UNIVERSITY OF CALIFORNIA

Los Angeles

Improving Recommender Systems via Multimodal Information

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Computer Science

by

Zeyu Li

2021

ABSTRACT OF THE DISSERTATION

Improving Recommender Systems via Multimodal Information

by

Zeyu Li
Doctor of Philosophy in Computer Science
University of California, Los Angeles, 2021
Professor Wei Wang, Chair

Recommender systems are the backbones of a variety of critical services provided by tech-heavy applications and companies. In social media applications such as Facebook, Instagram, TikTok, and Snapchat, recommender systems of different types are leveraged to suggest the next post, image, or video to users to their satisfaction. Online shopping websites, such as Amazon, eBay, and Taobao, recommend items to users so that they can immediately find what they favor without the need for intensive querying. Due to its outstanding significance, both academia and industry put great effort into developing more powerful recommendation engines.

In this dissertation, we aim at improving recommender systems via different ways of incorporating data from multiple modalities such as the graphical structure of the entity relations, the attributes of entities, and the textual reviews to items from users. We exemplify the process of incorporating multimodal data via five works completed during my Ph.D. study. In these works, we will demonstrate the incorporation of different data modalities for different recommendation scenarios. NeRank focuses on the question routing task that recommends experts to question raisers combining user expertise and structural relations of entities. InterHAt considers the polysemy of features to build an interpretable click-through rate predictor. GEAPR, specialized in point of interest recommendation, decomposes the user motivation by data modalities such as social network, attribute information, and geolocation. The framework of ASPE+APRE presents a possibility to objectively understand the preference of users through what they said rather than what they purchased, clicked, or viewed. Using the objective information, recommender systems can obtain a detailed and fine-grained picture of user interests and item properties. This framework handles the descriptive statements of reviews leaving the comparative statements unattended. Finally, we introduce SAECON that deals with comparative statements and analyzes the reviews with larger coverage.

The research effort demonstrates that incorporating data from multiple modalities can hugely improve the performance of recommendations. In addition, it provides recommendation engines with interpretability to decompose the motivation behind certain user behaviors when using the service. It can be envisioned that the fusion of multimodal data will inspire the development of recommender systems in both academic research and industrial practice.

The dissertation of Zeyu Li is approved.

University of California, Los Angeles

2021

To my parents,

who support me with deep and everlasting love.

To my girlfriend Xue,

who encourages me firmly and tirelessly without hesitation.

To my kittens Krissy and Pearl,

who accompany me through the dark days with pure trust.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

It was a fantastic experience to have my Ph.D. study at UCLA. I joined UCLA in September 2016. I can still remember my day one here which was hot and sweaty. I told myself to get extremely prepared for the upcoming adventure of research. Now five years have passed and the marvelous journey is coming to a beautiful end. There were countless stories I will remember forever and countless people who deserve a huge and sincere *Thank you!* from me. Below are the most outstanding ones of them.

First of all, I would like to thank my Ph.D. advisor, Prof. Wei Wang. Prof. Wang was the first person whom I heard from at UCLA. She offered me an invaluable opportunity to study at UCLA, to know and collaborate with such talented people, to live a colorful life in the *city of stars*, and to eventually graduate as her student with great honor and, of course, a Ph.D. degree. I appreciate her tolerance of me messing up projects as an ignorant new graduate student in my first year, her support for me when I decided to pursue the research direction of recommender systems, her suggestions and advice at different stages of my research, and her trust in me to drive projects and lead teams independently.

Secondly, I would like to thank my long-term trustworthy collaborator, Dr. Wei Cheng at NEC Labs America. We have been collaborating closely since 2018. It was he that helped me find my research interest, identify many valuable and potential research questions, and enhance the technical quality of the research works. He was rigorous in different aspects which resulted in the relatively high quality of our joint works. I deeply enjoyed chatting with him about those interesting research topics and industrial practices.

My appreciation also goes to the committee members, Prof. Yizhou Sun, Prof. Kai-Wei Chang, and Prof. Quanquan Gu for their constant support and enlightenment. Prof. Sun provided constructive advice to my first publication on the heterogeneous network. Endlessly pursuing higher quality, a precious lesson I learned from her, has been benefiting and will continue to benefit my study and career. Prof. Chang supervised my work on gender debiasing word embedding which also opened the gate of natural language processing for me. His ingenuity and wisdom in research shall always be an example of mine. Prof. Gu showed me the profound knowledge of theory of computation. He cared about my research progress although we did not co-work on any research projects. He also taught me to treat work with pure honesty and integrity.

The ScAi Lab is a warm family to me while I am thousands of miles away from my hometown. Lab members learn, discuss, work, and grow together. I would like to express my gratitude to Dr. Chelsea Ju, Dr. Jyun-Yu Jiang, Dr. Ruirui Li, Dr. Cheng Zheng, Dr. Yichao Zhou, Dr. Seungbae Kim, Guangyu Zhou, Xinxin Huang, Nathan LaPierre, Junheng Hao, Xiusi Chen, and Alexander Pelletier for all the shining memories we shared. When I first met these guys, their names did not have any prefixes. It was a great honor to witness how hard they worked to earn the doctoral title and how amazing their accomplishments were over the years of diligence. Special thanks to Madelen Hem and Sim-Lin Lau, who took extreme care of our administrative work such as filing the hiring paperwork and hosting retreats or dinners. You made ScAi Lab a warm home for all of us!

In addition to them, I would also like to express my sincere gratitude to my collaborators: Dr. Jieyu Zhao, Dr. Wenchao Yu, Dr. Haifeng Chen, Yang Chen, Haiqi Xiao, Reema Kshetramade, John Houser, Zihan Liu, Yilong Qin, Yanzhao Wang, Chengyao Zhang, Aristotle Henderson, and Prateek Sane. Without your help, I would never be able to make such achievements. During the several collaborations with the UCLA Medical School, Dr. Alex Bui, Dr. Panayiotis Petousis, Dr. Peipei Ping, Dr. Naveen Raja, Dr. James Wilson, and Dr. Mario Deng provided help in various forms which I deeply appreciate.

Outside UCLA, I was fortunate to have four opportunities to work with three different companies as an intern. Nicholas Stang and Henry Venturelli hosted my first internship at Intuit as a Data Scientist Intern where I learned how to properly handle industrial data processing and model training. Dr. Tong Sun and Dr. Handong Zhao hosted my internship at Adobe Research as a Research Intern. We read papers on differential privacy together and analyzed the possibility of applying such a technique to NLP algorithms. Yongshang Wu, Dongming Lei, Dr. Tiangao Gou, Dr. Yue Zhang, and Dr. Will Feng were my peers during my two internships with Facebook. Their kind help and support was necessary for my projects to succeed.

Finally, I would like to say thank you to my family. My father, Changfa Li, and my mother, Yanhua Liu, supported me throughout my study in computer science with their strong and everlasting love. In the darkest moments, their words can always cheer me up to overcome whatever obstacles with a mighty will. My girlfriend, Xue Xia, accompanied me through various gloomy occasions when loneliness hit during the pandemic and papers got rejected. Her tireless encouragement was the lighthouse that navigated me on the ocean of uncertainty, disappointment, nostalgia, exhaustion, and loneliness. My little furry friends, Krissy and Pearl, also loved me using their special ways. They trusted me, played with me, and added tons of fun to my life. I sincerely appreciate all their love and devotion which are an invaluable lifelong treasure for me.

I have never imaged that I could receive help from such many people along the journey. Hopefully, they are also proud of the works presented in this dissertation. Life is full of struggles and challenges as well as surprises after them which are worth fighting for. This dissertation only marks the end of my Ph.D. journey. The future is yet to come and I am ready to take on the fight.

VITA

2012 – 2016	B.S. in Computer Science and Technology, Harbin Institute of Technology, China
2016 – 2021	Teaching Assistant, Teaching Associate, Graduate Student Researcher, and Teaching Fellow, University of California, Los Angeles, CA
2018	(Summer) Data Scientist Intern, Intuit Inc., Woodland Hills, CA
2019	(Summer) Research Intern, Adobe Research, San Jose, CA
2020	(Summer) PhD Machine Learning Software Engineer Intern, Facebook Inc., Remote, CA
2021	(Summer) PhD Machine Learning Software Engineer Intern, Facebook Inc., Remote, CA

PUBLICATIONS

Zeyu Li, Yilong Qin*, Zihan Liu*, Wei Wang. *Powering Comparative Classification with Sentiment Analysis via Domain Adaptive Knowledge Transfer*, In The 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP'21)

Zeyu Li, Wei Cheng, Reema Kshetramade, John Houser, Haifeng Chen, Wei Wang. *Recommend for a Reason: Unlocking the Power of Unsupervised Aspect-Sentiment Co-Extraction*, In The 2021 Conference on Empirical Methods in Natural Language Processing: Findings (Findings of EMNLP'21)

Zeyu Li, Wei Cheng, Haiqi Xiao, Wenchao Yu, Haifeng Chen, Wei Wang. *You Are What and Where You Are: Graph Enhanced AttentionNetwork for Explainable POI Recommendation*, In Proceedings of The 30th ACM International Conference on Information and Knowledge Management (CIKM'21)

Jyun-Yu Jiang, **Zeyu Li**, Chelsea J.-T. Ju and Wei Wang. *MARU: Meta-context Aware Random Walks for Heterogeneous Network Representation Learning*, In Proceedings of The 29th ACM International Conference on Information and Knowledge Management (CIKM'20)

Zeyu Li, Wei Cheng, Yang Chen, Haifeng Chen, Wei Wang. *Interpretable Click-Through Rate Prediction through Hierarchical Attention*, 13th ACM International WSDM Conference (WSDM'20)

Zeyu Li, Jyun-Yu Jiang, Yizhou Sun, Wei Wang. *Personalized Question Routing via Network Embedding*, Thirty-Third AAAI Conference on Artificial Intelligence (AAAI'19)

Jieyu Zhao, Yichao Zhou, **Zeyu Li**, Wei Wang, Kai-Wei Chang. *Learning Gender-Neutral Word Embedding*, 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP'18)

Zeyu Li, Hongzhi Wang, Wei Shao, Jianzhong Li, Hong Gao. *Repairing Data through Regular Expressions*, 42nd International Conference on Very Large Data Bases (VLDB'16)

CHAPTER 1

Introduction

1.1 Recommender Systems

Mobile internet boomed in the 2010s. Innovative mobile devices were designed, created, and made cheap and easily accessible. More users connected to the Internet and participated in a wide variety of online services such as online shopping, social network, search engine, and online entertainment. With more users, high-tech companies were able to collect an enormous amount of data that can precisely record the sequences of user behaviors. These datasets then powered the massive scale machine learning-based *recommender systems*. These recommender systems, in turn, provided smart customized experiences to users and fueled the prosperity of business.

What are recommender systems? Recommender systems refer to intelligent platforms and software that learn user preference and predict users' interaction behaviors such as watching a video or a post, clicking an advertisement, purchasing an item, visiting a point of interest, or rating and reviewing a product after purchase. In this dissertation, we use *interaction* to denote the set of possible behaviors that happened between users and items.

Nowadays, recommender systems serve as the backbones of a wide range of applications or web services. Amazon, Taobao, and JD use different recommendation models for search, "item you may be interested in", "people also viewed", etc., which increase the visibility of similar items and elongate user engagement. Facebook, Instagram, TikTok, and Snapchat employ recommendation models to build user profiles and correspondingly customize the next batch of content to display so that users can interact with the application longer or more frequently. Yelp and Google Map learn user preference via recommender systems so that they can tailor the query results considering the diverse tastes of users.

The oil that powers the recommender systems to build user preference profile is data, which includes user-item interaction history records, the social relationships of users described by graphs, the static attribute information of users and items, the review text users write as feedback of items, etc. Intuitively, the more data involved in the development process of recommender systems, the better the recommendation performance will get. This idea has been the guiding principle for the research in recommendation in the past years.

Traditional recommender systems are based on the history of user interaction or engagement. The intuition supporting the idea is that users showing similar purchasing patterns tend to continue to buy the same group of items. This idea is typically defined as *collaborative filtering*. For example, matrix factorization (MF) decomposes a static interaction matrix of user and item into a user embedding matrix and an item embedding matrix. Follow-up works of matrix factorization employ additional regularizations or constraints to enhance accuracy or efficiency.

The downsides of matrix factorization-based are obvious. First, it cannot model temporal information and discards the dynamics of user interests and item supplies. Items being recommended are global regardless of the time or season of the query. Second, it completely fails for new users and new items unless being repeatedly retrained with high computational cost. Third, it is incapable of utilizing data from other domains such as attribute, network, or text. Fruitful research progress has been made attempting to solve the third problem by advanced recommendation algorithms. This dissertation also pursues that direction and demonstrates efforts in incorporating multimodal data into building better recommendation algorithms.

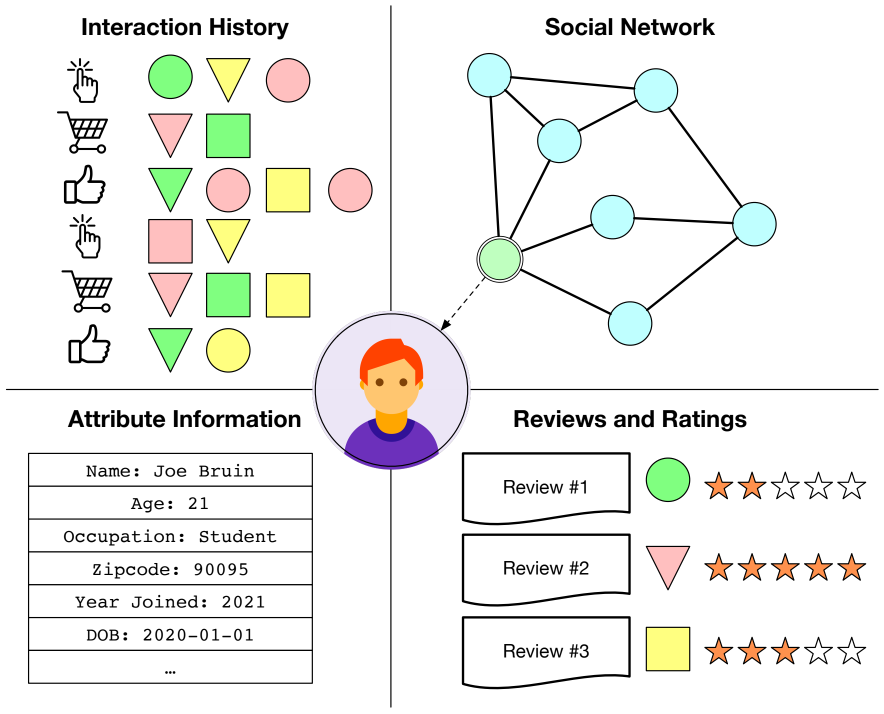


Figure 1.1: Example data modality in recommender systems.

1.2 Incorporating Multimodal Data

The recent development of deep learning provides effective tools for modeling multimodal data and fusing the information for a more accurate prediction. Figure 1.1 shows four example modalities of the recommender systems including *Interaction history*, *Social network*, *Attribute information*, and *Reviews and ratings*. Incorporating these four different types of data is comprehensively discussed in this dissertation such as the corresponding pioneering works, past milestones, our contribution in these threads, and the potential exploration directions for future effort. We introduce these data types as follows.

Interaction history Interaction history refers to the past engagement of users with items. According to the type of services of the platform, interactive behaviors can be watching videos or posts, clicking on advertisements, commenting on items, and visiting places, etc. Different kinds of behaviors represent different levels of preference. For example, for a news recommender, a possible descending order of likeness can be *retweet/share* = *comment* > *like* > *watch entirely* > *click but leave* > *non-click* > *dislike* > *hide/report*. Although applications may not have the whole spectrum of likeness buttons, they can still build algorithms with discriminative loss terms on different signals. These signals reveal the users' loves and hates and help the recommender suggest items users also like and avoid things they dislike.

Social network Social network has attracted extensive attention itself due to its diverse research directions such as node representation learning for account classification, edge prediction for "friends you may know" suggestion, clique detection for mining potential interest groups, etc. The topological structure can provide proximity information of the entities in the network. As some online services also contain social networks, it is worth being incorporated into the recommendation since, intuitively, the friends or connected users of a user can influence their behaviors. Frequently used techniques of network-related algorithms include network embedding and graph neural networks which will be discussed in later chapters.

Attribute information Attributes of items are typically static demographic features such as user name, age, and location (zip code). Attribute information can help the machine learning algorithm identify certain patterns of the sub-cohort of the user base. For example, young users tend to have similar behavioral patterns, and geographically close users can also have similar needs to those specific regions, etc. One successful attempt to leverage attribute information is factorization machines where different orders of interactions are computed for the features.

Reviews and ratings Different from the above three modalities, reviews and their associated ratings provide a subjective angle of user preference, an angle not based on what users do but they say. Understanding reviews requires techniques from natural language processing. Applying text mining or text representation learning to build recommender systems is not novel. But most of these work either only encodes the text into latent representation or assigns a fixed number of latent aspects without extracting the actual sentiments expressed by the text. Encoding the entire sentence or review can hardly provide any clues of interpreting the recommendation using the text data. Therefore, a method that can efficiently extract the entities in that text with associated user sentiments is strongly needed.

