like an information station, which allows its children (B- slot nodes) to share their semantics and features.

$\phi(e_{ij})$	$\tau(i)$	$\tau(j)$	$P(j\|i)$
r_1: i2i_stat_dep	intent	intent	$\lambda_1 \leq p < \lambda_2$
r_2: i2s_stat_dep	intent	slot-B	$\lambda_1 \leq p < \lambda_2$
r_3: s2s_stat_dep	slot-B	slot-B	$\lambda_1 \leq p < \lambda_2$
r_4: s2i_stat_dep	slot-B	intent	$\lambda_1 \leq p < \lambda_2$
r_5: i2i_stat_strong_dep	intent	intent	$p \geq \lambda_2$
r_6: i2s_stat_strong_dep	intent	slot-B	$p \geq \lambda_2$
r_7: s2s_stat_strong_dep	slot-B	slot-B	$p \geq \lambda_2$
r_8: s2i_stat_strong_dep	slot-B	intent	$p \geq \lambda_2$
r_9: b2i_hierarchy	slot-B	slot-I	\
r_{10}: i2b_hierarchy	slot-I	slot-B	\
r_{11}: parent2child_hierarchy	pseudo	slot-B	\
r_{12}: child2parent_hierarchy	slot-B	pseudo	\

Table 7.1: Illustration of the definition of the 12 relations in HLG. $\phi(e_{ij})$ denotes the relation of edge e_{ij} (from i to j). $\tau(i)$ denotes what kinds of label the node i corresponds to. $P(j|i)$ denotes the conditional probability of a sample having a label j when it has a label j.

7.4 ReLa-Net

Overview. The architecture of our ReLa-Net is shown in Fig. 7.4. Firstly, the initial label embeddings containing beneficial label correlations are generated by the proposed Heterogeneous Label Graph Transformations (HLGT) module, and the initial semantic hidden states are generated by the Self-Attentive Semantics Encoder. Then we allow the two tasks to interact recurrently. At each time step, for each task, the semantic hidden states and the label knowledge of both tasks are fused, and their correlations are learned in a BiLSTM and then a GAT. Then the designed Label-Aware Inter-Dependent Decoding mechanism produces the estimated labels via measuring the correlations between the hidden state and the label embeddings. The obtained labels are fed to the designed Dynamically-Masked Heterogeneous Label Graph Transformations (DM-HLGT) module, which dynamically derives the sample-specific label knowledge via operating on a DM subgraph of HLG. Then the obtained label knowledge is used at the next time step. Finally, after T step, the last produced labels are taken as the final predictions.

Next, we introduce the details of each module.

7.4.1 HLGT

To capture the correlations among the intent and slot labels, inspired by [62, 92], we conduct relation-specific graph transformations to achieve information aggregation on

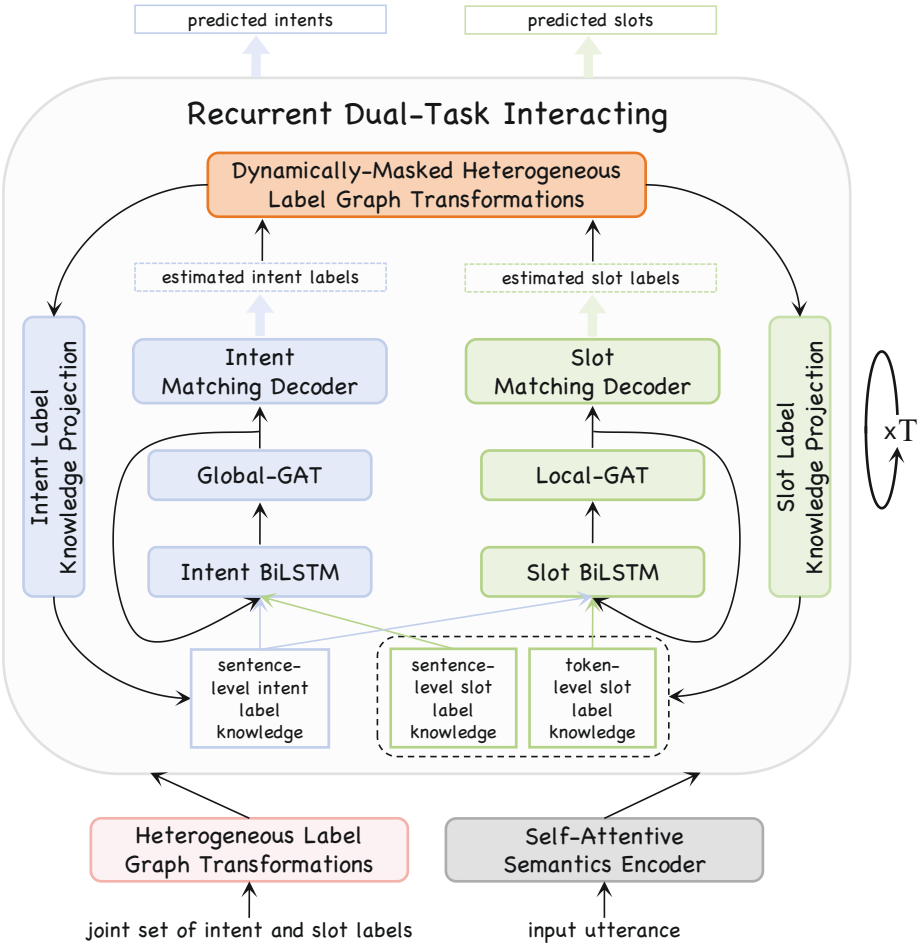


Figure 7.4: The network architecture of our ReLa-Net.

HLG:

$$e_i^l = \text{ReLU}(W_1 e_i^{l-1} + \sum_{r \in \mathcal{R}} \sum_{j \in \mathcal{N}_i^r} \frac{1}{|\mathcal{N}_i^r|} W_1^r e_j^{l-1}), \tag{7.1}$$

where l denotes the layer number; $\mathcal{N}_i^r$ denotes the set of node i's neighbors which are connected to it with r-type edges; W_1 is self weight matrix and W_1^r is relation-specific weight matrix. The initial node representations are obtained from an initialized matrix which is trained with the model.

After L layers, the initial intent label embedding matrix E^I and slot label embedding matrix E^S are obtained. Although E^I and E^S contain beneficial correlative information, the HLG is globally built on the whole train set. Therefore, they are not flexible enough for every sample, whose labels may form a local subgraph on HLG. And we believe capturing the sample-specific label correlations can further enhance the reasoning by discovering potential neighbor labels. To this end, we propose the Dynamically-Masked HLGT, which will be introduced in Sec. 7.4.3.4.

7.4.2 Self-Attentive Semantics Encoder

Following previous works, we adopt the self-attentive encoder [57, 58] to generate the initial semantic hidden states. It includes a BiLSTM [21] and a self-attention mechanism [77]. The input utterance's word embeddings are fed to the BiLSTm and self-attention separately. Then the two streams of word representations are concatenated as the output semantic hidden states.

