Ph.D. DISSERTATION

How to capture the important tokens and build sequential encoder for token-level classification models

토큰 단위 분류 모델을 위한 중요 토큰 포착 및 시퀀스 인코더 설계 방법

BY

KANG TAE-GWAN

AUGUST 2022

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Ph.D. DISSERTATION

How to capture the important tokens and build sequential encoder for token-level classification models

토큰 단위 분류 모델을 위한 중요 토큰 포착 및 시퀀스 인코더 설계 방법

BY

KANG TAE-GWAN

AUGUST 2022

DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

How to capture the important tokens and build sequential encoder for token-level classification models

토큰 단위 분류 모델을 위한 중요 토큰 포착 및 시퀀스 인코더 설계 방법

지도교수 정 교 민

이 논문을 공학박사 학위논문으로 제출함

2022년 8월

서울대학교 대학원

전기 컴퓨터 공학부

강 태 관

강태관의 공학박사 학위 논문을 인준함

2022년 8월

Abstract

With the development of the internet, a great volume of data has accumulated over time. Therefore, dealing with long sequential data can become a core problem in web services. For example, streaming services such as YouTube, Netflix, and Tictoc have used the user's viewing history sequence to recommend videos that users may like. Such systems have replaced the user's viewed video with each item or token to predict what item or token will be viewed next. These tasks have been defined as Token-Level Classification (TLC) tasks. Given the sequence of tokens, TLC identifies the labels of tokens in the required portion of this sequence. As mentioned above, TLC can be applied to various recommendation systems. In addition, most Natural Language Processing (NLP) tasks can also be formulated as TLC problem. For example, a sentence and each word within the sentence can be expressed as a token-level sequence. In particular, in the case of information extraction, it can be changed to a TLC task that distinguishes whether a specific word span in the sentence is information.

The characteristics of TLC datasets are that they are very sparse and long. Therefore, it is a very important problem to extract only important information from the sequences and properly encode them. In this thesis, we propose the method to solve the two academic questions of TLC in Recommendation Systems and information extraction: 1) How to capture important tokens from the token sequence and 2) How to encode a token sequence into the model. As deep neural networks (DNNs) have shown outstanding performance in various web application tasks, we design the RNN and Transformer-based model for recommendation systems, and information extractions.

In this dissertation, we propose novel models that can extract important tokens for recommendation systems and information extraction systems. In recommendation systems, we design a BART-based system that can capture an important portion of token sequence through self-attention mechanisms and consider both bidirectional and left-to-right directional information. In information systems, we present relation network-based models to focus on important parts such as opinion target and neighbor words.

keywords: Token, Token-level Sequence, Classification, Deep Learning
student number: 2016-20860

Contents

Abstract **i**

Contents **iii**

List of Tables **v**

List of Figures **vii**

1 INTRODUCTION **1**

2 Token level classification in Recommendation Systems **8**

2.1 Overview 8

2.1.1 Introduction to TLC in Recommendation Systems 8

2.1.2 Introduction to Hierarchical RNN 9

2.1.3 Introduction to E-BART4Rec 11

2.1.4 Related Works 13

2.1.5 Backgrounds 15

2.2 Hierarchical Recurrent Nural Network-based Recommendation Systems 19

2.2.1 Model Description 19

2.2.2 Experimental Results 23

2.3 Entangled Bidirectional Encoder to Auto-regressive Decoder for Sequential Recommendation 27

2.3.1 Model Description 27

2.3.2 Experimental Results 32
2.4 Conclusion 37

3 Token level classification in Information Extraction 39

3.1 Overview 39
3.1.1 Introduction to TLC in Information Extraction 39
3.1.2 Introduction to RABERT 40
3.1.3 Introduction to GRED 43
3.1.4 Related Works 46
3.1.5 Backgrounds 47
3.2 RABERT: Relation-Aware BERT for Target-Oriented Opinion Words Extraction 49
3.2.1 Model Description 49
3.2.2 Experimental Results 54
3.3 Gated Relational Target-aware Encoder and Local Context-aware Decoder for Target-oriented Opinion Words Extraction 58
3.3.1 Model Description 58
3.3.2 Experimental Results 66
3.4 Conclusion 77

4 Conclusion 79

4.1 On-going Researches 79
4.2 Conclusion 80

Abstract (In Korean) 95

Acknowlegement 97

List of Tables

2.1 Performance comparison on different methods on two datasets. We highlighted the best performance and the second best performance methods in bold and underlined fonts respectively. 36

3.1 Example of user reviews and their extracted pairs of *opinion targets* and *opinion words*. The red-colored words and the blue-colored words represent *opinion targets* and *opinion words*, respectively. 43

3.2 Statistics of the four datasets 54

3.3 Main Experimental Results(%). Performance comparison with the existing baselines on four datasets. 57

3.4 Two examples of BIO-scheme sentences for TOWE task. The underlined terms indicate *opinion targets*, and the bold style terms represent corresponding *opinion words*. 58

3.5 Main experimental results on the four benchmark datasets. Performance comparison with the baselines. 72

3.6 The F1 scores of various pooling methods of *opinion targets* on the four benchmark datasets. 73

3.7 Case study: the prediction of Encoder+Decoder model and GRED. The red-colored words and blue-colored words represent *opinion targets* and *opinion words*, respectively. 74

3.8 Case study: the prediction of Encoder+Decoder model and GRED. The red-colored words and blue-colored words represent *opinion targets* and *opinion words*, respectively. 75

List of Figures

1.1 Token-Level Prediction in recommendation systems and natural language process. 2

1.2 Summary of the proposed models 3

2.1 Illustration of general Feedforward Neural Network (FNN) architecture and Recurrent Neural Network (RNN). 15

2.2 Illustration of Long Short Term Memory (LSTM, left) and Gated Recurrent Unit (GRU, right): f, i and o are respectively the forget gate, input gate, and output gate. h and $\hat{h}$ are the hidden (cell) state and the candidate of hidden (cell) state in LSTM. In one LSTM cell, f, i, and o must be trained simultaneously, so LSTM must have too many parameters. However, x is the input in GRU. z denotes the update gate and h denotes the reset gate, there are only a few gates. h and $\hat{h}$ are respectively the hidden state and the candidate hidden state. In GRU, the hidden state and output are the same. In this case, h is the hidden state and also the output of the GRU. 16

2.3 This is a figure of our model with a hierarchical structure. Upper square nodes are the long-term layer, and the lower rounded-square nodes are the short-term layer. The short-term layer receives input whenever an event occurs. 19

2.4 This is a figure of one short-term layer block in hi-RNN. 20

2.5 This is a figure of a long-term layer block in hi-RNN. 21

2.6 Hit@10 for movielens1M dataset using the baseline methods and our method. 24

2.7 Hit@10 for Steam dataset using the baseline methods and our method. 25

2.8 Hit@10 for movielens1M dataset with various maximum sequence lengths. 26

2.9 Hit@10 for movielens1M dataset with various α parameters. 26

2.10 The overview of E-BART4Rec. The encoder takes noisy input sequence $i_1, \_, ...i_n$ and the decoder takes $(n-1)$ time input sequence with start token S. The Gate Network adjusts the final output between the encoder and the decoder. 31

2.11 The *influence ratio* and NDCG@10 (denoted as N@10) for validation sets over training epoch. 32

2.12 NDCG@10 comparison on SASRec, BERT4Rec, BART4Rec and E-BART4Rec on MovieLens with different maximum length. 33

2.13 Summary of Chapter 2 38

3.1 An example of TOWE. TOWE aims at extracting the corresponding opinion words (colored in blue) for the given targets (colored in orange). 40

3.2 The concept of GRED. The target-aware encoder finds target-dependent opinion words, and the local context-aware decoder seeks local context-dependent opinion words. Then, the gate network dynamically aggregates the outputs of these two networks. 44

3.3 The overview of RABERT architecture. RABERT consists of a Text Encoder, Target pooler, Target-sentence relation network, Neighbor-aware relation network, and a Conditional Random Field, which are denoted as Encoder, Pool, Target-Sent, Neighbor, and CRF respectively in the figure. Marking represents adding target markers to sentences. 53

3.4 Ablation study on the four datasets. The error bars represent the F1 score difference between each model and RABERT. 55

3.5 The overview of GRED. GRED comprises a target-aware context encoder module, a local context-aware decoder module, and a gate network. The target-aware context encoder module and the local context-aware decoder module contain the target relation module and the local context relation module, respectively. The gate network computes final representations based on outputs from both modules. 64

3.6 Structures of the two relation networks: (a) Target relation network and (b) Local context relation network. The target relation network compares the opinion target (*"bottles of wine"*) and each word (*"bottles"*, *"of"*, *"wine"*,*"are"*, and *"cheap"*) in the sentence (*"bottles of wine are cheap"*). The local context relation network captures relationship among the words (*"bottles"*, *"of"*, *"wine"*,*"are"*, and *"cheap"*) in the sentence (*"bottles of wine are cheap"*). 65

3.7 Ablation study of GRED. We report the F1 score of the variants of GRED on the four benchmark datasets. 68

3.8 Visualization of the gate weight α_u of each word. The opinion targets of each sentence are *"hot dogs"*, *"Slamwich"*, *"cocktail"* and *"selection of bottled beer"*, respectively. The extracted opinion words of each sentence are *"top notch"*, *"amazing"*, {*"great", "die", "for"*}, and {*"Not", "much"*}.The stronger the red words, the greater the value of α_u. 76

3.9 Summary of Chapter 3 78

4.1 The overview of information extracion in medical domain 80

4.2 The comparison of the proposed models 81

4.3 The summary of this dissertation 82

Chapter 1

INTRODUCTION

Recently, Natural Language Processing (NLP) has attracted much attention from academia and industries. Among the NLP fields, the demand for analyzing each word in the sentences is increasing to solve fine-grained tasks such as aspect-based sentiment analysis and entity retrieval tasks more accurately [1–3].

Therefore, Token-Level Classification (TLC) tasks have emerged as one of the core problems in NLP fields. In TLC, each word of a sentence can be treated as a token and the sentences can be represented as token-level sequences. Given the sequence of tokens, TLC predicts the labels of tokens in the required portion of this sequence. For example, given token sequences $\{i_0, i_1, ..., i_k, ...i_{n-1}, i_n\}$, TLC classifies the types of each token $\{i_0, i_1, ..., i_k, ...i_{n-1}, i_n\}$ as a label sequence $\{L_{i_0}, L_{i_1}, ..., L_{i_k}, ...L_{i_{n-1}}, L_{i_n}\}$. It is not necessary to classify all tokens, but only specific tokens are classified such as $i_k \rightarrow L_{i_k}$.

Furthermore, TLC has been applied to not only NLP fields but also recommendation systems. For example, in movie sequential recommendation systems, the movies the users watched are given as token and token sequences respectively. Given these histories, the TLC predicts which movie the next user will choose.

In this thesis, we focus on Deep Neural Networks (DNNs)-based TLC models in information extraction tasks of NLP fields and sequential recommendation tasks. In

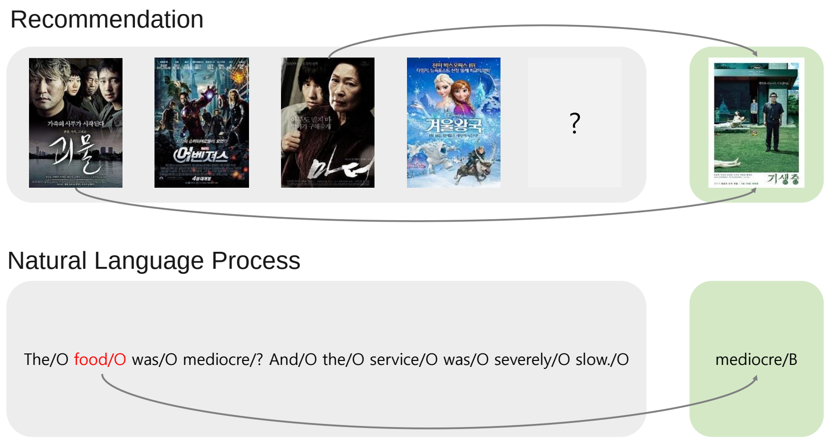
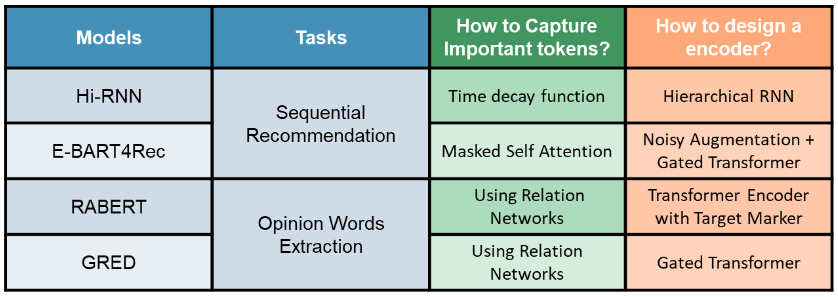

Recommendation
?
Natural Language Process
The/O food/O was/O mediocre/? And/O the/O service/O was/O severely/O slow./O
mediocre/B

Figure 1.1: Token-Level Prediction in recommendation systems and natural language process.

early researches, Recurrent Neural Networks (RNNs) have been widely used in various TLC tasks because they are useful for processing sequential information. However, RNNs have shown their vulnerability to long-term dependency problems, which cannot transfer gradient information in long-term sequence. Since Transformer has used parallel processes and has not suffered from long-term dependency problems, many recent works have developed transformer-based TLC models for Recommendation systems and NLP tasks.

The TLC deals with a long sequence composed of several various tokens. Therefore, it is very important to extract only meaningful tokens from the sequence. For example, in movie recommendation systems, selecting only the movies that the user actually enjoyed from the movie viewing history can enhance the recommendation model performance. In information extraction, it is necessary to capture a specific aspect within the sentence because information related to a specific aspect must be extracted from the sentence. Also, encoding the token-level sequence into the model is critical for TLC tasks. For example, the performance of TLC can be improved by sequencing and encoding information about time or target aspects together.



Models	Tasks	How to Capture Important tokens?	How to design a encoder?
Hi-RNN	Sequential Recommendation	Time decay function	Hierarchical RNN
E-BART4Rec		Masked Self Attention	Noisy Augmentation + Gated Transformer
RABERT	Opinion Words Extraction	Using Relation Networks	Transformer Encoder with Target Marker
GRED		Using Relation Networks	Gated Transformer

Figure 1.2: Summary of the proposed models

In this dissertation, we aim to solve two academic questions of TLC tasks: 1) How to capture important tokens from sequential information and 2) How to encode a token sequence. We propose the new recommendation systems and information extraction systems to address these two questions in chapter 2 and chapter 3, respectively. We summarize the proposed models in figure 1.2.

In chapter 2, we attempt to find the answers to the two academic questions of TLC tasks for sequential recommendation tasks. First, we present the new recommendation systems that incorporate the temporal properties of the user history using hierarchical RNNs.

Hierarchical Recurrent Neural Network-based Recommendation Systems (In Chapter 2.2). Nowadays, recommendation systems are widely used in various services. This system predicts what item a user will use next using large amounts of stored user history. Recommendation systems are commonly applied in various fields such as movies, e-commerce, and social services. However, previous researches on recommendation systems commonly overlooked the importance of item usage sequence and time intervals of the time series data from users. We provide a novel recommendation system that incorporates these temporal properties of the user history. We design a recurrent neural network (RNN) model with a hierarchical structure so that the sequence and time intervals of the user's item usage history can be considered. The model is divided into two layers: a layer for a long time and a layer for a short time. We conduct experiments on real-world data such as Movielens and Steam datasets, which have a long-time range, and show that our new model outperforms the previously widely used recommendation methods, including RNN-based models. We also conduct experiments to find out the influence of the length and time interval of sequences in our model. These experimental results show that both sequence length and time interval are influential, indicating that it is important to consider the temporal properties for long-term sequences.

This work [4] was published in Byeongjin Choe, Taegwan Kang, and Kyomin Jung *"Recommendation System With Hierarchical Recurrent Neural Network for Long-Term Time Series"*, in IEEE Access.

In the next, we present the new transformer-based recommendation systems that can capture important tokens using a self-attention mechanism and can encode sparse sequences effectively.

Entangled Bidirectional Encoder to Auto-regressive Decoder for Sequential Recommendation (In Chapter 2.3). Recently, BERT has shown overwhelming performance in sequential recommendation by using a bidirectional attention mechanism. Although the bidirectional model effectively captures dynamics from user interaction, its training strategy does not fit well to the inference stage in sequential recommendation which generally proceeds in a left-to-right way. To address this problem, we introduce a new recommendation system built upon BART, which is widely used in NLP tasks. BART uses a left-to-right decoder and injects noise into its bidirectional encoder, which can reduce the gap between training and inference. However, direct usage of BART for recommendation system is challenging due to its model property and domain difference. BART is an auto-regressive generative model, and its noising transformation techniques are originally developed for text sequence. In this paper, we present a novel sequential recommendation model, Entangled BART for Recommendation (E-BART4Rec) that entangles bidirectional encoder and auto-regressive decoder with noisy transformations for user interaction. Unlike BART, where the final output only depends on its output of the decoder, E-BART4Rec dynamically integrates the output of the bidirectional encoder and auto-regressive decoder based on a gating mechanism that calculates the importance of each output. We also employ noisy transformation that imitates the real users' behaviors, such as item deletion, item cropping, item reverse, and item infilling, to the input of the encoder. Extensive experiments on widely-used datasets demonstrate that our models significantly outperform the baselines.

This work [5] is published in Taegwan Kang, Hwanhee Lee, Byeongjin Choe, and Kyomin Jung *"Entangled Bidirectional Encoder to Auto-regressive Decoder for Sequential Recommendation"*, in Proceddings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21).

In chapter 3, we aim to solve the two academic questions of TLC tasks for the information extraction tasks. First, we design the RABERT that can fully utilize the power of BERT and consider the two important relations: target-aware relation and neighbor-aware relation.

RABERT: Relation-Aware BERT for Target-Oriented Opinion Words Extraction (In Chapter 3.2). Targeted Opinion Word Extraction (TOWE) is a subtask of aspect-based sentiment analysis, which aims to identify the corresponding opinion terms for given opinion targets in a review. To solve the TOWE task, recent works mainly focus on learning the target-aware context representation that infuses target information into context representation by using various neural networks. However, it has been unclear how to encode the target information to BERT, a powerful pre-trained language model. In this paper, we propose a novel TOWE model, RABERT (Relation-Aware BERT), that can fully utilize BERT to obtain target-aware context representations. To introduce the target information into BERT layers clearly, we design a simple but effective encoding method that adds target markers indicating the opinion targets to the sentence. In addition, we find that the neighbor word information is also important for extracting the opinion terms. Therefore, RABERT employs the target-sentence relation network and the neighbor-aware relation network to consider both the opinion target and the neighbor words information. Our experimental results on four benchmark datasets show that RABERT significantly outperforms the other baselines and achieves state-of-the-art performance. We also demonstrate the effectiveness of each component of RABERT in further analysis.

This work [6] is published in *"RABERT: Relation-Aware BERT for Target-Oriented Opinion Words Extraction"* written by Taegwan Kang, Minwoo Lee, Nakyeong Yang, and Kyomin Jung, in Proceedings of the 30th ACM International Conference on Information & Knowledge Management (CIKM '21).

Finally, we propose the new information extraction models that can dynamically the two important information - target information and local context information using a gate network.

Gated Relational Target-aware Encoder and Local Context-aware De- coder for Target-oriented Opinion Words Extraction (In Chapter 3.3). Target-oriented Opinion Word Extraction (TOWE) is a recently designed subtask of aspect-based sentiment analysis that aims to extract the opinion words corresponding to given opinion targets in text. To solve TOWE, it is important to consider the surrounding words of opinion words as well as the opinion targets. Although existing works have captured the opinion target, they cannot effectively utilize the local context, i.e. relationship among surrounding words of opinion words. In this paper, we propose a novel TOWE model, Gated Relational target-aware Encoder and local context-aware Decoder (GRED), which dynamically leverages the information of the opinion target and the local context. Intuitively, the target-aware encoder catches the opinion target information, and the local context-aware decoder obtains the local context information from the relationship among surrounding words. Then, GRED employs a gate mechanism to dynamically aggregate the outputs of the encoder and the decoder. In addition, we adopt a pre-trained language model BART, as the structure of GRED to improve the implicit language knowledge. Extensive experiments on four benchmark datasets show that GRED outperforms all the baseline models and achieves state-of-the-art performance. Furthermore, our in-depth analysis demonstrates that GRED properly leverages the information of the opinion target and the local context for extracting the opinion words.

Chapter 2

Token level classification in Recommendation Systems

2.1 Overview

2.1.1 Introduction to TLC in Recommendation Systems

In this chapter, we aim to find the answer to the academic questions of TLC in sequential recommendation systems: 1) How to capture important tokens from sequential information and 2) How to encode a token sequence.

Recently, recommendation systems have become widespread on many web service systems such as music, movie streaming sites, and e-commerce systems. In general, the recommendation systems present the items that the users are likely to choose based on their historical information. Therefore, personalizing the recommendation systems to each user is key to achieving good performance. Most of the recommendation systems have been proposed to predict users' personalized preferences and find items with the highest preference based on their past usage records. Especially, deep learning-based approaches have shown promising performance on the recommendation systems such as Recurrent Neural Network (RNN) [7, 8] and Transformer [9, 10].

In this dissertation, we have focused on RNN and Transformer-based recommendation systems. First, we propose the RNN-based recommendation systems that incorporate temporal properties (Chapter 2.2). However, Transformer-based recommendation systems are superior to RNN-based recommendation systems in recent works [9, 10]. Therefore, we also propose a Transformer-based recommendation system that entangles the bidirectional model and left-to-right model (chapter 2.3). All of our works have attempted to solve the academic questions. Finally, we summarize and compare our recommendation systems (chapter 2.4).