In summary, we see that the four data modalities can benefit the performance of recommender systems in many aspects. Therefore, we propose methods in this dissertation to utilize these data from different aspects to boost the recommendation performance. In the next section, we go through the research works in a nutshell and summarize their contribution.

1.3 Contribution of Dissertation

In this dissertation, we list three groups of work including NeRank, InterHAt and GEAPR, and ASPE+APRE and SAECON. We briefly summarize their contribution in advancing the incorporation of multimodel data into recommender systems.

First of all, NeRank is a novel idea for question routing marking the first work that considers the network structure of a community-based question answering forum such as Stack Overflow. Specifically, we model the different types of entities in the graph as well as their relations as a heterogeneous information network so that the diverse semantics attached to the edges can be accurately captured. In addition, NeRank fuses text data with network embedding through a text encoder via which the network embeddings learned are aligned with the text semantic. The expertise information is preserved in the embeddings too.

Secondly, InterHAt and GEAPR are two projects on interpretable recommender systems. InterHAt tries to explain a CTR prediction using a hierarchical attention network, which is a novel architectural design. With such a design, a recommender can compute the interactions of different features potentially coming from different modalities and output the salience of these interactions via attention scores. GEAPR explains visits to the point of interest by decomposing the potential motivation into several different factors such as geolocation, attributes, or network structures. Here the different types of data are combined to recommend new points of interest. The significant contributions of these works are building explainable models for recommendation scenarios and including attribute information in POI recommendations.

Finally, ASPE+APRE and SAECON are two examples of incorporating natural language in the recommender systems. ASPE+APRE consists of two parts including an Aspect-Sentiment Pair Extractor and an Attention-Property-aware Rating Estimator. ASPE extracts aspect-sentiment pairs (AS-pairs) which are the basic units of user preference on certain item properties. The sentiment terms indicate the emotion of users such as like or hate. APRE takes in the AS-pairs, pools the sentiments for the same aspects, and aggregates the detailed sentiments to predict the final rating score a user will give an item. However, this framework can only deal with descriptive statements but fails to handle comparative ones. This is where SAECON comes in. It can detect comparative sentences in reviews and predict which one of the two given entities wins the comparison.

The novelty of the ASPE+APRE framework is that it marks the first attempt to model user preference to aspects explicitly and explain the rating prediction by aspects with AS-pairs as the exact evidence. Of course, the performance is improved against all existing rating prediction baselines. For SAECON, our contribution is that the accuracy of the comparative preference classification (CPC) is enhanced with the help of the innovative domain adaptive knowledge transfer that borrows training signal from the aspect-based sentiment analysis (ABSA) data and task.

Overall, this dissertation presents novel efforts in incorporating data from different modalities into improving recommender systems in several distinct scenarios such as question routing, CTR prediction, POI recommendation, and rating prediction.

1.4 Overview

In the last section of this chapter, we present how the following parts of the dissertation are organized. Chapter 2 introduces the existing research effort relevant to this dissertation. We elaborate on relevant research problems and techniques separately. Chapter 3 introduces NeRank, a framework that recommends domain experts of a community-based question answering platform to users raising new questions. Chapter 4 introduces InterHAt, an interpretable click-through rate prediction model that predicts whether a user clicks a post or an advertisement and explains the prediction. Chapter 5 introduces a location-based recommendation framework to recommend points of interest to users taking into consideration structural context, attributes, geolocation, and immediate neighbors. Chapter 6 and Chapter 7 cover two natural language processing-based models for understanding user preference both in descriptive and comparative statements. With them, recommender systems can incorporate review data to decompose and interpret users' attention to the items.

CHAPTER 2

Related Work

This chapter discusses the related work for the dissertation. According to their topics, these sections are divided into two parts. The first part covers relevant research progress on the different recommendation scenarios such as existing work and their corresponding advantages and disadvantages. The second part covers the background of related techniques utilized in the following research projects.

2.1 Related Work for Recommendation Scenarios

2.1.1 Question Routing

Question routing, which will be discussed in Chapter 3, is defined as predicting whether a user in community-based question answering (CQA) will share knowledge and answer a given question [ZLK12]. The majority of previous works fall into two categories: feature engineering-based methods and matrix factorization-based methods.

Feature engineering algorithms [ZLK12, JW13, CP13] feed features extracted from users, questions, and their relations to models such as SVM [HDO98] and Linear Regression [CP13] to rank the user authority to make recommendations. However, they require carefully crafted features and the performance relies heavily on feature selection.

Matrix factorization models decompose feature matrices based on the *low-rank* assumption to discover users' expertise on particular words and compute ranking scores by the inner product of the user and question feature vectors [ZZH15]. It suffers from the limitation of the bag-of-word model which is unable to preserve sequential text semantics.

2.1.2 Click-Through-Rate (CTR) Prediction

CTR prediction models predict the probability of a user clicking on a certain item within a certain context. It has drawn great attention from both academia and industry [GZL17, Ren10, CKH16, QCR16, WZX18, XYH17, LLS16, SHJ16, ZZS18, LZZ18, WFF17, HC17, ZZZ18, ZMF18] due to its significant impact on online advertisements. The advancement of CTR prediction algorithms essentially shows a trend towards deeper model architectures since they are more powerful in feature interaction learning [SSX18].

Factorization Machine (FM) [Ren10] assigns a d-dimensional trainable continuous-valued representation to each distinct feature, learns the representations of distinct features, and makes predictions by a linear aggregation of first- and second-order features. Although FM can be generalized to high-order cases, it suffers from the computational cost of exponential complexity [BFU16] and the low model capability of shallow architecture. Field-aware Factorization Machine (FFM) [JZC16] assumes that features may have dissimilar semantics under distinct fields and extends the idea of FM by making the feature representation field-specific. Although it achieves better CTR results than FM, the parameter size and complexity are also increased and overfitting is easier to happen. Attentional Factorization Machine (AFM) [XYH17] extends FM with an *attention net* that improves not only the performance but also interpretability. The authors argue that the feature salience provided by the attention network greatly enhances the transparency of FM. That said, AFM can only learn up to the second-order attention-based salience due to the inherent architectural limit of FM.

Wide&Deep [CKH16] consists of a wide and a deep component, which are essentially a generalized linear model and a multi-layer perceptron (MLP), respectively. The CTR prediction is made by a weighted combination of the outcomes of the two components. Note that the deep component, i.e., the MLP, ruins the possibility of explaining the prediction because the layer-wise transformations are conducted on unit level instead of feature level, and individual unit level values can not carry concrete and complete semantic information of features. Deep&Cross Network (DCN) [WFF17] slightly differs from Wide&Deep in that DCN replaces the linear model with a *cross-product* transformation to integrate high-order information with non-linear deep features. DeepFM [GTY17] improves these two models by replacing the polynomial production with an FM component. The deep MLP component captures the high-order feature interaction and the FM analyzes the second-order feature interaction. xDeepFM [LZZ18] claims that MLP parameters are actually arbitrarily modeling the *implicit* feature interactions. The authors hence introduce the compressed interaction network (CIN) to model the *explicit* features alongside the implicit ones. Recent works from industry practice include DIN [ZZS18] and DIEN [ZMF18] that respectively model the static and dynamic shopping interest of users. Both work heavily rely on deep feed-forward networks which are typically unexplainable.

All aforementioned CTR prediction models depend heavily on deep neural networks and achieve ever-increasing performances. However, as a sword has two edges, deep learning algorithms suffer from potential risks in reliability and security. The weights and activations of hidden layers are hardly explainable and the causal relationships between the inputs and outputs are concealed and uncertain. They all fail to provide any feature-level clues that explain why such deep feature learning strategies enhance or diminish the CTR performance. Consequently, the predictions made thereby without clear explanations are considered untrustworthy.

2.1.3 Point of Interest (POI) Recommendation

Points of interest refer to locations that customers of online business directories or review forums are interested in. POI recommendation is a popular task since it directly affects the revenue and reputation of POI platforms. Research on this topic has been fruitful [LPC17, LJJ19, YBZ17, YYL11, LGH16, LWW16, ZC15, ZC13, ZCL14, LWS14, LZX14, LCL15, WYC20, YCG20, ZYZ19, JXZ19, ZZK17, WWT17, HPN18, LZX20, ZGH19, ZMZ19, SQC20, CLZ20, MZW18, CGC20, ZYL17, RAB20, LLW20, YZC16, LCZ20, YWW17, CZH17, YZZ18]. We categorize them into traditional POI models and deep learning-based ones and discuss their pros and cons by examples.

Traditional models USG [YYL11] is a collaborative filtering-based model for POI recommendation. It suggests that not only social connections but also geographical influences can help improve the accuracy of POI recommendations. Therefore, USG specifically looks at three complementary factors: user preference of POIs, social influences, and geographical influence. GeoSoCa [ZC15] digs deeper into POIs' property that the *category* of POI is taken into consideration. Authors argue that category is critical information and it affects user preference since people have different biases towards different types of POIs. Therefore, GeoSoCa firstly employs the biases measurement to build personalized POI popularity. ASMF and ARMF [LGH16] refer to augmented square error-based MF and augmented ranking error-based MF, respectively. Despite the minor difference in the selection of error function, they both focus on user relations from three dimensions which are generally defined as *friendships*, namely social friends, location friends, and neighbor friends. The emphases on user friendships strongly indicate that users' preferences can be greatly reshaped by and effectively learned from human-human connections.

Deep learning-based models PACE [YBZ17] utilizes a multi-task learning architecture that models user context, POI context, and user-POI interaction simultaneously. Technically, it assigns a learnable embedding vector to each user and POI to capture their latent features and use a feed-forward layers-based deep network to predict *user context*, *POI context*, and check-ins. SAE-NAD [MZW18] is composed of a self-attentive encoder (SAE) for user-POI interaction modeling and a neighbor-aware decoder (NAD) for geographical context modeling. SAE differentiates user preference degrees in multiple aspects by self-attention. NAD ensures only physically and preferentially nearby users' check-ins receive stronger weights in the POI recommendation. APOIR [ZYZ19] signifies the first application of generative adversarial network (GAN) [GPM14] on POI recommendation. The co-trained two sides of the mini-max game are the recommender aiming to suggest the most probable POI check-ins and the discriminator that separates the recommended POI from the true visits.

All aforementioned previous approaches carefully attend to user preferences mining and POI profiling in terms of categories and geolocations. However, we notice two major disadvantages that deserve some improvements. First, attributes of individuals have long been ignored even though recommendation models such as factorization machine (FM) [Ren10] have demonstrated the usefulness of user attributes to enhance accuracy. Existing algorithms have a delicate design on user preference and geolocation modeling [ZYL17, MZW18] but lack latent attribute learning for users. Second, all previous models, either deep learning-based or MF-based, preserve the information of users or POIs by latent representations without explicitly highlighting salient factors or signals. Different information sources are integrated by simple operations such as addition, concatenation, or multilayer perceptrons (MLP). Consequently, the trained models with unjustifiable parameters fail to explain why users favor or dislike certain POIs and what really causes a visit.

Sequential POI recommendation Sequential, or successive, POI recommendation (SPR) models [ZMZ19, ZZK17, YCG20, JXZ19, SQC20, WYC20] is a separate branch of location-based recommendations from general POI recommendations. They are essentially different use scenarios. General models emphasize the modeling of general user and POI characteristics whereas SPR models focus on time-sensitive check-in suggestions and temporal POI visit behavior mining.

2.1.4 Aspect-Based Sentiment Analysis

Aspect-based sentiment analysis (ABSA) [XLS20, WML18] predicts sentiments toward specific aspects of an item mentioned in the text. It has been utilized in recommendation scenarios to discover the user preference disclosed in product reviews. Traditional approaches of ABSA utilize SVM for classification [KZC14, WAC14, ZZL14] while neural network-based approaches employ variants of RNN [NS15, AG20], LSTM [TQF16, WHZ16, BLB19], GAT [WSY20], and GCN [PND20, XZL20].

More recent works widely use complex contextualized NLP models such as BERT [DCL19]. [SHQ19] transform ABSA into a Question Answering task by constructing auxiliary sentences. [PO20] and [TJL20] utilize contextualized language encoding to capture the context of aspect terms to predict the sentiments. [CSW20] focuses on the consistency of the emotion surrounding the aspects, and [DSW20] equips pre-trained BERT with domain-awareness of sentiments.

2.1.5 Aspect or Sentiment Terms Extraction

Aspect and sentiment terms extraction is a presupposition of ABSA. However, manually annotating data for training, which requires the hard labor of experts, is only feasible on small datasets in particular domains such as Laptop and Restaurant [PGP14, PGP15] which are overused in ABSA.

RINANTE [DS19] and SDRN [CLW20] automatically extract both terms using rule-guided data augmentation and double-channel opinion-relation co-extraction, respectively. However, the supervised approaches are too domain-specific to generalize to out-of-domain or open-domain corpora. Conducting domain adaptation from small labeled corpora to unlabeled open corpora only produces suboptimal results [WP18]. SKEP [TGX20] exploits an unsupervised PMI+seed strategy to coarsely label sentimentally polarized tokens as sentiment terms, showing that the unsupervised method is advantageous when annotated corpora are insufficient in the domain of interest.

2.1.6 Aspect-based Recommendation

The aspect-based recommendation is a category of recommendation algorithms that decompose the motivation of click or purchase into different latent aspects in pursuit of a fine-grained preference study. It is a relevant task with a major difference that specific terms indicating sentiments are not extracted. Only the aspects are needed [HYW19, GCH19, HJW20, CZJ18]. Some disadvantages are summarized as follows. Firstly, the aspect extraction tools are usually outdated and inaccurate such as LDA [HYW19], TF-IDF [GCH19], and word embedding-based similarity [HJW20]. Second, the representation of sentiment is scalar-based which is coarser than the embedding-based method used in our work.

2.1.7 Review-based Rating Prediction

Review-based rating prediction is an important task in the recommendation. Although the rating scores are typically discrete, the task is modeled as a regression task where the model predicts the scalar scores using the text of the review. Related approaches utilize text mining algorithms to build user and item representations and predict ratings [KPO16, ZNY17, CZL18, CZJ18, LLD19, BLT17]. However, the text features learned are latent and unable to provide explicit hints for explaining user interests.

2.1.8 Comparative Preference Classification

Comparative Preference Classification (CPC) originates from the task of Comparative Sentence Identification (CSI) [JL06]. CSI aims to identify the comparative sentences. [JL06] approaches this problem by Class Sequential Mining (CSR) and a Naive Bayesian classifier. Building upon CSI, [PBF19] proposes the task of CPC, releases CompSent-19 dataset, and conducts experimental studies using traditional machine learning approaches such as SVM, representation-based classification, and XGBoost. However, they neglect the entities in the comparative context [PBF19]. ED-GAT [MMW20], a more recent work, uses the dependency graph to better recognize long-distance comparisons and avoid falsely identifying unrelated comparison predicates. However, it fails to capture semantic information of the entities as they are replaced by *entityA* and *entityB*. Furthermore, having multiple GAT layers severely increases training difficulty.

2.2 Related Work for Techniques

2.2.1 Network Embedding

Network embedding models learn low-dimensional representations for nodes in a network that preserve the structural context of nodes [PAS14, GL16, TQW15, SHZ18, DCS17, CS17]. As the type of graph can be either homogeneous or heterogeneous depending on whether the graph has different node types, the network embedding methods also fall into these two categories. For heterogeneous information networks, the diversified node and edge types bring forth additional semantic information of networks which motivates the metapath-based network embedding algorithms [SHY11]. [CS17] proposes a task-specific and path-augmented model that jointly optimized the network-general and task-specific objectives. Metapaths are specifically selected for the task. *metapath2vec* and *metapath2vec++* [DCS17] combine metapaths with the word2vec model for heterogeneous embedding learning.

2.2.2 Attention Mechanism

Attention mechanism learns a function that weighs over intermediate features and manipulates the information that is visible to other modules of the machine learning algorithm. Due to its capability to pinpoint and amplify salient features that greatly affect the predictions [GBY18], attention mechanism is regarded as a reasonable and reliable way to explain the decision-making procedure in many tasks such as recommender systems [XYH17, YZZ18], health care systems [CBS16], computer vision [XXY15], visual question answering (VQA) [LYB16, HAA18], etc.

For example, RETAIN [CBS16] studies electric health records (EHR) of patients with a two-layer attention network that identifies and explains influential hospital visits and significant clinical diagnoses associated with the visits. Co-attention mechanism [HAA18] in VQA proposes question-guided visual attention and visual-guided question attention on word level, phrase level, and question level. Three levels of information are combined to predict the answer with improved performance while retaining the explainability of the outcomes.

In the natural language domain, language-specific and across-language attention networks based on linguistic hierarchy [YYD16, PP17] such as words and sentences are proposed for document classification tasks. Another form of attention in NLP is self-attention. Researchers from Google design Transformer [VSP17] based on multi-head self-attention in which tokens in a sentence attend to other tokens within the same sentence to learn the compound sentence semantics. Using the strong learning power of Transformer, BERT [DCL18], built by stacking a number of bi-directional Transformer layers, achieves state-of-the-art performance on 11 major NLP tasks. The success of BERT shows the outstanding feature interaction power of Transformer.

In summary, a variety of existing works have endorsed that utilizing attention mechanism improves both accuracy and transparency of the model. Being capable to identify important features and feature interactions makes attention mechanism a reliable way to explain the *thinking* of machine learning models [GBY18]. Therefore, attention is considered as the solution to *interpretablity* in various research scenarios including recommender systems [XYH17, YZZ18, LCC20], graph representation learning [VCC17], computer vision [XXY15], etc. Although the attention modules are not trained for generating human-readable prediction rationales, they can still reveal the salience distribution of information when the feature representations flow through the model architecture, which can serve as a form of explanation.