7.4.3 Recurrent Dual-Task Interacting

7.4.3.1 Semantics-Label Interaction

BiLSTM has been proven to be capable of sequence reasoning [30, 76, 111]. In this paper, we utilize two BiLSTMs for intent and slot to achieve the fusion and interactions among the semantics and label knowledge of both tasks. Specifically, the two BiLSTMs can be formulated as:

$$\begin{aligned} \hat{h}_t^{I,i} &= \text{BiLSTM}_\text{I}(h_t^{I,i} \| K_t^I \| K_t^S, \hat{h}_t^{I,i-1}, \hat{h}_t^{I,i+1}), \\ \hat{h}_t^{S,i} &= \text{BiLSTM}_\text{S}(h_t^{S,i} \| K_t^I \| K_t^{S,i}, \hat{h}_t^{S,i-1}, \hat{h}_t^{S,i+1}), \end{aligned} \tag{7.2}$$

where $\|$ denotes concatenation operation.

For Intent BiLSTM (BiLSTM_I), the input is the concatenation of intent semantic hidden states $h_t^{I,i}$, sentence-level intent label knowledge K_t^I and the sentence-level slot label knowledge K_t^S, where t denotes the step number of recurrent dual-task interacting. Here we use sentence-level label knowledge because multiple intent detection is on sentence-level. For Slot BiLSTM (BiLSTM_S), the input is the concatenation of slot semantic hidden states $h_t^{S,i}$, sentence-level intent label knowledge K_t^I and the token-level slot label knowledge $K_t^{S,i}$. Here we use token-level slot label knowledge because slot filling is a token-level task. And we use sentence-level intent label knowledge here because it can provide indicative clues of potential slot labels regarding the captured inter-task dependencies among intent and slot labels.

At the first step, there is no available label knowledge, so the inputs of the two BiLSTMs are the initial hidden states generated in Sec. 7.4.2. How to obtain $K_t^I, K_t^S, K_t^{S,i}$ is depicted in Sec. 7.4.3.5.

7.4.3.2 Graph Attention Networks

Global-GAT Since multiple intent detection is a sentence-level task, we believe it is beneficial to further capture the global dependencies among the words. We first construct a fully-connected graph on the input utterance. There are n nodes in this

graph and each one corresponds to a word. And we adopt the GAT [78] for information aggregation. The initial representation of node i is $\hat{h}_t^{I,i}$ generated by BiLSTM_I. After L layers, we obtain $\widetilde{H}_t^I = \{\widetilde{h}_t^{I,i}\}_{i=1}^N$.

Local-GAT Since slot filling is on token-level, we adopt a Local-GAT to capture the token-level local dependencies.The Local-GAT works on a locally-connected graph where node i is connected to nodes $\{i-w, ..., i+w\}$, where sliding window size w is a hyper-parameter. The initial representation of node i is $\hat{h}_t^{S,i}$ generated by BiLSTM_S. After L layers, we obtain $\widetilde{H}_t^S = \{\widetilde{h}_t^{S,i}\}_{i=1}^N$. And we adopt the GAT [78] for information aggregation.

7.4.3.3 Label-Aware Inter-Dependent Decoding

In this work, we design a novel label-aware inter-dependent decoding mechanism (shown in Fig. 7.5) to leverage the dual-task correlative information conveyed in the learned label embeddings to guide the decoding process. Concretely, the hidden state is first projected into the joint label embedding space, and then the dot products are conducted between it and all label embeddings of a specific task to obtain the correlation score vector. The larger correlation score indicates the shorter distance between the hidden state's projection and the label embedding. Thus the hidden state more likely belongs to the corresponding class. In this way, the joint label embedding space serves as a bridge that connects the decoding processes of the two tasks, and the beneficial correlative information conveyed in the learned label embeddings can guide the inter-dependent decoding for the two tasks. Next, we introduce the intent and slot decoders in detail.

Intent Decoder Following previous works, we conduct token-level multi-label classification. Firstly, the intent hidden state $\widetilde{h}_t^{I,i}$ is projected into the joint label embedding space via an MLP. Then the correlation score vector $C_t^{I,i}$ is calculated via dot products between it and all intent label embeddings. This process can be formulated as:

$$C_t^{I,i} = (W_I(\text{LeakyReLU}(W_h^I \widetilde{h}_t^{I,i} + b_h)) + b_I) \ \widehat{E}_t^I, \tag{7.3}$$

where $C_t^{I,i}$ is a N^I-dimensional vector; $\widehat{E}_t^I$ is the sample-specific intent label embedding matrix at step t; W_* and b_* are model parameters.

Then $C_t^{I,i}$ is fed to the sigmoid function to obtain the token-level intent probability vector. And a threshold (0.5) is used to select the intents. Finally, we obtain the predicted sentence-level intents by voting for all tokens' intents [57].

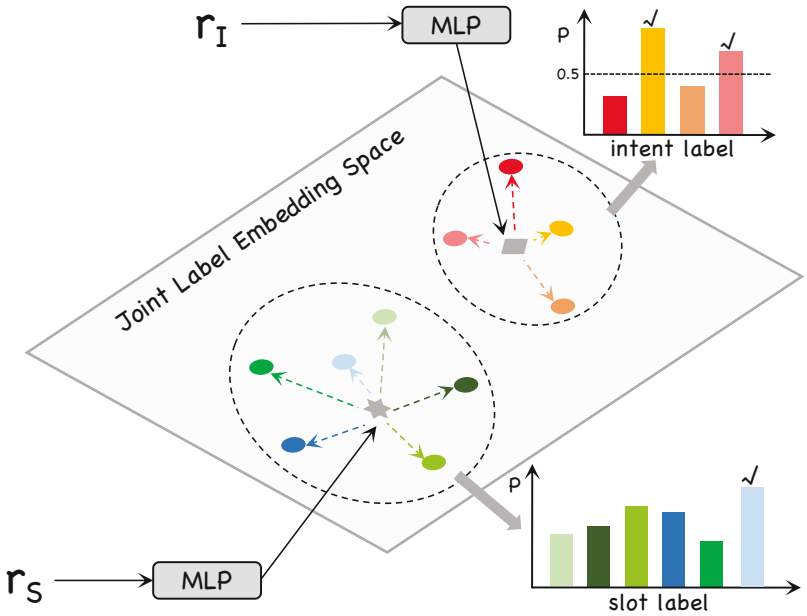


Figure 7.5: Illustration of our proposed label-aware co-decoding mechanism. Blue/green circles denote slot labels and red/yellow circles denote intent labels. The dash arrow denotes the distance between the hidden state projection and the label. The shorter the distance, the greater the probability that the label is correct. ✓denotes that the label is selected as the prediction.