2.1.2 Introduction to Hierarchical RNN

In a recommendation system, temporal information in the users' past usage record is one of the important elements. Items recently purchased and items purchased long ago have different influences on current users' preferences. If someone who used to enjoy horror movies in the past has recently watched romance movies, he or she is now more likely to prefer romance movies to horror movies. Order information is also part of temporal information. For example, people usually buy a keyboard after buying a computer, but it will be rare to buy a computer after buying a keyboard.

However, the importance of temporal information has not been carefully considered in previous recommendation systems. Classical recommendation methods [11–13], such as content-based filtering and collaboration filtering, fail to take advantage of temporal information, looking at all records equally. Recently, several studies have been conducted using order information to make use of temporal information. Hidasi et al. [7, 8] suggest a recurrent neural network (RNN)-based recommendation model, called the Session-based RNN recommendation system. RNN is a well-known model for processing sequential data and is widely applied to the language model and speech recognition. Session-based RNN uses implicit feedback – click, visit, and any other session information – as input sequence of RNN.

However, many previous studies lacked attempts to utilize the time interval between the use of items. In a short sequence, the time interval between item use may not be an important factor, but in a sequence over a long period of time, the time interval becomes more important. In many previous studies, there is no difference between watching the two films one year apart and one day apart. Moreover, a week ago for a user who watched a movie every day for a week, will be much further away in order than a year ago for a user who watches a movie only once a year.

In this study, we present an RNN-based hierarchical model to consider both long sequence lengths and time interval information. A typical RNN-based model struggles to differentiate between events that occur at a certain interval of time and events that occur irregularly. Our model applied a hierarchical structure to effectively use time information to handle different time intervals differently in the model.

Our model has two layers: a short-term layer and a long-term layer. The short-term layer deals with short-term sequences that were commonly dealt with in previous models. In this paper, we use a basic RNN structure. Our model views the entire sequence as a bunch of short-term sequences. The long-term layers remember information from previous short-term sequences and deliver it over a long time considering the time interval.

We did experiments on datasets of Movielens with 1 million movie reviews and Steam. These datasets deals with several years of time and is, therefore, suitable for experiments on long-term time series. Experimental results show our model outperforms previous RNN models considering sequential order as well as models that do not consider sequential order. To further analyze our model, we conducted an experiment on sequence length and the influence of a time interval. In the experiment on sequence length, our model does not show impressive performance compared to baselines when the sequence length is very short, but is powerful when sequence length is long. We also conducted an ablation study on how much time interval should be considered between items. This showed that when the time interval is long, it is helpful to moderately weaken the influence of the item whose order in the item usage sequence is most recent. This shows why the model presented in this study is particularly strong in long-term sequential datasets.

2.1.3 Introduction to E-BART4Rec

In recent years, providing a personalized recommendation system has become essential for web services such as e-commerce, music, and video streaming services. Since user interaction with web services changes dynamically over time, catching it well is the core of the sequential recommendation system. As Deep Neural Networks (DNNs) have demonstrated their strong performance on capturing sequential information in many applications [14–16], various DNN models such as Recurrent Neural Networks (RNNs) [7], Convolutional Neural Networks (CNNs) [17, 18] and Transformers [10] have been employed as a model architectures of sequential recommendations. Among them, Bidirectional Encoder Representation from Transformer for sequential Recommendation (BERT4Rec) [9] has produced a great performance by facilitating a bidirectional attention mechanism to model sequential user interactions.

Despite the powerful performance of bidirectional modeling in BERT4Rec, bidirectional modeling has a gap between training and inference: bidirectional modeling is assumed to see all user interaction, but an inference stage in sequential recommendation only allows to see the past interaction as follows:

$$\begin{aligned} &Train : \{i_1, i_2, mask, i_4, i_5\} \rightarrow \{i_1, i_2, i_3, i_4, i_5\} \\ &Inference : \{i_1, i_2, mask, \underbrace{\underline{i_4}, \underline{i_5}}_{\text{unseen}}\} \rightarrow \{i_1, i_2, i_3\} \end{aligned} \tag{2.1}$$

To address this problem, we employ a model combining Bidirectional and Auto-Regressive Transformer (BART) [19], which has shown remarkable achievement in a wide range of Natural Language Processing (NLP) tasks. BART consists of a bidirectional encoder and a left-to-right decoder, which is pre-trained with noisy texts. Specifically, the encoder takes noisy input and the decoder is optimized to reconstruct original input based on the encoder representation of the noisy input. In other words, BART incorporates **(1) Left-to-right modeling** and **(2) Injecting noise** into bidirectional modeling, which can reduce the gap between training and inference in sequential recommendation: **(1) Left-to-right modeling** only uses the past information for modeling sequential information, hence it has no difference from inference. **(2) Injecting noise** to data samples can reduce an reliance on the future information (e.g. deleting future sequence in training: $\{i_1, i_2, mask, \_, , i_5\}$).

Nevertheless, there are still many restrictions on adapting BART for sequential recommendation. First, BART is not only a left-to-right model but also *auto-regressive generative* model. Generative models require dense datasets to elaborately model the data distribution, but the datasets for sequential recommendations are usually sparse. Second, the noisy transformation method used in BART is originally developed for text, and cannot be applied directly to sequential recommendation.

To overcome these limitations of the original BART, we propose a novel recommendation model, **E-BART4Rec** that **entangles** bidirectional model and auto-regressive model, and adopts noisy transformation for user interaction. BART computes its final output only using the output of auto-regressive decoder. Unlike BART, the final output of E-BART4Rec dynamically aggregates a bidirectional encoder and an auto-regressive decoder according to the characteristics of the user interaction. Specifically, as illustrated in Figure 2.10, we first obtain each output from the encoder and decoder. Then, we employ Gate Network to calculate an influence weight, β that determines how much the encoder and decoder are used to compute the next interaction. Finally, the next interaction is inferred by summing both the outputs with β. Additionally, we inject noise into the encoder using four types of noisy transformations such as item deletion, item cropping, item reverse, and item infilling. We conduct experiments on two widely-used datasets and the results of experiments demonstrate that our model outperforms the other baselines. Furthermore, qualitative analysis shows that our proposed E-BART4Rec can effectively aggregate the encoder and decoder based on the data characteristics.

2.1.4 Related Works

We introduce a number of past studies related to the recommendation system. This covers from recommendation system studies where sequence is not considered to recent RNN-based studies.

Collaborative Filtering (CF) is the general recommendation system method [11]. CF is based on the notion that people who have the same preferences might choose similar items. CF analyzes a user-item matrix to look for similar neighbors of users and then proposes items through these neighbors. However, item-to-item CF focuses on item–item matrices. The idea behind item-to-item CF is that items that have similar properties might be chosen by the same people. One of the other CF methods is Content-Boosted Collaborative Filtering (CBCF) [20], which uses ratings as a vector form directly.

Matrix Factorization (MF) gives insight to methods that utilize item ratings. MF maps user-latent factors and item-latent factors to joint dimensions. The goal of MF is to ensure that the product of these latent factors converges on the rating. The pure singular-value-decomposition-based (PureSVD) matrix factorization method and weighted regularized matrix factorization (WRMF) are examples of MF methods. MF is also used for recommendations with implicit feedback. BPR [21] is applied to MF to provide item prediction from implicit positive-only feedback. This method formulates the recommendation problem as a ranking problem. In addition, some BPR-based recommendation systems try to add social network information to basic features [22], and some BPR-based systems grade items to consider user preferences [23].

Non-sequential Neural Networks are also used in recommendation systems. Most of the deep-learning models used in recommendation systems are based on CF or MF, because deep-learning models can find more precise similarities in CF and latent factors in MF. The restricted Boltzmann machine (RBM) exhibits good performance in model phoneme recognition [24] and the classification problem [25], because it can easily obtain accurate features. Similarly, the RBM can be applied to CF-based recommendation systems. The RBM automatically extracts similarities or latent factors for items. Auto-encoders-based recommendation systems also have strengths in finding associations in features. The stacked denoising auto encoder (SDAE) [26] analyzes relationships among items and can obtain more accurate relations than the general CF model. collaborative deep learning (CDL) [27] jointly performs encoding and rating considering MF processes with feedforward networks. The collaborative denoising auto encoder (CDAE) [28] matches with top-N recommender systems using an auto encoder. NeuMF [29] combines a MF-based method and a multi-layer perceptron-based model to predict the item that will be used by a particular user. However, since the number of users affects the input of the model, there is a problem that if a new user comes in, a whole new learning is required. The key to these deep-learning models is that they can easily extract better latent factors than non-deep-learning models. However, since these models only change the existing CF/MF models into a deep-learning form, they still have limitations such as failure to consider temporal properties.

Recurrent Neural Networks (RNNs) work well with sequential data. RNNs are generally applied to language modelsfancyhdr [30] and speech recognition [31]. Hidasi et al. suggested an RNN-based recommendation model called a session-based RNN recommendation system [7]. The session-based RNN uses only implicit feedbacks – clicks or watch history – as the RNN's input sequence. Based on the implicit feedbacks, the session based RNN learned to rate latent scores of the items like BPR. Wu et al. proposed an RNN-based recommendation system [32] that exploits user-profile data compounded with session information data and guesses the items users want.

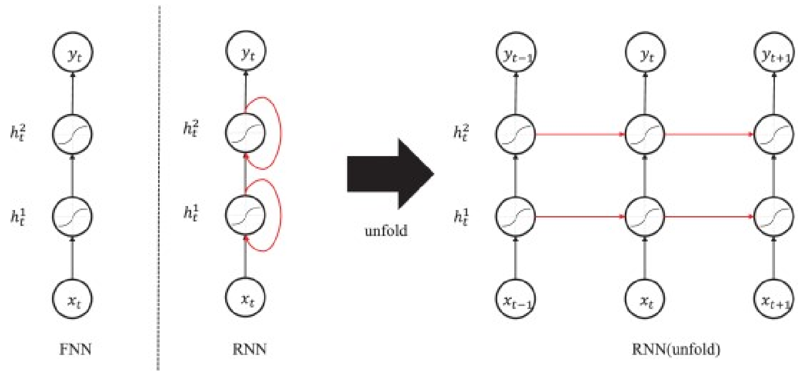


Figure 2.1: Illustration of general Feedforward Neural Network (FNN) architecture and Recurrent Neural Network (RNN).

2.1.5 Backgrounds

Recommendation System

The goal of a recommendation system is to propose the items that the users are most likely to prefer to use. In the recommendation problem that we want to solve, an event is a user's use of a specific item. One event contains information about which users used which items and when, which is a tuple of (user, item, timestamp). Each user has multiple events with different timestamps, and listing them in temporal order generates a sequence of events. If a sequence of events $S = \{x_1, \ldots, x_n\}$ is given for any user, the recommendation system finds an item of the next incoming event x_{n+1} by using given sequence S . More specifically, the recommendation system computes the probabilities of each item used in the next event and makes a ranked list of the probabilities that are sorted in ascending order for each user.

It is a very difficult problem for users to pinpoint which item to use next among so many items. In addition, it is also very useful to present multiple strong candidates to users rather than a single answer in the practical recommendation system application. Therefore, to increase convenience and efficiency, recommendation systems that propose the top-N number of ranked items in the ranked list to the user, referred to as

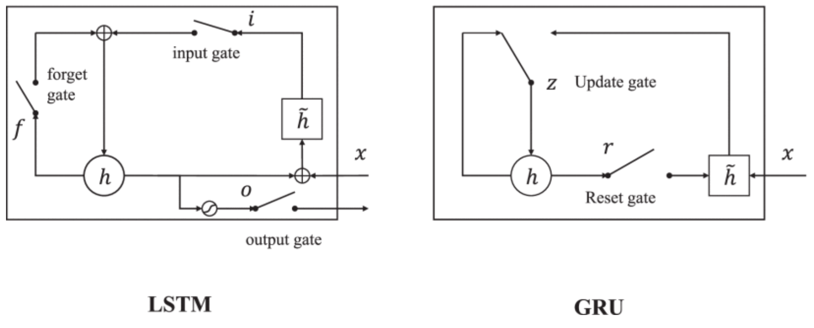


Figure 2.2: Illustration of Long Short Term Memory (LSTM, left) and Gated Recurrent Unit (GRU, right): f, i and o are respectively the forget gate, input gate, and output gate. h and $\hat{h}$ are the hidden (cell) state and the candidate of hidden (cell) state in LSTM. In one LSTM cell, f, i, and o must be trained simultaneously, so LSTM must have too many parameters. However, x is the input in GRU. z denotes the update gate and h denotes the reset gate, there are only a few gates. h and $\hat{h}$ are respectively the hidden state and the candidate hidden state. In GRU, the hidden state and output are the same. In this case, h is the hidden state and also the output of the GRU.

top-N recommendation systems, are widely used. For evaluating the model in a test phase, the researchers compare the ranked list with the actually used items in the test set.

Recurrent Neural Network

The RNN works well in many sequential data analysis tasks. In this section, we describe the architecture of the RNN. The RNN is a modified version of a feedforward neural network. In Figure 2.1, the general feedforward network (left) assumes that all the inputs are independent from each other, but the hidden layer of the RNN is dependent on previous information caused by the red-colored edges. These red-colored edges are called “recurrent edges,” and they start from the hidden layers and come in the same hidden layers. Given the input dataset, the RNN can renew the hidden unit state every time using the past hidden unit state (recurrent) and new input. In this manner, the RNN can induce time-dependency on sequential data. In summary, the formulas updating the hidden state h and output y in the RNN structure are as follows:

$$\begin{aligned} h_t =& f(U_t x_t + W_t h_{t-1} + b_h), \\ y_t =& g(W_y h_t + b_y) \end{aligned}$$

where the weight parameters are given by U and W , and the bias parameters are given by b.

However, this basic RNN structure has a practical problem referred to as the vanishing gradient problem [33]. The LSTM [34] is a particularly popular structure in the RNN-based deep-learning architectures that solves the vanishing gradient problem. Cho et al. suggested the GRU [35], which simplifies the LSTM by using reset gates and update gates that are similar to the switch structure. The reset gate determines how to combine previous hidden state information with input information, and the update gate determines the extent to which the next hidden state reflects the previous hidden state information and new input information. Formulas of the GRU are as follow:

$$\begin{aligned} z =& \sigma(x_t U^z + s_{(t-1)} W^z) \\ r =& \sigma(x_t U^r + s_{(t-1)} W^r) \\ h =& tanh(x_t U^h) + (s(t-1) \cdot r) W^h) \\ s_t =& (1-z) \cdot h + z \cdot s_(t-1) \end{aligned}$$

where the reset gate is given by r , the update gate is given by z , the activation function is given by σ and the state vector is given by s (Figure 2.2). These structures are currently widely used in practice, and this study also builds models using GRU.

Transformer

In this section, we briefly review the transformer architecture in recommendation systems [6]. Transformer is first proposed in [16]. The core part of transformer is the multi-head attention which can diversify the model representation. In Multi-head attention, we first denote $h_t^l \in R^d$ the hidden representations at each layer l for each position t and $H^l \in R^{n_u \times d}$ denote a matrix $(h_1^l, ..., h_{n_u}^l)^T$. Then, each single attention head linearly projects H^l to M subspaces and aligns projected matrix with itself, thereby calculating a latent representations between projected matrix itself:

$$\begin{aligned} Attention(Q, K, V) &= \text{softmax}(\frac{QK^T}{\sqrt{d/m}})V \\ x_m^l &= Attention(H^l W_m^Q, H^l W_m^K, H^l W_m^V), \end{aligned} \tag{2.2}$$

where the $x_m^l \in R^{n_u \times d}$ are the latent representations for each head m, and $W_m^Q \in R^{d \times d/M}$, $W_m^K \in R^{d \times d/M}$ and $W_m^V \in R^{d \times d/M}$ are linear projection matrices. The outputs of the Multi-head attention network are produced by concatenating all the latent representations in each single attention head:

$$X^l = Concat(x_1^l, x_2^l, ..., x_M^l)W^X, \tag{2.3}$$

where the $W^X \in R^{d \times d}$ are weights of the Multi-head attention networks. The Point-wise feed-forward network adds non-linearity and additional relation between dimension of the Multi-head representation X^l.

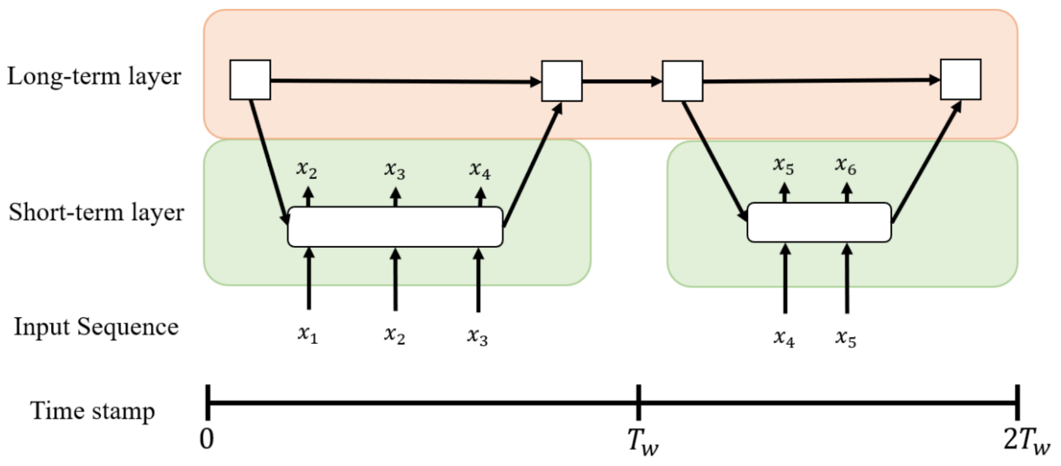


Figure 2.3: This is a figure of our model with a hierarchical structure. Upper square nodes are the long-term layer, and the lower rounded-square nodes are the short-term layer. The short-term layer receives input whenever an event occurs.

2.2 Hierarchical Recurrent Nural Network-based Recommendation Systems

2.2.1 Model Description

In this section, we explain our new recommended system model for long-term time-series data, Hi-RNN. We describe how our model can handle the long-term time series data with various time intervals.

The RNN structure described in background section is a neural network specialized in dealing with sequential data. However, as shown above, the basic RNN structure lacks consideration of the interval between inputs. If time intervals between item usage events are all the same, we can ignore the time interval information and use basic RNN. However, if time intervals are irregular, the basic RNN cannot handle them accurately. In addition, the long-term sequence data we use can vary in time intervals from minutes to years. Therefore, we constructed a hierarchical model consisting of two layers: short-term and long-term, which solves this problem.

(Short-term Layer) A short-term layer is a layer for events within a short period. Our model deals with sequence $S = \{i_1, \ldots, i_n\}$ divided into T_w intervals. If the time

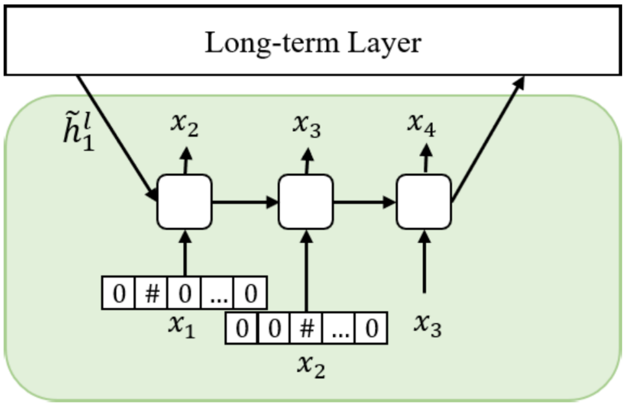


Figure 2.4: This is a figure of one short-term layer block in hi-RNN.

stamp of i_1 is 0, the first sub-sequence S_1 contains item usage records(events) from 0 to T_w . In this way, the whole sequence is divided by sub-sequences $\{S_1, \ldots, S_m\}$, where $m = \lceil T_n/T_w \rceil$ is the number of sub-sequences.

The short-term layer consists of small basic RNN structures that receive a sub-sequence as input. The time interval between events within the same sub-sequence is very short compared to the overall data length. Because of the short time interval, the influence of the time interval is significantly reduced, and only the sequential order of the inputs is sufficient. Thus, the RNN structure that does not consider time intervals can also sufficiently handle events within a short period.

Prediction of the next item is made in the short-term layer. To predict the next item, the basic RNN uses item information of the current time and hidden state up to the previous time. In the short-term layer, the current state of the long-term layer ${h^l}_{t`1}$ is additionally brought in to get item usage information from the distant past.

$$h_t^s = f(U^s x_t + W^s h_{(t-1)}^s + W^l h_{(t-1)}^l + b^s)$$

where x_t is the vector in which the item information of the event number t is converted into one-hot encoding. The last node's hidden state is used in long-term layers as a summary of the current sub-sequence. In other words, m hidden states can be obtained from m sub-sequences through the short-term layer.

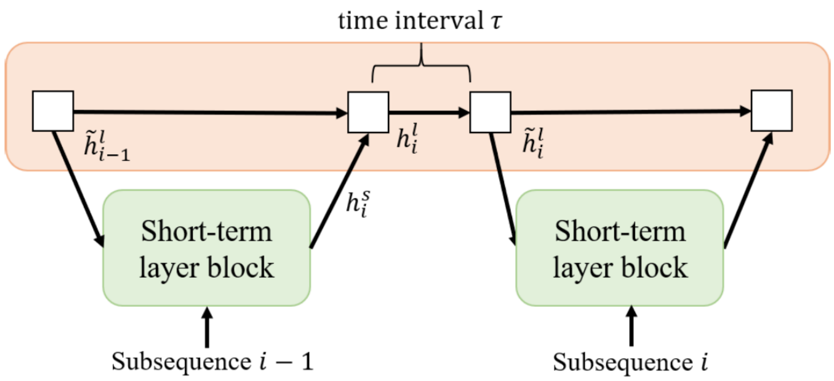


Figure 2.5: This is a figure of a long-term layer block in hi-RNN.