CHAPTER 3

Personalized Question Routing via Heterogeneous Network Embedding

Question Routing (QR) on Community-based Question Answering (CQA) websites aims at recommending answerers that have high probabilities of providing the *accepted answers* to new questions. The existing question routing algorithms simply predict the ranking of users based on query content. As a consequence, the question raiser information is ignored. On the other hand, they lack learnable scoring functions to explicitly compute ranking scores.

To tackle these challenges, we propose NeRank in this chapter that, firstly, jointly learns representations of question content, question raiser, and question answerers by a heterogeneous information network embedding algorithm and a long short-term memory (LSTM) model. The embeddings of the three types of entities are unified in the same latent space. Secondly, NeRank conducts question routing for personalized queries, i.e., queries with two entities `(question content, question raiser)`, by a convolutional scoring function taking the learned embeddings of all three types of entities as input. Using the scores, NeRank routes new questions to high-ranking answerers that are skillfulness in the question domain and have similar backgrounds to the question raiser.

Experimental results show that NeRank significantly outperforms competitive baseline question routing models that ignore the raiser information in three ranking metrics.

3.1 Motivation and Background

Community-based question answering (CQA) such as Stack Overflow[1] is rapidly gaining popularity and becoming an important type of social media for sharing and spreading knowledge. Through CQA websites, users with questions are able to quickly locate answers provided by experts. A user can also create a new post if relevant and satisfactory QA records do not exist, and then wait for answers from the community. After several responses are gathered, the question raiser reviews the answers and selects one that he/she is the most satisfied with as the *accepted answer*. The answer collection can be unacceptably time-consuming due to the lack of an efficient way to find the *domain experts*. As a result, a large number of questions remain poorly addressed.

One of the solutions to promote answer collection is to automatically identify users that tend to contribute high-quality answers and then send answer invitations to them. The answer collection is consequently accelerated since these users are able to immediately spot the questions of their expertise. Such task is also known as *question routing* and is previously addressed by feature engineering-based approaches [ZLK12, JW13, CP13]. Features exploited include the statistics of users, the language modeling features of question content, and the relationships between users and questions. All of them focus on estimating users' authority level and identifying the skillful users for recommendation.

However, the feature engineering-based strategies have at least three limitations as follows. First, they are not personalized, i.e., they cannot customize recommendations for questions raised by users with diverse characteristics due to ignoring the background and preference of the question raisers. Second, they lack explicit definitions of scoring functions for queries with multiple entities and, therefore, have trouble computing scores for new question routing queries. Third, they model question content by language model or topic model features which are unable to capture the complex semantics of question content. Also,

[1]https://stackoverflow.com

their representation power is undermined when handling questions with new topics that are unobserved or underrepresented in the training set.

In order to overcome the above limitations, we propose *NeRank*, which stands for Network embedding-augmented Ranking for question routing. NeRank assesses the personalized authority, i.e., the authority of an answerer with respect to not only the question content but also the background of the question raiser, for question routing. In addition, the recommended answerers are also expected to have similarities with the question raiser in domains of interest to fulfill the personalization requirement so that their responses conform to the raiser's anticipation.

In particular, NeRank models CQA websites as heterogeneous information networks (HIN), namely *CQA networks*, and applies a metapath-based heterogeneous network embedding algorithm to CQA networks to learn representations for question raisers and question answerers. A long short-term memory (LSTM) component is specifically utilized to learn question content representation. Using network embedding, the proximity information between the entities in a CQA network is preserved.

NeRank models the question routing task as a ranking problem and utilizes a convolutional neural network (CNN) to compute the ranking score of an answerer given a query `(question raiser, question content)`. Such ranking score measures the probability of the answerer providing the accepted answer to this question. Compared with previous frameworks [ZLK12, ZZH15, JW13], our ranking function explicitly computes the ranking scores taking advantage of rich non-linear information of the three entities.

We summarize our contributions as follows:

- We propose NeRank for personalized question routing on CQA websites. Compared with existing models, NeRank considers question raiser's profile in addition to question content. To the best of our knowledge, this is the first work on personalized question routing on CQA websites.

- We learn representations of entities by an HIN embedding method and compute ranking scores for answerers by an explicitly defined CNN scoring function given the learned representations as input. It is a novel attempt to apply an embedding-based method to question routing.
- We conduct extensive experiments on two real CQA datasets and evaluate the routing performance of NeRank via three ranking metrics. Our model outperforms the baselines in these metrics. Results also show that the HIN embedding algorithm and CNN scoring function improve the ranking performance.

3.2 Preliminaries and Problem Statement

A CQA network is built upon a static archive of a CQA website conserving all question-answer sessions accumulated over time. We create question $\underline{\text{R}}$aiser set $R = \{r_1, r_2, \ldots, r_m\}$ and $\underline{\text{A}}$nswerer set $A = \{a_1, a_2, \ldots, a_k\}$ where m is the number of users who have asked questions, i.e. question raisers, and k is the number of users who have answered questions, i.e. question answerers, in this CQA website. Note that we only model users that have asking or answering records in the dataset. Hence, each user of the CQA website may have one or two embeddings associated with the role(s) they played. We create $\underline{\text{Q}}$uestion set $Q = \{q_1, q_2, \ldots, q_l\}$ where l denotes the number of questions. There exist two relations among these entities, namely "raises a question" between entities in R and Q and "answers a question" between entities in A and Q.

A CQA network is defined as a heterogeneous information network $G = (V, E, T, \phi)$, where $V = R \cup Q \cup A$ denotes the node set; E denotes the edge set; T denotes the set of three entity types involved [ZLZ17]; $\phi : V \to T$ is a labeling function that maps an entity into its type $t \in T$. Each edge in a CQA network symbolizes an asking or answering record. Note that entities of R and A do not directly interact with each other and hence there is no connection between them. Figure 3.1 shows a toy example of a CQA network. Node r_2 is linked to q_2 and q_3, meaning that r_2 poses q_2 and q_3. q_3 is linked to a_3 and a_4 since a_3 and a_4 answer q_3. a_1 and a_2 have strong similarity since they both answer q_1 and q_2.

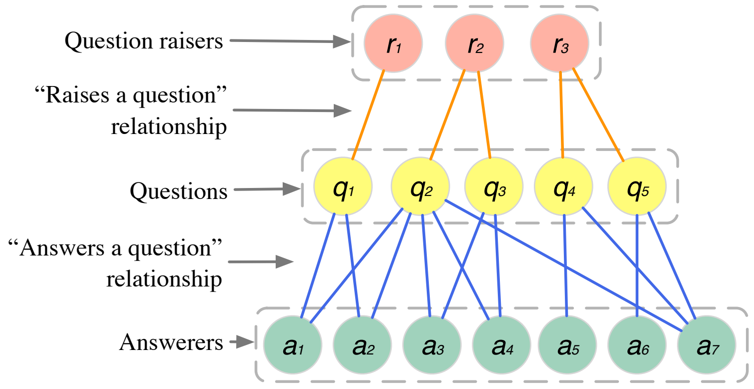

Question raisers
"Raises a question" relationship
Questions
"Answers a question" relationship
Answerers
r1
r2
r3
q1
q2
q3
q4
q5
a1
a2
a3
a4
a5
a6
a7

Figure 3.1: A heterogeneous network view of a CQA website.

Using above notations, we define the personalized question routing as the following: Given a CQA network $G = (V, E, T, \phi)$ and a query `(question raiser, question content)` denoted by $\gamma = (r, q)$ where $r \in R$ is a question raiser and $q \in Q$ is a new question, compute the ranking scores for answerers $a \in A$ and select the answerer with the highest ranking score as the predicted provider of the accepted answer.

3.3 NeRank

In this section, we demonstrate the technical details of NeRank. The list of notations is provided in Table 3.1 in advance for the convenience of later discussion.

3.3.1 Problem Overview

We formalize the personalized question routing problem as a ranking task in NeRank which ranks the probabilities of potential answerers contributing the accepted answers using the embeddings of entites. Specifically, it has two steps: modeling entity-wise similarity and computing the ranking scores of answerers given `(question raiser, question content)`

Table 3.1: Frequently used notations for NeRank.

Notation(s)	Definition
G	The CQA network.
E, V, T	The edge set, node set, and type set of G.
d	The dimension of entity embeddings.
$\boldsymbol{v}_e$, $\boldsymbol{u}_e$	The d-dimensional embedding of entity e.
n, c	The center and context entity [MSC13].
$\mathcal{P}$	A metapath.
$w_{\mathcal{P}}$	A walk generated according to $\mathcal{P}$.
τ_j	The entity type of the j-th element of $\mathcal{P}$.
$e_t^{(i)}$	The i-th entity of $w_{\mathcal{P}}$ with type τ_t.
$\phi(e)$	The entity type of the entity e.
D	The corpus of all (n, c) pairs, positive samples.
$\boldsymbol{\Theta}$	The parameter set of NeRank.

queries. The trained representations for the three types of entities are expected to preserve both the entity-wise *proximity* information and the question-raiser-specific *expertise* information.

In the following subsections, we explain how the NeRank pipeline (shown in Figure 3.2) acquires the embeddings with proximity and enterprise information and computes the rankings scores. There are two steps in the training procedure. Step 1 learns the entity embeddings using an LSTM-equipped metapath-based network embedding algorithm. Step 2 computes the ranking scores using a convolutional scoring function and finally outputs a ranked list of answerers.

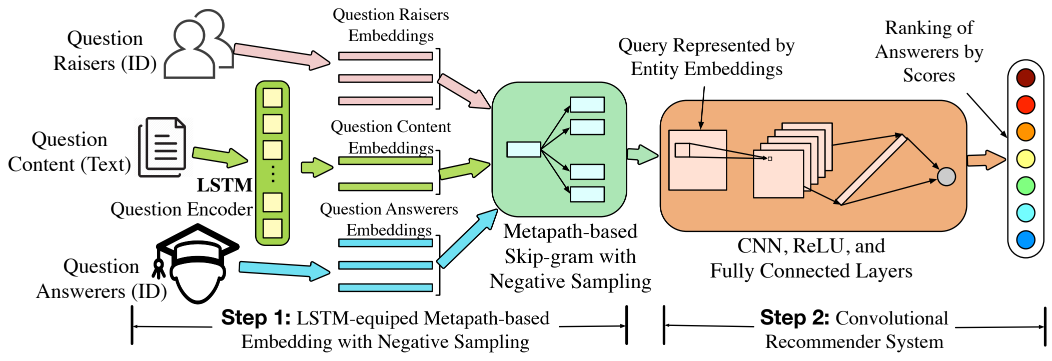

Question Raisers (ID)
Question Content (Text)
Question Answerers (ID)
LSTM
Question Encoder
Question Raisers Embeddings
Question Content Embeddings
Question Answerers Embeddings
Metapath-based Skip-gram with Negative Sampling
Query Represented by Entity Embeddings
CNN, ReLU, and Fully Connected Layers
Ranking of Answerers by Scores
Step 1: LSTM-equiped Metapath-based Embedding with Negative Sampling
Step 2: Convolutional Recommender System

Figure 3.2: The pipeline of NeRank with two training steps.

3.3.2 LSTM-equiped Metapath-based Embedding with Negative Sampling

To capture proximity information, we learn embeddings of heterogeneous entities using an LSTM-equipped metapath-based heterogeneous network embedding model. We first explain the metapath-based Skip-gram on HIN and then show the jointly optimized LSTM model for question content representation learning.

3.3.2.1 Metapaths for HIN Embedding.

HIN owns various node types, which differs from homogeneous networks. Simply applying the original random walk-based Skip-gram to HINs results in biases towards certain types of nodes [SHY11]. To create a bias-free walk corpus for Skip-gram on HINs, [DCS17] generate walks according to the patterns specified by *metapaths*. It has been proved that HIN embedding models benefit from metapaths in reducing biases [DZT15, SH12, SNH13].

A metapath $\mathcal{P}$ is a sequence of objects linked by relations in form of $\tau_1 \xrightarrow{\pi_1} \tau_2 \xrightarrow{\pi_2} \ldots \tau_t \xrightarrow{\pi_t} \tau_{t+1} \ldots \xrightarrow{\pi_{l-1}} \tau_l$. $\pi = \pi_1 \circ \pi_2 \circ \cdots \circ \pi_{l-1}$ denotes the composite relations between node types τ_1 and τ_l, $\tau_i \in T$. For example, metapath $A \xrightarrow{\text{answers}} Q \xrightarrow{\text{raises}^{-1}} R \xrightarrow{\text{raises}} Q \xrightarrow{\text{answers}^{-1}} A$ means that two answerers each solves a question raised by the same person. τ^{-1} denotes the inverse relationship of τ, e.g. raises^{-1} represents "is raised by". Apparently, the metapath preserves semantic and structural correlations of entities in HIN which will be encoded in the representations by the Skip-gram model. We will omit the relations in metapath notations (e.g., $AQRQA$) in the following discussion since the relation type between each given pair of entities is unique.

Metapath $\mathcal{P}$ guides the walk generation as follows. Randomly select an entity e_1 of type τ_1 as the initial entity, and then cycle from nodes of type τ_2 to τ_l until $w_{\mathcal{P}}$ grows to the desired length L. An example walk in Figure 3.1 is $a_5q_4r_3q_5a_7q_2r_2q_3a_4$ given $\mathcal{P} = AQRQA$ and $L = 9$. The transition from τ_t-type entity $e_t^{(i)}$ to entity $e^{(i+1)}$ is governed by the transition probability $p(e^{(i+1)}|e_t^{(i)}, \mathcal{P})$:

$$p(e^{(i+1)}|e_t^{(i)}, \mathcal{P}) = \begin{cases} \frac{1}{N} & (e^{(i+1)}, e_t^{(i)}) \in E, \phi(e^{(i+1)}) = \tau_{t+1} \\ 0 & \text{otherwise.} \end{cases},$$

where N denotes the count of τ_{t+1}-type neighbors of $e_t^{(i)}$.

3.3.2.2 Skip-gram with Negative Sampling on HINs.

Negative sampling is an approximation strategy to relieve the expensive computational cost of softmax function [MCC13]. Skip-gram with negative sampling maximizes the likelihood of D, the positive sample set generated from the metapath walk corpus by the sampling method in [MCC13], and minimizes the likelihood of the negative samples $D' = \{(n, c)|n, c \in V \wedge (n, c) \notin D\}$. The overall likelihood $\mathcal{L}(D, D'|\boldsymbol{\Theta})$ to maximize is:

$$\mathcal{L}(D, D'|\boldsymbol{\Theta}) = \sum_{D} \log(\sigma(\boldsymbol{v}_n \cdot \boldsymbol{u}_c)) + \sum_{D'} \log(-\sigma(\boldsymbol{v}_n \cdot \boldsymbol{u}_c)). \tag{3.1}$$

In Equation (3.1), $\boldsymbol{v}_n$ and $\boldsymbol{u}_c$ are representations of center entity n and context entity c. They are parts of the model parameter $\boldsymbol{\Theta}$. $\sigma(\cdot)$ is the sigmoid function. Equation (3.1) is consistent with word2vec that each entity has two versions of embeddings. We select the "center entity" version embedding as the input of the CNN recommender model.

Here we emphasize the necessity of the HIN embedding component without which the pure CNN scoring function has limited capability of analyzing user-user correlations and similarity.

3.3.2.3 LSTM for Question Representation.

Different from the question raisers and answerers whose embeddings are parts of the parameter $\mathbf{\Theta}$, the embeddings of question content, $\boldsymbol{v}_q$, are not in $\mathbf{\Theta}$ but directly obtained from question text through an LSTM model. $\boldsymbol{v}_q$ is then sent to Equation (3.1) together with the corresponding $\boldsymbol{v}_r$ and $\boldsymbol{v}_a$ for training and ranking score calculations.

LSTM is powerful in learning sequential features such as the semantics of text and has been applied to a variety of tasks such as text classification [ZSL15] and machine translation [BCB14]. We skip the mathematical details of LSTM since they have been frequently discussed in previous literature.

The generation of $\boldsymbol{v}_q$ of question q with L_q words is as follows. The input is the word embedding matrix of question $\mathbf{X} \in \mathbb{R}^{L_q \times k}$ composed of k-dimensional word vectors. The LSTM cell at the t-th time step receives the t-th word embedding vector $\boldsymbol{x}_t \in \mathbb{R}^k$ in $\mathbf{X}$ as well as the hidden state $\boldsymbol{h}_{t-1}$ from the previous time step. The output at time t is the hidden state $\boldsymbol{h}_t \in \mathbb{R}^d$ which contains the accumulated semantic information from $\boldsymbol{x}_0$ to $\boldsymbol{x}_t$. Therefore, we use the hidden state output of the last time unit, $\boldsymbol{h}_{L_q}$, as the text semantic representation $\boldsymbol{v}_q$ for the textual content of q.

It is worth mentioning that given the trained $\mathbf{\Theta}$ and the word sequence of a new question q_{new}, the representation learning of q_{new} is independent of the training data and the CQA network structure. Therefore, the LSTM component tackles the challenge of cold start issues for new questions.