Slot Decoder Following [57], we conduct parallel decoding. Similar to Eq. 7.3, the slot correlation score vector $C_t^{S,i}$ is obtained by:

$$C_t^{S,i} = (W_S(\text{LeakyReLU}(W_h^S \widetilde{h}_t^{S,i} + b_S)) + b_S)\ \widehat{E}_t^S, \tag{7.4}$$

where $C_t^{S,i}$ is a N^S-dimensional vector; $\widehat{E}_t^S$ is the sample-specific slot label embedding matrix at step t; W_* and b_* are model parameters.

Then $C_t^{I,i}$ is fed to the softmax function to obtain the slot probability distribution. Finally, the argmax function is used to select the predicted slot.

7.4.3.4 DM-HLGT

After decoding, we obtain the predicted intents and slots at step t. To capture the sample-specific label correlations, which we believe further benefit the reasoning in the next step, we first construct a DM subgraph of HLG for each sample. The process is simple: the nodes of predicted intent labels and slot labels and their first-order neighbors are reserved, while other nodes on HLG are masked out. Then relation-specific graph transformations are conducted on this graph to achieve information aggregation, whose formulation is similar to Eq. 7.1. At the first step, the $\widehat{E}_0^I$ and $\widehat{E}_0^S$ are E^I and E^S.

7.4.3.5 Label Knowledge Projection

To provide label knowledge for next time step, the predicted labels should be projected into vectors. In Sec. 7.4.3.1, we use three kinds of label knowledge: $K_t^I, K_t^S, K_t^{S,i}$. K_t^I and K_t^S are sentence-level intent and slot label knowledge, respectively. K_t^I is the sum of label embeddings of the predicted intents and K_t^S is the sum of label embeddings of the predicted slots while the `O` slot is excluded. $K_t^{S,i}$ is token-level slot label knowledge, which is the label embedding of the predicted slot of token t.

7.4.4 Optimization

Following previous works, the standard loss function for intent task ($\mathcal{L}_I$) and slot task ($\mathcal{L}_S$) are:

$$\begin{aligned} l(\hat{y}, y) &= \hat{y}\log(y) + (1-\hat{y})\log(1-y), \\ \mathcal{L}_I &= \sum_{t=1}^{T}\sum_{i=1}^{n}\sum_{j=1}^{N^I} l(\hat{y}_i^I[j], y_i^{I,t}[j]), \\ \mathcal{L}_S &= \sum_{t=1}^{T}\sum_{i=1}^{n}\sum_{j=1}^{N^S} \hat{y}_i^S[j]\log(y_i^{[S,t]}[j]). \end{aligned} \tag{7.5}$$

Besides, we design a constraint loss ($\mathcal{L}_{cst}$) to push ReLa-Net to generate better label distributions at step t than step $t-1$:

$$\mathcal{L}_{cons}^I = \sum_{t=2}^{T}\sum_{i=1}^{n}\sum_{j=1}^{N^I} \hat{y}_i^I[j]\ \max(0, y_i^{[I,t-1]}[j] - y_i^{[I,t]}[j]). \tag{7.6}$$

And $\mathcal{L}_{cst}^S$ can be derived similarly.

Then the final training objective is:

$$\mathcal{L} = \gamma_I\left(\mathcal{L}_I + \beta_I\mathcal{L}_{cst}^I\right) + \gamma_S\left(\mathcal{L}_S + \beta_S\mathcal{L}_{cst}^S\right), \tag{7.7}$$

where $\gamma_I = 0.1$ and $\gamma_S = 0.9$ balance the two tasks; $\beta_I = 0.01$ and $\beta_S = 1.0$ balance the standard loss and constraint loss for the two tasks.

7.5 Experiments

7.5.1 Settings

Datasets. Following previous works, we conduct experiments on two benchmarks: MixATIS and MixSNIPS [8, 20, 58]. In MixATIS, the split of train/dev/test set

is 13162/756/828 (utterances). In MixSNIPS, the split of train/dev/test set is 39776/2198/2199 (utterances).

Evaluation Metrics. Following previous works, we evaluate multiple intent detection using accuracy (Acc), slot filling using F1 score, and sentence-level semantic frame parsing using overall Acc. Overall Acc denotes the ratio of utterances for which both intents and slots are predicted correctly.

Implementation Details. Following previous works, the word and label embeddings are randomly initialized and trained with the model. Due to limited space, the experiments using pre-trained language model is presented in the Appendix. The dimension of the word/label embedding is 128 on MixATIS and 256 on MixSNIPS. For HLG, $\lambda_1 = 0.4$ and $\lambda_2 = 0.9$. The hidden dim is 200. The max layer number of all GNNs is 2. Max time step T is 2. We adopt Adam [34] optimizer to train our model using the default setting. For all experiments, we select the best model on the dev set and report its results on the test set. Our source code will be released.

7.5.2 Main Results

We compare our model with: (1) Attention BiRNN [42]; (2) Slot-Gated [16]; (3) Bi-Model [84]; (4) SF-ID Network [11]; (5) Stack-Propagation [54]; (6) Joint Multiple ID-SF [74]; (7) AGIF [58]; (8) GL-GIN [57]. Table 7.2 lists the results on the test sets. We can observe that:

1. Our Rela-Net consistently outperforms all baselines by large margins on all datasets and tasks. Specifically, compared with Gl-GIN, the previous best model, ReLa-Net achieves an absolute improvement of 9.2% in terms of Overall Acc on MixATIS, a relative improvement of over 20%.

2. ReLa-Net gains larger improvements on the intent task than slot task. The reason is that ReLa-Net is the first model to allow the two tasks to interact with each other, while previous models only leverage the predicted intents to guide slot filling.

3. Generally, the results on overall Acc are obviously worse than slot F1 and intent Acc, indicating that it is hard to align the correct prediction of intents and slots. Compared with baselines, our ReLa-Net achieves especially significant improvements on overall Acc and effectively reduces the performance gap between overall Acc and slot F1/intent Acc. This can be attributed to the fact that our model can capture sufficient and beneficial label correlations from HLG, and the designed label-aware inter-dependent decoding mechanism can leverage the label correlations to guide decoding, making the two tasks' decoding processes correlative. As a result, the correct predictions of the two tasks are better aligned.

Models	MixATIS			MixSNIPS		
	Overall (Acc)	Slot (F1)	Intent (Acc)	Overall (Acc)	Slot (F1)	Intent (Acc)
Attention BiRNN	39.1	86.4	74.6	59.5	89.4	95.4
Slot-Gated	35.5	87.7	63.9	55.4	87.9	94.6
Bi-Model	34.4	83.9	70.3	63.4	90.7	95.6
SF-ID	34.9	87.4	66.2	59.9	90.6	95.0
Stack-Propagation	40.1	87.8	72.1	72.9	94.2	96.0
Joint Multiple ID-SF	36.1	84.6	73.4	62.9	90.6	95.1
AGIF	40.8	86.7	74.4	74.2	94.2	95.1
GL-GIN	43.0	88.2	76.3	73.7	94.0	95.7
ReLa-Net (ours)	**52.2**	**90.1**	**78.5**	**76.1**	**94.7**	**97.6**

Table 7.2: Main results. ReLa-Net obtains statistically significant improvements over baselines with $p < 0.01$.