(Long-term Layer) The long-term layer transmits past information to the short-term layer block by using an RNN-based structure. The long-term layer receives m last hidden states of the m short-term layer blocks as follows:

$$h^l_t = f(U^l h^s_t + V^l h^l_{(t-1)} + b^l)$$

where h^s_t is a short-term layer's hidden state and $h^l_{(t-1)}$ is a long-term layer's previous hidden state. These m inputs are a sequence with a constant time interval of T_w . That is, the short-term layer transforms irregular time interval data to equal time interval data. Therefore, the long-term layer can process data without losing information about time intervals, even if they are made up of an RNN structure. The long-term layer also reduces the problem of RNNs, which are more likely to lose historical information as the input length increases, because multiple inputs are grouped and the length of the entire input is shorter.

However, depending on the time distribution of events in the sequence, there may be an empty subsequence. In this case, there is no input on the node in the long-term layer corresponding to the empty subsequence. To solve this problem, our model deletes long-term layer nodes with empty inputs. However, if a node is simply deleted, the time interval is not considered correctly, and the recent time becomes the same as that long ago. However, the longer the time interval, the lesser the importance of sequencing information about what items were just before and the greater the need for a comprehensive review of past records. Therefore, the node in the long-term layer takes into account the time interval and uses the following $\hat{h_t^l}$ instead of the last node's hidden state, h_t^l

$$\hat{h_t^l} = \alpha^\tau h_t^l + (1 - \alpha^\tau) h_{(mean,t)}^l$$

where τ is a time interval between two consecutive nodes, α is a constant, and $h_{(mean,t)}^l$ is the average of the hidden states in the long-term layer up to the previous time of t . This allows the hidden state of the long-term layer to gradually return to its mean over time. If $\leq T_w$, then $\hat{h_t^l} = h_t^l$ because it is normal for no node to be deleted. That is,

$$\hat{h_t^l} = \begin{cases} \alpha^\tau h_t^l + (1 - \alpha^\tau) h_{(mean,t)}^l & \tau > T_w \\ h_t^l & \tau \leq T_w \end{cases} .$$

2.2.2 Experimental Results

A. Dataset Our experiments used two real-world datasets: Movielens 1 million movie review dataset [36] and Steam game platform dataset [10]. We filter out users that have less than 5 events. Movielens dataset contains 6040 users, 3416 items, and 163.5 average review sequence length per user. Steam dataset contains 334730 users, 13047 items, and 12.26 average sequence length per user. We used only the implicit feedback from the dataset except for the rating, review contents, etc. That is, in the dataset we use, each user has a sequence consisting only of (item ID, timestamp) tuples. For each user, we used the most recent item for test, same as [13]. We used the second most recent item for verification and the rest for training.

B. Baselines We used common baseline methods, which were generally used in the recommendation system and compared them with our model. As described in Figure 2.6, the baseline methods were POP, BPR, FMC, FPMC, GRU4Rec and GRU4Rec+. Pop and BPR are methods of not considering sequences, FMC and FPMC are methods of considering the last visited item, and GRU4Rec(+) are RNN-based models.

Pop: This model depends on how often each item was used previously. It only counts the total number of previous item uses.

BPR [21]: This method applies Bayesian Personalized Ranking to matrix factorization.

FMC: This method uses the last used item and a first-order marcov chain to recommend the next item.

FPMC [37]: This method uses a combination of matrix factorization and markov chain. Therefore, unlike FMC, FPMC can reference not only the last item used but also the further past item.

GRU4Rec [7]: This Session-based RNN method applies RNN to e-commerce to predict the items that the user will click on next through three layers (i.e. Embedding-GRU-Feedforward), based on items that the user has clicked on in the past; it uses a loss function based on item ranking. We use this for the movie dataset.

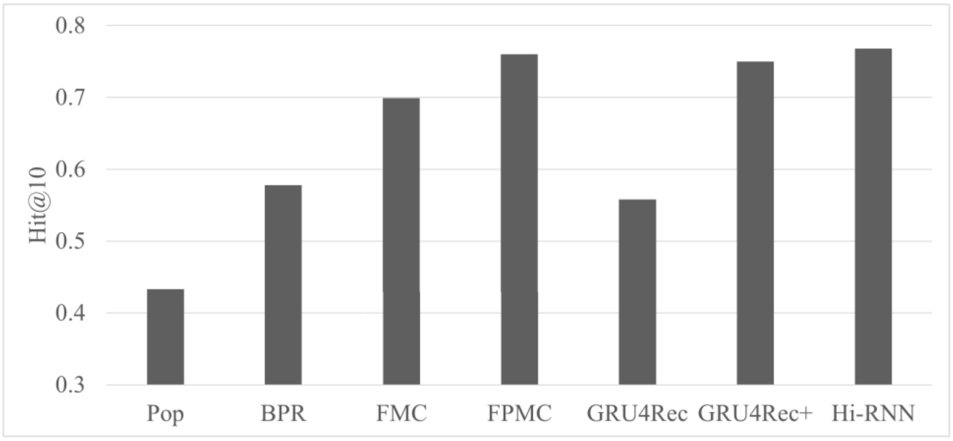


Figure 2.6: Hit@10 for movielens1M dataset using the baseline methods and our method.

GRU4Rec+ [8]: This model improves GRU4Rec by changing a loss function and a sampling strategy.

C. Results Here we use Hit@k as the measurement for comparison because Recall is well adapted to top-N recommendation system evaluation. Each method presents k candidates that are likely to be used next and we check the accuracy of whether the next item actually used was in the candidate group. Because this study predicts which item will be used at the next time, the correct answer is only one item. When predicting k candidate items, $Hit = \#correct/\#user$. To reduce the amount of calculation, we compared ground truth with 100 negative items that were randomly sampled instead of comparing it with all items [29].

Figure 2.6 shows the results of comparing the baseline models and the models presented in this study. We observe that the experimental results of our proposed model, Hi-RNN that consider the long-term user histories are significantly higher than those of other models. The hit@10 of the existing methods did not exceed 76%, so that of the model of this study was 78%. It is more than 10% different from Pop and BPR, which do not use any temporal property. GRU4Rec and GRU4Rec+ which consider the characteristics of sequence data by using RNN performs better than Pop and BPR,

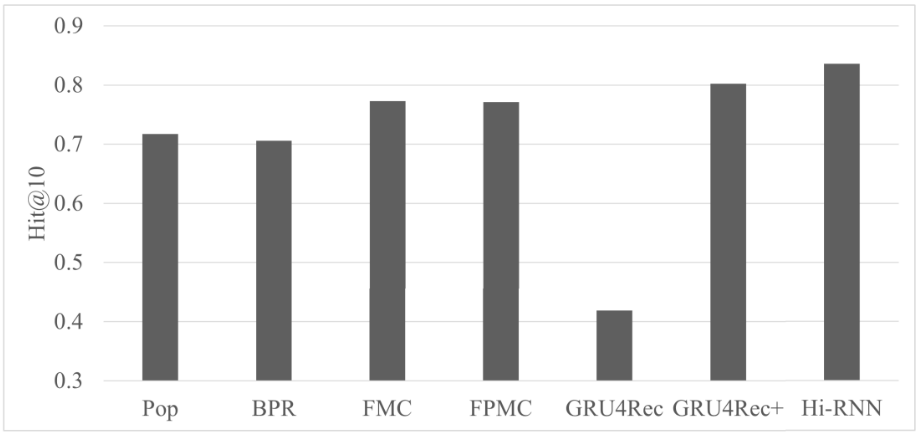


Figure 2.7: Hit@10 for Steam dataset using the baseline methods and our method.

but the performance of the Hi-RNN model which consider time interval is better. The performance difference from Pop and BPR was not significant on the Steam dataset compared to the Movielens dataset. We guess that since the average sequence length is longer in the Movielens dataset than in the Steam dataset, the importance of the temporal property increases as the sequence becomes longer, resulting in this difference.

We conducted an additional experiment to further examine the difference in the importance of the temporal property according to the sequence length. Figure 2.7 is the result of a comparative experiment with the variable maximum sequence lengths on the Movielens dataset. This experiments show a significant decrease in hit rate to 0.6 when Hi-RNN looks only the last 10 records. This hit rate is more similar to BPR, which does not consider the sequence. Its performance is lower than that of FMC, which only looks at its last visit. However, as we saw above, when long periods of data were used together, hi-RNN's performance was the best. This shows that considering the temporal property is more meaningful in the long-term sequence. In addition, if we examine the last 200 items, we obtain a very slightly higher hit rate, of 78%, than 100 items. We guess that because the data that are too old are not properly reflected in the user's current characteristics and the current item issues.

Experiments on various α values as Figure 2.8 show the importance of time interval. The closer the α is to 1, the greater the reflection of the data just before, re-

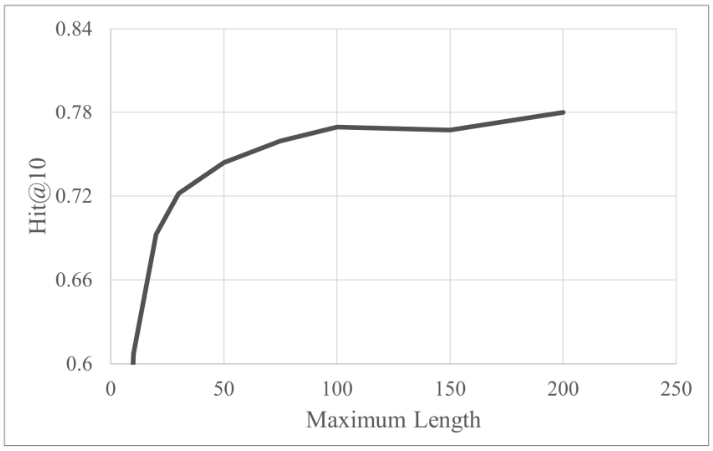


Figure 2.8: Hit@10 for movielens1M dataset with various maximum sequence lengths.

gardless of time interval. Conversely, if close to zero, the longer the time interval, the importance of the input just before becomes more like that just one of the past inputs. Experiments show the most improved performance when α is 0.5. This result means tl

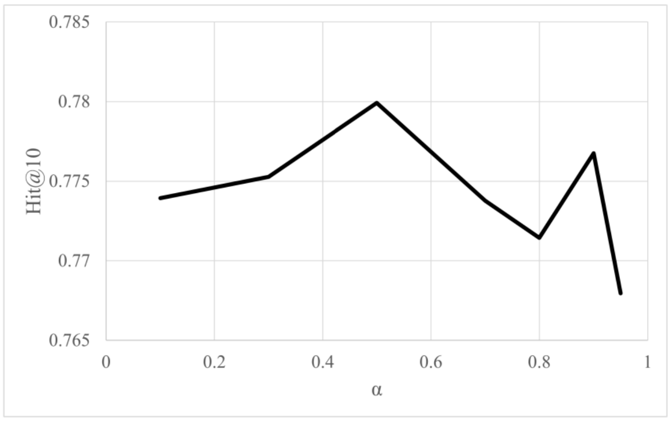


Figure 2.9: Hit@10 for movielens1M dataset with various α parameters.

2.3 Entangled Bidirectional Encoder to Auto-regressive Decoder for Sequential Recommendation

2.3.1 Model Description

Problem Statement

We denote set of users as $U = \{u_1, u_2, ..., u_{N_U}\}$, a set of items as $S = \{s_1, s_2, ..., s_{N_S}\}$ and the user interaction as $I_u = \{i_1, i_2, ..., i_{n_u}\}$ in chronological order for $u \in U$ where $i_t \in S$ is the item interacted with $u \in U$ and n_u is the length of user interaction u. The goal of sequential recommendation is to predict the next item i_{n_u+1} from the user interaction I_u.

Model Architecture

As illustrated in Figure 2.10, E-BART4Rec consists of a Bidirectional encoder, an Auto-regressive decoder from BART, and Gate Network.

Encoder: Bidirectional Encoder Representations from Transformer: The encoder network is composed of two sub-layers, a Multi-head attention network and a Point-wise feed-forward network, which is the same structure as [9]. For the Multi-head attention network in the encoder, let $h^l_t \in R^d$ denote the item hidden representations at each layer l for each position t and $H^l \in R^{n_u \times d}$ denote a matrix $(h^l_1, ..., h^l_{n_u})^T$, each single attention head linearly projects H^l to M subspaces and aligns projected matrix with itself, thereby calculating a latent representations between projected matrix itself:

$$\begin{aligned} Attention(Q, K, V) &= \text{softmax}(\frac{QK^T}{\sqrt{d/m}})V \\ x^l_m &= Attention(H^l W^Q_m, H^l W^K_m, H^l W^V_m), \end{aligned} \tag{2.4}$$

where the $x^l_m \in R^{n_u \times d}$ are the latent representations for each head m, and $W^Q_m \in R^{d \times d/M}, W^K_m \in R^{d \times d/M}$ and $W^V_m \in R^{d \times d/M}$ are linear projection matrices. The outputs of the Multi-head attention network are produced by concatenating all the latent representations in each single attention head:

$$X^l = Concat(x_1^l, x_2^l, ..., x_M^l)W^X, \tag{2.5}$$

where the $W^X \in R^{d \times d}$ are weights of the Multi-head attention networks. The Point-wise feed-forward network adds non-linearity and additional relation between dimension of the Multi-head representation X^l.

We also employ Dropout and Add & Norm layers, which consist of residual connection and layer-normalization followed by each sub-layers. Finally, the encoder output $O_E \in R^{n_u \times d}$ is computed through two sub-layers, dropout and Add & Norm layers.

Decoder: Auto-regressive model based on Transformer: The decoder network is composed of three sub-layers, a Masked Multi-head attention network, a Multi-head attention network, and a Point-wise feed-forward network. The Masked Multi-head attention network considers only the first t items when computing $(t + 1)$-th latent representations. Therefore, we mask the attention between $t^{'}$-th element in Q and t-th element in K $(t > t^{'})$ for the Masked Multi-head attention network.

To capture the relation between the latent representation of the encoder and decoder, the Multi-head attention of the decoder computes the latent representations between the latent representation of Masked Multi-head attention $\hat{H}^l$ and the encoder output O_E:

$$\begin{gathered} \hat{x}_i^l = Attention(\hat{H}^l \hat{W}_m^Q, O_E \hat{W}_m^K, O_E \hat{W}_m^V) \\ \hat{X}^l = Concat(\hat{x}_1^l, \hat{x}_2^l, ..., \hat{x}_M^l)W^{\hat{X}} \end{gathered} \tag{2.6}$$

We also use Dropout and Add & Norm layer followed by each sub-layers in the decoder for calculating the output of decoder $O_D \in R^{n_u \times d}$.

Entangled Encoder and Decoder using Gate Network: Gate network calculates how much the outputs of the encoder and decoder will be used to compute the next interaction [38, 39]. Both outputs are linearly transformed and summed in element-wise. Then, the influence weight $\beta \in R^{n_u \times d}$ is obtained by taking sigmoid function to this value as follows:

$$\beta = \sigma(O_E W_E + O_D W_D + b_l), \tag{2.7}$$

where $W_E \in R^{d \times d}$, $W_D \in R^{d \times d}$ and $b_l \in R^{n_u \times d}$ are learnable parameters, and σ is sigmoid function. The entangled output of the encoder and decoder $O \in R^{n_u \times d}$ is computed as weighted sum of O_D and O_E with β:

$$O = \beta * O_D + (1 - \beta) * O_E, \tag{2.8}$$

where $*$ denotes element-wise multiplication. Finally, we apply a feed-forward Network to O and compute output distribution over target items:

$$P(i_t | \{i_1, ..., i_{t-1}\}) = \text{softmax}(FFN(o_{t-1})E^T + b^F), \tag{2.9}$$

where $E \in R^{N_s \times d}$ is embedding matrix, b^F is bias term, o_t is the t-th position element of O.

Noisy Augmentation

Our model takes noisy data as an encoder input and then reconstructs original data in the decoder. We make noisy data by exploiting the four transformations, which consider users' behaviors in the real-world (e.g. skip behavior and loss of historical records). The four transformations are summarized as follows.

Item Deletion: We randomly delete items from the user interaction sequence:

$$Encoder : \{i_1, i_2, i_3, i_4, i_5\} \rightarrow \{i_1, i_3, i_5\} \tag{2.10}$$

Item Cropping: We randomly choose the index i between 1 and the length of the user interaction sequence. Then, we delete items from beginning to i-th in the user interaction sequence.

$$Encoder : \{i_1, i_2, i_3, i_4, i_5\} \rightarrow \{i_3, i_4, i_5\} \tag{2.11}$$

Item Reverse: We reverse the user interaction sequence, which is used in [40].

$$Encoder : \{i_1, i_2, i_3, i_4, i_5\} \rightarrow \{i_5, i_4, i_3, i_2, i_1\} \tag{2.12}$$

Item Infiling: Item spans are sampled from user interaction sequence, with span lengths drawn from a Poisson distribution. Then we replace the item spans with one mask token.

$$Encoder : \{i_1, i_2, i_3, i_4, i_5\} \rightarrow \{i_1, i_2, mask, i_5\} \tag{2.13}$$

In [19], noise transformation is used in pre-training, but we use noisy transformation to augment data and apply *Cloze* task to the noisy data.

Training Objective

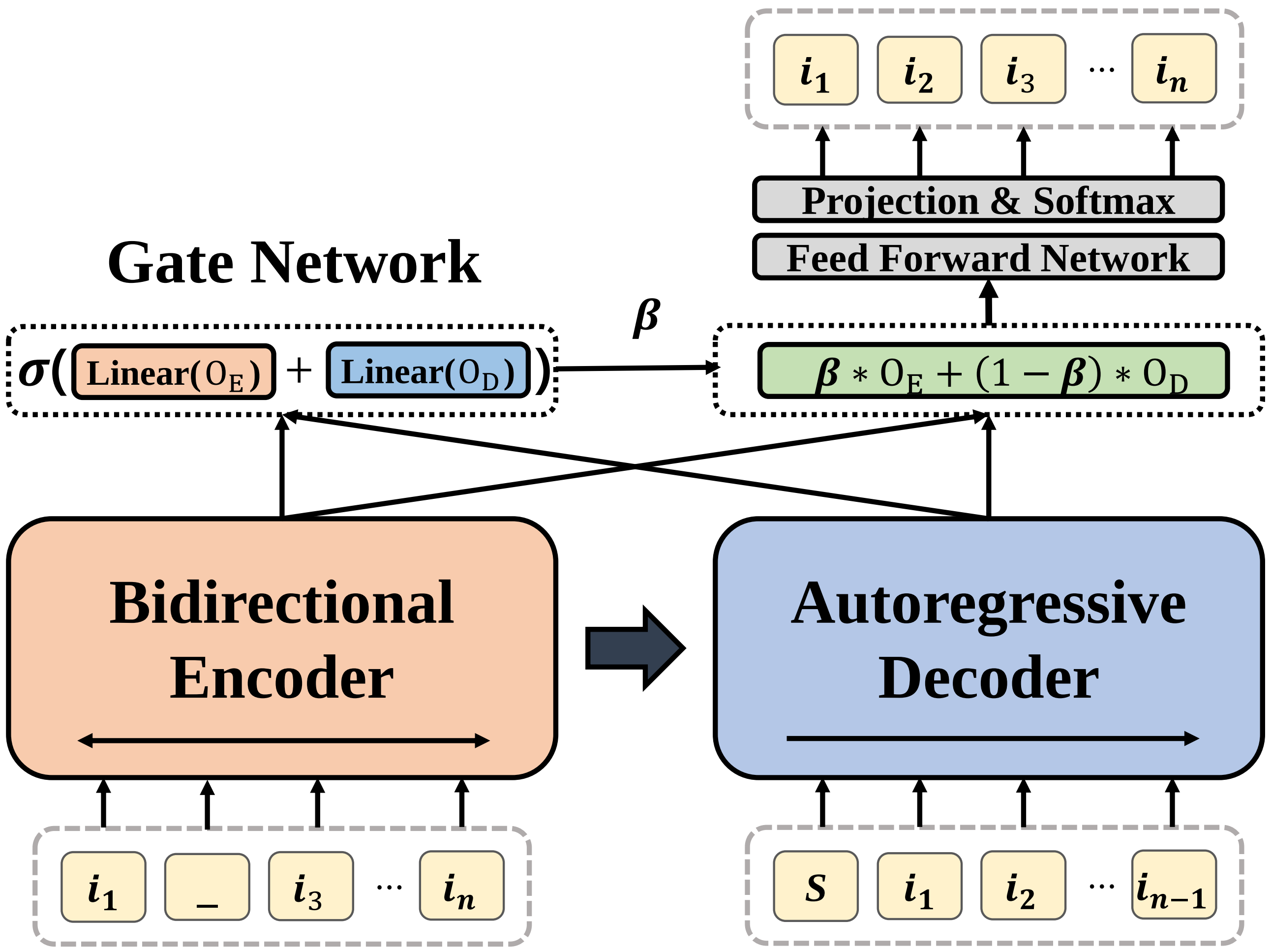
We jointly train a bidirectional encoder over noisy inputs and an auto-regressive decoder. For auto-regressive style, we use noisy data as an input of the encoder, $(n_u - 1)$ sequence as an input of the decoder, and the decoder is optimized to reconstruct the original sequence:

$$\begin{aligned} Input &: \{i_1, i_3, mask, i_5\}\ (\textit{Encoder}) + \{S, i_1, i_2, i_3, i_4\}\ (\textit{Decoder}) \\ target &: \{i_1, i_2, i_3, i_4, i_5\} \end{aligned} \tag{2.14}$$

In training, we define the loss for the user interaction sequence I_u as the negative log-likelihood of the targets:

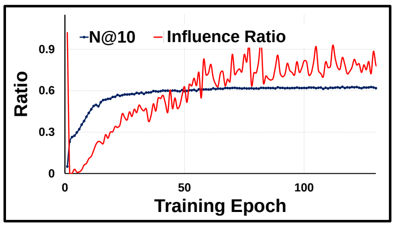
$$L = \frac{1}{|I_u|} \sum_{i_k \in I_u} - \log P(i_k = i_k^* | I_u^{'}), \tag{2.15}$$

where $I_u^{'}$ is the converted version for the encoder input and decoder input such as noisy augmentation. In inference, we substitute the last item in the encoder input with mask token and use $(n_u - 1)$ sequence as the decoder input.