3.3.3 Convolutional Recommender System

In this section, we present a convolutional neural network ranking model F that comprehensively analyzes the correlations between the three entities and computes the ranking score. Considering the properties of a ranking score, we rationally assume the following two partial order constraints based on our intuition and observation made on the dataset: (1) The best answerer has the highest score among all answerers to the query $\gamma = (r, q)$; (2) Answerers who answered q have higher scores than those who did not.

Using entity representation $\boldsymbol{v}_r$, $\boldsymbol{v}_q$, and $\boldsymbol{v}_a$, we translate the above constraints and formalize the scoring function $F(\boldsymbol{v}_r, \boldsymbol{v}_q, \boldsymbol{v}_a)$ as follows:

$$\begin{gathered}\forall a^*, a \in A_\gamma, \forall a_n \in A \text{ and } a_n \notin A_\gamma, \\ [!ht] F(\boldsymbol{v}_r, \boldsymbol{v}_q, \boldsymbol{v}_{a^*}) \geq F(\boldsymbol{v}_r, \boldsymbol{v}_q, \boldsymbol{v}_a), \\ F(\boldsymbol{v}_r, \boldsymbol{v}_q, \boldsymbol{v}_a) \geq F(\boldsymbol{v}_r, \boldsymbol{v}_q, \boldsymbol{v}_{a_n}),\end{gathered} \tag{3.2}$$

where A_γ is the set of answerers of $\gamma = (r, q)$, $a^* \in A_\gamma$ is the accepted answerer, and a_n is an answerer that is not involved in γ.

The reason of building a CNN-based scoring function is as follows: CNN has a strong capability of extracting hidden correlations of entities represented by static feature maps such as images [HZR16] and text [Kim14]. Compared with some straightforward scoring functions such as the dot-product, CNN is more powerful in preserving sophisticated correlations in the embedding matrices. Therefore, we design F as a CNN since the ranking score produced by $F(\boldsymbol{v}_r, \boldsymbol{v}_q, \boldsymbol{v}_a)$ can be considered as a hidden feature of the combination of r, q, and a.

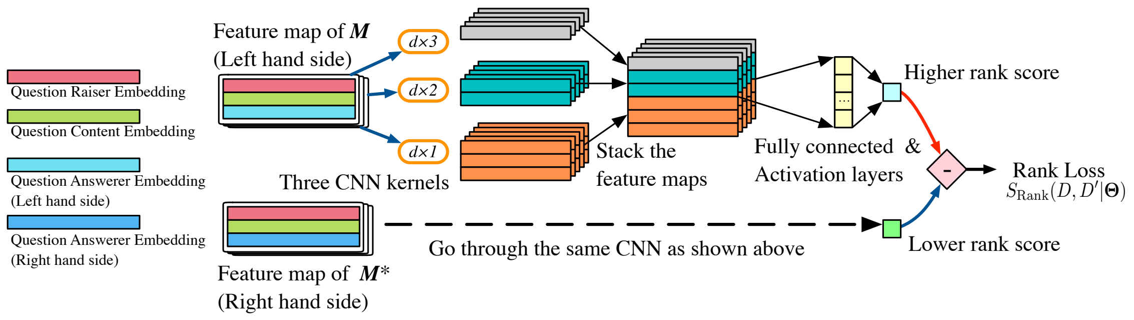

Feature map of $\boldsymbol{M}$
(Left hand side)
$d\times3$
$d\times2$
$d\times1$
Question Raiser Embedding
Question Content Embedding
Question Answerer Embedding
(Left hand side)
Question Answerer Embedding
(Right hand side)
Three CNN kernels
Stack the feature maps
Fully connected & Activation layers
Higher rank score
-
Rank Loss
$S_{\text{Rank}}(D, D'|\Theta)$
Lower rank score
Go through the same CNN as shown above
Feature map of $\boldsymbol{M}^*$
(Right hand side)

Figure 3.3: The CNN-based ranking module of NeRank to learn the ranking scores.

The computation of ranking scores is depicted in Figure 3.3. Given a query $\gamma = (r, q)$ and an answerer a to compute ranking score for, we stack their embeddings to construct the feature map $\boldsymbol{M}$ as $\boldsymbol{M} = [\boldsymbol{v}_r, \boldsymbol{v}_q, \boldsymbol{v}_a]$, $\boldsymbol{M} \in \mathbb{R}^{d\times3}$. Three convolutional kernels $k_1 \in \mathbb{R}^{d\times1}$,

$k_2 \in \mathbb{R}^{d\times 2}$, and $k_3 \in \mathbb{R}^{d\times 3}$ are applied to the input matrices. The intermediate hidden features go through two fully-connected layers and ReLU layers before deriving the ranking score. k_1 extracts the hidden features within the vector of each entity. k_2 captures the correlation between (1) $\boldsymbol{v}_r$ and $\boldsymbol{v}_q$ and (2) $\boldsymbol{v}_q$ and $\boldsymbol{v}_a$ since they have direct interactions. k_3 extracts the overall correlations across the three entities. Therefore, the aggregation of k_1, k_2, and k_3 is able to comprehensively utilize the hidden features in $\boldsymbol{M}$ and measure the score of a given γ.

The inequalities in Equation (3.2) hold for all two groups of triplets: (1) the *accepted* triplets versus the corresponding *answered but unaccepted* triplets; (2) the *answered* triplets versus the random *unanswered* triplets. Therefore, we model the ranking as the maximization of $S_{\text{Rank}}(D, D'|\boldsymbol{\Theta})$ which is defined as the summation of all differences between the two sides of the inequalities in Equation (3.3):

$$\begin{aligned}
&S_{\text{Rank}}(D, D'|\boldsymbol{\Theta}) \\
&= \sum_{(a^*,q),(a,q)\in D} \left(F(\boldsymbol{v}_r, \boldsymbol{v}_q, \boldsymbol{v}_{a^*}) - F(\boldsymbol{v}_r, \boldsymbol{v}_q, \boldsymbol{v}_a)\right) \\
&+ \sum_{(a,q)\in D,(a_n,q)\in D'} \left(F(\boldsymbol{v}_r, \boldsymbol{v}_q, \boldsymbol{v}_a) - F(\boldsymbol{v}_r, \boldsymbol{v}_q, \boldsymbol{v}_{a_n})\right),
\end{aligned} \tag{3.3}$$

We only select (a, q) pairs from D and D' since (a, q) pairs provide sufficient coverage over all question instances for training in the CQA datasets.

3.3.4 Objectives and Optimization

Cost Functions. We need to optimize the parameter $\boldsymbol{\Theta}$ that contains four parts: all embeddings of question raisers, all embeddings of question answerers, the parameters of the LSTM, and the parameters of the CNN-based scoring component. Since the optimal $\boldsymbol{\Theta}$ should maximize both Equation (3.1) and Equation (3.3), we alternatively maximize the two objective functions by gradient-based algorithms and back-propagation.

When NeRank converges, the embeddings and deep models in the optimum state have the following properties: (1) The entity embeddings contain proximity and expertise information to achieve personalized question routing; (2) The LSTM question encoder maps the content of new questions to a latent space where two additional aspects of information (expertise and proximity) are assessable in addition to text semantics. (3) The CNN recommender generates ranking scores using all three entity embeddings to measure the scores of answerers providing the accepted answer.

Complexity. Suppose that single LSTM and CNN computations have T_1 and T_2 atomic operations respectively, a training batch has b instances, and the embedding dimension is d. The forward time complexity is $O(b(T1 + T2 + bd))$ per iteration.

Avoid Overfitting. We have the following mechanisms to prevent overfitting from happening on LSTM and CNN. (1) The LSTM is simplified to single-directional and single-layer to prevent over-parameterization; (2) Early stopping is utilized so the training terminates when the losses reach plateaux, which is shown in Figure 3.5; (3) The two objective terms are alternatively optimized towards different directions. They function as each other's regularizer that avoids overfitting.

3.4 Experiments

In this section, we introduce the experiment settings, show experimental results, and demonstrate effectiveness and efficiency of NeRank.

3.4.1 Experimental Setup

Two datasets of two real-world CQA websites with specific topics are employed to evaluate NeRank: Biologyand English. Each dataset[2] contains all questions raised before December, 2017 and all users' historical asking and answering records. The datasets differ in sizes (see

[2]Available at: `https://archive.org/details/stackexchange`

Table 3.2) and are mutually exclusive in topics so that NeRank can be comprehensively tested. Other CQA datasets, such as Yahoo! Answers, are not selected for evaluation since they do not provide accepted answers that are needed to serve as the ground truth.

Table 3.2: Statistics of the datasets for the evaluation of NeRank.

Dataset	# of users	# of r	# of q	# of a
Biology	5,071	3,696	2,224	21,613
English	35,713	19,743	22,753	209,543

CQA networks are built from 90% of questions and the corresponding users to generate training walks. The rest 10% of questions and the corresponding raisers and answerers for testing. Users in the test set should have at least 5 asking or answering records to avoid cold starts. In each test query $\gamma = (r, q)$, we create a candidate answerer set of 20 answerers that includes all answerers of q in the dataset and some other users randomly selected from the top 10% most responsive users. We choose the answerer with the highest predicted ranking score as the recommendation. The owners of the accepted answers are the ground truth.

The walks are generated from metapath $AQRQA$ with the default length of 13 (three cycles) and the default node coverage of 20 (each node is covered at least 20 times in walk generation). The window size of Skip-gram model is set as 4; we use 3 negative samples per positive sample; and the dimension of learned embeddings is set as 256. We use a 64-channel CNN for ranking and the 300-dimensional *GoogleNews* pretrained word2vec model[3] to build the embedding matrix $\boldsymbol{x}$ for questions. NeRank is prototyped by Python 3.6.4 and PyTorch 0.4.0[4]. All experiments are conducted on a single 16GB-memory Tesla V100 GPU in an 512GB memory Nvidia DGX-1.

[3]Available at: `https://code.google.com/archive/p/word2vec/`

[4]Available at: `https://github.com/zyli93/NeRank`

3.4.2 Experimental Results

This section reports the experimental results and analyses of the effectiveness and efficiency of NeRank. Note that three metrics, including Mean Reciprocal Rank (*MRR*), Hit at K (*Hit@K*), and Precision at 1 (*Prec@1*), are applied to evaluate the ranking performance.

3.4.2.1 Effectiveness of NeRank

We compare NeRank with three baseline models shown as below.

Score A trivial method that recommends the answerer that has the largest number of accepted answer.

NMF Non-negative Matrix Factorization [GNH11] uses matrix decomposition to solve the ranking problem.

L2R SVM-based and RankingSVM-based learning to rank algorithms [JW13] that extract features from user-question relations to predict the ranking.

The performances of NeRank and the baselines are shown in Table 3.3. NeRank significantly outperforms all baseline algorithms on both datasets in terms of all metrics. On the Biology dataset, NeRank achieves a Prec@1 of 0.387 and a Hit@K of 0.806, meaning that around 38.7% of the predictions are correct and the ground truth can be found in the top-5 ranked answerers in around 80.6% of the predictions. On the English dataset, NeRank achieves similar performances that the Prec@1 is 0.372 and Hit@k is 0.833. MRR in both dataset are around 0.56 indicating a huge improvement over the baselines in terms of overall ranking performance. All improvements of NeRank over the best baseline, NMF, are significant at 99% confidence in a *paired t-test*.

Some entries in Table 3 show that Biology has better results than English. Although, generally speaking, larger training sets may lead to better performance, the properties of the datasets may also play a role. In Biology, there exist a small group of proficient users with particular expertise. However, the English community has a larger proportion of skilled users since language is a common knowledge. Therefore, the performance may show small variance across datasets.

In summary, NeRank has a strong ability in discovering experts that provide the accepted answer. The advantages of NeRank lie in the following facts: (1) NeRank considers question raiser information in addition to question content and question answerer, which can better conduct question routing. (2) NeRank utilizes deep neural network models that preserve the complex information of text semantic features and entity correlation features.

Table 3.3: Performance comparisons between NeRank and three baseline models.

Dataset	Biology			English		
Metric	MRR	Hit@K	Prec@1	MRR	Hit@K	Prec@1
Score	0.27	0.412	0.105	0.203	0.379	0.065
NMF	0.375	0.643	0.177	0.458	0.737	0.225
L2R	0.169	0.158	0.050	0.101	0.058	0.024
NeRank	0.563	0.806	0.387	0.567	0.833	0.372

3.4.2.2 Effectiveness of Metapath-based Embeddings

We compare NeRank with two of its variants that employs, instead of metapath-based HIN embedding model, Deepwalk [PAS14] (denoted by "NeRank-DW") and LINE [TQW15] (denoted by "NeRank-LINE") for embedding learning. Other configurations remain unchanged. We show that metapath benefits represention learning on HINs and improve the performance of NeRank.

Figure 3.4 shows the experimental results. We observe that NeRank achieves better results than NeRank-DW and NeRank-LINE on both datasets on all metrics. The reason

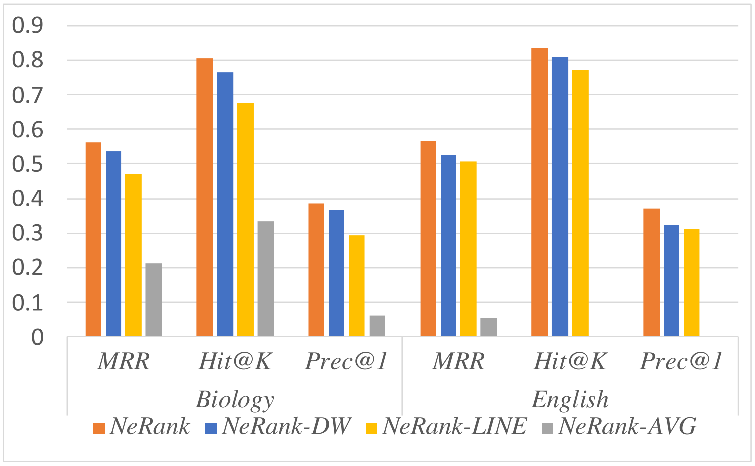


Figure 3.4: Performance comparison between NeRank and three variants.

is that Deepwalk and LINE are designed for homogeneous networks whereas CQA networks are heterogeneous that contain rich semantic information in diverse node and edge types. Metapath-based models take advantage of the semantic information and thus helps enhance the performance.

Although not particularly designed for HIN, Deepwalk and LINE are also capable of discovering the proximity relations between entities since both the metapath-based model and homogeneous network embedding models assume that connected entities have similarity. This accounts for the insignificance of the performance drop.

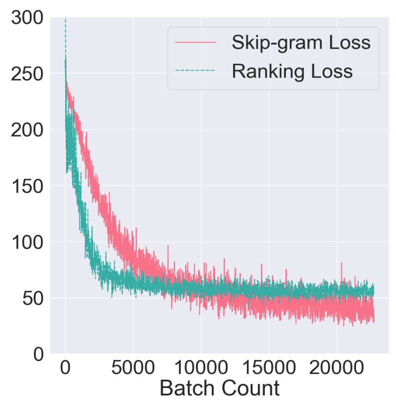
3.4.2.3 Effectiveness of Scoring Function

We compare NeRank with another variant that replaces the CNN scoring function by $\boldsymbol{v}_a \cdot \frac{\boldsymbol{v}_r + \boldsymbol{v}_q}{2}$, another combination of query `(question raiser, question content)`. The dot product of $\boldsymbol{v}_a$ and the numeric average of $\boldsymbol{v}_r$ and $\boldsymbol{v}_q$ is considered as the ranking score. Other settings are the same. We illustrate that our scoring function can effectively extract the latent expertise information and accurately generate ranking scores. The results are demonstrated in Figure 3.4 in which we denote the variant as "NeRank-AVG".

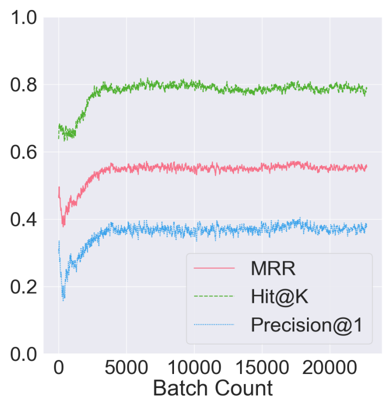
We observe that NeRank significantly outperforms NeRank-AVG by at least two folds. The performance difference is maximized in Prec@1 where NeRank-AVG can only correctly predict for 6.05% of the queries on Biology dataset and 0.04% on English dataset.

The huge performance gap indicates the strong ability of the CNN scoring function to capture the expertise information from the correlations of entity representations and make accurate predictions.

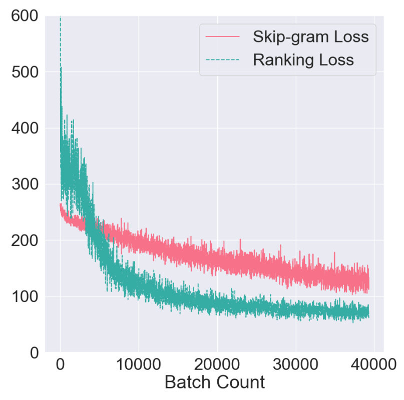
3.4.2.4 Convergence rate of NeRank



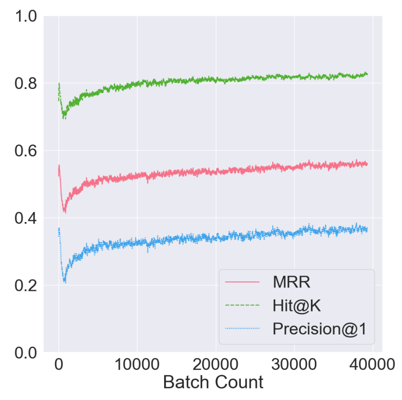
(a) Losses on Biology



(b) Metrics on Biology



(c) Losses on English



(d) Metrics on English

Figure 3.5: Convergence rate study of NeRank.