4. ReLa-Net gains more significant improvements on MixATIS dataset than MixSNIPS. The reason is that MixATIS dataset includes much more labels than MixSNIPS: MixATIS includes 17 intent labels and 117 slot labels, while MixSNIPS only include 7 intent labels and 72 slot labels. More labels in MixATIS dataset cause it is much harder for correct predictions, which can be proved by the lower Overall Acc scores of all models on MixATIS than MixSNIPS. However, the core strength of our model is leveraging the label topologies and relations. Therefore, more labels in MixATIS dataset result in larger improvements.

7.5.3 Ablation Study

We conduct two groups of ablation experiments on HLG and ReLa-Net to verify the necessities of their components. Table 7.3 shows the results.

HLG. ***w/o stat_dep*** denotes the statistical dependencies are removed, which means there are no edges between intent and slot label nodes on HLG. We can observe sharp

Variants	MixATIS		
	Overall (Acc)	Slot (F1)	Intent (Acc)
ReLa-Net	**52.2**	**90.1**	**78.5**
w/o stat_dep	45.3 (↓6.9)	87.5(↓2.6)	76.9(↓1.6)
w/o hierarchy	48.4 (↓3.8)	87.9(↓2.2)	78.1(↓0.4)
w/o relation	45.1(↓7.1)	88.2(↓1.9)	75.4(↓3.1)
w/o matching	45.2(↓7.0)	88.1(↓2.0)	77.4(↓1.1)
w/o GATs	47.6(↓4.6)	88.3(↓1.8)	77.2(↓1.3)
w/o DM-HLGT	50.2(↓2.0)	89.2(↓0.9)	78.0(↓0.5)

Table 7.3: Results of ablation experiments.

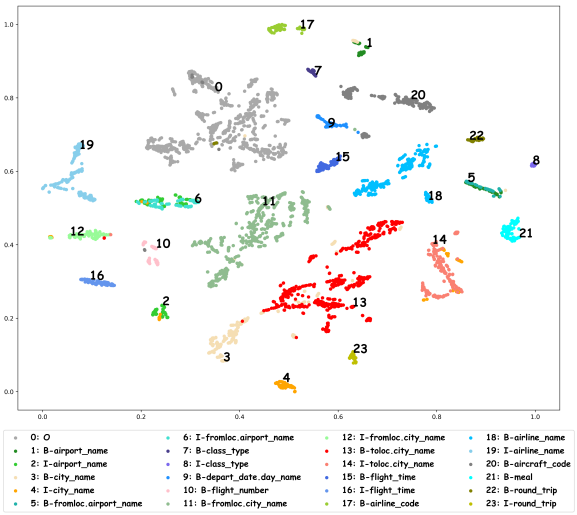


Figure 7.6: Visualization of ReLa-Net's slot clusters.

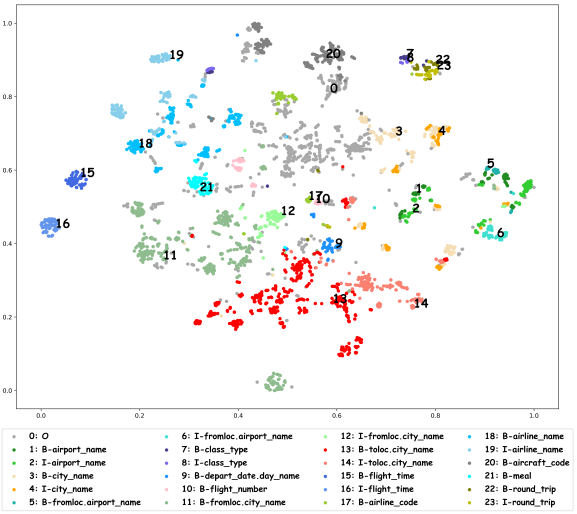


Figure 7.7: Visualization of GL-GIN's slot clusters.

drops in the performances of two tasks and the semantic frame parsing. This proves that statistical dependencies can effectively capture beneficial inter-task dependencies. ***w/o hierarchy*** denotes the slot hierarchies are removed. We can find although Intent Acc is not seriously affected, the Slot F1 and Overall Acc are both significantly reduced. This proves that slot hierarchies can effectively bring improvements via capturing the beneficial structural dependencies among slot labels. ***w/o relation*** denotes HLG collapses into a homogeneous graph without specific relations. Its poor performances prove that the defined rich relations on HLG are crucial for capturing the label correlations, which play key roles in dual-task reasoning and label-aware inter-dependent decoding. **ReLa-Net.** ***w/o matching*** denotes the designed matching

decoders in ReLa-Net are replaced with the linear decoders used in previous models. Its worse results prove that our designed matching decoders can effectively improve the decoding quality via leveraging the beneficial label correlations captured from HLG. ***w/o GATs*** denotes the Global-GAT and Local-GAT are removed. Its worse results prove that the sentence-level global dependencies and the token-level local dependencies are important for the intent task and slot task, respectively. However, even if its semantics and semantics-label interactions are only modeled via LSTM, it still outperforms all baselines. This can be attributed to the fact that ReLa-Net can learn beneficial label correlations from HLG and sufficiently leverage them for decoding. ***w/o DM-HLGT*** denotes the DM-HGT is removed. And its results verify that DM-HLGT can effectively enhance the performance via capturing the sample-specific label correlations and discovering potential neighbor labels.

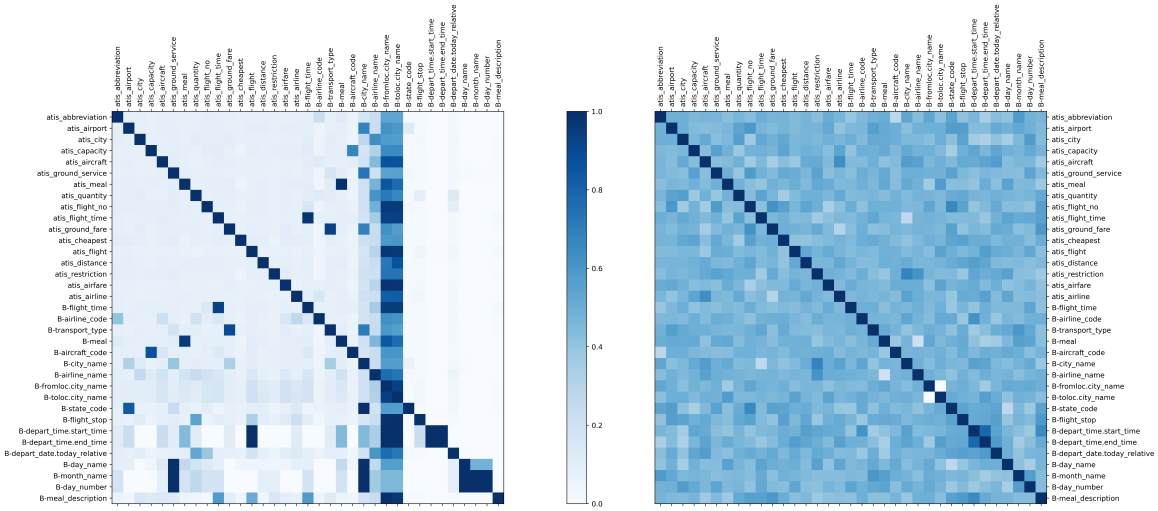
7.5.4 Visualization of Hidden State Clusters