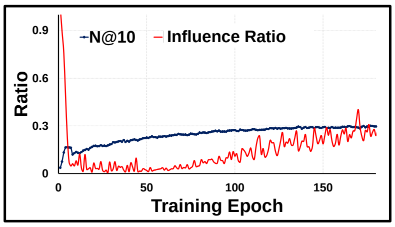


i_1 i_2 i_3 … i_n
Projection & Softmax
Feed Forward Network
Gate Network
β
$\sigma(\text{Linear}(O_E) + \text{Linear}(O_D))$
$\beta * O_E + (1-\beta) * O_D$
Bidirectional Encoder
Autoregressive Decoder
i_1 _ i_3 … i_n
S i_1 i_2 … i_{n-1}

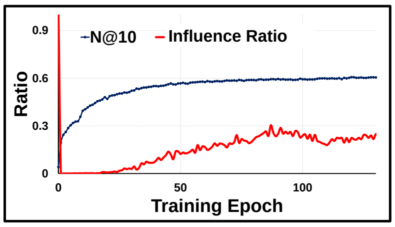
Figure 2.10: The overview of E-BART4Rec. The encoder takes noisy input sequence $i_1, \_, ...i_n$ and the decoder takes $(n-1)$ time input sequence with start token S. The Gate Network adjusts the final output between the encoder and the decoder.



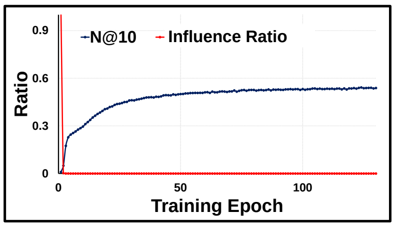
(a) MovieLens



(b) Last.fm



(c) ML-50



(d) ML-10

Figure 2.11: The *influence ratio* and NDCG@10 (denoted as N@10) for validation sets over training epoch.

2.3.2 Experimental Results

In this section, we conduct extensive experiments to answer the following research questions:

•**RQ 1**: Is E-BART4Rec effective for recommendation system compared to other baselines?

• **RQ 2**: How can the gate network improve the performance of E-BART4Rec compared to BART4Rec?

Experimental Settings

Datasets We evaluate our proposed methods on two widely-used real-world datasets: **(1)MovieLens**[1]: This is created by MovieLens for movie recommendation systems. **(2)Last.fm**[2]: This dataset is a music recommendation dataset obtained through the Last.fm. We filter out users that interacted with items less than 5 times and set the

[1]https://grouplens.org/datasets/movielens/
[2]http://millionsongdataset.com/lastfm/

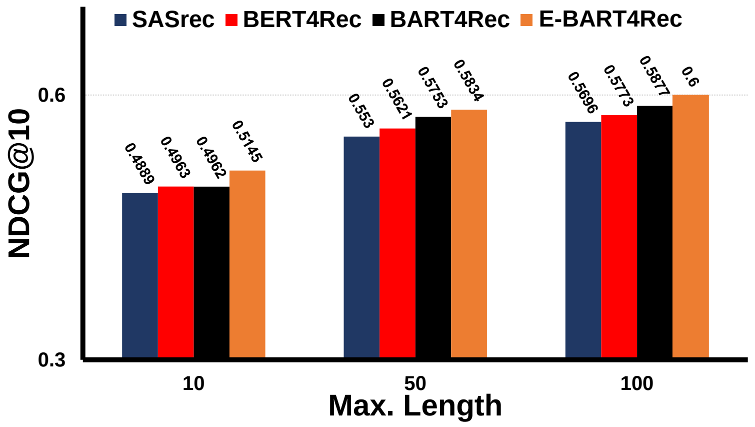


Figure 2.12: NDCG@10 comparison on SASRec, BERT4Rec, BART4Rec and E-BART4Rec on MovieLens with different maximum length.

maximum length as 100 for MovieLens and 50 for Last.fm respectively. After filtering, MovieLens contains 447407 interactions with 3517 items, and Last.fm contains 27368 interactions with 3622 items. The data sparsity of MovieLens and Last.fm are 97.90% and 99.31% respectively. **Evaluation metrics** We adopt NDCG and Hit-Ratio to evaluate the performance of the models. We employ NDCG@5, NDCG@10, Hit-Ratio@5, and Hit-Ratio@10 to report the results. We denote Hit-Ratio@K as Hit@K.

Baseline methods We compare our methods with the following methods: a popularity-based method (**POP**), four deep learning methods (**GRU4Rec** [7], **Caser** [18], **SASRec** [10] and **BERT4Rec** [9]), and two generative deep learning methods (**Seq2Seq** [16] and **BART4Rec** [19]).

Parameter setting We fix the batch size of all models to 128 on MovieLens and 256 on Last.fm. We set the layer size as 128, the number of layers as 4, and the number of heads as 4. For other methods, we set the hyper-parameters as the suggestions from the original papers.

Performance Comparison To answer RQ 1, we evaluate the performance of all the methods on both the datasets. The overall performances are summarized in Table 2.1, and the best results and second-best results are highlighted in a bold style and underlined respectively.

First of all, E-BART4Rec apparently achieves the best performance among the baselines on both the datasets, which supports that E-BART4Rec captures the dynamics from sequential information well. As compared to Seq2Seq and BART4Rec, we observe that E-BART4Rec consistently outperforms them across all the datasets. It confirms that our entangling method can enhance the auto-regressive model. BART4Rec also achieves a great performance all over the scores. In particular, BART4Rec outperforms Seq2seq on both the datasets, which indicates that noisy augmentations are effective.

Next, we note that SASRec and BERT4Rec consistently show a great performance across all scores. Moreover, BERT4Rec outperforms SASRec, which demonstrates that the bidirectional model captures the sequential information well. For a popularity-based method, POP shows relatively poor performance all over the scores. For deep learning models, we observe that performance order is consistent across all the scores (i.e. BERT4Rec $>$ SASRec $>$ Caser $>$ GRU4Rec). We note that Caser achieves better performance than GRU4Rec. Since Caser employs a noisy objective that substitutes the very next item target with the later one, Caser is robust to skip behaviors. This result demonstrates that injecting noise to data samples enhances the performance of recommendation models.

Analysis on Gate Network

To answer the RQ2, we compare $|\beta * O_D|$ and $|(1-\beta) * O_E|$, which indicate the influences of the encoder and decoder respectively, on four datasets: MovieLens, MovieLens with the maximum sequence length 50 (ML-50), MovieLens with the maximum sequence length 10 (ML-10), and Last.fm. In Figure 2.11, we report a mean value of $\frac{|\beta * O_D|}{|(1-\beta) * O_E|}$, which is called the *influence ratio*, and *NDCG@10* for validation set over training epoch. Surprisingly, the *influence ratios* converge to a certain value on all the datasets, especially zero on ML-10, which indicates that the auto-regressive decoder is not involved in inference at all. We also observe that the *influence ratio* on Movie-

Lens is relatively high compared to Last.fm, ML-50, and ML-10. These observations indicate that the gate network reduces the influence of the decoder when data is sparse or short in length. Additionally, we report the performance of SASRec, BERT4Rec, BART4Rec, and E-BART4Rec on MovieLens, ML-50, and ML-10 in Figure 2.12. We note that the performance order is inconsistent on BART4Rec and BERT4Rec on ML-10, which shows that BART4Rec does not fit with sparse or short-length data. However, E-BART4Rec shows robust performance on ML-10 by reducing the influence of the decoder. Therefore, we demonstrate that the gate network can enhance the robustness of E-BART4Rec by controlling the influence of the encoder and decoder.

Table 2.1: Performance comparison on different methods on two datasets. We highlighted the best performance and the second best performance methods in bold and underlined fonts respectively.

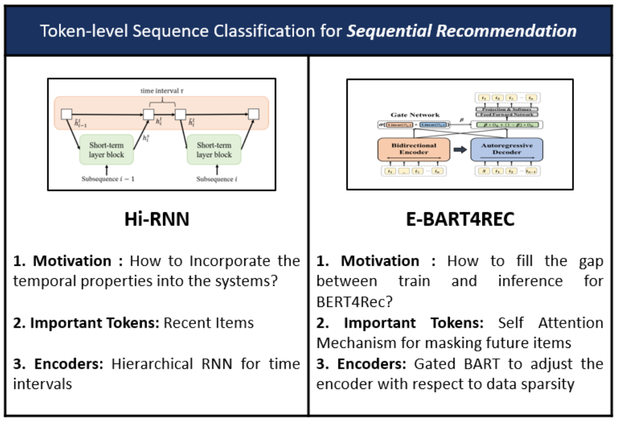
Dataset	Metric	*POP*	*GRU4Rec*	*Caser*	*SASRec*	*BERT4Rec*	*Seq2Seq*	*BART4Rec*	*E-BART4Rec*
MovieLens	NDCG@5	0.1898	0.4210	0.4789	0.5375	0.5468	0.5413	<u>0.5578</u>	**0.5686**
	NDCG@10	0.2366	0.5078	0.5288	0.5696	0.5773	0.5749	<u>0.5877</u>	**0.6000**
	Hit@5	0.2886	0.6157	0.6428	0.6897	0.6960	0.6839	<u>0.7042</u>	**0.7076**
	Hit@10	0.4368	0.7343	0.7721	0.7896	0.7900	0.7876	<u>0.7962</u>	**0.8043**
Last.fm	NDCG@5	0.1317	0.1122	0.1869	0.2151	0.2188	0.2122	<u>0.2295</u>	**0.2496**
	NDCG@10	0.1626	0.1477	0.2321	0.2627	0.2699	0.2649	<u>0.2737</u>	**0.3008**
	Hit@5	0.1940	0.1802	0.2839	0.3148	0.3283	0.3181	<u>0.3396</u>	**0.3578**
	Hit@10	0.3004	0.2708	0.4326	0.4735	<u>0.4886</u>	0.4815	0.4764	**0.5162**

2.4 Conclusion

In this dissertation, we have found the answer of the academic questions of TLC in sequential recommendation systems: 1) How to capture important tokens from sequential information and 2) How to encode a token sequence.

We have suggested new RNN and Transformer-based recommendation systems. First, we have proposed the RNN-based recommendation system considers the order and time interval for the recommendation system. We construct the hierarchical model based on RNN that is effective in a long-term time series. We show that the model suggested in this study has high performance compared with several baseline models from experiments. The experiments indicates that it is important to use temporal properties in long-term time series datasets. As for future work, we believe that the use of additional information together will further improve the model. For example, a user relationship graph-based information between users called ‘‘trust’’ can be used together. Previously, there have been many studies that have used trust together make more personalized and accurate recommendations [41–44]. However, these studies did not consider temporal properties such as sequential information and temporal interval. We expect that advancing the short-term layer of the Hi-RNN model, which currently use the vanilla RNN structure, will allow the model to consider not only the information of one user but also the information of both that user and the highly trust users together. In addition, we believe that other information such as review content and ratings can also be further considered to enhance the model.

In the next, we proposed a novel recommendation system, E-BART4Rec that entangles a bidirectional encoder and an auto-regressive decoder, and employs noisy transformation for user interaction. Bidirectional Encoder Representation from Transformer for sequential Recommendation (BERT4Rec) [9] has achieved good performance by utilizing a bidirectional attention mechanism to model sequential user interactions. Despite its successful performance, bidirectional modeling has a gap between training and inference. To address this problem, we employ Bidirectional and



Token-level Sequence Classification for ***Sequential Recommendation***

Hi-RNN	E-BART4REC
1. Motivation : How to Incorporate the temporal properties into the systems?	**1. Motivation :** How to fill the gap between train and inference for BERT4Rec?
2. Important Tokens: Recent Items	**2. Important Tokens:** Self Attention Mechanism for masking future items
3. Encoders: Hierarchical RNN for time intervals	**3. Encoders:** Gated BART to adjust the encoder with respect to data sparsity

Figure 2.13: Summary of Chapter 2

Auto-Regressive Transformer (BART) [19] which can incorporate Left-to-right modeling and Injecting noise into bidirectional modeling. Nevertheless, there are still many restrictions on adapting BART for sequential recommendation such as demanding dense datasets. To overcome these limitations, our E-BART4Rec entangles bidirectional model and auto-regressive model, and adopts noisy transformation for user interaction. Our extensive experiments on real-world datasets demonstrate the effectiveness of E-BART4Rec. Furthermore, the qualitative analysis shows that E-BART4Rec controls the influence of bidirectional encoder and auto-regressive decoder according to data sparsity and length.

Finally, we summarized the academic questions and their corresponding answers of our proposed models in Figure 2.13. For capturing important tokens, HI-RNN focuses on recent items and E-BART4Rec uses self-attention mechanisms for masking future items. For designing encoder, Hi-RNN employ hierarchical architectures for time intervals, and E-BART4Rec uses gated BART to adjust the encoder with respect to data sparsity.

Chapter 3

Token level classification in Information Extraction

3.1 Overview

3.1.1 Introduction to TLC in Information Extraction

In this chapter, we aim to solve the answer of the academic questions of TLC in information extraction systems: 1) How to capture important tokens from sequential information and 2) How to encode a token sequence.

Aspect-based Sentiment Analysis (ABSA) is one of the important task in Natural Language Processing (NLP) [1–3,45]. Generally, ABSA attempts to find the sentiment polarity of aspect in the review sentence. The key point of ABSA is to find the opinion words that describe the aspect target. Therefore, Target Opinion Words Extraction (TOWE) is emerged to extract the opinion words. Recently, many deep learning-based models have been proposed for addressing TOWE task such as Recurrent Neural Network (RNN) [46, 47] and Graph Neural Network (GNN) [48, 49].

In this dissertation, we have proposed Transformer-based information extraction systems. First, we propose the BERT-based TOWE models that can fully utilize the BERT (Chapter 3.2). Much relation information in sentence is important for TOWE task. Therefore, we also propose new information extraction system that can consider target relation and neighbor relation (chapter 3.3). All of our works have attempted to

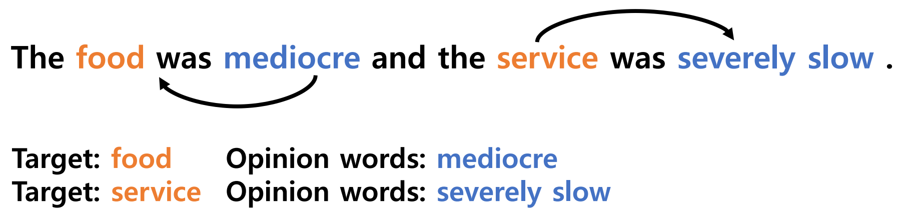

The food was mediocre and the service was severely slow .
Target: food Opinion words: mediocre
Target: service Opinion words: severely slow

Figure 3.1: An example of TOWE. TOWE aims at extracting the corresponding opinion words (colored in blue) for the given targets (colored in orange).

solve the academic questions. Finally, we summarize and compare the our recommendation systems (chapter 3.4).

3.1.2 Introduction to RABERT

Targeted Opinion Word Extraction (TOWE) [46] is a newly emerged subtask of aspect-based sentiment analysis (ABSA) [45,50,51]. Given a review and a target word within the review, the object of TOWE is to identify opinion words corresponding to the target. Opinion targets are the words or phrases representing aspects or entities towards which users show their attitude in the review, and opinion words refer to the terms used to express attitudes or opinions of the users about the opinion targets explicitly. As shown in Figure 3.1, in the sentence *"The food was mediocre and the service was severely slow."*, *"mediocre"* is the corresponding opinion word for the target *"food"* while the opinion word span *"severely slow"* corresponds to target *"service"*.

The important point for addressing the TOWE task is that even the same sentences can vary in the opinion words depending on the opinion target. Therefore, a core challenge of the TOWE task is to learn a target-aware context representation, which incorporates opinion target information into a contextualized sentence representation. [46] has proposed the IO-BiLSTM which encodes the left context and the right context of the opinion target in the sentence separately to indicate the position of the opinion targets. Subsequent works have obtained the target-aware context representation by introducing the opinion target indicator embeddings into various Deep Neural Networks (DNNs) model such as BiLSTM [47] and Graph Neural Networks (GNNs) [48].

Nevertheless, it has been unclear whether we can introduce the target information into BERT [52], a powerful pre-trained language model, to learn the target-aware context representation. There have been some approaches to use BERT for addressing the TOWE task [48, 49]. However, they simply use BERT for encoding sentences without any opinion target information and just concatenate the target indicator embedding to the output of BERT. Thus, BERT cannot be fully utilized to capture the relation information between the opinion targets and the sentence.

In this paper, to tackle this limitation, we propose a BERT-based target-aware context representation model for the TOWE task. We first add the markers indicating opinion target words to the sentence in pre-process and obtain context representations by encoding this sentence into BERT layers. This simple marker can help retain the target information in BERT layers. Furthermore, our model employs a novel target-context relation network to catch the relation information between the opinion target and the sentence. To extract a target representation from the context representation, we collect a part corresponding to the operation target word of the context representation and pool the collected representation. Then, the target-sentence relation network merges the target representation and the context representation and produces the target-aware context representation by considering the relation between them.

In addition to the relation between the opinion targets and the sentence, the relation among the neighbor words, which are left and right side words for a given word, is also important for identifying opinion words. For example, as shown in figure 3.1, *"mediocre"* and *"slow"* have a dependency on their corresponding target *"food"* and *"service"* respectively, but *"severely"* is mainly dependent on *"slow"* which is neighbor word of *"severely"*. Therefore, learning only the target-aware context representation cannot be enough to solve the TOWE task.

To capture the neighbor words information, we also design the neighbor-aware relation network which extracts the relation among the neighbor words. Specifically, we make a neighbor representation by shifting each word of context representation one by one, right or left. Then the neighbor-aware relation network merges the context representation and its neighbor representation and obtains the neighbor-aware context representation from them.

By combining these two methods, we propose a powerful target-oriented opinion words extraction model, named **RABERT** (**R**elation-**A**ware **BERT**) that considers both the relations between opinion targets and a sentence and the relations among neighbor words in a sentence. RABERT can fully utilize the power of BERT to obtain target-aware context representation, and capture the two important relations using the target-sentence relation network and the neighbor-aware relation network. We conduct experiments on four benchmark dataset, *14res*, *14lap* [1], *15res* [2] and *16res* [3]. The results of our experiments demonstrate that our model significantly outperforms the other baselines and achieves a new state-of-the-art performance of the TOWE task. In addition, further analysis validates the effectiveness of each component of our model.

Table 3.1: Example of user reviews and their extracted pairs of *opinion targets* and *opinion words*. The red-colored words and the blue-colored words represent *opinion targets* and *opinion words*, respectively.

User Reviews:
”The *service* is *amazing* and *food* is *out of this world*. I ordered the *Clams Oreganato appetizer*. It was *great* and *top notch*. The Parm lunch special came with pasta and choice of soup or salad. I had a *delicious meal* with *great service*. Awesome! ”
Extracted *opinion targets* - opinion pairs:
1. *service* - *amazing*
2. *food* - *out of this world*
3. *Clams Oreganato appetizer* - *great, top notch*
4. *meal* - *delicious*
5. *service* - *great*

3.1.3 Introduction to GRED

Target-oriented Opinion Word Extraction (TOWE) [46] is a recently designed task from aspect-based sentiment analysis (ABSA) [45, 50, 51]. In TOWE, entities or objects towards which users show their attitudes are regarded as the opinion targets. Correspondingly, those terms explicitly expressing the attitudes are defined as the opinion words. Given a sentence and opinion targets, the goal of TOWE is to extract the opinion words describing the opinion targets. For example, in the sentence *”The service is amazing and food is out of this world.”*, TOWE identifies the word *”amazing”* as the opinion word for *”service”*, and the terms *”out of this world”* as the opinion words for *”food”*. Table 3.1 shows more example pairs of opinion targets and opinion words.

Extracting opinion targets or opinion words has been widely used in various Nat-

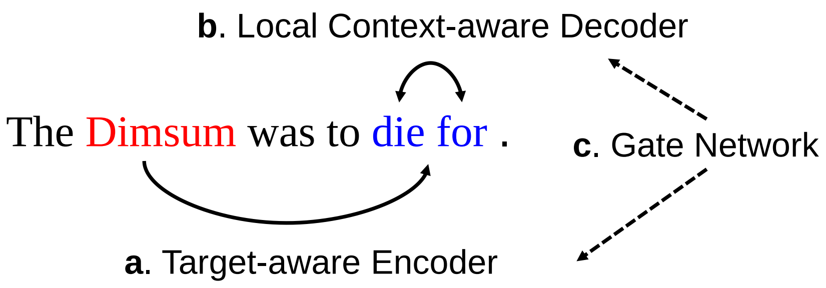


Figure 3.2: The concept of GRED. The target-aware encoder finds target-dependent opinion words, and the local context-aware decoder seeks local context-dependent opinion words. Then, the gate network dynamically aggregates the outputs of these two networks.

ural Language Processing (NLP) tasks such as sentiment analysis [53–55] and text mining [56–58]. In this trend, TOWE has become more important because it explicitly informs the correlations between opinion targets and opinion words. To address this task, Fan *et al.* [46] have released four benchmark datasets across different domains. Furthermore, they have formalized TOWE as a sequence labeling problem for a sentence with given opinion targets.

To solve TOWE, previous works have primarily focused on how to incorporate the opinion target information into the sentence representations. In this literature, there have been various attempts to encode the opinion target such as using an additional BiLSTM-based target encoder [46] or adopting a target position embedding [47, 48]. With the development of Pre-trained Language Models (PLMs) [19, 52, 59], recent works have added a special token indicating the opinion targets to exploit the power of PLMs, which have shown promising results [60, 61].