We plot the trends of losses, i.e., negative objectives (Skip-gram objective in Equation (3.1) and Ranking objective in Equation (3.3)), and metrics as a function of the training iterations in Figure 3.5. These trends give us insight to the convergence rate of NeRank. These experiments are run in the default configurations.

It is observed that the NeRank converges at around 5,000 iterations (batch count) on the Biology dataset and at around 10,000 iterations on the English dataset. The convergences of the metrics happen before 5,000 iterations on Biology and before 10,000 iterations on English. Such converge rate is fast given the complex CNN and LSTM hybrid architecture of NeRank and a single GPU core, which demonstrates the model's learning efficiency and scalability. Convergences happen earlier on smaller dataset (Biology) and later on larger dataset (English). The reason is that, with the same batch size, a larger proportion of entities in smaller networks participates in training due to negative sampling.

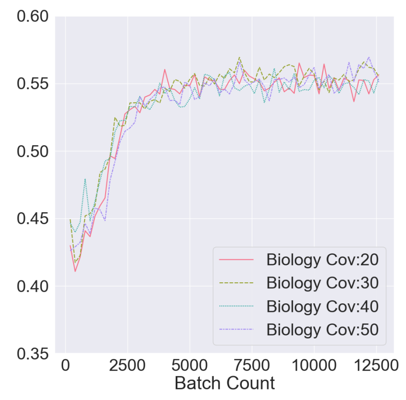
3.4.2.5 Parameter Sensitivity

We also evaluate the sensitivity of NeRank to node coverage and walk length. Node coverage refers to the number of times a certain node is covered by the training walks. We report the trends of MRR in Figure 3.6.

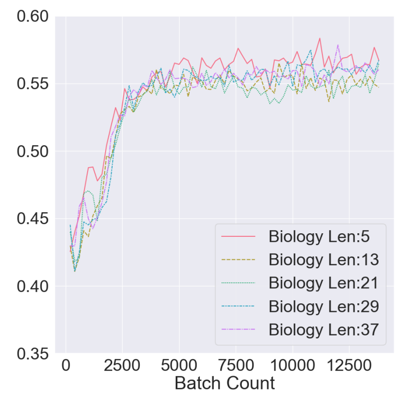
It is observed that the curves almost coincide in the four subfigures, meaning that NeRank converges to very similar states at a similar speed although given different node coverages and walk lengths. Therefore, NeRank is robust to the changes of these hyperparameters.

3.5 Summary

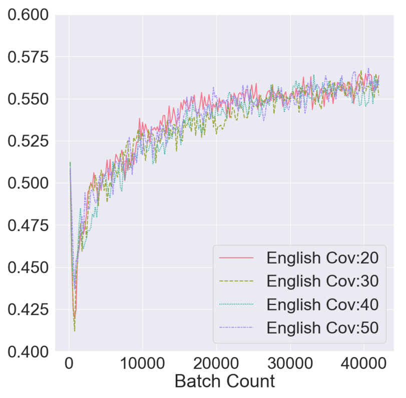
In this chapter, we propose NeRank, a framework for personalized question routing based on the question content and question raisers. NeRank learns representations of entities by heterogeneous network embedding and LSTM. Using the embeddings, the convolutional scoring model computes the ranking scores to predict the answerer that most probably



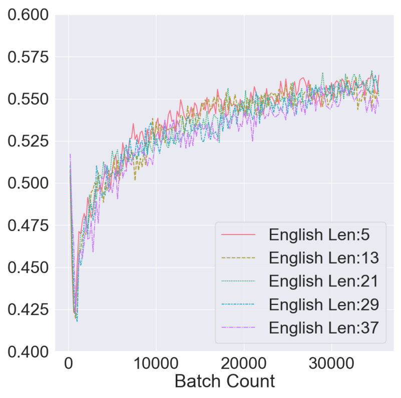
(a) MRR, Cov (Biology)



(b) MRR, Len (Biology)



(c) MRR, Cov (English)



(d) MRR, Len (English)

Figure 3.6: Parameter sensitivity study of NeRank w.r.t. MRR.

contribute the accepted answer. We test NeRank on two real-world CQA datasets. NeRank achieves a high performance and outperforms the state-of-the-art models.

CHAPTER 4

Click-Through Rate Prediction via Hierarchical Attention

Click-through rate (CTR) prediction is a critical task in online advertising and marketing. For this problem, existing approaches, with shallow or deep architectures, have three major drawbacks. First, they typically lack persuasive rationales to explain the outcomes of the models. Unexplainable predictions and recommendations may be difficult to validate and thus unreliable and untrustworthy. In many applications, inappropriate suggestions may even bring severe consequences. Second, existing approaches have poor efficiency in analyzing high-order feature interactions. Third, the polysemy of feature interactions in different semantic subspaces is largely ignored.

In this chapter, we introduce InterHAt that employs a Transformer with multi-head self-attention for feature learning. On top of that, hierarchical attention layers are utilized for predicting CTR while simultaneously providing interpretable insights of the prediction results. InterHAt captures high-order feature interactions by an efficient attentional aggregation strategy with low computational complexity. Extensive experiments on four public real datasets and one synthetic dataset demonstrate the effectiveness and efficiency of InterHAt.

4.1 Motivation and Background

Click-through rate (CTR) is defined as the probability of a user clicking through a particular recommended item or an advertisement on a web page. It plays a significant role in recommender systems, such as online advertising, since it directly affects the revenue of advertising agencies [RDR07, GZL17, WZX18, ZZS18, RDR07, HC17, ZMF18, JZC16, HPJ14]. Consequently, CTR prediction, which attempts to accurately estimate the CTR given information describing a user-item scenario, is critical for achieving precise recommendations and increasing good revenue for enterprises.

The development of deep learning provides a new machine learning paradigm that utilizes deeper neural network structure to capture more complex information from the training data. Therefore, the architectural and computational complexity of existing CTR prediction models has been ever increasing in order to learn the joint effect of multiple features, i.e., high-order features (a.k.a. cross features), and attain better prediction accuracy. Specifically, a *k-th order feature* ($k \in \mathbb{N}$) refers to a latent variable that is a k-th degree polynomial of the raw features [CKH16, WFF17]. Deep neural networks provide strong capability to capture rich high-order information due to the large number of layers and units. For example, DeepFM [GTY17] and xDeepFM [LZZ18] learn high-order features by multi-layer feed-forward neural networks (FNN) and multi-block compressed interaction networks (CIN).

However, the ever-growing model complexity has two drawbacks: *impaired interpretability* and *poor efficiency*. For interpretability, the prediction-making processes are hard to be reasonably explained since the weights and activations of the neural network layers are usually deemed unexplainable. For example, the wide component of Wide&Deep [CKH16] applies cross-product transformations to feature embeddings but fails to quantify and justify its effectiveness to the actual click-through rate prediction performance. The lack of persuasive rationales for the predictions of the models casts shadow on their reliability and security. In many applications, e.g., medication recommendation [LXM13] and financial services [Zib16], untrustworthy and unreliable advertisements can mislead users to click through the statistically popular but actually useless or even harmful links which can result in serious consequences such as economic or health losses.

The second defect of existing approaches is the poor efficiency since the high-order interaction feature generation by deep neural networks involves extremely heavy matrix computations in deep neural networks (DNN). For example, the compressed interaction network (CIN) in xDeepFM [LZZ18] computes the $(k+1)$-th order feature matrix by an outer product layer and a fully-connected layer which entails a cubic complexity to the embedding dimension. The deep component in Wide&Deep has a number of fully-connected layers each of which involves a quadratic number of multiplications.

In real applications, the efficiency issue is prevalent and critical. Advertising agencies prefer prompt click recommendation provision to slow or costly ones especially under the pressure of massive real-time recommendation queries. For example, Criteo, which is an Internet advertisement company, handles over 4 billion click-throughs in 24 days[1]. Despite the large data volume, new features, such as new users and items, are emerging rapidly, to which the recommender systems must quickly adapt for better user experience. Therefore, learning the representations of an enormous number of existing or emerging features can be computationally intractable with existing approaches.

In addition to the interpretability and efficiency issues, we point out another impediment that can degrade the performance of detecting important cross-feature interactions: different cross-features may have conflicting influences on CTR that have to be comprehensively analyzed. For example, a movie recommendation record `movie.genre` = `horror`, `user.age` = `young`, `time` = `8am` has conflicting factors: the combination of the first two encourages the click-through whereas the combination of the latter two inhibits it since movie watching usually happens at night. Such conflict problem is caused by the *polysemy* of feature

[1]`https://ailab.criteo.com/criteo-releases-new-dataset/`

interactions in different semantic subspaces. In this example, the polysemic interactions of `user.age` cause opposite impacts on CTR when `user.age=young` is combined with two different attributes, `movie.genre` and `time`. However, this problem is largely ignored by the existing methods.

To address the above issues, in this chapter, we propose an **Inter**pretable CTR prediction model with **H**ierarchical **At**tention (InterHAt) that efficiently learns salient features of different orders as interpretative insights and accurately predicts CTR simultaneously in an end-to-end fashion. Specifically, InterHAt explicitly quantifies the impacts of feature interactions of arbitrary orders by a novel hierarchical attention mechanism, aggregates the important feature interactions for efficiency purposes, and explains the recommendation decision according to the learned feature salience. Different from the hierarchical attention network by [YYD16] that studies the linguistic hierarchy (word and sentence), InterHAt uses the hierarchical attention on feature orders, and the high-order features are generated based on the lower ones.

To accommodate the polysemy of feature interactions in different semantic subspaces, InterHAt leverages a Transformer [VSP17] with multi-head self-attention to comprehensively study different possible feature-wise interactions. Transformer has been popularly employed in natural language processing tasks such as sentiment analysis, natural language inference [DCL18], and machine translation [TC18]. The multiple attention heads can capture the manifold mutual effects of words that jointly compose the semantics of text from different latent subspaces. We utilize this great property of Transformer to detect the complex polysemy of feature interactions and learn a polysemy-augmented feature list which serves as the input of hierarchical attention layers. Note that despite the strong capability of Transformer in feature learning, the model efficiency is retained according to [VSP17].

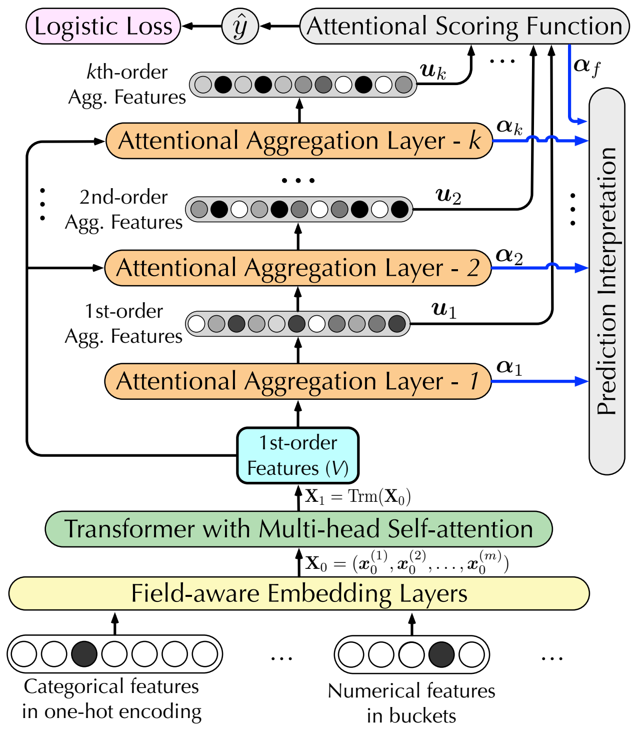
We summarize the contributions of this chapter as follows.

- We propose InterHAt for CTR prediction. Particularly, InterHAt employs hierarchical attention to pinpoint the significant single features or different orders of interactive features that have great contributions to the click-through. Then, InterHAt can compose a corresponding attention-based explanation for the CTR prediction based upon the various orders of feature interactions.

- InterHAt utilizes a Transformer with multi-head self-attention to thoroughly analyze possible interactive relations between features in different latent semantic subspaces. To our knowledge, InterHAt is the first approach that employs the Transformer with multi-head self-attention to learn the polysemy of latent features for CTR prediction.
- InterHAt predicts CTR without using deep multilayer perceptron networks that entail heavy computational cost. It aggregates the features instead and hence saves the expense of enumerating the exponential size of feature interactions. As a result, it is more efficient in handling high-order features than existing algorithms.
- Extensive experiments are conducted to evaluate InterHAt for interpretability, efficiency, and effectiveness on three major CTR benchmark datasets (Criteo, Avazu, and Frappe), one popular recommender system dataset (MovieLens-1M), and one synthetic dataset. Results show that InterHAt explains the decision-making process, achieves a huge improvement on training time, and still has comparable performance with the state-of-the-art models.

4.2 InterHAt

In this section, we elaborate the pipeline of InterHAt depicted in Figure 4.1 and CTR prediction interpretation method according to the attentional weights. The inputs are categorical and numerical features at the bottom and the outputs are a prediction $\hat{y}$ and a cross entropy loss. The black arrows explain the data flow for training and prediction, the blue arrows illustrate the collection of attentions for interpretation.



Logistic Loss
$\hat{y}$
Attentional Scoring Function
$\boldsymbol{u}_k$
$\boldsymbol{\alpha}_f$
kth-order Agg. Features
Attentional Aggregation Layer - k
$\boldsymbol{\alpha}_k$
2nd-order Agg. Features
$\boldsymbol{u}_2$
Attentional Aggregation Layer - 2
$\boldsymbol{\alpha}_2$
1st-order Agg. Features
$\boldsymbol{u}_1$
Attentional Aggregation Layer - 1
$\boldsymbol{\alpha}_1$
Prediction Interpretation
1st-order Features (V)
$\mathbf{X}_1 = \mathrm{Trm}(\mathbf{X}_0)$
Transformer with Multi-head Self-attention
$\mathbf{X}_0 = (\boldsymbol{x}_0^{(1)}, \boldsymbol{x}_0^{(2)}, \ldots, \boldsymbol{x}_0^{(m)})$
Field-aware Embedding Layers
Categorical features in one-hot encoding
Numerical features in buckets

Figure 4.1: The architecture of InterHAt with prediction interpretation generation.

4.2.1 Embedding Layer

Feature embedding is a prerequisite for CTR prediction since the click-through records contain discrete categorical terms that are not directly applicable to numerical computations [GTY17, QCR16, SHJ16, WFF17].

A click-through record contains a set of *fields* F and a binary label y as the ground truth representing whether a click-through is actually made. Each field $f \in F$ has either a categorical or a numerical value. Distinct values are defined as different *features*. For categorical fields, we apply *multi-field* one-hot encoding to field-aware embedding layers for low-dimensional real-valued feature representations. Specifically, each distinct feature value v of a field is assigned a trainable d-dimensional continuous vector as its representation. If a particular feature appears in a click-through record, the corresponding embedding of that feature is considered as the field representation. For numerical fields, we assign one vector to each field as its embedding. Given v_f as the normalized value of a numerical field f and $\boldsymbol{x}^{(f)}_{\text{num},0} \in \mathbb{R}^d$ as the trainable representation associated with this field, the representation of the feature, $\boldsymbol{x}^{(f)}_{\text{num}} \in \mathbb{R}^d$, is derived by $\boldsymbol{x}^{(f)}_{\text{num}} = v_f \cdot \boldsymbol{x}^{(f)}_{\text{num},0}$. The initial input representation matrix $\mathbf{X}_0 \in \mathbb{R}^{d \times m}$ is then $\mathbf{X}_0 = \left(\boldsymbol{x}^{(1)}_0, \boldsymbol{x}^{(2)}_0, \ldots, \boldsymbol{x}^{(m)}_0\right)$ where $m = |F|$.

4.2.2 Multi-head Transformer

Transformer is prevalent in natural language processing thanks to the outstanding power to learn the co-effects to the text semantics of word pairs within a sentence or across sentences regardless of the orders and distances of the words. In the context of CTR prediction, we define the co-effects of the features, i.e., feature interactions, towards different polarity as the "polysemy". Therefore, we equip InterHAt with a multi-head self-attention based Transformer to capture the rich pair-wise feature interactions and learn the diversified polysemy of feature interactions in different semantic subspaces, i.e., diversified implications towards the CTR in different click-through contexts.

Given the input matrix $\mathbf{X}_0$ that contains the learnable embeddings of features of a training CTR record, the latent representation $\mathbf{H}_i$ of Transformer head i is obtained by a scaled dot-product attention [VSP17],

$$\mathbf{H}_i = \text{softmax}_i\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_K}}\right)\mathbf{V},$$
$$\mathbf{Q} = \mathbf{W}^{(Q)}_i \mathbf{X}_0, \quad \mathbf{K} = \mathbf{W}^{(K)}_i \mathbf{X}_0, \quad \mathbf{V} = \mathbf{W}^{(V)}_i \mathbf{X}_0.$$

Matrices $\mathbf{W}^{(Q)}_i \in \mathbb{R}^{d_K \times d}$, $\mathbf{W}^{(K)}_i \in \mathbb{R}^{d_K \times d}$, and $\mathbf{W}^{(V)}_i \in \mathbb{R}^{d_K \times d}$ are weight parameters to learn for head i and d_K denotes the dimension of $\mathbf{K}$ and $\mathbf{H}_i \in \mathbb{R}^{d_K \times m}$.