To better understand the promising results of ReLa-Net, we perform TSNE [75] visualization on the final hidden states of the utterances words in the test set of MixATIS. It is hard to visualize intent states because a single hidden state usually corresponds to multiple intents. Therefore, we visualize the final slot hidden state of each word in the utterances. Since there are more than 100 slot labels, to simplify the visualization, we rank the slot labels regarding their frequencies in the test set, and we show the results of the top-24 labels. Besides, most slot labels in the test set are `O`, and we randomly pick 500 ones. The slot clusters visualization of our ReLa-Net is shown in Fig. 7.6.

From Fig. 7.6, we can observe that the boundaries of the clusters are clear, and the data points in the same cluster are tightly closed to each other. And we can find that the B- slot clusters and their corresponding I- slot clusters are clearly separated. Besides, there is nearly no incorrect I- slot data point. The high accuracy of I- slot labels prove that our ReLa-Net effectively resolves the uncoordinated slot problem (e.g., `B-singer` followed by `I-song`). This is attributed to the beneficial slot hierarchies we discovered and leveraged.

For comparison, we visualize GL-GIN's slot clusters in the same way, as shown in Fig. 7.7. We can observe that GL-GIN's slot clusters boundaries are not clear. Some data points of '`O`' slot are close to other clusters. And some B- clusters and their I- clusters even significantly overlap each other.

The above observations prove that our method can learn better hidden states via effectively capturing and leveraging the label correlations.



(a) Label Co-occurence Probability Matrix

(a) Learned Label Correlation Matrix

Figure 7.8: Visualization of label co-occurrence matrix and learned label correlation matrix.

7.5.5 Visualization of Label Correlations

To further look into the label correlations, we visualize the label co-occurrence probability matrix constructed from MixATIS training set and the correlation matrix of ReLa-Net's learned label embeddings, as shown in Fig. 7.8 (a) and (b). Label correlations of two labels are measured by the cosine similarity of their embeddings. For simplification, we visualize all intent labels (17 ones) and some slot labels (18 ones). From Fig.7.8 (a), we can observe that there exist many distinct co-occurrence patterns among the labels of the two tasks. The core of this work is capturing and leveraging the beneficial label correlations, and promising improvements have been observed from the experiment results, while previous works neglect this point. From Fig.7.8 (b), we can find that our ReLa-Net learns non-trivial label embeddings. From the patterns of label correlations in Fig. 7.8 (b), we can hardly observe the label co-occurrence patterns in Fig. 7.8 (a). This proves that ReLa-Net can comprehensively leverage the label co-occurrences to learn robust and effective label embeddings, rather than just mechanically memorize them.

Chapter 8

Conclusion and Future work

In this thesis, we improve natural language understanding from the perspective of multi-graph architectures, which have not been paid enough attention to. Generally, we argue that in both of single-task and multi-task scenarios, the graph architectures that combine multi-source information can be constructed for more efficient and sufficient interactions, which can enhance the inference.

For single-task scenarios, we focus on aspect or target sentiment analysis. In Chapter 5, we point out that existing works omit the integration of external knowledge and the joint consideration of both global and local syntactic information. To solve the issue, we propose KaGRMN to adaptively and recurrently incorporate the required aspect knowledge from external knowledge base. Besides, we propose DSG to integrate the two kinds of syntactic information for comprehensive reasoning. In Chapter 6, we argue that previous graph-based models cannot effectively solve the issue of noisy information aggregation and Loss of distant correlations. To this end, we propose a syntax-graph pruning model to remove the noisy word's node from both the syntax graph and the self-attention-derived semantics graph. And we combine the two pruned graphs to guarantee the graph connectivity and the distant beneficial correlations can be introduced via first-order connections.

For multi-task scenarios, we focus on two dual-task language understanding task: 1) joint dialog sentiment classification and act recognition; 2) joint multiple intent detection and slot filling. In Chapter 7, we argue that previous model for joint dialog sentiment classification and act recognition neglect the prediction-level interactions between the two tasks and the temporal relational information among the utterances. To solve the issue, we propose DARER and DARER^2 to model the sufficient temporal relational interactions among the semantics and prediction information of the two tasks. In Chapter 8, we argue that not only the guidance from intent to slot, but also the one from slot to intent is also beneficial, while previous works ignore this.

To solve the issue, we propose a two-stage framework, Co-guiding Net, which first predicts the initial label distributions of the two tasks, and then leverage the encoded prediction information to achieve mutual guidances between the two tasks. And we propose the semantics-label heterogeneous graph and heterogeneous graph attention networks to model the sufficient and relational interactions between the two tasks. In Chapter 9, we argue that previous works pay little attention to the label dependencies between the two tasks, which can be leveraged to enhance the label embeddings and then benefit dual-task reasoning. To solve the issue, we propose ReLa-Net, which can encode the dual-task label dependencies into the label embeddings and leverage them for inter-dependent dual-task decoding.

More recently, prompt-based learning has achieved stunning performances in various of NLP tasks. In the future, I will investigate the multi-task and multi-graph framework based on prompt-based learning. For multi-task learning, via designing specific prompting templates, the large language models can be guided to comprehensively understand the semantics and capture the correlations between different tasks. For multi-graph learning, multi-graph architectures can be leveraged to enhance the semantics understanding in the fine-tuning procedure, and prompt learning can be used in the inference procedure to guide the large language model to sufficiently reach its potential.

Bibliography

[1] Dario Bertero, Farhad Bin Siddique, Chien-Sheng Wu, Yan Wan, Ricky Ho Yin Chan, and Pascale Fung. Real-time speech emotion and sentiment recognition for interactive dialogue systems. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1042–1047, Austin, Texas, November 2016. Association for Computational Linguistics.

[2] Yixin Cao, Zhiyuan Liu, Chengjiang Li, Zhiyuan Liu, Juanzi Li, and Tat-Seng Chua. Multi-channel graph neural network for entity alignment. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1452–1461, Florence, Italy, July 2019. Association for Computational Linguistics.