Nevertheless, these approaches have a critical limitation that they cannot effectively utilize the surrounding words of the opinion words. Understanding such local context as well as the opinion target information is helpful in extracting the opinion words. For example, in the sentence *"The Dimsum was to die for"*, a human can easily identify the word *"for"* as the opinion words by considering the opinion target *"Dimsum"* and the surrounding word *"die"* together. Although [61] have attempted to incorporate this local context information into PLMs, they merely put the target information with the local context information together, resulting in congested sentence representation. Consequently, their performance has shown limited improvements only where the local context information has less importance.

In this paper, we propose a **G**ated **R**elational target-aware **E**ncoder and local context-aware **D**ecoder based sequence labeling model (**GRED**), which dynamically leverages the opinion target information and the local context information for TOWE. Specifically, the target-aware encoder first obtains the opinion target information by using the target relation network. Simultaneously, local context-aware decoder captures the local context information from the relationships among surrounding words by using the local context relation network. Then, GRED employs the gate network to aggregate the outputs of the encoder and the decoder. The role of the gate network is to determine how much those outputs will impact on the final prediction. Therefore, GRED can properly mix the opinion target information and the local context information. The concept of GRED is illustrated in Fig. 3.2. In addition, to improve the language knowledge of both the encoder and the decoder, we adopt a pre-trained language model BART [19] as the structure of GRED.

We evaluate our GRED on the four benchmark datasets, *14res*, *14lap* [1], *15res* [2] and *16res* [3]. The results of our extensive experiments demonstrate that GRED outperforms the baselines and achieves new state-of-the-art performance for TOWE task. Additionally, further comprehensive analyses validate the effectiveness of each component of GRED. In summary, the main contributions of this work are as follows:

- We propose GRED for TOWE, which dynamically leverages a target-aware encoder and a local context-aware decoder via a gate network.

- We show that GRED achieves the state-of-the-art performances on four widely-used benchmark datasets through our extensive experiments.

- With our comprehensive analysis on GRED, We demonstrate the effectiveness of GRED in various perspectives.

3.1.4 Related Works

Extracting opinion targets and opinion words have been principal tasks for natural language processing. One line of this research has focused on opinion target extraction (OTE), which aims to seek the opinion target aspect terms in the sentences [62–64]. In other approaches, opinion words extraction (OWE) has attempted to locate the words expressing the users' attitude [65, 66]. Recently, several works have proposed a co-extraction framework that extracts the opinion targets and opinion words jointly. They have detected the targets and opinion words jointly by utilizing a word alignment model [67] or multi-task learning [68–70]. However, all these works have still not considered the relationship between opinion targets and opinion words.

To study this, there have been researches conducted on the task of extracting corresponding opinion words for the given opinion targets. Classical methods have been designed to seek corresponding opinion terms based on word distance [71] and dependency parsing tree [56]. However, these methods require external knowledge and show vulnerability to diverse patterns of data. Subsequent works have integrated opinion target information into the context and extracted the corresponding opinion words using deep neural networks such as RNN [46,47] and GCN (Graph Convolutional Network) [48, 49]. In these works, [46] have first proposed an end-to-end neural network model using IOG (Inward-Outward LSTM + Global context) to fuse opinion target information with the global context but have suffered from high model complexity. In contrast to IOG, [47–49, 72] have employed position embedding of opinion targets, resulting in not increasing the model complexity excessively.

Nevertheless, the above methods cannot fully utilize the powerful pre-trained language models to address TOWE task. Thus, recent works have adopted the pre-trained language models and achieved promising results [60, 61, 73]. They have attempted to incorporate opinion target information into PLMs by modifying the input sentence with explicitly marking the opinion targets. Inspired by the previous works, our proposed method GRED also adopts this strategy to obtain the opinion target information. However, GRED is different from the pre-trained language model-based methods as GRED dynamically mixes the two important information; opinion target information and local context information.

3.1.5 Backgrounds

Target-oriented Opinion Words Extraction

In TOWE, we regard entities or objects towards which users show their attitudes as the opinion targets. Also, we define terms explicitly expressing the attitudes as the opinion words. Given a sentence and opinion targets, TOWE attempts to extract the opinion words describing the opinion targets. In detail, TOWE can be formalized as a problem of sequence labeling for given targets. Given a review sentence $\mathbf{s} = \{w_1, w_2, ..., w_n\}$ consisting of n words, and an opinion target word in the sentence **s**, the goal of TOWE task is to tag each words as $y_i \in \{B, I, O\}$ (B: Beginning, I: Inside, O: Others).

Transformer

In this section, we briefly review the transformer architecture [16] in TOWE systems. In Multi-head attention, we first denote $h_t^l \in R^d$ the hidden representations at each layer l for each position t and $H^l \in R^{n_u \times d}$ denote a matrix $(h_1^l, ..., h_{n_u}^l)^T$. Then, each single attention head linearly projects H^l to M subspaces and aligns projected matrix with itself, thereby calculating a latent representations between projected matrix itself:

$$\begin{aligned} Attention(Q, K, V) &= \text{softmax}(\frac{QK^T}{\sqrt{d/m}})V \\ x_m^l &= Attention(H^l W_m^Q, H^l W_m^K, H^l W_m^V), \end{aligned} \tag{3.1}$$

where the $x_m^l \in R^{n_u \times d}$ are the latent representations for each head m, and $W_m^Q \in R^{d \times d/M}, W_m^K \in R^{d \times d/M}$ and $W_m^V \in R^{d \times d/M}$ are linear projection matrices. The outputs of the Multi-head attention network are produced by concatenating all the latent representations in each single attention head:

$$X^l = Concat(x_1^l, x_2^l, ..., x_M^l)W^X, \tag{3.2}$$

where the $W^X \in R^{d \times d}$ are weights of the Multi-head attention networks. The Point-wise feed-forward network adds non-linearity and additional relation between dimension of the Multi-head representation X^l.

In this dissertation, we employ the transformer-based Pre-trained Language Model (PLM) such as BERT [15] and BART [19]. For BERT, cloze tasks and *Is Next Sentence* problem are used to pre-train the model. BERT only uses the encoder part of transformer. For BART, various noisy masked language model problems are used to pre-train the model. BART employs the encoder and decoder parts of the transformer.

3.2 RABERT: Relation-Aware BERT for Target-Oriented Opinion Words Extraction

3.2.1 Model Description

Task Formalization

We formalize TOWE as a problem of sequence labeling for given targets. Given a review sentence $\mathbf{s} = \{w_1, w_2, ..., w_n\}$ consisting of n words, and an opinion target word in the sentence $\mathbf{s}$, the goal of TOWE task is to tag each words as $y_i \in \{B, I, O\}$ (B: Beginning, I: Inside, O: Others). For example, the sentence in figure 3.1, *"The food was mediocre and the service was severely slow ."* is tagged as *"The/O food/O mediocre/B and/O the/O service/O was/O severely/O slow/O ./O"* for the given target as *"food"* as target and *"The/O food/O mediocre/O and/O the/O service/O was/O severely/B slow/I ./O"* for the given target as *"service"*.

RABERT architecture

As illustrated in figure 3.3, RABERT consists of a Text Encoder, Target pooler, Target-sentence relation network, neighbor-aware relation network, and a Conditional Random Field. For ease of explanation, we assume that the opinion target words span contains two words: $\mathbf{W}_{span} = \{w_t, w_{t+1}\}$.

Opinion Target Marker and Text Encoder. The text encoder takes the sentence with the given target word span and produces the context representations . We use the pre-trained language model BERT [52] for the text encoder. In previous works [47–49], to incorporate the given opinion target into the sentence, they have used target indicator embeddings with word embeddings. Instead, we simply add a target marker referring to the position of opinion target to the sentence to retain target information in BERT layer. The target markers are added before and after the opinion target words span such as $\mathbf{s}' = \{w_1, ..., \langle \mathbf{s} \rangle, w_t, w_{t+1}, \langle \mathbf{s} \rangle, ..., w_n\}$, where $\langle \mathbf{s} \rangle$ is the target markers. Then, the text encoder generates the context representations $\mathbf{H} = \{h_1, ..., h_t, h_{t+1}, ..., h_n\}$ given $\mathbf{s}'$.

Target Representation. We make target representation by extracting target words span information from the context representation. First, we collect the part corresponding to the opinion target word of the context representations, $\mathbf{H}_{span} = \{h_t, h_{t+1}\}$, and then apply pooling layer to $\mathbf{H}_{span}$. In previous works [74, 75], it has shown that *Log-SumExp* (**LSE**) is effective in representing target or entity mention within a sentence . Thus, we employ **LSE** to pool the collected target word representations as follows:

$$h_{tar} = \mathbf{LSE}(\mathbf{H}_{span}), \tag{3.3}$$

where $\mathbf{LSE}(\mathbf{H}_{span}) = \log \sum_{h_i \in \mathbf{H}_{span}} \exp(h_i)$ and $h_{tar} \in \mathbb{R}^d$ is a target representation.

Target-Sentence Relation Network. To capture the relation between the opinion target words and the sentence, we collate the target representation h_{tar} with each word representation in the context representation $\mathbf{H}$. For each $h_i \in \mathbf{H}$, we concatenate h_{tar} and h_i, and apply a *Position-wise Feed-Forward Network* (**PFFN**) to it. Then, the target-aware context representation $\overline{\mathbf{H}} \in \mathbb{R}^{n \times d}$ is computed by capturing the relation between h_{tar} and $\mathbf{H}$ as:

$$\begin{aligned} \overline{\mathbf{H}} &= \mathbf{PFFN}(\{h_1 \oplus h_{tar}, h_2 \oplus h_{tar}, ..., h_n \oplus h_{tar}\}), \\ \mathbf{PFFN}(\{x_1, x_2, ..., x_n\}) &= \{\mathbf{FFN}(x_1), \mathbf{FFN}(x_2), ..., \mathbf{FFN}(x_n)\}, \\ \mathbf{FFN}(x) &= \tanh(\mathbf{A}x + \mathbf{b}), \end{aligned} \tag{3.4}$$

where $\oplus$ is concatenate operator, and $\mathbf{A} \in \mathbb{R}^{d \times 2d}$ and $\mathbf{b} \in \mathbb{R}^d$ are learnable parameters.

Neighbor-Aware Relation Network. The Neighbor-Aware relation network extracts the relations between the word and its neighbor words within the sentence. Since the sentence can be represented as an un-directed chain graph, the neighbor words only contain a left side word and a right side word of a given word. Thus, we make neighbor representations $\overline{\mathbf{H}}_{-1}$ and $\overline{\mathbf{H}}_{+1}$, which are shifting the elements of $\overline{\mathbf{H}}$ by one to right and left respectively as:

$$\overline{\mathbf{H}}_{-1} = \{\overline{h}_0, \overline{h}_1, ..., \overline{h}_{n-1}\},$$
$$\overline{\mathbf{H}}_{+1} = \{\overline{h}_2, \overline{h}_3 ..., \overline{h}_n, \overline{h}_0\}, \tag{3.5}$$

where $\overline{h}_0 \in \mathbb{R}^d$ is zero vector. Then, we collate $\overline{\mathbf{H}}$ with $\overline{\mathbf{H}}_{-1}$ and $\overline{\mathbf{H}}_{+1}$ respectively. A representation of within sentence relation are computed using **PFFN** as:

$$\mathbf{L}_{-1} = \mathbf{PFFN}(\overline{\mathbf{H}} \oplus \overline{\mathbf{H}}_{-1}),$$
$$\mathbf{L}_{+1} = \mathbf{PFFN}(\overline{\mathbf{H}} \oplus \overline{\mathbf{H}}_{+1}), \tag{3.6}$$

where $\mathbf{L}_{-1} \in \mathbb{R}^{n \times d}$ and $\mathbf{L}_{+1} \in \mathbb{R}^{n \times d}$ are the representations of within sentence relation. Then we combine each representation of within sentence relation, and obtain the final neighbor-aware context representation as:

$$\mathbf{R} = \mathbf{PFFN}(\overline{\mathbf{L}}_{-1} \oplus \overline{\mathbf{L}}_{+1}), \tag{3.7}$$

where $\mathbf{R} = \{r_1, ..., r_n\}$ is final representation of RABERT. The final representation **R** entangles the target word span information and the neighbor word information. Next, we use the decoder for sequence labeling based on **R**.

Conditional Random Field (CRF). We use Conditional Random Field (**CRF**) to capture structural dependency of the sentence as in [46]. Specifically, we employ a multiple linear-chain **CRF**s and score the sequence labeling as conditional probability:

$$p(\mathbf{y}|\mathbf{R}) = \frac{\exp(s(\mathbf{R}, \mathbf{y}))}{\sum_{y' \in Y} \exp(s(\mathbf{R}, \mathbf{y}))}, \tag{3.8}$$

where $\mathbf{y} = \{y_1, ..., y_n\}$ is corresponding label sequence and $y_i \in \{B, I, O\}$, Y is the set of all possible label sequences and $s(\mathbf{R}, \mathbf{y}) = \sum_{i=0}^{n}(\mathbf{T}_{y_{i-1}, y_i} + \mathbf{E}_{i, y_i})$ is the score function. $\mathbf{T}_{y_{i-1}, y_i}$ is the transition score from label y_{i-1} to y_i and $\mathbf{E}_{i, y_i} = \mathbf{A}_E\mathbf{R} + \mathbf{b}_E$ is the emission score. **CRF**s use the negative log likelihood as the loss function:

$$Loss(\mathbf{s}) = -\log(p(\mathbf{y}|\mathbf{R})) \tag{3.9}$$

We minimize the $Loss(\mathbf{s})$ for training. For inference, the model generates the tag sequences that maximize $p(\mathbf{y}|\mathbf{R})$ based on the Viterbi algorithm.

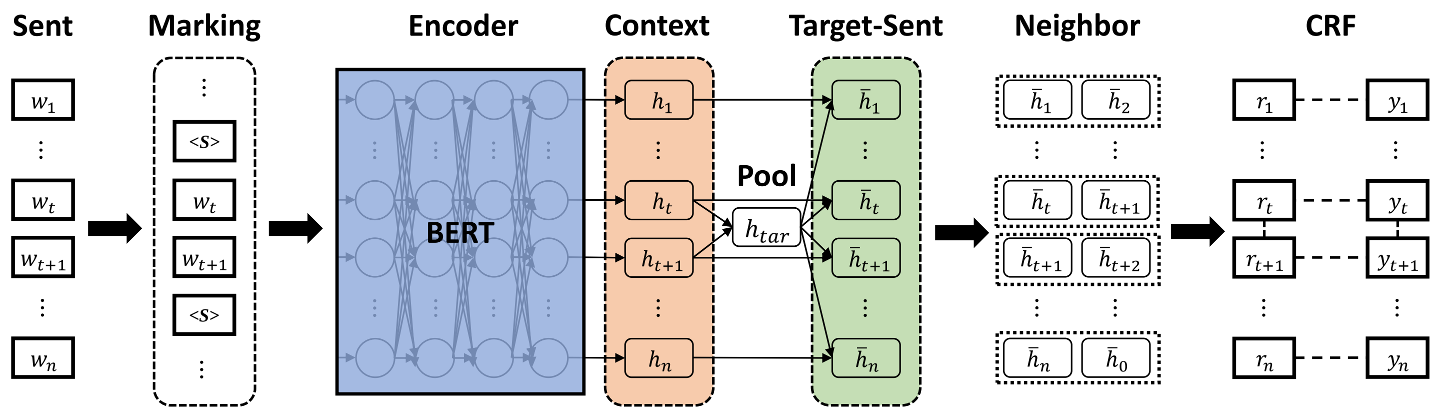


Figure 3.3: The overview of RABERT architecture. RABERT consists of a Text Encoder, Target pooler, Target-sentence relation network, Neighbor-aware relation network, and a Conditional Random Field, which are denoted as Encoder, Pool, Target-Sent, Neighbor, and CRF respectively in the figure. Marking represents adding target markers to sentences.

Table 3.2: Statistics of the four datasets

Datasets		# of sentences	# of targets
14res	Train	1627	2643
	Test	500	865
14lap	Train	1158	1634
	Test	343	482
15res	Train	754	1076
	Test	325	436
16res	Train	1079	1512
	Test	329	457

3.2.2 Experimental Results

Experimental Setup

Datasets. We conduct experiments on four datasets: ***14res***, ***15res*** and ***16res*** are the restaurant review datasets from SemEval challenge 2014 [1], 2015 [2], and 2016 [3] respectively. ***14lap*** is the laptop review datasets from SemEval challenge 2014 [1]. To formulate the TOWE task, we use newly annotated datasets from [46]. The statistics of all the datasets are summarized in Table 3.2.

Evaluation metrics. Following the previous works [46, 47], precision (denoted as P), recall (denoted as R), and F1 score (denoted as F1) are adopted as the metrics to measure the performance of models. We regard extracted opinion words span as a correct prediction when the starting and ending positions of the predicted span are both the same as those of the golden span.

Baselines. We compare our methods with the following methods: Rule-based methods (Distance-rule and Dependency-rule), Recurrent Neural Network (RNNs) based methods (LSTM, BiLSTM, Pipeline, TC-BiLSTM, IOG [46], PE-BiLSTM and LOTN

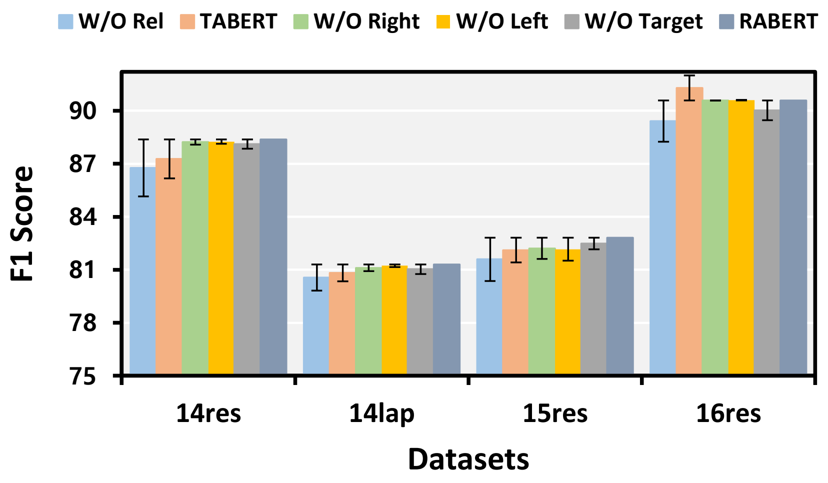


Figure 3.4: Ablation study on the four datasets. The error bars represent the F1 score difference between each model and RABERT.

[47]) and other DNN + BERT based methods (SDRN+BERT [76], ONG [48] and ARGCN+BERT [49]). We also report the performance of TABERT, a model without the relation network in RABERT.

Parameter setting. We fix the batch size as 32 and the maximum sequence length as 128. We consider the dropout rate from $\{0.1, 0.15, 0.2, 0.25, 0.3, 0.35\}$. We use *Adam* optimizer and consider the learning rate of BERT layer from $\{2 \times 10^{-5}, 3 \times 10^{-5}, 4 \times 10^{-5}, 5 \times 10^{-5}\}$ and **PFFN** from $\{1 \times 10^{-4}, 2 \times 10^{-4}, 3 \times 10^{-4}, 4 \times 10^{-4}\}$.

Experimental Results

Comparison with the other baselines. To demonstrate effectiveness of our model, we evaluate the performance of all the methods on the four benchmark datasets. The overall performances are summarized in Table 3.3. The best results are highlighted in bold style. First of all, we can observe that RABERT and TABERT are superior to the other baselines, and achieve the new state-of-the-art scores on most of the metrics. The experiments results show that RABERT achieves the best F1 score on *14res*, *14lap*, and *15res*, while TABERT achieves the best on *16res*. Comparing RABERT with the existing baselines, RABERT significantly outperforms the state-of-the-art model, ARGCN+BERT by 2.95%, 4.94%, 4.57%, and 3.89% F1 score respectively on all the datasets. These observations demonstrate that our models can effectively capture the target representations and the relations among the neighbor words. Next, we note that BERT-used methods (SDRN+BERT, ONG, and ARGCN+BERT) show a great performance across all the scores. For the rule-based methods, Distance-rule and Dependency-rule show relatively poor scores on most of the datasets because they cannot catch the sequential information of sentences. We also observe that the performances of the deep learning-based model with the target information are better than the methods without the target information (i.e. PE-biLSTM and TC-BiLSTM ¿ LSTM and BiLSTM). In particular, target indicator embedding methods such as PE-BiLSTM and LOTN obtain improved performance all over the scores. These observations show how important it is for TOWE tasks to make target information well introduced to models.

Ablation study. To validate the effectiveness of each component in RABERT, we eliminate different components from RABERT. The results of ablation study are illustrated in Figure 3.4. *W/O Rel*, *W/O Right*, *W/O Left*, and *W/O Target* denote removing both the relation networks, $\overline{\mathbf{H}}_{-1}$, $\overline{\mathbf{H}}_{+1}$, and the target-sentence relation network from RABERT, respectively. RABERT shows relatively good performance on most of the datasets, but not on *16res* because the model performance on *16res* has already been sufficiently convergent and no additional information is needed. The performance comparison between RABERT and *W/O Rel* validates the effectiveness of the relation network. *W/O Target* gets worse scores than RABERT on all the datasets, which indicates that the target-sentence relation network can catch the relation between the target and the sentence. Also, we demonstrate the effectiveness of the neighbor-aware relation network from the observation that TABERT, *W/O Right* and *W/O Left* show worse performance than RABERT on most of the datasets.

Table 3.3: Main Experimental Results(%). Performance comparison with the existing baselines on four datasets.