A combination of hidden features $\mathbf{H}_i$ forms an augmented representation matrix $\mathbf{X}_1$ that preserves both the intrinsic and polysemic information of each feature. Computationally, we use concatenation followed by a feed-forward layer and a ReLU for the combination to learn the non-linearity of the combined information as

$$\mathbf{X}_1 = \text{ReLU}(\text{FeedForward}(\mathbf{W}_m[\mathbf{H}_1; \mathbf{H}_2; \ldots; \mathbf{H}_h])),$$

where $\mathbf{W}_m \in \mathbb{R}^{d \times hd_k}$ contains the weights and h is the number of attention heads and ";" denotes the concatenation of matrices. The $\mathbf{X}_1 \in \mathbb{R}^{d \times m}$ is the matrix with polysemy-augmented features and ready to be sent to the hierarchical attention layer for explainable CTR prediction.

4.2.3 Hierarchical Attention

The augmented feature matrix $\mathbf{X}_1$ is served as the input of the hierarchical attention layers which learn the feature interaction and generate interpretations simultaneously. However, computing the high-order multi-feature interactions by enumerating all possible combinations is expensive due to the combinatorial explosion. Such potential expense motivates the aggregation of the current order before proceeding to the computation of the higher order. That is, in order to generate the $(i+1)$-th order cross-features $\mathbf{X}_{i+1}$, we first aggregate the i-th layer hidden features to $\boldsymbol{u}_i$ as a summarization of $\mathbf{X}_i$. The interaction between $\mathbf{X}_i$ and $\mathbf{X}_1$, from which we derive $\mathbf{X}_{i+1}$, is computed by the proxy of $\mathbf{X}_i$, i.e., the attentional aggregation $\boldsymbol{u}_i$ from Equation (4.1), and $\mathbf{X}_1$. Mathematically, given the i-th feature matrix $\mathbf{X}_i = \left(\boldsymbol{x}_i^{(1)}, \ldots, \boldsymbol{x}_i^{(m)}\right)$, its attentional aggregation representation $\boldsymbol{u}_i$ is

$$\boldsymbol{u}_i = \text{AttentionalAgg}(\mathbf{X}_i) = \sum_{j=1}^{m} \alpha_i^{(j)} \boldsymbol{x}_i^{(j)}, \tag{4.1}$$

where $\alpha_i^{(j)} \in \mathbb{R}$ denotes the attention on the j-th field in the i-th attentional aggregation layer. $\alpha_i^{(j)}$ is computed by

$$\alpha_i^{(j)} = \frac{\exp\left(\boldsymbol{c}_i^T \text{ReLU}(\mathbf{W}_i \boldsymbol{x}_i^{(j)}\right)}{\sum_{j' \in F} \exp\left(\boldsymbol{c}_i^T \text{ReLU}(\mathbf{W}_i \boldsymbol{x}_i^{(j')})\right)}, \tag{4.2}$$

where $\mathbf{W}_i \in \mathbb{R}^{s \times d}$ is the weight of layer i, $\boldsymbol{c}_i \in \mathbb{R}^s$ is the context vector of layer i, and s denotes the attention space size. Note that other attention mechanisms can also be adopted here, such as the gated attention mechanism [ITW18]. Using $\boldsymbol{u}_i$ and $\mathbf{X}_i$, we derive $\boldsymbol{x}_{i+1}^{(j)}$ in $\mathbf{X}_{i+1}$ by a cross-product transformation [CKH16, HZR16]

$$\boldsymbol{x}_{i+1}^{(j)} = \boldsymbol{u}_i \circ \boldsymbol{x}_1^{(j)} + \boldsymbol{x}_i^{(j)}, \quad j \in \{1, \ldots, m\}, \tag{4.3}$$

where $\circ$ denotes the Hadamard product of two vectors.

Recurrently applying Equation (4.1) and Equation (4.3) produces $\boldsymbol{u}_i$ and $\mathbf{X}_i$ for feature orders from the 1st order to the k-th, the highest cross-feature order to analyze, by a series of attentional aggregation layers. These layers composite a hierarchy that extracts features from low order to higher ones and the lower ones contribute to the construction of one-order higher features using the proposed attentional aggregation and cross-product transformation.

As the last step, we combine attentional aggregations $\mathbf{U} = (\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_k)$ to predict the probability of click-through. $\mathbf{U}$ gathers all combinatorial feature semantics of k orders. By modifying k, InterHAt is able to capture *arbitrary* order of feature interactions, and yet avoids the exponential cardinality of high-order feature combinations.

4.2.4 Objective Function and Optimization

The final CTR prediction function $g(\mathbf{U}) = \hat{y} \in [0, 1]$ maps $\mathbf{U}$ to a probability that quantifies the CTR. $g(\mathbf{U})$ is implemented as the following. It first computes the attentional aggregation of $\mathbf{U}$ by Equation (4.4) and Equation (4.5) to obtain its aggregation $\boldsymbol{u}_f \in \mathbb{R}^d$ and attention $\boldsymbol{\alpha}_f \in \mathbb{R}^k$,

$$\boldsymbol{u}_f = \text{AttentionalAgg}(\mathbf{U}) = \sum_{j=1}^{k} \alpha_f^{(j)} \boldsymbol{u}_j, \tag{4.4}$$

$$\alpha_f^{(j)} = \frac{\exp\left(\boldsymbol{c}_f^T \text{ReLU}(\mathbf{W}_f \boldsymbol{u}_j\right)}{\sum_{j' \in \{1,\ldots,k\}} \exp\left(\boldsymbol{c}_f^T \text{ReLU}(\mathbf{W}_f \boldsymbol{u}_{j'})\right)}, \tag{4.5}$$

where $\boldsymbol{\alpha}_f$ is the importance distribution across k feature orders, $\boldsymbol{c}_f$ and $\mathbf{W}_f$ are learnable parameters. Finally, the prediction $\hat{y}$ is then made by

$$\hat{y} = \text{sigmoid}(\text{MLP}(\boldsymbol{u}_f))$$

where $\text{MLP}(\cdot)$ refers to a *shallow* multilayer perceptron that reduces the output dimension from d to 1. The objective function, Equation (4.6), of InterHAt is a cross entropy loss of binary classification.

$$\mathcal{L}(\Theta) = \sum_{t \in \mathcal{D}} [-y_t \log(\hat{y}_t) - (1 - y_t) \log(1 - \hat{y}_t)] + \lambda ||\Theta||_2. \tag{4.6}$$

$\mathcal{D}$ denotes the training set and Θ includes all trainable parameters, namely feature embeddings and the parameters of Transformer and hierarchical layers. An L_2 regularization weighted by λ is applied to Θ to prevent overfitting. We optimize Equation (4.6) by Adam gradient descent optimizer [KB14].

4.2.5 Interpretation

This section elaborates how to "understand" the attentions in the hierarchy as important factors that trigger the prediction of CTR. Note that the attention mechanism only highlights the salience of features so it is not expected to generate completely human readable interpretations. This assumption is consistent with other attention-based interpretable models [GBY18].

Here is a walk-through of the interpretation using the salience distribution $(\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \ldots, \boldsymbol{\alpha}_k)$ and $\boldsymbol{\alpha}_f$. $\boldsymbol{\alpha}_f$ contains the significance of all k orders of features and signifies the feature orders that are influential to the ultimate CTR prediction. Dominant weights in $\boldsymbol{\alpha}_f \in \mathbb{R}^k$ pinpoint the $\mathbf{X}_i$'s that contain significant i-th order features. According to $\boldsymbol{\alpha}_f$, we learn the numbers of orders, i.e., the numbers of interacting features, that have the strongest impact to encourage the user to click through the recommended ads.

The attention weights in corresponding $\boldsymbol{\alpha}_i$ identify the candidate individual features that participate in the contributory i-th order features. For example, if the attention weights of features of fields f_1 and f_2, i.e., $\boldsymbol{\alpha}_i[f_1]$ and $\boldsymbol{\alpha}_i[f_2]$, outweigh the rest of the features in $\boldsymbol{\alpha}_i$, we learn that features of field f_1 and f_2 both contribute to an i-th order feature since they actively interact with the $i-1$ order aggregation features. Finally, following the above steps, we can identify all features in different orders. The actual click-through is interpreted by identifying salient features layer by layer and order by order.

4.3 Experiments

4.3.1 Experimental Setup

Datasets We evaluate InterHAt on three publicly available datasets, namely *Criteo*[2], *Avazu*[3], and *Frappe* [BCK15]. Criteo and Avazu contain chronologically ordered click-through records from Criteo and Avazu which are two online advertisement companies. We use their top 30% records for evaluation. Frappe dataset contains context-aware app usage log. Table 4.1 shows the statistics of the datasets. The ratio of train, test, and validation set sizes is 8:1:1.

Table 4.1: Statistics of Criteo, Avazu, and Frappe datasets for the evaluation of InterHAt.

Dataset	Criteo	Avazu	Frappe
#. of features (C + N)	22 + 14	21 + 0	7 + 0
#. of total records	13.8M	12.1M	288K
#. of distinct features	605.7K	23.8K	5,382

Baseline models and metrics The performance of InterHAt is compared with the following state-of-the-art approaches specifically designed for CTR tasks:

FM [Ren10] Factorization Machine that uses linear combination of first-order and second-order (dot-product of feature vectors) to compute CTR.

Wide&Deep [CKH16] An ensemble method of general linear model and an unexplainable deep MLP.

[2]`https://www.kaggle.com/c/criteo-display-ad-challenge`

[3]`https://www.kaggle.com/c/avazu-ctr-prediction`

DCN [SHJ16] An ensemble method of a cross-product transformation for high-order features and a deep MLP.

PNN [QCR16] A production based feature engineering algorithm that uses an architecture composed by simple inner product, outer product, and non-linear activation functions for CTR prediction.

DeepFM [GTY17] A combination of a deep MLP and a factorization machine to compute CTR.

xDeepFM [LZZ18] A combination of a deep MLP and a novel compress information network module that more thoroughly studies the subtle implicit features for CTR.

We argue that the baseline models considered are strong enough to present the state-of-the-art performance on CTR prediction, especially on Criteo and Avazu which are dedicated for CTR prediction evaluation and have been utilized in the most of the above works.

We focus on metrics *Logloss*, i.e., the cross entropy loss, and AUC which is the shorthand of Area Under the ROC Curve. These two metrics are widely adopted by CTR prediction evaluations. A smaller Logloss or a larger AUC represents better performance. We present the experimental results of InterHAt on its efficiency, effectiveness, and interpretability.

Default hyperparameters The default settings of each dataset are listed in Table 4.2 for reproducibility purposes. The settings vary across the three datasets due to different dataset sizes. The prototype of InterHAt is implemented by Python 3.7 + TensorFlow 1.12.0 and run with a 16GB Nvidia Tesla V100 GPU.

4.3.2 Efficiency and Effectiveness

We illustrate the comparison of InterHAt with baseline models and its variant to show its efficiency and effectiveness.

Table 4.2: Default hyperparameter settings of InterHAt for datasets.

Dataset	Criteo	Avazu	Frappe
Embedding size (d)	12	8	12
Attention size (s)	30	20	16
#. of heads	12	8	4
Regularization weight (λ)	2e-4	2e-4	2e-3

4.3.2.1 Efficiency

Figure 4.2 demonstrates a comparison on the runtime between InterHAt and five state-of-the-art models with GPU implementations on Criteo and Avazu. Frappe is not used for the efficiency test since its size is relatively small and the computational overhead accounts for most of the runtime. FM is also not used since only CPU-based implementation is available. The y-axis shows an average runtime per epoch over five training epochs after which all models start to converge observably. The hardware settings are identical to what mentioned in the experiment setting session. From the figure, we observe that InterHAt displays an outstanding efficiency by spending the minimum time for each epoch among the six models.

Two properties of InterHAt enable the huge speedup: (1) The attentional aggregation operations across the features reduce the problem scale from exponential to linear by avoiding the enumeration of all possible feature combinations in the k orders; (2) Only shallow MLP layers are involved in InterHAt in contrast with the deep MLP used in the baseline models. Deep neural network can drastically slow down the computation due to the humongous parameter sizes.

4.3.2.2 Effectiveness

In CTR prediction task, a 10^{-3} magnitude of performance gain on AUC or Logloss is considered as a **huge improvement**. We observe from Table 4.3 that InterHAt outperforms

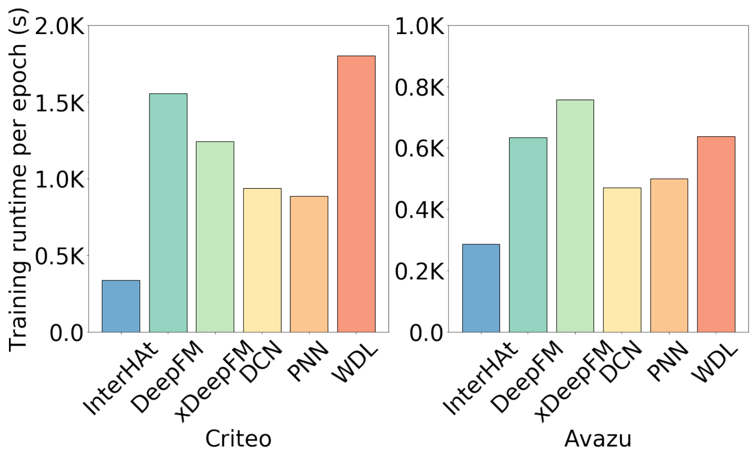


Figure 4.2: Efficiency study of InterHAt and the baselines on average runtime per epoch.

all models on Frappe and Avazu on both metrics, and attains comparable performance on Criteo. Therefore, the effectiveness of InterHAt is substantiated despite the fact that InterHAt is structurally simpler compared with other models. InterHAt-S refers to the variant of InterHAt that has the multi-head self-attention module removed as an ablation study. The decreased performance of InterHAt-S proves the contribution of the multi-heads based Transformer.

The reason that InterHAt virtually ties other models on Criteo is that the features of Criteo are more complicated in semantics as opposed to Avazu and Frappe. Competing models use non-explainable deep fully-connected (FC) layers to capture the complex implicit information and improve the performance. However, InterHAt is free of deep FC layers that damage the model interpretability. In addition, the current field-aware embedding strategy, in which numerical fields only have a single embedding $\boldsymbol{x}^{(f)}_{\text{num},0}$, undermines the ability of InterHAt to parameterize numerical-numerical and categorical-numerical feature interactions. We leave the exploration towards proper feature representation and parameterization scheme for future work.

Table 4.3: Performance comparisons of InterHAt and baseline models on Logloss and AUC

Dataset	Criteo		Avazu		Frappe	
Metrics	Logloss	AUC	Logloss	AUC	Logloss	AUC
FM	0.4814	0.7525	0.3951	0.7508	0.4480	0.8625
Wide&Deep	0.4577	0.7845	0.3920	0.7564	0.2571	0.9500
DCN	0.4590	0.7826	0.3921	0.7564	0.2335	0.9616
PNN	**0.4547**	**0.7887**	0.3916	0.7569	0.2177	0.9642
DeepFM	0.4560	0.7866	0.3920	0.7561	0.2410	0.9520
xDeepFM	0.4563	0.7874	0.3917	0.7569	0.2043	0.9694
InterHAt-S	0.4608	0.7820	0.3919	0.7577	0.2151	0.9616
InterHAt	0.4577	0.7845	**0.3910**	**0.7582**	**0.2026**	**0.9696**

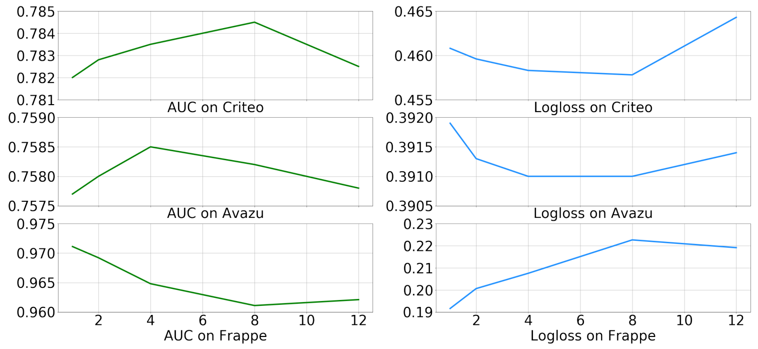


Figure 4.3: Parameter sensitivity study on number of heads in Transformer.