[3] Christophe Cerisara, Somayeh Jafaritazehjani, Adedayo Oluokun, and Hoa T. Le. Multi-task dialog act and sentiment recognition on mastodon. In Emily M. Bender, Leon Derczynski, and Pierre Isabelle, editors, *COLING*, pages 745–754. Association for Computational Linguistics, 2018.

[4] Luefeng Chen, Min Li, Wanjuan Su, Min Wu, Kaoru Hirota, and Witold Pedrycz. Adaptive feature selection-based adaboost-knn with direct optimization for dynamic emotion recognition in human–robot interaction. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 5(2):205–213, 2021.

[5] Peng Chen, Zhongqian Sun, Lidong Bing, and Wei Yang. Recurrent attention network on memory for aspect sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 452–461, Copenhagen, Denmark, 2017. Association for Computational Linguistics.

[6] Zhuang Chen and Tieyun Qian. Transfer capsule network for aspect level sentiment classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 547–556, Florence, Italy, 2019. Association for Computational Linguistics.

[7] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, 2014. Association for Computational Linguistics.

[8] Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces, 2018.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, 2019. Association for Computational Linguistics.

[10] Li Dong, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou, and Ke Xu. Adaptive recursive neural network for target-dependent Twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 49–54, Baltimore, Maryland, 2014. Association for Computational Linguistics.

[11] Haihong E, Peiqing Niu, Zhongfu Chen, and Meina Song. A novel bi-directional interrelated model for joint intent detection and slot filling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5467–5471, Florence, Italy, 2019. Association for Computational Linguistics.

[12] Feifan Fan, Yansong Feng, and Dongyan Zhao. Multi-grained attention network for aspect-level sentiment classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3433–3442, Brussels, Belgium, 2018. Association for Computational Linguistics.

[13] Rashmi Gangadharaiah and Balakrishnan Narayanaswamy. Joint multiple intent detection and slot labeling for goal-oriented dialog. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational*

Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 564–569, Minneapolis, Minnesota, 2019. Association for Computational Linguistics.

[14] Deepanway Ghosal, Navonil Majumder, Rada Mihalcea, and Soujanya Poria. Exploring the role of context in utterance-level emotion, act and intent classification in conversations: An empirical study. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1435–1449, Online, August 2021. Association for Computational Linguistics.

[15] Deepanway Ghosal, Navonil Majumder, Soujanya Poria, Niyati Chhaya, and Alexander F. Gelbukh. Dialoguegcn: A graph convolutional neural network for emotion recognition in conversation. In *EMNLP/IJCNLP (1)*, pages 154–164. Association for Computational Linguistics, 2019.

[16] Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. Slot-gated modeling for joint slot filling and intent prediction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 753–757, New Orleans, Louisiana, 2018. Association for Computational Linguistics.

[17] Dilek Hakkani-Tür, Gokhan Tur, Asli Celikyilmaz, Yun-Nung Chen, Jianfeng Gao, Li Deng, and Ye-Yi Wang. Multi-domain joint semantic frame parsing using bi-directional rnn-lstm. In *Interspeech 2016*, pages 715–719, 2016.

[18] Devamanyu Hazarika, Soujanya Poria, Amir Zadeh, Erik Cambria, Louis-Philippe Morency, and Roger Zimmermann. Conversational memory network for emotion recognition in dyadic dialogue videos. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2122–2132, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[19] Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. Exploiting document knowledge for aspect-level sentiment classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 579–585, Melbourne, Australia, 2018. Association for Computational Linguistics.

[20] Charles T. Hemphill, John J. Godfrey, and George R. Doddington. The ATIS spoken language systems pilot corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27,1990*, 1990.

[21] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[22] Binxuan Huang and Kathleen Carley. Parameterized convolutional neural networks for aspect level sentiment classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1091–1096, Brussels, Belgium, 2018. Association for Computational Linguistics.

[23] Binxuan Huang and Kathleen Carley. Syntax-aware aspect level sentiment classification with graph attention networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5469–5477, Hong Kong, China, 2019. Association for Computational Linguistics.

[24] Binxuan Huang and Kathleen Carley. Syntax-aware aspect level sentiment classification with graph attention networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5469–5477, Hong Kong, China, 2019. Association for Computational Linguistics.

[25] Lianzhe Huang, Xin Sun, Sujian Li, Linhao Zhang, and Houfeng Wang. Syntax-aware graph attention network for aspect-level sentiment classification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 799–810, Barcelona, Spain (Online), 2020. International Committee on Computational Linguistics.

[26] N. Inui, T. Ebe, B. Indurkhya, and Y. Kotani. A case-based natural language dialogue system using dialogue act. In *Systems, Man, and Cybernetics, 2001 IEEE International Conference on*, volume 1, pages 193–198 vol.1, 2001.

[27] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *ICLR (Poster)*, 2017.

[28] Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. Target-dependent Twitter sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 151–160, Portland, Oregon, USA, 2011. Association for Computational Linguistics.

[29] Wenxiang Jiao, Michael R. Lyu, and Irwin King. Real-time emotion recognition via attention gated hierarchical memory network. In *AAAI*, pages 8002–8009. AAAI Press, 2020.

[30] Arzoo Katiyar and Claire Cardie. Investigating LSTMs for joint extraction of opinion entities and relations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 919–929, Berlin, Germany, 2016. Association for Computational Linguistics.

[31] Guolin Ke, Di He, and Tie-Yan Liu. Rethinking positional encoding in language pre-training. In *ICLR*, 2021.

[32] Byeongchang Kim, Seonghan Ryu, and Gary Geunbae Lee. Two-stage multi-intent detection for spoken language understanding. *Multimedia Tools and Applications*, 76(9):11377–11390, 2017.

[33] Minkyoung Kim and Harksoo Kim. Integrated neural network model for identifying speech acts, predicators, and sentiments of dialogue utterances. *Pattern Recognition Letters*, 101:1–5, 2018.

[34] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[35] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[36] Changliang Li, Liang Li, and Ji Qi. A self-attentive model with gate mechanism for spoken language understanding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3824–3833, Brussels, Belgium, 2018. Association for Computational Linguistics.

[37] Jingye Li, Hao Fei, and Donghong Ji. Modeling local contexts for joint dialogue act recognition and sentiment classification with bi-channel dynamic convolutions. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 616–626, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics.

[38] Ruifan Li, Hao Chen, Fangxiang Feng, Zhanyu Ma, Xiaojie Wang, and Eduard Hovy. Dual graph convolutional networks for aspect-based sentiment analysis. In *ACL*, pages 6319–6329, 2021.

[39] Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. DailyDialog: A manually labelled multi-turn dialogue dataset. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 986–995, Taipei, Taiwan, November 2017. Asian Federation of Natural Language Processing.