Models	*14res*			*14lap*			*15res*			*16res*		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Distance-rule	58.39	43.59	49.92	50.13	33.86	40.42	54.12	39.96	45.97	61.90	44.57	51.83
Dependency-rule	64.57	52.72	58.04	45.09	31.57	37.14	65.49	48.88	55.98	76.03	56.19	64.62
LSTM	52.64	65.47	58.34	55.71	57.53	56.52	57.27	60.69	58.93	62.46	68.72	65.33
BiLSTM	58.34	61.73	59.95	64.52	61.45	62.71	60.46	63.65	62.00	68.68	70.51	69.57
Pipeline	77.72	62.33	69.18	72.58	56.97	63.83	74.75	60.65	66.97	81.46	67.81	74.01
TC-BiLSTM	67.65	67.67	67.61	62.45	60.14	61.21	66.06	60.16	62.94	73.46	72.88	73.10
IOG	82.38	78.25	80.23	73.43	68.74	70.99	72.19	71.76	71.91	84.36	79.08	81.60
PE-BiLSTM	80.10	76.51	78.26	72.01	64.20	67.83	70.36	65.73	67.96	82.27	74.95	78.43
LOTN	84.00	80.52	82.12	77.08	67.62	72.02	76.61	70.29	73.29	86.57	80.89	83.62
SDRN+BERT	**91.14**	76.37	83.10	**84.37**	65.42	73.69	83.57	70.33	76.38	91.13	80.34	85.40
ONG	83.23	81.46	82.33	73.87	77.78	75.77	76.63	**81.14**	78.81	87.72	84.38	86.01
ARGCN+BERT	87.32	83.59	85.42	75.83	76.90	76.36	78.81	77.69	78.24	88.49	84.95	86.69
TABERT	87.35	87.18	87.27	81.77	**79.89**	80.82	83.76	80.53	82.11	**91.73**	**90.86**	**91.29**
RABERT	87.82	**88.93**	**88.37**	83.33	79.37	**81.30**	**85.68**	80.12	**82.81**	90.49	90.67	90.58

Table 3.4: Two examples of BIO-scheme sentences for TOWE task. The underlined terms indicate *opinion targets*, and the bold style terms represent corresponding *opinion words*.

Targets	**BIO-scheme sentence**
duck confit	”The/O duck/O confit/O is/O always/O **[amazing/B]** and/O the/O foie/O gras/O terrine/O with/O figs/O was/O out/O of/O this/O world/O ./O”
foie gras terrine	”The/O duck/O confit/O is/O always/O amazing/O and/O the/O foie/O gras/O terrine/O with/O figs/O was/O **[out/B of/I this/I world/I]** ./O”

3.3 Gated Relational Target-aware Encoder and Local Context-aware Decoder for Target-oriented Opinion Words Extraction

3.3.1 Model Description

Task Formalization

TOWE task can be formalized as a problem of sequence labeling for opinion target specified sentences. For a given a sentence $\mathbf{s} = \{w_1, w_2, ..., w_n\}$ consisting of n words, an opinion target $\mathbf{a} = \{w_i, w_{i+1}, ..., w_{i+j}\}$, and an opinion word $\mathbf{o} = \{w_k, w_{k+1}, ..., w_{k+m}\}$ (a and o are sub-sequences of s), the goal of TOWE task is to tag each words as $y_i \in \{B, I, O\}$ (B: Beginning, I: Inside, O: Others) based on the probabilities $p(\mathbf{o}|\mathbf{s}, \mathbf{a})$. The sequence of B and I indicates the corresponding opinion term o for the opinion target a. Table 3.4 illustrates the BIO-scheme sentence for given opinion targets.

Overall Framework

The structure of our **G**ated **R**elational target-aware **E**ncoder and local context-aware **D**ecoder (GRED) is illustrated in Fig. 3.9. GRED consists of a target-aware context encoder module and a local context-aware decoder module with a gate network. The overall framework of GRED is as follows. We first add a special token to indicate the opinion targets within the sentences. Next, these modified sentences are encoded into the target-aware encoder module and the local context-aware decoder module. These modules aim to capture opinion target information and local context information via the relation networks, respectively. The target-aware encoder module extracts the target-aware representation using multi-head self-attention layers and a target relation network. In the local context-aware decoder module, the local context relation network catches the local context representation from the surrounding words, and then the gate network outputs final representations by aggregating these two representations. Finally, Conditional Random Field (CRF) layer determines the tags of the sequence based on the final representation.

Target-aware Input Preprocessing

To fully utilize the pre-trained language model, we use the symbol ”*” as a target indicating token instead of the separator token ”[SEP]” used in the BERT-based models [60,61]. We insert the symbol ”*” before and after opinion target $a = \{w_i, \ldots, w_{i+j}\}$. Then, the sentence s is pre-processed as $s^* = \{w_1, ..., w_i, ..., w_{i+j}, ..., w_n\}$. GRED also uses a BART-style sequence classification scheme, which processes right-shifted input for the decoder. Thus, the input sentence of the decoder s_d^* is modified as $s_d^* = \{$¡s¿$, w_1, ..., *, w_i, ..., w_{i+j}, *, ..., w_n,$ ¡/s¿$\}$.

Target-aware Encoder Module

The target-aware encoder module consists of a text encoder and a target relation network. The text encoder receives the modified sentence s^* and produces the embedding of each word of those sentences. For each word in the sentence, the target relation network generates target-aware word representations based on the relationships between the opinion targets and each word in the sentence.

Text Encoder Sentence Embeddings. The text encoder adopts a structure of transformer encoder in Vaswani *et al.* [16], which have increased expressive power by using a multi-head self attention mechanism. To exploit the power of the pre-trained language model, we use the encoder part of BART for the text encoder. The text encoder takes the modified sentence s^* and generates context embedding of s^*:

$$H^{enc} = \text{BART}_{enc}(s^*), \tag{3.10}$$

where BART_{enc} is the encoder layers of BART and $H^{enc} = \{h^e_1, ..., h^e_i, ..., h^e_{i+j}, ..., h^e_n\}$ is the sentence embedding of s^*.

Opinion Target Embedding. To obtain the opinion target embedding, we first gather the parts that correspond to the opinion target from the sentence embedding H^{enc} and then apply a pooling layer to these parts. For the pooling layer, various pooling methods such as max-pooling, mean-pooling, and *LogSumExp* (**LSE**)-pooling are used to compute the opinion target embeddings:

$$\begin{aligned} h^e_{target} &= Max(\{h^e_i ..., h^e_{i+j}\}), \ \text{(Max-pool)}, \\ h^e_{target} &= \frac{1}{j+1}\sum_{k=i}^{i+j} h^e_i \qquad \text{(Mean-pool)}, \\ h^e_{target} &= \textbf{LSE}(\{h^e_i, ..., h^e_{i+j}\}) \ \text{(LSE-pool)}, \end{aligned} \tag{3.11}$$

where $\textbf{LSE}(\{h^e_i, ..., h^e_{i+j}\}) = \log \sum_{k=i}^{i+j} \exp(h_k)$ and $h^e{}_{target}$ is an opinion target embedding.

Target-aware Representation. Target-aware representations are derived from the relationships between each word in the sentence embedding and the opinion target embedding. Thus, we employ the relation network to compute these relationships. As shown in Fig. 3.6 (a), the target relation network takes the opinion target and each word in the sentence and then produces the target-aware representation. In this work, we use **MLP** as a structure of the target relation network. Given the opinion target embedding h^c_{target} and the sentence embedding H^{enc}, the target-aware representation is calculated as follows:

$$\begin{aligned} r^{Tar}_u &= \textbf{TRN}(\mathcal{C}^{Tar}(h^e_u, h^e_{target})), \quad u = 1, 2, ..., n \\ R^{Tar} &= \{r^{Tar}_1, r^{Tar}_2, ..., r^{Tar}_n\}, \end{aligned} \tag{3.12}$$

where **TRN** is the target relation network, $\mathcal{C}^{Tar}$ is combined function and R^{Tar} is target-aware representation. We use concatenation operator as $\mathcal{C}^{Tar}$, but other methods such as element-wise sum and multiplication are possible.

Local Context-aware Decoder

Local context-aware decoder is composed of a text decoder and a local context relation network. The text decoder takes the right-shifted sentence s^*_d and generates causal sentence embedding based on the encoder output and the previous words. The local context relation network produces the local context-aware representation by exploring the relationships among surrounding words in the given sentence. Finally, the gate network dynamically fuses the local context-aware representation into target-aware representation for predicting labels.

Text Decoder Sentence Embedding. The text decoder has the same structure as the BART decoder. Unlike the text encoder, the text decoder uses masked multi-head attention. Thus, the right-shifted sentence s^*_d is passed to the decoder. Given the encoder sentence embedding H^{enc} and the sentence s^*_d, then the text decoder computes the text decoder sentence embedding as follows:

$$H^{dec} = \text{BART}_{dec}(s^*_d, H^{enc}), \tag{3.13}$$

where BART_{dec} is the decoder layers of BART and $H^{dec} = \{h^d_1, ..., h^d_i, ..., h^d_{i+j}, ..., h^d_n\}$ is the sentence embedding of s^*_d.

Local Context-aware Representation. The local context-aware representations are obtained from contextualizing each word and its surrounding words in the sentence.

We use the local context relation network to explore the relationship among surrounding words in the given sentence. As illustrated in Fig. 3.6 (b), the local context relation network calculates these relationships based on the word, its left-side word, and its right-side word. In this work, we consider tri-grams as the local context (e.g. $\{w_{u-1}, w_u, w_{u+1}\}$). Thus, the local context-aware representation is computed as follows:

$$\begin{gathered} r_u^{Loc} = \textbf{LRN}(\mathcal{C}^{Loc}(h_{u-1}^d, h_u^d, h_{u+1}^d)), \qquad u = 1, 2, ..., n \\ R^{Loc} = \{r_1^{Loc}, r_2^{Loc}, ..., r_n^{Loc}\}, \end{gathered} \tag{3.14}$$

where **LRN** is the local context relation network, $\mathcal{C}^{Loc}$ is combined function and R_u^{Loc} is local context-aware representation. $\mathcal{C}^{Loc}$ is composed of two different **MLP** and elementwise-sum (i.e. $\mathcal{C}^{Loc}(x, y, z) = A(x, y) + B(y, z)$; A and B are **MLP**).

Gate Network. The final representations aggregate the target-aware representation from the encoder and the local context-aware representation from the decoder. Instead of simply combining the two representations, GRED leverages the gate network to decide how both the representations play a part in sequence labeling. Given the target-aware representation and the local context-aware representation, the gate network computes the aggregated representation as follows:

$$\begin{aligned} \alpha_u &= \sigma(W_u^{Tar} r_u^{Tar} + W_u^{Loc} r_u^{Loc} + b_u), \\ r_u &= \alpha_u r_u^{Tar} + (1 - \alpha_u) r_u^{Loc}, \\ R &= \{r_1, r_2, ..., r_n\}, \end{aligned} \tag{3.15}$$

where α_u is the gate weight of each word, σ is sigmoid function, W_u^{Tar} and W_u^{Loc} are weight matrices, b_u is bias vector and R is final representation.

Decoding strategy and loss function

Given the final representation R, we decode the sequence label $Y = \{y_1, ..., y_n\}$ based on the probability $p(Y|R)$. In this work, we adopt Conditional Random Field (**CRF**) as our decoding strategy because it can capture the word structural dependency of the sentence and correlations between labels. Specifically, the score function of **CRF** can be defined as:

$$S(R,Y)=\sum_{u=0}^{n}(\mathbf{A}_{y_{u-1},y_u}+\mathbf{Q}_{u,y_u}),$$
$$Q=RW_q+b_q \tag{3.16}$$

where the A measures transition score between two adjacent labels and the matrix Q is emission score.

Then, we can compute the probability using the score function:

$$p(\mathbf{Y}|\mathbf{R})=\frac{\exp(S(\mathbf{R},\mathbf{Y}))}{\sum_{\hat{y}\in\bar{Y}}\exp(S(\mathbf{R},\hat{y}))}, \tag{3.17}$$

where $\bar{Y}$ is the set of all possible sequential labels. **CRF** uses the negative log likelihood as the loss function. Thus, the loss of a given sentence can be calculated by the negative log-likelihood of the probability:

$$Loss(\mathbf{s})=-\log(p(\mathbf{Y}|\mathbf{R})) \tag{3.18}$$

We minimize this loss function $Loss(\mathbf{s})$ for training. For a decoding process, the model generates the label sequence, which maximize $p(\mathbf{Y}|\mathbf{R})$ via Viterbi algorithm.

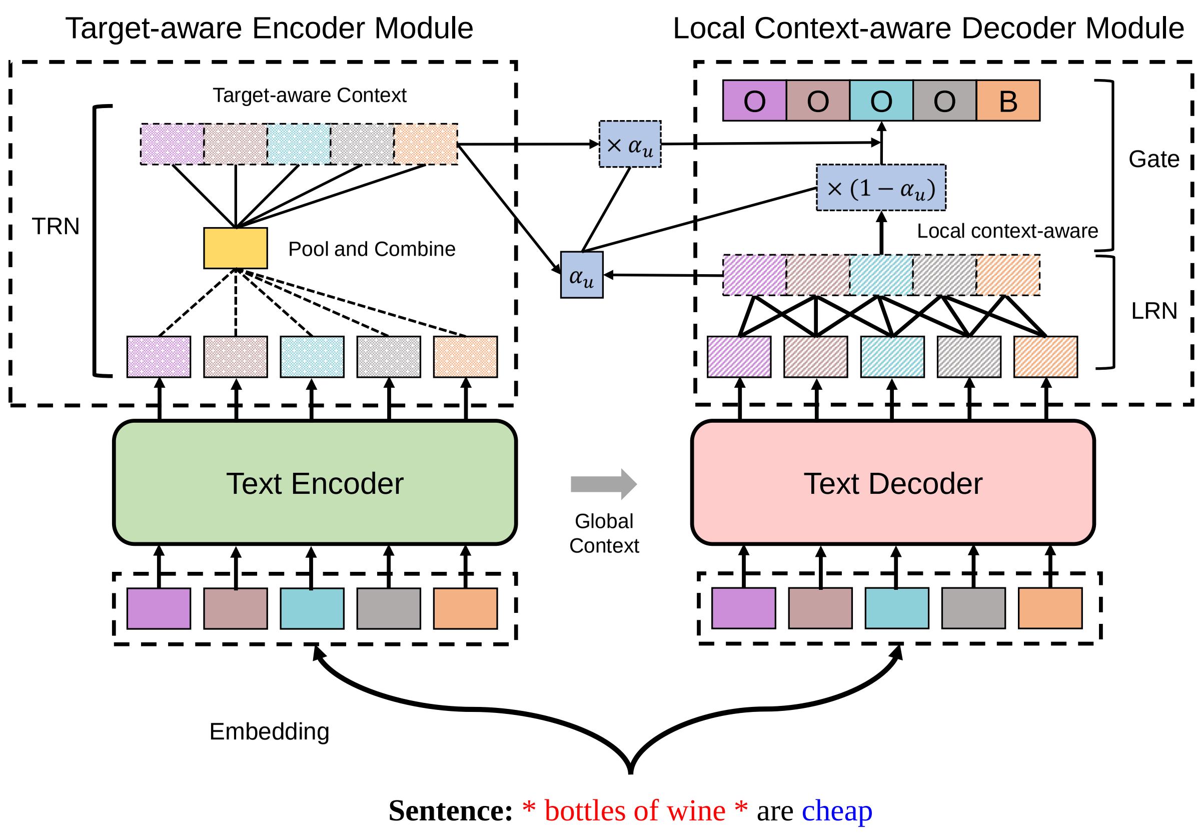
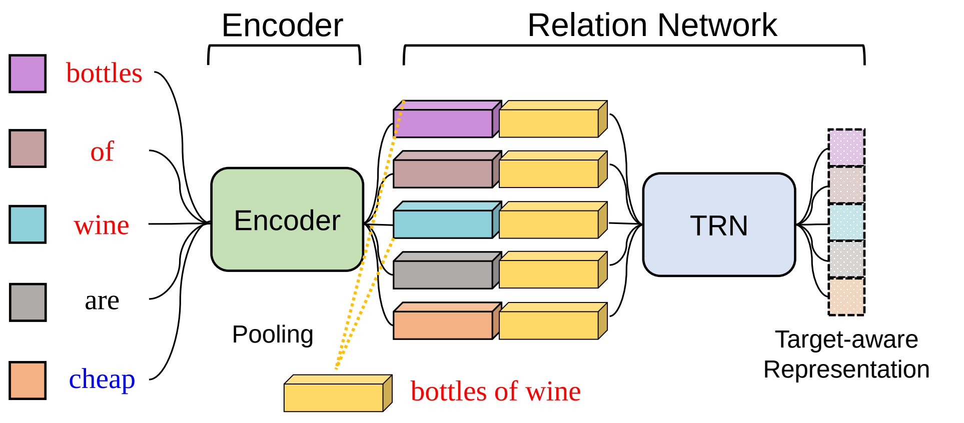
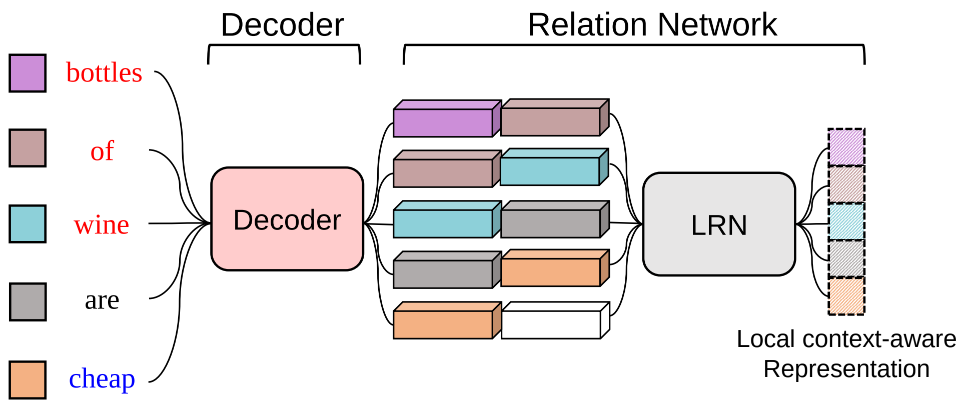


Figure 3.5: The overview of GRED. GRED comprises a target-aware context encoder module, a local context-aware decoder module, and a gate network. The target-aware context encoder module and the local context-aware decoder module contain the target relation module and the local context relation module, respectively. The gate network computes final representations based on outputs from both modules.



(a)



(b)

Figure 3.6: Structures of the two relation networks: (a) Target relation network and (b) Local context relation network. The target relation network compares the opinion target (*"bottles of wine"*) and each word (*"bottles"*, *"of"*, *"wine"*,*"are"*, and *"cheap"*) in the sentence (*"bottles of wine are cheap"*). The local context relation network captures relationship among the words (*"bottles"*, *"of"*, *"wine"*,*"are"*, and *"cheap"*) in the sentence (*"bottles of wine are cheap"*).

3.3.2 Experimental Results

Experimental Setup

Datasets To verify the effectiveness of GRED, we conduct extensive experiments on four benchmark TOWE datasets: ***14res***, ***15res***, ***16res***, and ***14lap***. These datasets were bulit by [46] for TOWE task based on the SemEval Challenge 2014 Task 4 [1], SemEval Challenge 2015 task 12 [2] and SemEval Challenge 2016 task 5 [3] respectively. ***14res, 15res*** and ***16res*** are collected from review sentencess in restaurant domain. ***14 lap*** contains the review sentence in laptop domain. The statistics of these datasets are summarized in Table 3.2.

Baselines For the comprehensive and comparative analysis of GRED, we compare it with the following methods:

1. Rule-based methods. We adopt Distance-rule and Dependency-rule as our baselines. Distance-rule method utilizes the distance and Part-Of-Speech (POS) tags to extract the opinion words. Dependency-rule method uses the dependency tree of the sentence to determine the opinion words.

2. Deep Neural Network (DNNs)-based methods. We choose BiLSTM, TC-BiLSTM, IOG [46], PE-BiLSTM and LOTN [47] for our DNN-based baseline methods. These methods employ the Recurrent Neural Networks (RNNs) to capture the dependency between the opinion target and its corresponding opinion words. IOG uses an Inward-Outward BILSTM to learn opinion target-aware representation and global context representation and then fuses these representations to predict labels. LOTN utilizes the additional position embeddings for indicating opinion targets and transfers the latent opinion knowledge from resource-rich datasets to TOWE task model.

3. Pre-trained Language Model (PLM) based methods. We also adopt SDRN [76], ONG [48], ARGCN [49], TSMSA [60], RABERT [61] and UNI-GEN [73] for baselines. SDRN employs a BERT-based encoder with an opinion entity extraction unit, a relation detection unit, and a synchronization unit for the Aspect Opinion Pair Extraction (AOPE) task. ONG incorporates the syntactic structures of the sentence into deep learning models using Graph Convolution Networks (GCNs). ARGCN consists of BiLSTM-based sequential layers and attention-based relational GCN layers to consider semantic and syntactic relevance between words simultaneously. TSMSA uses a multi-head self attention mechanism to specify opinion target in the sentence. RABERT integrates the relation network into BERT layer to capture the relationship between words. UNI-GEN converts all ABSA subtasks into a unified generative formulation and exploits BART to solve all these tasks.

Hyper parameter setting. We implement our proposed GRED with pytorch library [77] and Hugging Face Transformers[1]. In our experiments, the batch size is set to 8, and the maximum sequence length is set to 128. We train the model using *Adam* optimizer and learning rate decay strategy with $\beta_1 - 0.9$ and $\beta_2 = 0.999$. We also set the warmup steps to 100 steps. The dropout rate is selected from $\{0.3, 0.4, 0.5\}$ based on the performances on validation sets. The learning rate of the encoder layer and decoder layer is set to 5×10^{-5}. We set the learning rate of **TRN** and **LRN** to 1×10^{-4}. We adopt pre-trained BART from [19], which consists of 12 layers for the encoder and the decoder, respectively. **TRN** and the gate network consist of 1-layer Feed-Forward Networks (FFNs). And **LRN** is composed of two parallel 1-layer FFNs and one additional FFN. GRED could fit in a single NVIDIA GTX 1080ti GPU. For a fair comparison with the baselines, we randomly select 20% of the training set as the dev set using the same random seeds as [46].