4.3.2.3 Sensitivity on Transformer heads

This section illustrates the hyperparameter sensitivity study on Transformer head numbers as an ablation study. The Logloss and AUC of InterHAt with different numbers of heads are given in Figure 4.3. We change the number of heads from 1 to 12, keep other settings fixed, and train the model until convergence. For Criteo and Avazu, the optimal options of the number of heads are 8 and 4, respectively. For Frappe, the optimal head number falls on 1, which is consistent with our observation that the semantics of Frappe fields is

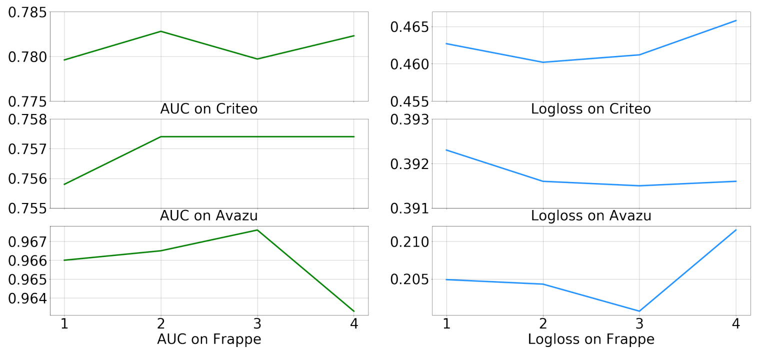


Figure 4.4: Parameter sensitivity study on the highest feature orders.

isolated from each other without any potential interactions. The results prove the existence of the multiple aspects of semantics, i.e., the feature polysemy, in the click-through records in complex datasets and justify the usage of multi-head Transformer. As the number of heads increases, the performances descend due to over-parameterization.

4.3.2.4 Highest feature order

We evaluate InterHAt with different highest feature order, i.e., different k, on three datasets. The k changes from 1 to 4. We use cross-features from the first- to the k-th-order in these experiments. The results are shown in Figure 4.4. On large datasets, Criteo and Avazu, the AUC and Logloss have marginal fluctuations when the order increases. However, in Frappe datasets, overfitting comes into existence after the order is greater than 3. In general, InterHAt has a stable performance on high-order learning.

4.3.3 Interpretability

Interpretation is generated in company with the predictions which is one of the major contributions of InterHAt. In this section, we demonstrate the interpretations by visualizing the learned salient low- or high-order features. However, the actual content of the click-through records in the two public real-world benchmark datasets, Criteo and Avazu, are encrypted for privacy-preserving issues, which makes it impossible to justify the interpretation constructed by InterHAt. Therefore, in order to comprehensively test the explanation generation of InterHAt, we use a real-world dataset and a synthetic dataset to simulate real click-through records. In the following subsections, we discuss data collection and results based on the two datasets.

4.3.3.1 Evaluate on real dataset

Dataset The real semantic meaning of the features in Criteo and Avazu are encrypted. Other datasets that are also in recommemder system domain are appropriate substitutes. Therefore, we select MovieLens-1M [HK16] dataset for this tasks. MovieLens-1M has plain-text[4] attributes and is also extensively employed to evaluate recommender systems [SSX18]. It is composed of around 1M anonymous movie ratings given by 6,040 MovieLens users. Each records has user profile, movie genres, and a rating ranging from 1 to 5. User profiles include *Age*, *Gender*, and *Profession* and movie attributes include *Release year* and 18 different genres. We consider a "rate" action in MoiveLens-1M as a click-through in CTR prediction, i.e., the positive samples with labels as 1. We create a negative records with the same amount as the positive ones by randomly sampling pairs of movies and users and label them as 0. The positive and negative datasets are disjoint to each other.

Results We plot the heat maps of the attention weights from the first-order to the third-order, that is, the $\boldsymbol{\alpha}_i$ in Equation (4.2) with $i \in \{1, 2, 3\}$. We select $k = 3$ since few higher order features are found significant. The $\boldsymbol{\alpha}_f$ of the following cases are not presented in the interest of space. The k-order example we select for visualization has a largest $\boldsymbol{\alpha}_f[k]$ among all weights in the corresponding $\boldsymbol{\alpha}_f$. The darker cells in Figure 4.5, 4.6, and 4.7 signify greater feature importance that InterHAt learns from the rating records. The movie genres

[4]`https://grouplens.org/datasets/movielens/1m/`

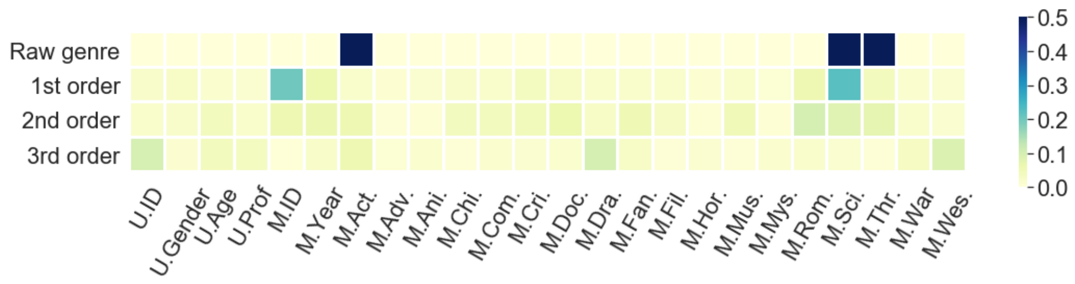


Figure 4.5: Attention weights of a first-order salient feature example (*The Terminator*, 1984)

in the figures have been shortened to three letters[5]. In the *Raw genre* rows, black cells mean the movie has the corresponding genre attributes in the raw data, i.e., the training data.

Figure 4.5 shows a rating to the movie *The Terminator* (1984), which reports the largest aggregation attention weight on the first-order features. In this record, we observe that `M.ID` and `M.Sci.` significantly outweigh other cells in the *1st-order* row due to the high reputation of the movie itself and its outstanding characteristic as a Sci-Fi (Science Fiction). InterHAt also detects that the other two genre labels, *Action* and *Thriller*, are not as accurate and hence not highlighted. Higher order interactions are not observed as strong since people may already make the decision to watch The Terminator by its great reputation as a Sci-Fi movie.

Figure 4.6 demonstrates a second-order interaction dominated case in a rating towards *Léon: The Professional* (1994). We observe one first-order feature and two second-order features with more "heat". For the two second-order features, *Crime* and *Romance* interaction is captured due to the moving love and criminal story that the movie tells. The combined affect of the two characteristics increases the probability of this movie being watched and rated. A first-order feature `U.ID` is highlighted since InterHAt discovers from the training data that this particular user frequently rates movies. InterHAt then believes a rate is likely to happen when he or she is present. This is consistent with logic of attention-based model

[5]Please refer to `http://files.grouplens.org/datasets/movielens/ml-1m-README.txt` for the full names.

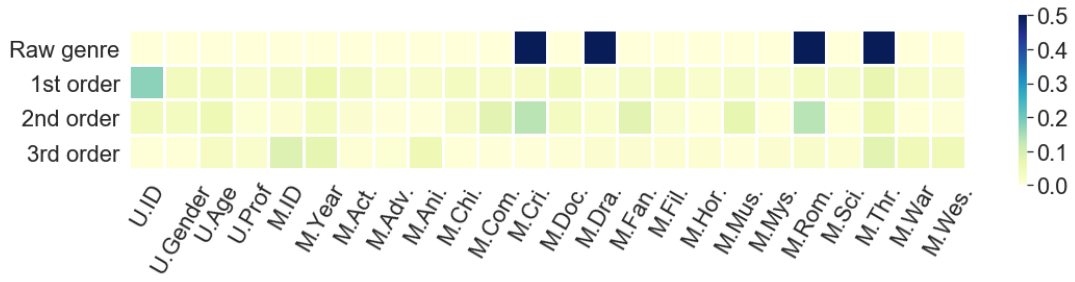


Figure 4.6: Attention weights of a second-order salient feature example (*Léon: The Professional*, 1994)

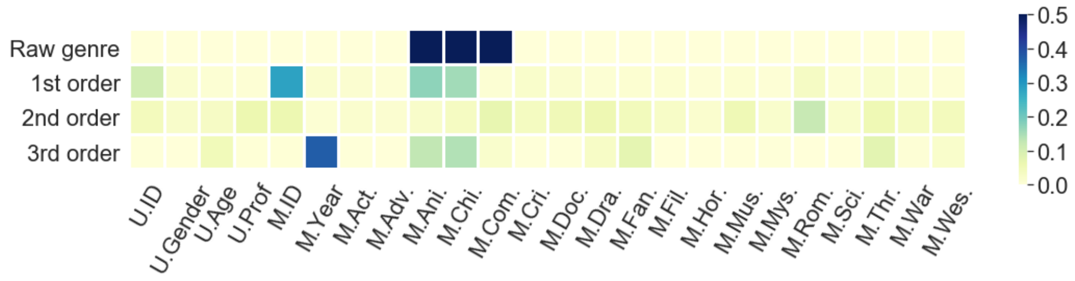


Figure 4.7: Attention weights of a third-order salient feature example (*Toy story 2*, 1999)

interpretation in Section 2.2.2 that it is only able to highlight the steering of information flow in the model but unable to create an intuitive human-readable story of predictions.

An example of the third-order interaction dominated case is given in Figure 4.7 where the feature importance of a rating of *Toy story 2* (1999) is depicted. We observe a three-feature interaction, *Release year*, *Animation*, and *Children*, in which we are curious about how *Release year* interacts with the other two closely related features. It turns out that the year 1999 is important for animated movies and the total amount of tickets sold reaches a maximum between 1995 and 2000 according to a movie market survey[6].

[6]`https://m.the-numbers.com/market/production-method/Animation-and-Live-Action`

4.3.3.2 Evaluate on synthetic dataset

Dataset Considering that MovieLens-1M is genuinely rating data rather than click-through data, we conduct a set of experiments using synthetic data to show the interpretability. The synthetic data contains 100K synthesized click-through records with 10 fields $F = [f_1, \ldots, f_{10}]$ simulating real click-through records. Each field is created independently and can take values from $[\beta_1, \ldots, \beta_{10}]$. The synthetic instance labels are decided by the feature groups using the rules in Table 4.4 as a simulation of groups of feature(s) solely or jointly affecting the CTR prediction. The labels are decided as follows. Given a feature group G, $y = 1$ representing the click-through happens, and $y = 0$ as the opposite,

$$\Pr(y = 1|F, G) = \begin{cases} p_1 & \text{if } \forall f_i \in G, f_i.val = \beta_i; \\ p_2 & \text{otherwise.} \end{cases} \tag{4.7}$$

For example, enabling *Rule 2* in Table 4.4 implies that the synthetic label has p_1 probability to be 1 and $1 - p_1$ to be 0 when the conditions hold that $f_3.val = \beta_3$ and $f_4.val = \beta_4$. Otherwise, the label will be set to 1 by p_2 probability and o by $1 - p_2$ probability. We set p_1 to 0.9 and p_2 to 0.2 to represent high and low probabilities of click-through. Without loss of generality, we evaluate features from the first-order to the third-order.

Table 4.4: Rules for creating the synthetic dataset for interpretation study.

Index	k-th order	Feature group G
1	First-order	$\{f_1\}$
2	Second-order	$\{f_3, f_4\}$
3	Third-order	$\{f_5, f_6, f_7\}$

Results We present the salient features by heat maps of the attentions in each layer. Each cell of order i in the following heat maps represents a normalized *average* of aggregation attention $\boldsymbol{\alpha}_i$ of all records that satisfy the rule, i.e., $f_i.val = \beta_i$.

Figure 4.8 depicts the heat map of the first order by enacting *Rule 1*. We observe that f_1 draws the largest attention among all features which is consistent to *Rule* 1. An additional observation is that the variance from the attentions is small, meaning that using first-order only for learning and predicting has a low stability.

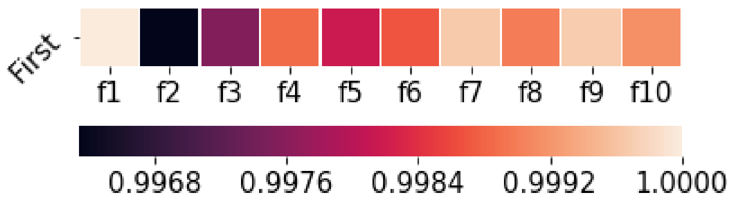


Figure 4.8: First-order attention heat map for the interpretation on synthetic data.

We plot the second-order heat map in Figure 4.9 to visualize the second-order feature interactions by *Rule 2*. The learned attention values on f_3 and f_4 are notably greater than other cells as they have lighter colors in contrast with the black ones. Although the cells of f_3 and f_4 have different colors, they are still numerically close to each other. Therefore, the results in Figure 4.9 also demonstrate the ability of InterHAt to extract salient features and interpret click-through predictions.

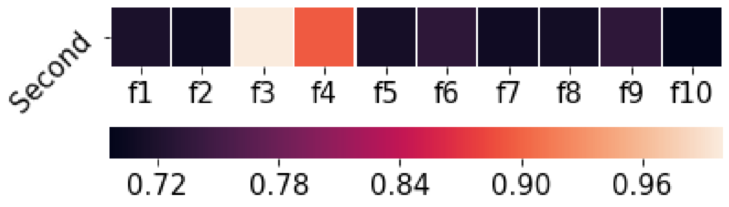


Figure 4.9: Second-order attention heat map for the interpretation on synthetic data.

Rule 3 exemplifies the interpretability in high-order scenarios. We include the heat maps from the first-order to the four-order in Figure 4.10. From the top three rows, we spot the process of InterHAt acquiring feature interaction knowledge from the dataset. In the first-order, f_6 and partial f_5 information is learned. Next, f_5 and partial f_7 are captured in addition to f_6 in the row of the second-order. Then, the third-order finished acquiring all the interaction information. Finally, the fourth-order features show uniform attention values with marginal variability, which demonstrates that the high-order feature learning terminates at the third-order and no greater order features are present in the dataset.

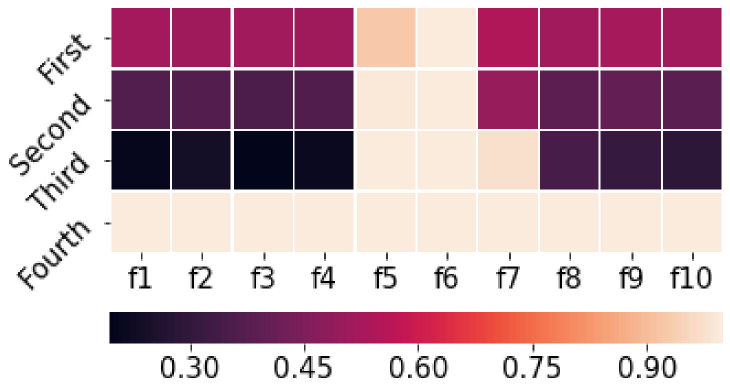


Figure 4.10: Third-order attention heat maps for the interpretation on synthetic data.

In summary, we comprehensively evaluated the ability of InterHAt to generate rationales while predicting the CTR using a real-world dataset and a synthesized dataset. The heat map visualizations of both datasets can be reasonably explained in alignment with human perception, which endorses the interpretability of InterHAt.

4.4 Summary

In this chapter, we proposed InterHAt, an interpretable, efficient, and effective CTR predictor. InterHAt leverages a multi-head Transformer to learn the polysemy of feature interactions and leverages a hierarchical attention structure to learn the importance of different orders of features. The explanation is inferred according to the learned importance distribution. Moreover, InterHAt achieves a relatively low computational cost compared with other models. Comprehensive experiments show that InterHAt can learn interpretable importance for feature interactions, runs faster than state-of-the-art models meaning a high efficiency on CTR prediction, and achieves comparable or even better performances.

Here are a few aspects for future effort: (1) A better embedding learning paradigm of numerical features is needed to boost the performance; (2) Explainable deep neural networks, such as MLP and outer products-based networks, are in demand to achieve high accuracy and interpretability.

CHAPTER 5

Point-of-Interest Recommendation via Graph Enhanced Attention Network

Point-of-interest (POI) recommendation is an emerging area of research on location-based social networks to analyze user behaviors and contextual check-in information. For this problem, existing approaches, with shallow or deep architectures, have two major drawbacks. First, for these approaches, the attributes of individuals have been largely ignored. Therefore, it would be hard, if not impossible, to gather sufficient user attribute features to have complete coverage of possible motivation factors. Second, most existing models preserve the information of users or POIs by latent representations without explicitly highlighting salient factors or signals. Consequently, the trained models with unjustifiable parameters provide few persuasive rationales to explain why users favor or dislike certain POIs and what really causes a visit. To overcome these drawbacks, we propose GEAPR, a POI recommender that is able to interpret the POI prediction in an end-to-end fashion. Specifically, GEAPR learns user representations by aggregating different factors, such as structural context, neighbor impact, user attributes, and geolocation influence. GEAPR takes advantage of a triple attention mechanism to quantify the influences of different factors for each resulting recommendation and performs a thorough analysis of the model interpretability. Extensive experiments on real-world datasets demonstrate the effectiveness of the proposed model. GEAPR is deployed and under test on an internal web server.

5.1 Motivation and Background

Point of interest (POI) recommendation is a critical component in the recommender system family. *Point of interest* refers to locations that customers of online business directories or review forums are interested in. Such directories or forums are typically named as *location-based social network* (LBSN), e.g., Yelp and Foursquare, since users interact with each other in various ways such as co-reviewing, co-visiting, or direct connecting via friendship relations.

POI recommendation has a wide coverage of scenarios in which the advertised items have significant spatial attributes that strongly influence the user decisions. Properly recommending POI replies on precisely understanding user taste, POI's property, geolocation, and their correlations. Varying from simple to sophisticated, existing algorithms are painstakingly customized for more precise user preference modeling, POI profiling, and user-POI relevance estimation. In other words, the development of POI recommendation systems witnesses the utilization of multiple modalities of data to achieve more satisfactory POI recommendations.