[40] Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. KagNet: Knowledge-aware graph networks for commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2829–2839, Hong Kong, China, November 2019. Association for Computational Linguistics.

[41] Zhouhan Lin, Minwei Feng, Cícero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A structured self-attentive sentence embedding. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[42] Bing Liu and Ian Lane. Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling. In *Proc. Interspeech 2016*, pages 685–689, 2016.

[43] Yijin Liu, Fandong Meng, Jinchao Zhang, Jie Zhou, Yufeng Chen, and Jinan Xu. CM-net: A novel collaborative memory network for spoken language understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1051–1060, Hong Kong, China, 2019. Association for Computational Linguistics.

[44] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[45] Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. Interactive attention networks for aspect-level sentiment classification. In Carles Sierra, editor, *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 4068–4074. ijcai.org, 2017.

[46] Saif Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 321–327, Atlanta, Georgia, USA, 2013. Association for Computational Linguistics.

[47] Duy-Kien Nguyen and Takayuki Okatani. Improved fusion of visual and language representations by dense symmetric co-attention for visual question answering. In *CVPR*, pages 6087–6096. IEEE Computer Society, 2018.

[48] Jinjie Ni, Tom Young, Vlad Pandelea, Fuzhao Xue, Vinay Adiga, and Erik Cambria. Recent advances in deep learning based dialogue systems: A systematic survey. *arXiv preprint arXiv:2105.04387*, 2021.

[49] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, 2014. Association for Computational Linguistics.

[50] Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. SemEval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 486–495, Denver, Colorado, June 2015. Association for Computational Linguistics.

[51] Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. SemEval-2014 task 4: Aspect based

sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland, 2014. Association for Computational Linguistics.

[52] Soujanya Poria, Iti Chaturvedi, Erik Cambria, and Amir Hussain. Convolutional mkl based multimodal emotion recognition and sentiment analysis. In Francesco Bonchi, Josep Domingo-Ferrer, Ricardo Baeza-Yates, Zhi-Hua Zhou, and Xindong Wu, editors, *ICDM*, pages 439–448. IEEE Computer Society, 2016.

[53] Libo Qin, Wanxiang Che, Yangming Li, Minheng Ni, and Ting Liu. Dcr-net: A deep co-interactive relation network for joint dialog act recognition and sentiment classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8665–8672. AAAI Press, 2020.

[54] Libo Qin, Wanxiang Che, Yangming Li, Haoyang Wen, and Ting Liu. A stack-propagation framework with token-level intent detection for spoken language understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2078–2087, Hong Kong, China, 2019. Association for Computational Linguistics.

[55] Libo Qin, Zhouyang Li, Wanxiang Che, Minheng Ni, and Ting Liu. Co-gat: A co-interactive graph attention network for joint dialog act recognition and sentiment classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13709–13717, 2021.

[56] Libo Qin, Tailu Liu, Wanxiang Che, Bingbing Kang, Sendong Zhao, and Ting Liu. A co-interactive transformer for joint slot filling and intent detection. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8193–8197, 2021.

[57] Libo Qin, Fuxuan Wei, Tianbao Xie, Xiao Xu, Wanxiang Che, and Ting Liu. GL-GIN: Fast and accurate non-autoregressive model for joint multiple intent detection and slot filling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 178–188, Online, 2021. Association for Computational Linguistics.

[58] Libo Qin, Xiao Xu, Wanxiang Che, and Ting Liu. AGIF: An adaptive graph-interactive framework for joint multiple intent detection and slot filling. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1807–1816, Online, 2020. Association for Computational Linguistics.

[59] Vipul Raheja and Joel Tetreault. Dialogue Act Classification with Context-Aware Self-Attention. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3727–3733, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[60] Mirco Ravanelli, Philemon Brakel, Maurizio Omologo, and Yoshua Bengio. Light gated recurrent units for speech recognition. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(2):92–102, 2018.

[61] Tulika Saha, Aditya Patra, Sriparna Saha, and Pushpak Bhattacharyya. Towards emotion-aided multi-modal dialogue act classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4361–4372, Online, July 2020. Association for Computational Linguistics.

[62] Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *ESWC*, pages 593–607, 2018.

[63] K. Schouten and F. Frasincar. Survey on aspect-level sentiment analysis. *IEEE Transactions on Knowledge and Data Engineering*, 28(3):813–830, March 2016.

[64] Guokan Shang, Antoine Tixier, Michalis Vazirgiannis, and Jean-Pierre Lorré. Speaker-change aware CRF for dialogue act classification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 450–464, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics.

[65] Weizhou Shen, Siyue Wu, Yunyi Yang, and Xiaojun Quan. Directed acyclic graph network for conversational emotion recognition. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1551–1560, Online, August 2021. Association for Computational Linguistics.

[66] Jiaxin Shi, Shulin Cao, Lei Hou, Juanzi Li, and Hanwang Zhang. TransferNet: An effective and transparent framework for multi-hop question answering over relation graph. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4149–4158, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[67] Youwei Song, Jiahai Wang, Tao Jiang, Zhiyue Liu, and Yanghui Rao. Attentional encoder network for targeted sentiment classification. 2019.

[68] Kai Sun, Richong Zhang, Samuel Mensah, Yongyi Mao, and Xudong Liu. Aspect-level sentiment analysis via convolution over dependency tree. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5679–5688, Hong Kong, China, 2019. Association for Computational Linguistics.

[69] Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. Effective LSTMs for target-dependent sentiment classification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3298–3307, Osaka, Japan, 2016. The COLING 2016 Organizing Committee.

[70] Hao Tang, Donghong Ji, Chenliang Li, and Qiji Zhou. Dependency graph enhanced dual-transformer structure for aspect-based sentiment classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6578–6588, Online, 2020. Association for Computational Linguistics.

[71] Jialong Tang, Ziyao Lu, Jinsong Su, Yubin Ge, Linfeng Song, Le Sun, and Jiebo Luo. Progressive self-supervised attention learning for aspect-level sentiment analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 557–566, Florence, Italy, 2019. Association for Computational Linguistics.

[72] Yuanhe Tian, Guimin Chen, and Yan Song. Aspect-based sentiment analysis with type-aware graph convolutional networks and layer ensemble. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, pages 2910–2922, 2021.

[73] Yuanhe Tian, Guimin Chen, and Yan Song. Enhancing aspect-level sentiment analysis with word dependencies. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3726–3739, Online, April 2021. Association for Computational Linguistics.

[74] Gokhan Tur and Renato De Mori. *Spoken language understanding: Systems for extracting semantic information from speech.* John Wiley & Sons, 2011.

[75] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008.

[76] Ashish Vaswani, Yonatan Bisk, Kenji Sagae, and Ryan Musa. Supertagging with LSTMs. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 232–237, San Diego, California, 2016. Association for Computational Linguistics.