Evaluation metrics. To be consistent with the previous works [46,47,60,61], we adopt precision, recall, and F1 score as the evaluation metrics to compare the performance of models. We consider the predicted opinion words span to be a correct prediction when the starting and ending positions of them are both the same as those of the golden span.

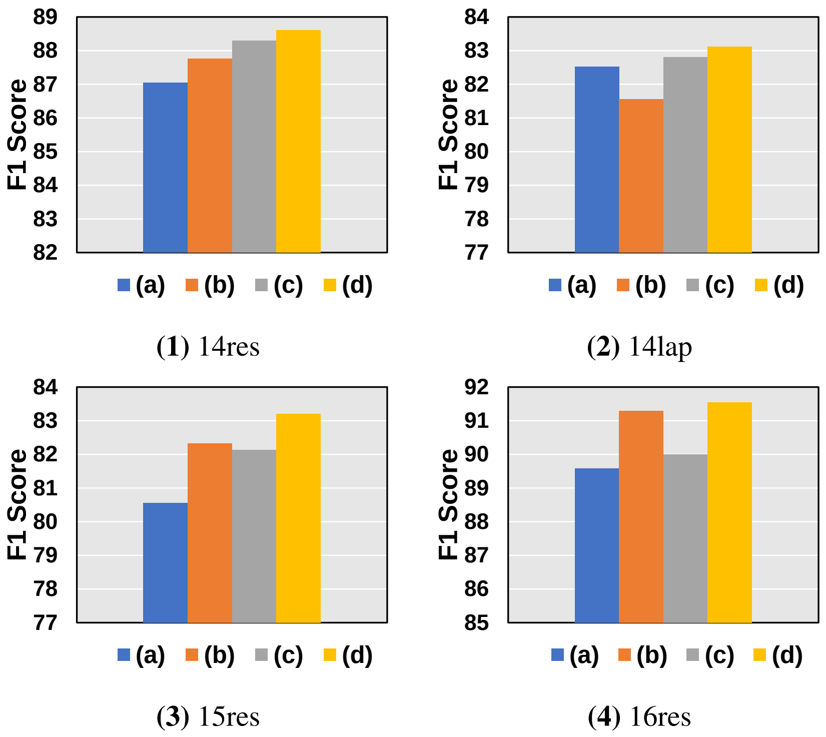


Figure 3.7: Ablation study of GRED. We report the F1 score of the variants of GRED on the four benchmark datasets.

Experimental Results

Performance comparison on the benchmark datasets. Here, we focus on the TOWE task performance comparison between the proposed GRED and the existing models on the four benchmark datasets. All the experimental results are reported in Table 3.5. The best scores are highlighted in bold style. First of all, compared with the other baselines, the proposed GRED obtains superior performance and achieves the new state-of-the-art performance on all of the datasets. In detail, we find that GRED outperforms the state-of-the-art scores by 0.24%, 1.71%, 0.39%, and 0.97% for F1 scores on the four datasets, respectively. These results validate the effectiveness of the proposed GRED for TOWE task.

[1]https://huggingface.co/transformers

For the results of rule-based methods, the dependency-rule performs better than the distance rule, which indicates that word dependency is critical to solving TOWE task. And we note that both the rule-based methods (Distance-rule and Dependency-rule) show poor performance across all the scores. This reveals that the rules cannot handle the diverse patterns of TOWE task. On the other hand, DNN-based methods achieve relatively better performance than the rule-based methods because of their model expressive power. Next, we observe that IOG, PE-BiLSTM, and LOTN obtain an F1-score with about 25% improvement over LSTM and BiLSTM methods, which demonstrates the effectiveness of using opinion target information. Thus, all of these results reveal that capturing both word dependency and opinion target is important for TOWE.

Finally, pre-trained language model-based methods achieve great performance on all the scores. In particular, RABERT and TSMSA show relatively better performance than other pre-trained language model-based methods such as SDRN, ONG, ARGCN, and UNI-GEN. These experimental results demonstrate that the target indicating token can be effective in fully utilizing the pre-trained language model for TOWE task. Also, RABERT achieves the previous state-of-the-art performance. This validates the effectiveness of capturing target information and local context information together.

Ablation study. To investigate the effectiveness of each part of GRED, we evaluate the variants of GRED: **(a) Encoder only**: only using the target-aware encoder to predict labels, **(b) Decoder only**: only feeding the output of the decoder to compute final representation, **(c) Encoder+Decoder**: uniformly combining the outputs of the encoder and the decoder without the gate network, and **(d) GRED**.

The results of the ablation study are depicted in Fig. 3.7. First, Encoder only model is inferior to the other models on most of the datasets. This demonstrates that using only opinion target information cannot extract the diverse patterns of opinion words in TOWE task. Second, naive aggregation of the encoder and the decoder does not ensure performance improvement. The performance drops of Encoder+Decoder on ***15res*** and

16res confirm the suitability of our proposed gate mechanism. Overall, GRED shows the best performance than all the other models on all the datasets. This reveals that all three components of GRED are critical to solve TOWE: 1) capturing the opinion words, 2) utilizing the local context, and 3) dynamically leveraging by our gate mechanism.

Next, we also report the performance of various opinion target pooling methods in Table 3.6: Max-pool, Mean-pool, and LSE-pool from Equation 2. We observe that the Max-pooling method shows relatively poor performance, but the Mean-pool and LSE-pool perform almost identical performance, which indicates that the Max-pool can lose more target information than the other methods. Comparing the Mean-pool and LSE-pool, the performance of LSE-pool is more robust in the four benchmark datasets. Thus, we adopt LSE-pool for the target pooling method of GRED in this work.

Case study. In order to validate the effectiveness of our proposed GRED, we extract some TOWE examples of GRED and Encoder+Decoder model from the two different domains, restaurant and laptop (Table 3.8). In a simple case such as Sentence 1, we can observe that both GRED and Encoder+Decoder model give the correct prediction for the given sentence and opinion targets. However, in Sentence 2, Sentence 3, and Sentence 4 which are more complicated than Sentence 1, only GRED can extract the correct opinion terms successfully. We also note that, even in the wrong prediction examples, Sentence 5 and Sentence 6, GRED gives the prediction closer to the gold label answers than Encoder+Decoder model. These results demonstrate that the proposed GRED can leverage the gate network to improve the effectiveness for complicated patterns of TOWE task.

We also visualize the weight of the gate network α_u to investigate its role for predicting labels in Fig. 3.8. In the first example, the value of α_u increases at the word *"top notch"*, which is corresponding opinion words for *"hot dogs"*. However, in the second example, the highest value of α_u is at the word *"amazing"*. These results indicate that the target-aware encoder mainly focuses on the opinion target-dependent words. In the third example, we can observe that the value of α_u goes up at the words, *"great"*, *"die"*, and *"for"*, which indicates the gate network can capture the multiple opinion words simultaneously. In particular, comparing the words *"die for"* and *"Not much"* in the third and fourth example, and the words *"top notch"* in the first example, the value of α_u at *"much"* and *"for"* are lower than others *"die"*, *"Not"* and *"top notch"*. Since the opinion words *"for"* and *"much"* are dependent on the surrounding words *"die"* and *"Not"* respectively as well as their corresponding opinion targets, the gate network reduces the influence of the target-aware encoder. Therefore, these results demonstrate that the gate network can effectively regulate the target-aware encoder and the local context-aware decoder.

Table 3.5: Main experimental results on the four benchmark datasets. Performance comparison with the baselines.

Models	14res			14lap			15res			16res		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Distance-rule	58.39	43.59	49.92	50.13	33.86	40.42	54.12	39.96	45.97	61.90	44.57	51.83
Dependency-rule	64.57	52.72	58.04	45.09	31.57	37.14	65.49	48.88	55.98	76.03	56.19	64.62
LSTM	52.64	65.47	58.34	55.71	57.53	56.52	57.27	60.69	58.93	62.46	68.72	65.33
BiLSTM	58.34	61.73	59.95	64.52	61.45	62.71	60.46	63.65	62.00	68.68	70.51	69.57
Pipeline	77.72	62.33	69.18	72.58	56.97	63.83	74.75	60.65	66.97	81.46	67.81	74.01
TC-BiLSTM	67.65	67.67	67.61	62.45	60.14	61.21	66.06	60.16	62.94	73.46	72.88	73.10
IOG	82.38	78.25	80.23	73.43	68.74	70.99	72.19	71.76	71.91	84.36	79.08	81.60
PE-BiLSTM	80.10	76.51	78.26	72.01	64.20	67.83	70.36	65.73	67.96	82.27	74.95	78.43
LOTN	84.00	80.52	82.12	77.08	67.62	72.02	76.61	70.29	73.29	86.57	80.89	83.62
SDRN	**91.14**	76.37	83.10	84.37	65.42	73.69	83.57	70.33	76.38	91.13	80.34	85.40
ONG	83.23	81.46	82.33	73.87	77.78	75.77	76.63	81.14	78.81	87.72	84.38	86.01
ARGCN	87.32	83.59	85.42	75.83	76.90	76.36	78.81	77.69	78.24	88.49	84.95	86.69
TSMSA	87.86	85.05	86.43	83.63	80.46	82.01	80.91	83.80	82.33	89.31	89.14	89.23
RABERT	87.82	88.93	88.37	83.33	79.37	81.30	**85.68**	80.12	82.81	90.49	90.67	90.58
UNI-GEN	86.01	84.76	85.38	83.11	78.13	80.55	80.12	80.93	80.52	89.22	86.67	87.92
GRED	88.10	**89.13**	**88.61**	**84.24**	**82.01**	**83.11**	82.05	**84.38**	**83.20**	**91.29**	**91.81**	**91.55**

Table 3.6: The F1 scores of various pooling methods of *opinion targets* on the four benchmark datasets.

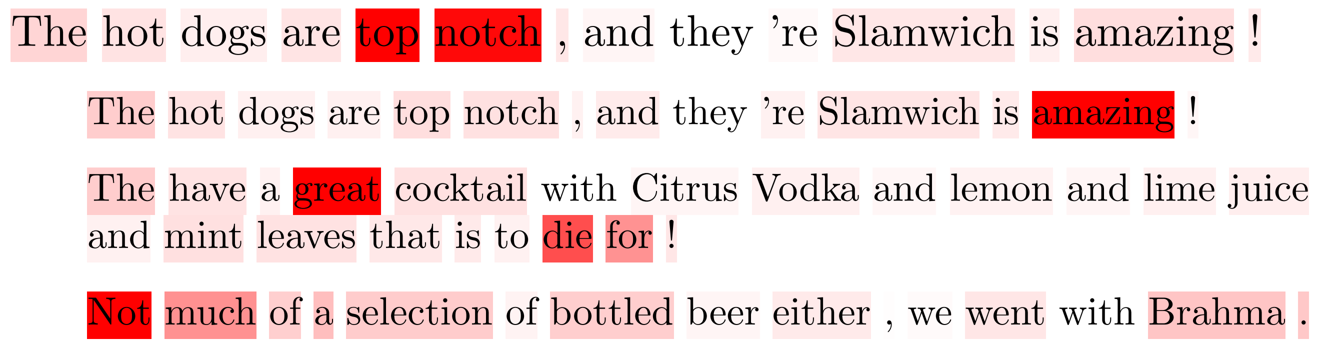
Methods	Datasets				
	14res	14lap	15res	16res	Avg
Max-pool	87.83	81.25	81.50	91.27	85.46
Mean-pool	**88.97**	81.94	**83.33**	90.65	86.22
LSE-pool	88.61	**83.11**	83.20	**91.55**	**86.62**

Table 3.7: Case study: the prediction of Encoder+Decoder model and GRED. The red-colored words and blue-colored words represent *opinion targets* and *opinion words*, respectively.

Sentence 1: They have great rolls, the triple color and norwegetan rolls , are awesome and filling .
Encoder+Decoder: They have great rolls, the triple color and norwegetan rolls , are awesome and filling . ✓
GRED: They have great rolls, the triple color and norwegetan rolls , are awesome and filling . ✓
Sentence 2: I did swap out the hard drive for a Samsung 830 SSD which I highly recommend .
Encoder+Decoder:I did swap out the hard drive for a Samsung 830 SSD which I highly recommend . ✗
GRED:I did swap out the hard drive for a Samsung 830 SSD which I highly recommend . ✓
Sentence 3: My daughter 's wedding reception at Water 's Edge received the highest compliments from our guests .
Encoder+Decoder: My daughter 's wedding reception at Water 's Edge received the highest compliments from our guests .✗
GRED: TMy daughter 's wedding reception at Water 's Edge received the highest compliments from our guests .✓

Table 3.8: Case study: the prediction of Encoder+Decoder model and GRED. The red-colored words and blue-colored words represent *opinion targets* and *opinion words*, respectively.

Sentence 4: Then the system would many times not power down without a forced power-off .
Encoder+Decoder: Then the system would many times not power down without a forced power-off . ✗
GRED: Then the system would many times not power down without a forced power-off . ✓
Sentence 5: The veal and the mushrooms were cooked perfectly .
Encoder+Decoder: The veal and the mushrooms were cooked perfectly . ✗
GRED: The veal and the mushrooms were cooked perfectly . ✗
Sentence 6: this Mac Mini does not have a built-in mic , and it would seem that its Mac OS 10.9 does not handle external microphones properly .
Encoder+Decoder: this Mac Mini does not have a built-in mic , and it would seem that its Mac OS 10.9 does not handle external microphones properly . ✗
GRED: this Mac Mini does not have a built-in mic , and it would seem that its Mac OS 10.9 does not handle external microphones properly . ✗



The hot dogs are top notch , and they 're Slamwich is amazing !

The hot dogs are top notch , and they 're Slamwich is amazing !

The have a great cocktail with Citrus Vodka and lemon and lime juice and mint leaves that is to die for !

Not much of a selection of bottled beer either , we went with Brahma .

Figure 3.8: Visualization of the gate weight α_u of each word. The opinion targets of each sentence are *"hot dogs"*, *"Slamwich"*, *"cocktail"* and *"selection of bottled beer"*, respectively. The extracted opinion words of each sentence are *"top notch"*, *"amazing"*, {*"great", "die", "for"*}, and {*"Not", "much"*}.The stronger the red words, the greater the value of α_u.

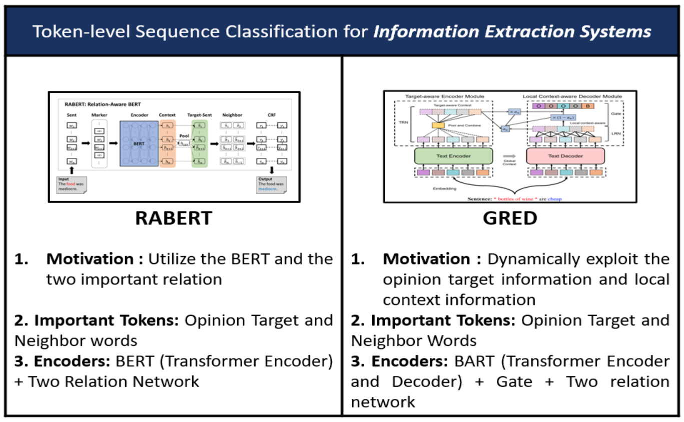
3.4 Conclusion

In this dissertation, we have found the answer of the academic questions of TLC in information extraction systems: 1) How to capture important tokens from sequential information and 2) How to encode a token sequence.

First, we propose a novel target-oriented opinion words extraction model, RABERT that can consider both the relations between opinion targets and a sentence and the relations among neighbor words. RABERT can fully leverage BERT to obtain target-aware context representation, and extract the two important relations using the target-sentence relation network and the neighbor-aware relation network. Our experimental results on widely-used datasets show that RABERT achieves new state-of-the-art performance. Furthermore, we demonstrate the effectiveness of each component of RABERT in our extensive analysis.

In the next, we also propose a novel encoder-decoder model named GRED for target-oriented opinion words extraction. Our proposed GRED is able to fully utilize the information of the opinion target and the local context. The target-aware encoder leverages the information of opinion target and the local context-aware decoder captures the local context using the relation networks. Then, the gate network dynamically combines the outputs of the encoder and the decoder to predict the label sequences. To validate our proposed GRED, we conduct extensive experiments on the four widely used benchmark datasets. Our GRED outperforms all of the baseline methods and achieves the new state-of-the-art performance. Also, in-depth analyses in multiple perspectives and qualitative studies demonstrate that GRED properly leverages the information of the opinion target and the local context. As a future work, we plan to design more efficient relation modules and investigate more informative relationships for TOWE task.

Finally, we summarized the academic questions and their corresponding answers of our proposed models in Figure 3.9. RABERT can utilize the BERT and just put the two important relations together. However, GRED uses the BART and dynami-



Token-level Sequence Classification for ***Information Extraction Systems***

RABERT	GRED
1. **Motivation :** Utilize the BERT and the two important relation	1. **Motivation :** Dynamically exploit the opinion target information and local context information
2. Important Tokens: Opinion Target and Neighbor words	**2. Important Tokens:** Opinion Target and Neighbor Words
3. Encoders: BERT (Transformer Encoder) + Two Relation Network	**3. Encoders:** BART (Transformer Encoder and Decoder) + Gate + Two relation network

Figure 3.9: Summary of Chapter 3

cally exploit the opinion target information and local context information. The both RABERT and GRED consider opinion target and neighbor words as important tokens. For designing encoder, RABERT employs BERT and two relation networks. GRED uses BART and the two gated relation neworks.

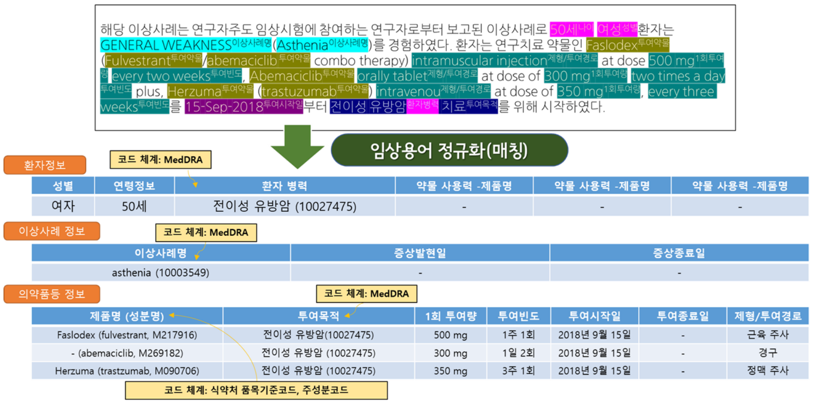
Chapter 4

Conclusion

4.1 On-going Researches

We are conducting a task to extract more detailed information by further expanding the Information Extraction Task. In particular, we perform the task of extracting the relationship between drugs and adverse effects and important arguments in the medical domain. For this task, we divide it into the five token-level classification tasks: argument extraction, concerned label classification, occurred label classification, key sentence extraction tasks, and relation extraction tasks. In argument extraction tasks, each word in the document is classified as various types of arguments such as named entity recognition tasks. In occurred label classification and concerned label classification, each word in the document is classified as a binary label. Finally, relation extraction takes pair of argument tokens, and classifies the relation types for the given pair. We also attempt to find the answers to two questions in this task. To find the important tokens, we utilize self-attention mechanisms to capture the semantic dependency in the sentence. We also use a domain-specific BERT model to encode the medical domain documents.

In the future, we will focus on few-shot learning-based TLC tasks because the few-shot learning matches well with the real world. For NLP fields, large-scaled language



해당 이상사례는 연구자주도 임상시험에 참여하는 연구자로부터 보고된 이상사례로 50세$^{\text{나이}}$ 여성$^{\text{성별}}$환자는 GENERAL WEAKNESS$^{\text{이상사례명}}$(Asthenia$^{\text{이상사례명}}$)를 경험하였다. 환자는 연구치료 약물인 Faslodex$^{\text{투여약물}}$ (Fulvestrant$^{\text{투여약물}}$/abemaciclib$^{\text{투여약물}}$ combo therapy) intramuscular injection$^{\text{제형/투여경로}}$ at dose 500 mg$^{\text{1회투여량}}$ every two weeks$^{\text{투여빈도}}$, Abemaciclib$^{\text{투여약물}}$ orally tablet$^{\text{제형/투여경로}}$ at dose of 300 mg$^{\text{1회투여량}}$ two times a day$^{\text{투여빈도}}$ plus, Herzuma$^{\text{투여약물}}$ (trastuzumab$^{\text{투여약물}}$) intravenou$^{\text{제형/투여경로}}$ at dose of 350 mg$^{\text{1회투여량}}$, every three weeks$^{\text{투여빈도}}$를 15-Sep-2018$^{\text{투여시작일}}$부터 전이성 유방암$^{\text{환자병력}}$ 치료$^{\text{투여목적}}$를 위해 시작하였다.

환자정보

코드 체계: MedDRA

성별	연령정보	환자 병력	약물 사용력 -제품명	약물 사용력 -제품명	약물 사용력 -제품명
여자	50세	전이성 유방암 (10027475)	-	-	-

이상사례 정보

코드 체계: MedDRA

이상사례명	증상발현일	증상종료일
asthenia (10003549)	-	-

의약품등 정보

코드 체계: MedDRA

제품명 (성분명)	투여목적	1회 투여량	투여빈도	투여시작일	투여종료일	제형/투여경로
Faslodex (fulvestrant, M217916)	전이성 유방암(10027475)	500 mg	1주 1회	2018년 9월 15일	-	근육 주사
- (abemaciclib, M269182)	전이성 유방암 (10027475)	300 mg	1일 2회	2018년 9월 15일	-	경구
Herzuma (trastzumab, M090706)	전이성 유방암 (10027475)	350 mg	3주 1회	2018년 9월 15일	-	정맥 주사

코드 체계: 식약처 품목기준코드, 주성분코드

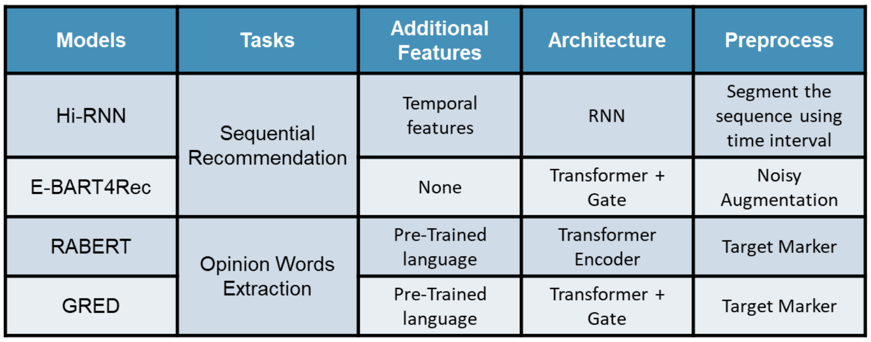
Figure 4.1: The overview of information extracion in medical domain

models can solve the few-shot learning settings, so we attempt to apply large-scaled language models to TLC tasks.

4.2 Conclusion

In this dissertation, we have aimed to solve the two academic questions of TLC tasks in the sequential recommendation and information extraction: 1) How to find the important tokens from token level sequence and, 2) How to design a good encoder. We compared and summarized the models proposed in our thesis in Figure 4.2 and Figure 4.3.

In recommendation systems, we first present the new recommendation systems that incorporate the temporal properties of the user history using hierarchical RNNs (Hi-RNN). Hi-RNN can consider the recent items as important tokens. Also, we utilize Hi-RNN to encode the temporal features of the items. Next, we propose the BART-



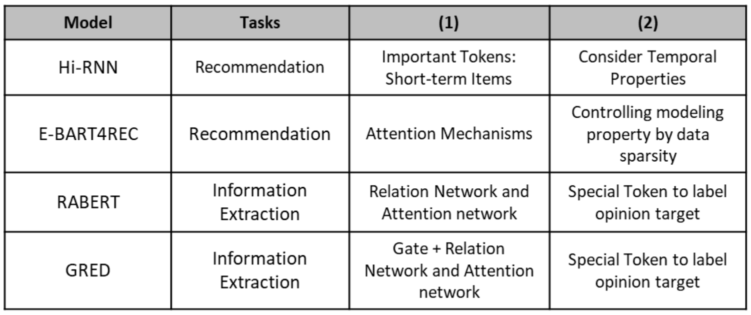
Models	Tasks	Additional Features	Architecture	Preprocess
Hi-RNN	Sequential Recommendation	Temporal features	RNN	Segment the sequence using time interval
E-BART4Rec		None	Transformer + Gate	Noisy Augmentation
RABERT	Opinion Words Extraction	Pre-Trained language	Transformer Encoder	Target Marker
GRED		Pre-Trained language	Transformer + Gate	Target Marker

Figure 4.2: The comparison of the proposed models

based recommendation system, E-BART4Rec that entangles bidirectional models and auto-regressive models. E-BART4Rec can capture the important items using a self-attention mechanism and effectively exploit the datasets regardless of data sparsity. We conduct experiments on real-world datasets such as movie recommendations and music recommendations, which verify the effectiveness of our recommendation models.

We also aim to solve the two academic questions of TLC tasks for the information extraction tasks. First, we design the RABERT that can fully utilize the power of BERT and consider the two important relations. Finally, we propose the new information extraction models that can dynamically the two important information - target information and local context information using a gate network. In the experimental results, we discover that not only opinion target tokens but also local-context tokens are important to solve information extraction tasks.

In conclusion, we find that the two academic questions are closely related to the



Model	Tasks	(1)	(2)
Hi-RNN	Recommendation	Important Tokens: Short-term Items	Consider Temporal Properties
E-BART4REC	Recommendation	Attention Mechanisms	Controlling modeling property by data sparsity
RABERT	Information Extraction	Relation Network and Attention network	Special Token to label opinion target
GRED	Information Extraction	Gate + Relation Network and Attention network	Special Token to label opinion target

Figure 4.3: The summary of this dissertation

performance of TLC tasks in the real world datasets. Our extensive experimental results on various datasets of the two tasks demonstrate that our proposed methods can be effective to solve the two questions. We also suggest qualitative analysis to verify the effectiveness of our models.

Bibliography

Bibliography

[1] M. Pontiki, D. Galanis, J. Pavlopoulos, H. Papageorgiou, I. Androutsopoulos, and S. Manandhar, "SemEval-2014 task 4: Aspect based sentiment analysis," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, (Dublin, Ireland), pp. 27–35, Association for Computational Linguistics, Aug. 2014.

[2] M. Pontiki, D. Galanis, H. Papageorgiou, S. Manandhar, and I. Androutsopoulos, "SemEval-2015 task 12: Aspect based sentiment analysis," in *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, (Denver, Colorado), pp. 486–495, Association for Computational Linguistics, June 2015.

[3] M. Pontiki, D. Galanis, H. Papageorgiou, I. Androutsopoulos, S. Manandhar, M. AL-Smadi, M. Al-Ayyoub, Y. Zhao, B. Qin, O. De Clercq, V. Hoste, M. Apidianaki, X. Tannier, N. Loukachevitch, E. Kotelnikov, N. Bel, S. M. Jiménez-Zafra, and G. Eryiğit, "SemEval-2016 task 5: Aspect based sentiment analysis," in *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, (San Diego, California), pp. 19–30, Association for Computational Linguistics, June 2016.

[4] B. Choe, T. Kang, and K. Jung, "Recommendation system with hierarchical recurrent neural network for long-term time series," *IEEE Access*, vol. 9, pp. 72033–72039, 2021.

[5] T. Kang, H. Lee, B. Choe, and K. Jung, "Entangled bidirectional encoder to autoregressive decoder for sequential recommendation," in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1657–1661, 2021.

[6] T. Kang, M. Lee, N. Yang, and K. Jung, "Rabert: Relation-aware bert for target-oriented opinion words extraction," in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pp. 3127–3131, 2021.

[7] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," *arXiv preprint arXiv:1511.06939*, 2015.

[8] B. Hidasi and A. Karatzoglou, "Recurrent neural networks with top-k gains for session-based recommendations," in *Proceedings of the 27th ACM international conference on information and knowledge management*, pp. 843–852, 2018.

[9] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang, "Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer," in *Proceedings of the 28th ACM international conference on information and knowledge management*, pp. 1441–1450, 2019.

[10] W.-C. Kang and J. McAuley, "Self-attentive sequential recommendation," in *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 197–206, IEEE, 2018.

[11] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th international conference on World Wide Web*, pp. 285–295, 2001.

[12] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *2008 Eighth IEEE international conference on data min-*

ing, pp. 263–272, Ieee, 2008.

[13] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme, "Factorizing personalized markov chains for next-basket recommendation," in *Proceedings of the 19th international conference on World wide web*, pp. 811–820, 2010.

[14] C. Zhou, C. Sun, Z. Liu, and F. Lau, "A c-lstm neural network for text classification," *arXiv preprint arXiv:1511.08630*, 2015.

[15] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[17] Q. Liu, F. Yu, S. Wu, and L. Wang, "A convolutional click prediction model," in *Proceedings of the 24th ACM international on conference on information and knowledge management*, pp. 1743–1746, 2015.

[18] J. Tang and K. Wang, "Personalized top-n sequential recommendation via convolutional sequence embedding," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pp. 565–573, 2018.

[19] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," *arXiv preprint arXiv:1910.13461*, 2019.

[20] P. Melville, R. J. Mooney, R. Nagarajan, *et al.*, "Content-boosted collaborative filtering for improved recommendations," *Aaai/iaai*, vol. 23, pp. 187–192, 2002.

[21] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, “Bpr: Bayesian personalized ranking from implicit feedback,” *arXiv preprint arXiv:1205.2618*, 2012.

[22] T. Zhao, J. McAuley, and I. King, “Leveraging social connections to improve personalized ranking for collaborative filtering,” in *Proceedings of the 23rd ACM international conference on conference on information and knowledge management*, pp. 261–270, 2014.

[23] L. Lerche and D. Jannach, “Using graded implicit feedback for bayesian personalized ranking,” in *Proceedings of the 8th ACM Conference on Recommender systems*, pp. 353–356, 2014.

[24] G. Dahl, M. Ranzato, A.-r. Mohamed, and G. E. Hinton, “Phone recognition with the mean-covariance restricted boltzmann machine,” *Advances in neural information processing systems*, vol. 23, 2010.

[25] H. Larochelle and Y. Bengio, “Classification using discriminative restricted boltzmann machines,” in *Proceedings of the 25th international conference on Machine learning*, pp. 536–543, 2008.

[26] F. Strub and J. Mary, “Collaborative filtering with stacked denoising autoencoders and sparse inputs,” in *NIPS workshop on machine learning for eCommerce*, 2015.

[27] H. Wang, N. Wang, and D.-Y. Yeung, “Collaborative deep learning for recommender systems,” in *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1235–1244, 2015.

[28] Y. Wu, C. DuBois, A. X. Zheng, and M. Ester, “Collaborative denoising auto-encoders for top-n recommender systems,” in *Proceedings of the ninth ACM international conference on web search and data mining*, pp. 153–162, 2016.

[29] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th international conference on world wide web*, pp. 173–182, 2017.

[30] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur, "Recurrent neural network based language model.," in *Interspeech*, vol. 2, pp. 1045–1048, Makuhari, 2010.

[31] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*, pp. 6645–6649, Ieee, 2013.

[32] S. Wu, W. Ren, C. Yu, G. Chen, D. Zhang, and J. Zhu, "Personal recommendation using deep recurrent neural networks in netease," in *2016 IEEE 32nd international conference on data engineering (ICDE)*, pp. 1218–1229, IEEE, 2016.

[33] S. Hochreiter, Y. Bengio, P. Frasconi, J. Schmidhuber, *et al.*, "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies," 2001.

[34] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[35] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.

[36] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *Acm transactions on interactive intelligent systems (tiis)*, vol. 5, no. 4, pp. 1–19, 2015.

[37] S. Rendle, "Factorization machines with libfm," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 3, no. 3, pp. 1–22, 2012.

[38] Z. Tu, Y. Liu, Z. Lu, X. Liu, and H. Li, “Context gates for neural machine translation,” *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 87–99, 2017.

[39] H. Chen, Z. Lin, G. Ding, J. Lou, Y. Zhang, and B. Karlsson, “Grn: Gated relation network to enhance convolutional neural network for named entity recognition,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 6236–6243, 2019.

[40] F. Yuan, X. He, H. Jiang, G. Guo, J. Xiong, Z. Xu, and Y. Xiong, “Future data helps training: Modeling future contexts for session-based recommendation,” in *Proceedings of The Web Conference 2020*, pp. 303–313, 2020.

[41] D. Rafailidis, “Modeling trust and distrust information in recommender systems via joint matrix factorization with signed graphs,” in *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, pp. 1060–1065, 2016.

[42] W. Li, X. Zhou, S. Shimizu, M. Xin, J. Jiang, H. Gao, and Q. Jin, “Personalization recommendation algorithm based on trust correlation degree and matrix factorization,” *IEEE Access*, vol. 7, pp. 45451–45459, 2019.

[43] P. D. Meo, “Trust prediction via matrix factorisation,” *ACM Transactions on Internet Technology (TOIT)*, vol. 19, no. 4, pp. 1–20, 2019.

[44] H. Parvin, P. Moradi, S. Esmaeili, and N. N. Qader, “A scalable and robust trust-based nonnegative matrix factorization recommender using the alternating direction method,” *Knowledge-Based Systems*, vol. 166, pp. 92–107, 2019.

[45] M. Pontiki, D. Galanis, H. Papageorgiou, S. Manandhar, and I. Androutsopoulos, “Semeval-2015 task 12: Aspect based sentiment analysis,” in *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pp. 486–495, 2015.

[46] Z. Fan, Z. Wu, X. Dai, S. Huang, and J. Chen, "Target-oriented opinion words extraction with target-fused neural sequence labeling," 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)*, pp. 2509–2518, 2019.

[47] Z. Wu, F. Zhao, X.-Y. Dai, S. Huang, and J. Chen, "Latent opinions transfer network for target-oriented opinion words extraction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, pp. 9298–9305, 2020.

[48] A. P. B. Veyseh, N. Nouri, F. Dernoncourt, D. Dou, and T. H. Nguyen, "Introducing syntactic structures into target opinion word extraction with deep learning," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 8947–8956, 2020.

[49] J. Jiang, A. Wang, and A. Aizawa, "Attention-based relational graph convolutional network for target-oriented opinion words extraction," in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pp. 1986–1997, 2021.

[50] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 168–177, 2004.

[51] B. Liu, "Sentiment analysis and opinion mining," *Synthesis lectures on human language technologies*, vol. 5, no. 1, pp. 1–167, 2012.

[52] J. Devlin, M.-W. Chang, K. Lee, and K. 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)*, pp. 4171–4186, 2019.

[53] H. Xu, B. Liu, L. Shu, and P. S. Yu, "Double embeddings and CNN-based sequence labeling for aspect extraction," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, (Melbourne, Australia), pp. 592–598, Association for Computational Linguistics, July 2018.

[54] Z. Gao, A. Feng, X. Song, and X. Wu, "Target-dependent sentiment classification with bert," *IEEE Access*, vol. 7, pp. 154290–154299, 2019.

[55] D. Tang, B. Qin, X. Feng, and T. Liu, "Effective LSTMs for target-dependent sentiment classification," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, (Osaka, Japan), pp. 3298–3307, The COLING 2016 Organizing Committee, Dec. 2016.

[56] L. Zhuang, F. Jing, and X.-Y. Zhu, "Movie review mining and summarization," in *Proceedings of the 15th ACM International Conference on Information and Knowledge Management*, CIKM '06, (New York, NY, USA), p. 43–50, Association for Computing Machinery, 2006.

[57] G. Qiu, B. Liu, J. Bu, and C. Chen, "Opinion word expansion and target extraction through double propagation," *Computational Linguistics*, vol. 37, pp. 9–27, Mar. 2011.

[58] F. Li, C. Han, M. Huang, X. Zhu, Y.-J. Xia, S. Zhang, and H. Yu, "Structure-aware review mining and summarization," in *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, (Beijing, China), pp. 653–661, Coling 2010 Organizing Committee, Aug. 2010.

[59] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, "Electra: Pre-training text encoders as discriminators rather than generators," *arXiv preprint arXiv:2003.10555*, 2020.

[60] Y. Feng, Y. Rao, Y. Tang, N. Wang, and H. Liu, "Target-specified sequence labeling with multi-head self-attention for target-oriented opinion words extraction," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1805–1815, 2021.

[61] T. Kang, M. Lee, N. Yang, and K. Jung, *RABERT: Relation-Aware BERT for Target-Oriented Opinion Words Extraction*, p. 3127–3131. New York, NY, USA: Association for Computing Machinery, 2021.

[62] S. Poria, E. Cambria, and A. Gelbukh, "Aspect extraction for opinion mining with a deep convolutional neural network," *Knowledge-Based Systems*, vol. 108, pp. 42–49, 2016.

[63] Y. Yin, F. Wei, L. Dong, K. Xu, M. Zhang, and M. Zhou, "Unsupervised word and dependency path embeddings for aspect term extraction," *arXiv preprint arXiv:1605.07843*, 2016.

[64] H. Xu, B. Liu, L. Shu, and P. S. Yu, "Double embeddings and cnn-based sequence labeling for aspect extraction," *arXiv preprint arXiv:1805.04601*, 2018.

[65] S. S. Htay and K. T. Lynn, "Extracting product features and opinion words using pattern knowledge in customer reviews," *The Scientific World Journal*, vol. 2013, 2013.

[66] I. Shamshurin, "Extracting domain-specific opinion words for sentiment analysis," in *Mexican International Conference on Artificial Intelligence*, pp. 58–68, Springer, 2012.

[67] K. Liu, H. L. Xu, Y. Liu, and J. Zhao, "Opinion target extraction using partially-supervised word alignment model.," in *IJCAI*, vol. 13, pp. 2134–2140, 2013.

[68] X. Li and W. Lam, “Deep multi-task learning for aspect term extraction with memory interaction,” in *Proceedings of the 2017 conference on empirical methods in natural language processing*, pp. 2886–2892, 2017.

[69] W. Wang, S. J. Pan, D. Dahlmeier, and X. Xiao, “Recursive neural conditional random fields for aspect-based sentiment analysis,” *arXiv preprint arXiv:1603.06679*, 2016.

[70] W. Wang, S. J. Pan, D. Dahlmeier, and X. Xiao, “Coupled multi-layer attentions for co-extraction of aspect and opinion terms,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, 2017.

[71] M. Hu and B. Liu, “Mining and summarizing customer reviews,” in *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD ’04, (New York, NY, USA), p. 168–177, Association for Computing Machinery, 2004.

[72] S. Mensah, K. Sun, and N. Aletras, “An empirical study on leveraging position embeddings for target-oriented opinion words extraction,” *arXiv preprint arXiv:2109.01238*, 2021.

[73] H. Yan, J. Dai, X. Qiu, Z. Zhang, *et al.*, “A unified generative framework for aspect-based sentiment analysis,” *arXiv preprint arXiv:2106.04300*, 2021.

[74] W. Zhou, K. Huang, T. Ma, and J. Huang, “Document-level relation extraction with adaptive thresholding and localized context pooling,” *arXiv preprint arXiv:2010.11304*, 2020.

[75] W. Jung and K. Shim, “T-rex: A topic-aware relation extraction model,” in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pp. 2073–2076, 2020.

[76] S. Chen, J. Liu, Y. Wang, W. Zhang, and Z. Chi, “Synchronous double-channel recurrent network for aspect-opinion pair extraction,” in *Proceed-*

ings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 6515–6524, 2020.

[77] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, “Automatic differentiation in pytorch,” 2017.

초 록

인터넷의 발달로, 많은 양의 데이터가 시간이 지남에 따라 축적되었다. 이로인해 긴 순차적 데이터를 처리하는 것은 웹 서비스의 핵심 문제가 되었다. 예를 들어, 유튜브, 넷플릭스, 틱톡과 같은 스트리밍 서비스는 사용자의 시청 기록 시퀀스를 사용하여 사용자가 좋아할 만한 비디오를 추천한다. 이러한 시스템은 다음에 어떤 항목이나 토큰을 볼 것인지를 예측하기 위해 사용자가 본 비디오를 각 항목 또는 토큰으로 대체하여 사용할 수 있다. 이러한 작업은 토큰 수준 분류(TLC) 작업으로 정의한다. 토큰 시퀀스가 주어지면, TLC는 이 시퀀스의 필요한 부분에서 토큰의 라벨을 식별한다. 이렇게와 같이, TLC는 다양한 추천 시스템에 적용될 수 있다. 또한 대부분의 자연어 처리(NLP) 작업은 TLC 문제로 공식화될 수 있다. 예를 들어, 문장과 문장 내의 각 단어는 토큰 레벨 시퀀스로 표현될 수 있다. 특히 정보 추출의 경우 문장의 특정 단어 간격이 정보인지 여부를 구분하는 TLC 작업으로 바뀔 수 있다.

TLC 데이터 세트의 특징은 매우 희박(Sparse)하고 길다는 것이다. 따라서 시퀀스에서 중요한 정보만 추출하여 적절히 인코딩하는 것은 매우 중요한 문제이다. 본 논문에서는 권장 시스템과 정보 추출에서 TLC의 두 가지 학문적 질문- 1) 토큰 시퀀스에서 중요한 토큰을 캡처하는 방법 및 2) 토큰 시퀀스를 모델로 인코딩하는 방법 을 해결하는 방법을 제안한다. 심층 신경망(DNN)이 다양한 웹 애플리케이션 작업에서 뛰어난 성능을 보여 왔기 때문에 추천 시스템 및 정보 추출을 위한 RNN 및 트랜스포머 기반 모델을 설계한다. 먼저 우리는 자기 주의 메커니즘을 통해 토큰 시퀀스의 중요한 부분을 포착하고 양방향 및 좌우 방향 정보를 모두 고려할 수 있는 BART 기반 추천 시스템을 설계한다. 정보 시스템에서, 우리는 의견 대상과 이웃 단어와 같은 중요한 부분에 초점을 맞추기 위해 관계 네트워크 기반 모델을 제시한다.

주요어: 토큰, 토큰시퀀스, 분류작업, 딥러닝

학번: 2016-20860

ACKNOWLEGEMENT

안녕하세요. 서울대학교 정교민 교수님 연구실 강태관입니다. 저는 6년 반 석박사 통합과정을 마치고 학위 논문을 제출하게 되었습니다. 그 동안 저에게 도움을 주셨던 분들이 매우 많습니다. 직접 말씀을 드려서 감사의 인사를 전달 드리기도 하였지만 이렇게 졸업 논문에서도 많은 분들께 감사의 말씀을 전하고자 합니다.

가장 먼저 누구보다도 많은 저의 연구실 기간 동안 저어게 도움을 주시고 언제나 지켜봐주셨던 지도교수님이었던 정교민 교수님께 감사를 드립니다. 정교민 교수님이 없으셨다면 이렇게 박사 연구를 진행할 수 없을 정도로 힘든 순간이 많았었습니다만 많은 도움을 주셔서 이렇게 제가 박사 학위를 수여 받을 수 있었습니다. 또한 마지막까지 저의 졸업 심사를 봐주시면서 많은 조언을 아끼지 않아주신 최진영 교수님, 이정우 교수님, 문태섭 교수님, 그리고 임성수 교수님께도 많은 감사의 인사를 드리겠습니다.

다음으로 연구실 동료분들에게 연구실 기간 동안 좋은 연구 및 좋은 영감을 주셔서 정말 감사하다는 말씀을 드리고 싶습니다. MILAB에 있는 기간은 제 인생에서 있어서 소중한 순간 중 하나였습니다. 또한 대학원 동기들에게도 많은 감사의 인사를 드리겠습니다. 마지막으로 언제나 저의 편이 되어주신 부모님에게 감사의 인사를 드립니다.