[77] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017.

[78] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[79] Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. Relational graph attention network for aspect-based sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3229–3238, Online, 2020. Association for Computational Linguistics.

[80] Ruili Wang, Wanting Ji, and Baoyan Song. Durable relationship prediction and description using a large dynamic graph. *World Wide Web*, 21(6):1575–1600, 2018.

[81] Shuai Wang, Sahisnu Mazumder, Bing Liu, Mianwei Zhou, and Yi Chang. Target-sensitive memory networks for aspect sentiment classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 957–967, Melbourne, Australia, 2018. Association for Computational Linguistics.

[82] Yan Wang, Jiayu Zhang, Jun Ma, Shaojun Wang, and Jing Xiao. Contextualized emotion recognition in conversation as sequence tagging. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 186–195, 1st virtual meeting, July 2020. Association for Computational Linguistics.

[83] Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. Attention-based LSTM for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 606–615, Austin, Texas, 2016. Association for Computational Linguistics.

[84] Yu Wang, Yilin Shen, and Hongxia Jin. A bi-model based RNN semantic frame parsing model for intent detection and slot filling. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 309–314, New Orleans, Louisiana, 2018. Association for Computational Linguistics.

[85] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. 2014. cite arxiv:1410.3916.

[86] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, 1992.

[87] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics.

[88] Di Wu, Liang Ding, Fan Lu, and Jian Xie. SlotRefine: A fast non-autoregressive model for joint intent detection and slot filling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1932–1937, Online, 2020. Association for Computational Linguistics.

[89] Yufeng Xiao, Huan Zhao, and Tingting Li. Learning class-aligned and generalized domain-invariant representations for speech emotion recognition. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 4(4):480–489, 2020.

[90] Bowen Xing, Lejian Liao, Dandan Song, Jingang Wang, Fuzheng Zhang, Zhongyuan Wang, and Heyan Huang. Earlier attention? aspect-aware LSTM for aspect-based sentiment analysis. In Sarit Kraus, editor, *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5313–5319. ijcai.org, 2019.

[91] Bowen Xing, Lejian Liao, Dandan Song, Jingang Wang, Fuzheng Zhang, Zhongyuan Wang, and Heyan Huang. Earlier attention? aspect-aware lstm for aspect-based sentiment analysis. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5313–5319. International Joint Conferences on Artificial Intelligence Organization, 7 2019.

[92] Bowen Xing and Ivor Tsang. DARER: Dual-task temporal relational recurrent reasoning network for joint dialog sentiment classification and act recognition. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3611–3621, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[93] Bowen Xing and Ivor Tsang. DARER: Dual-task temporal relational recurrent reasoning network for joint dialog sentiment classification and act recognition. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3611–3621, 2022.

[94] Bowen Xing and Ivor Tsang. Dignet: Digging clues from local-global interactive graph for aspect-level sentiment classification. *arXiv preprint arXiv:2201.00989*, 2022.

[95] Bowen Xing and Ivor Tsang. Neural subgraph explorer: Reducing noisy information via target-oriented syntax graph pruning. In *Proceedings of the Thirty-First*

International Joint Conference on Artificial Intelligence, IJCAI-22, pages 4425–4431. International Joint Conferences on Artificial Intelligence Organization, 7 2022. Main Track.

[96] Bowen Xing and Ivor W Tsang. Out of context: A new clue for context modeling of aspect-based sentiment analysis. *Journal of Artificial Intelligence Research*, 74:627–659, 2022.

[97] Bowen Xing and Ivor W. Tsang. Understand me, if you refer to aspect knowledge: Knowledge-aware gated recurrent memory network. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 6(5):1092–1102, 2022.

[98] Bowen Xing and Ivor W Tsang. Understand me, if you refer to aspect knowledge: Knowledge-aware gated recurrent memory network. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 6(5):1092–1102, 2022.

[99] Bowen Xing and Ivor W Tsang. Co-evolving graph reasoning network for emotion-cause pair extraction. *arXiv preprint arXiv:2306.04340*, 2023.

[100] Fuzhao Xue, Aixin Sun, Hao Zhang, and Eng Siong Chng. Gdpnet: Refining latent multi-view graph for relation extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14194–14202, 2021.

[101] Wei Xue and Tao Li. Aspect based sentiment analysis with gated convolutional networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2514–2523, Melbourne, Australia, 2018. Association for Computational Linguistics.

[102] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[103] Steve Young, Milica Gašić, Blaise Thomson, and Jason D. Williams. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179, 2013.

[104] Bowen Zhang, Xutao Li, Xiaofei Xu, Ka-Cheong Leung, Zhiyao Chen, and Yunming Ye. Knowledge guided capsule attention network for aspect-based

sentiment analysis. *IEEE ACM Trans. Audio Speech Lang. Process.*, 28:2538–2551, 2020.

[105] Chen Zhang, Qiuchi Li, and Dawei Song. Aspect-based sentiment classification with aspect-specific graph convolutional networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4568–4578, Hong Kong, China, 2019. Association for Computational Linguistics.

[106] Chenwei Zhang, Yaliang Li, Nan Du, Wei Fan, and Philip Yu. Joint slot filling and intent detection via capsule neural networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5259–5267, Florence, Italy, 2019. Association for Computational Linguistics.

[107] Lei Zhang, Shuai Wang, and Bing Liu. Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1253, 2018.

[108] Mi Zhang and Tieyun Qian. Convolution over hierarchical syntactic and lexical graphs for aspect level sentiment analysis. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3540–3549, Online, 2020. Association for Computational Linguistics.

[109] Xiaodong Zhang and Houfeng Wang. A joint model of intent determination and slot filling for spoken language understanding. In Subbarao Kambhampati, editor, *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 2993–2999. IJCAI/AAAI Press, 2016.

[110] Suncong Zheng, Feng Wang, Hongyun Bao, Yuexing Hao, Peng Zhou, and Bo Xu. Joint extraction of entities and relations based on a novel tagging scheme. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1227–1236, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[111] Suncong Zheng, Feng Wang, Hongyun Bao, Yuexing Hao, Peng Zhou, and Bo Xu. Joint extraction of entities and relations based on a novel tagging scheme. In *Proceedings of the 55th Annual Meeting of the Association for Computational*

Linguistics (Volume 1: Long Papers), pages 1227–1236, Vancouver, Canada, 2017. Association for Computational Linguistics.

[112] Peixiang Zhong, Di Wang, and Chunyan Miao. Knowledge-enriched transformer for emotion detection in textual conversations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 165–176, Hong Kong, China, November 2019. Association for Computational Linguistics.

[113] Lixing Zhu, Gabriele Pergola, Lin Gui, Deyu Zhou, and Yulan He. Topic-driven and knowledge-aware transformer for dialogue emotion detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1571–1582, Online, August 2021. Association for Computational Linguistics.