That being said, we point out two prevalent shortcomings of existing models: (1) inadequate interpretable motivation analysis for POI visits, and (2) absent attribute study for users with a diverse background.

First, for motivation analysis, the ranking functions of existing approaches merely fuse the multi-modal information without explicitly quantifying or explaining which modalities are comparatively more important than the others and which are less relevant. However, quantitatively comprehending the key causes of the check-ins is valuable because it is able to measurably interpret the users' mind-sets on choosing the next POI to visit. For example, some users always check in places their friends have checked in or have suggested, while others tend to visit places that their peer group favors. Such numerical motivation importance measurements can also reasonably provide a clear answer to the following debate. Tobler's first law of geography [Tob70], frequently cited by previous work [ZC15, LGH16, YYL11], states that: "Everything is related to everything else, but near things are more related than distant things." But authors of GeoMF state the opposite: a user's visit to certain POI implies exactly her indifference to those nearby, otherwise she would have visited them instead in the first place [LZX14]. With numerical motivation analysis, it becomes easy to capture and interpret the primary causes of user check-ins, i.e. the motivations, which also benefits LBSN on explaining their recommendations. In contrast, existing approaches are not adaptive enough to learn different motivations in a transparent way. They instead simply use unweighted additions [MZW18, ZZK17] or feature vector concatenations [YBZ17] to mingle the intermediate information and produce recommendations. Motivation importance is hardly revealed by these operations. Such discrepancy calls for an effective architecture that is elaborately developed for interpretable motivation analysis with explicit salience distribution on different motivation factors.

Second, existing POI recommendation methods largely ignore user attribute study which, however, is of great importance. The extensive literature of item-based recommender systems, e.g., movies and books, have demonstrated the potential of user profile, demographics, and their complex joint effects to enhance recommendation accuracy [Ren10, GTY17, XYH17, BFU16, LCC20, HLZ17]. Such potential is also plausible in the context of POI recommendation. For example, the young population loves to try different restaurants in different locations while the seniors may have distance concerns. However, user attribute information has been underestimated even by the recent deep learning-based POI recommendation models [MZW18, ZYZ19, ZZK17, ZMZ19, ZYL17] although deeper models have superior ability to fuse different information modalities and capture the corresponding importance. Therefore, it is necessary to incorporate the user attribute features to comprehensively cover possible motivation factors.

To address the two aforementioned concerns, we propose a Graph Enhanced Attention network for explainable POI Recommen-dation (short for *GEAPR*) in this chapter that recommends POIs in an adaptive and interpretable way. GEAPR leverages not only geographical and social information but also user personal attributes and provides an end-to-end justification of the recommendation in the meantime. Specifically, we decompose the possible motivating causes into four factors:

Structural Context. A check-in can be motivated by neighboring users with high structural proximity in the social network since they have a similar social context. This type of stimulus is typically ignored as it is latent and implicit. We argue and experimentally demonstrate that social structural context is a critical cause of visits.

Neighbor Impact. Impact from direct neighbors, i.e., friends, is another factor of interest since people are likely to trust their friends' suggestions and check-in POIs their friends did before. Previous works characterize neighbor impact by MF-based methods which fail to generate explanations simultaneously.

User Attributes. Check-in behaviors can also be spontaneous due to users' characteristics such as age, religion, income level, etc. For example, young users may choose to check-in the POIs that other young people love without external stimulus such as friends. GEAPR proposes to understand the underlying correlation between check-in behavior and attributes in a novel manner. Factorization machines-based models are dedicated to learning from attribute data. Therefore, GEAPR utilizes a factorization machines equipped with the attention mechanism to learn attribute features.

Geolocation Influence. Geolocation influence has a particularly strong impact on POI recommendations because it is intuitive that people are more aware of nearby restaurants, supermarkets, or museums, etc. than distant ones. In GEAPR, we fix the POI influence distribution parameterized by Manhattan distance and learn the user preference for each geographical unit.

Altogether, GEAPR takes advantage of the attention mechanism to quantify the influences of different factors for each resulting recommendation and performs a thorough analysis of the model interpretability. Some literature [JW19] states that attentions lack robustness to serve as an explanation. We acknowledge the statement but argue that interpretability reveals the *salience* of the factors the model captures from the complex statistics of training data. Also, to the best of our knowledge, generating a fully human-readable explanation as a by-product of the ranking score is yet technically infeasible since even users themselves are unable to articulate the exact reasons that motivate a visit to a POI.

The geolocation feature encoding is decoupled from the three other factors that only focus on the user's personal motivation. The main rationale is for the compatibility: although GEAPR is applied to the POI recommendation, it can be painlessly transplanted to geolocation-irrelevant recommendation scenarios by simply detaching the geolocation module. Examples include movie recommendation [ZYL18], question routing [LJS19], and new friend recommendations, etc. We summarize the major contributions:

- We propose GEAPR, a POI recommender that is able to interpret the POI prediction in an end-to-end fashion. It specifically focuses on four factors, namely structural context, neighbor impact, user attributes, and geolocation influence, and quantifies their influences by numeric values as the feature salience indicators.

- User attributes are taken into consideration in GEAPR. To the best of our knowledge, this is the **first** work that incorporate attributes to POI recommendation.

- Attention mechanism is used to address the recommendation interpretability by means of finding significant factors which are more influential in POI recommendation compared with other features.

- Extensive experiments are conducted on two real-world datasets from Yelp. Experimental results demonstrate the effectiveness of the proposed model. Testing results demonstrate the effectiveness of GEAPR.

5.2 GEAPR

5.2.1 Architecture Overview

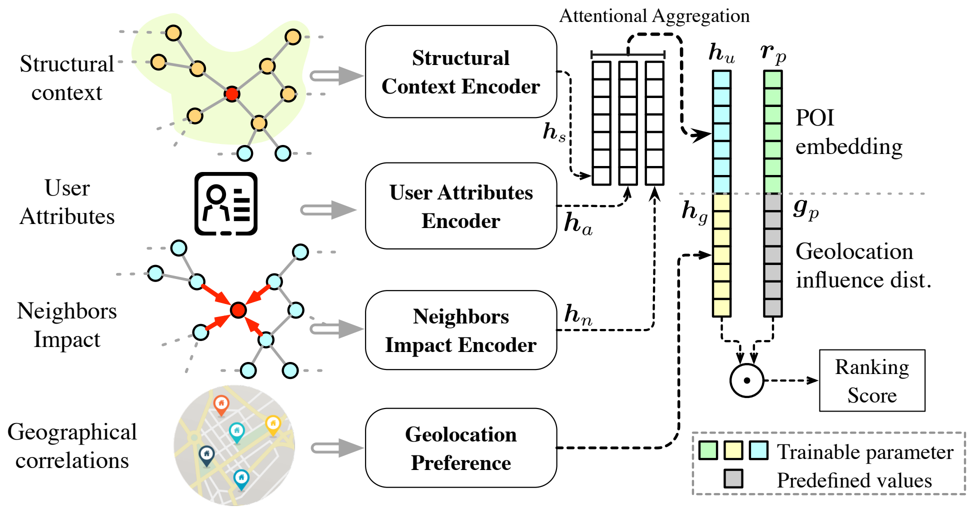


Figure 5.1: The overall pipeline of GEAPR.

The architecture of GEAPR is shown in Figure 5.1. Some important notations are summarized in Table 5.1. The inputs of GEAPR include the adjacency matrix $\mathbf{M}_a$ of the friendship graph of LBSN, structural context $\mathbf{M}_s$, the users' attributes $\mathcal{F}$, and the POI influence scores.

GEAPR uses three different architectures customized for the three factors on the user motivation side. Specifically, a dense neural network-based structural context encoder is utilized to learn the **structural context**, a graph neural network-based attentional friendship encoder is utilized to model the **neighbor impact**, and an attention-based latent factorization machine is utilized for preserving the **attribute interactions**.

These three sub-modules will generate three hidden feature representations individually as $\boldsymbol{h}_s$, $\boldsymbol{h}_n$, and $\boldsymbol{h}_a$. The information from three sources is then merged by an attentional aggregation [LCC20] strategy which is able to reveal the relative salience among them. The merged motivation representation is then combined with geolocation features as constraints so that

Table 5.1: Critical notations for GEAPR that are essential.

Notation(s)	Definition
U, P, E	The sets of users, POIs, and friendship relations in an LBSN.
n_u, n_p	The total numbers of users and POIs in an LSBN.
G	The friendship graph for users. $G = \{U, E\}$.
$\mathcal{N}_G(u)$	The set of neighbors of user u in G and $\mathcal{N}_G(u) = \{v \mid (u, v) \in E\}$.
$\mathcal{F}$, m	The set of fields with m fields of user attributes. $m = \lvert\mathcal{F}\rvert$.
$\mathbf{M}_a$	The adjacency matrix of G, $\mathbf{M}_a \in \{0, 1\}^{n_u \times n_u}$.
$\mathbf{M}_s$	The structural context matrix based on $\mathbf{M}_a$, $\mathbf{M}_s \in \mathbb{R}^{n_u \times n_u}$.
$\boldsymbol{h}_s$, $\boldsymbol{h}_n$, $\boldsymbol{h}_a$	Hidden vectors of structural context, neighbor impact, and attributes.
$\boldsymbol{h}_u$	The attentional aggregation of $\boldsymbol{h}_s$, $\boldsymbol{h}_n$, and $\boldsymbol{h}_a$.
$\boldsymbol{h}_g$	The geolocation preference of the user.
$\boldsymbol{g}_p$	The predefined geographical influence scores of POI to grids on a map.
$\boldsymbol{r}_p$	The hidden representation of POI semantics.
$s_{u,p}$	The score representing how likely user u will visit POI p.

strongly relevant but distant POIs will be removed from the recommendation. GEAPR then takes the dot-product of the graph-enhanced user embedding and the POI embedding to generates a scalar score $s_{u,p}$ representing the likelihood of a user u visiting a POI p in the future.

In order to preserve reliable interpretability while making an accurate recommendation, the building blocks of GEAPR focus on attention-based algorithms. Although literature argues that attention lacks the potential to provide an "explanation" that agrees with human perception, it still reveals the *distribution of salience* which can be considered as a form of explanation. In addition, it is worth noting that unlike the tasks where ground truth is typically defined and easily accessible, formal explanations are unavailable from LBSN or public datasets for interpretable POI recommendation as ground truth. Therefore, measuring the "correctness" of the generated interpretation is impracticable.

5.2.2 Structural Context Factor

The structural context tries to model the commonality of the close neighbors of a certain user. Intuitively, the proximity of user characters and preferences can be propagated through a few hops of social connections to form cliques within the network. In order to capture social context from network structure, GEAPR utilizes Random Walk with Restart (RWR), a popular method widely used for learning community proximity [NCL18]. The structural context features of a user are learned based upon his or her RWR representation.

Mathematically, given a network G of n_u nodes represented by its adjacency matrix $\mathbf{M}_a$ with $\mathbf{M}_{a,ij} = 1$ if nodes i and j are connected and otherwise 0, a starting user u_0 in U, the r-step RWR vector $\boldsymbol{p}^{(r)} \in \mathbb{R}^{n_u}$ is computed by

$$\boldsymbol{p}^{(r)} = \gamma \boldsymbol{p}^{(0)} + (1-\gamma)\boldsymbol{p}^{(r-1)}[\mathbf{D}^{-1}\mathbf{M}_a],$$

where γ denotes the probability that the random walk generator restarts from u_0, $\boldsymbol{p}^{(0)}$ denotes the corresponding row of u_0 in $\mathbf{M}_a$, and $\mathbf{D}$ denotes a diagonal matrix with $\mathbf{D}_{ii} = \sum_{j=1}^{n_u} \mathbf{M}_{a,ij}$.

Let R denote the maximum step of the RWR process, the summation of $\boldsymbol{p}^{(r)}$ is considered as the structural context. R is usually set as a small value such as 2 or 3 to make sure only *local* information is preserved in $\boldsymbol{h}'_s$ and $\boldsymbol{h}'_s = \sum_{r=1}^{R} \boldsymbol{p}^{(r)}$, $\boldsymbol{h}'_s \in \mathbb{R}^{n_u}$.

However, one problem of encoding the local context is the enormous dimension: the size of $\boldsymbol{h}'_s$ is the same scale as the user numbers. Therefore, GEAPR conducts dimension reduction to $\boldsymbol{h}'_s$ to generate $\boldsymbol{h}_s$, the latent features of structural context, by a multi-layer dense neural network with $\mathrm{ReLU}(x) = \max(0, x)$ as the activation function (using two layers as an example):

$$\boldsymbol{h}_s = \mathrm{ReLU}(\mathbf{W}_2^T(\mathrm{ReLU}(\mathbf{W}_1^T\boldsymbol{h}'_s + \boldsymbol{b}_1)) + \boldsymbol{b}_2), \quad \boldsymbol{h}_s \in \mathbb{R}^d,$$

where d is the dimension of hidden representations and $\{\mathbf{W}_i, \boldsymbol{b}_i\}$ are trainable parameters. ReLU$(\cdot)$ introduces non-linearity and enhances the representation learning capacity for structural context.

5.2.3 Neighborhood Impact Factor

The second aspect of potential visit stimuli is the direct friends since one may naturally check in the POIs suggested by friends. We thereby focus on the understanding of impact from neighbors $\mathcal{N}_G(u)$ of a user u. Graph attention network (GAT) [VCC17] provides an effective way to aggregate information from direct neighbors and compute the attention to pinpoint significant neighbors. Therefore, we encode neighborhood impact using an attention-based graph neural-network. Given a user u and the friends of u, $\mathcal{N}_G(u)$, the hidden neighbor feature $\boldsymbol{h}_n$ is

$$\boldsymbol{h}_n = \sigma\left(\sum_{j\in\mathcal{N}_G(u)} \alpha_{uj}\mathbf{W}_n\boldsymbol{v}_j\right).$$

$\sigma(\cdot)$ is typically a non-linear function such as ReLU$(\cdot)$ or $\tanh(\cdot)$. $\mathbf{W}_n \in \mathbb{R}^{d_n\times d_p}$ is a learnable weight matrix for the attention network that maps all neighbor embeddings to a common space. $\boldsymbol{v}_j$ is derived by the average POI embeddings that user j visited before. The scalar α_{uj} is the weight from user j to u and GEAPR computes α_{uj} by Equation (5.1) where *LeakyReLU* advances ReLU in that it allows shrunk negative signal to flow through, "||" denotes concatenation along an existing dimension, and $\boldsymbol{a} \in \mathbb{R}^{2d}$ is a trainable vector that helps compute the attention logits and $\mathbf{W} \in \mathbb{R}^{d\times d_p}$.

$$\alpha_{uj} = \frac{\exp\left(\text{LeakyReLU}\left(\boldsymbol{a}^T[\mathbf{W}\boldsymbol{v}_u||\mathbf{W}\boldsymbol{v}_j]\right)\right)}{\sum_{i\in\mathcal{N}_G(u)}\exp\left(\text{LeakyReLU}\left(\boldsymbol{a}^T[\mathbf{W}\boldsymbol{v}_u||\mathbf{W}\boldsymbol{v}_i]\right)\right)} \tag{5.1}$$

The set of attention weights α_{uj} demonstrates the influential neighbors. In addition to concatenation, other ways can also produce attention logits such as dot-product of $\boldsymbol{v}_j$ and $\boldsymbol{v}_u$, the matrix-dot-product $\boldsymbol{v}_j^T\mathbf{W}\boldsymbol{v}_u$, or the non-linear MLP of concatenation of $\boldsymbol{v}_j$ and $\boldsymbol{v}_u$. The original GAT [VCC17] can handle multiple attention heads and multiple neighborhood hops. Increased head numbers can preserve information in more sub-spaces, and enlarged scopes of direct or nearby users bring in more local context, which both benefit the performance. However, in consideration of the interpretability, we simplify the settings to a single head and one-hop neighbors.

5.2.4 Attribute Interactive Factor

Apart from the effects of social structural context and direct neighbors, the personal attributes are also important factors to motivate the user to visit particular POIs. The combinatorial possibilities of feature interactions create diverse influences on the users' preference towards POIs, which has been thoroughly studied in feature-based recommender systems such as factorization machines (FM) [Ren10], DeepFM [GTY17], and xDeepFM [LZZ18], etc. In GEAPR, we combine feature-based FM methods with attention mechanism [XYH17] to analyze feature interaction and maintain interpretability.

Embedding the categorical and numerical features into a lower-dimensional space is a prerequisite [Ren10, GTY17, XYH17, LCC20, LZZ18]. User attributes can be written as m fields $\{F_1, \ldots, F_m\}$ with different values, also known as *features*. We assign a trainable vector to each distinct feature $\boldsymbol{f}_c \in \mathbb{R}^{d_a}$ for categorical field and discretize the continuous value by bucketing and then treat the converted alternative as a categorical feature for the numerical field.

Given the feature embeddings, user attribute impact is model as

$$\boldsymbol{h}_a = \boldsymbol{w}_0 + \sum_{i=1}^{m} \beta_i \boldsymbol{f}_i + \sum_{i=1}^{m} \sum_{i=j+1}^{m} \lambda_{ij} \boldsymbol{f}_i \odot \boldsymbol{f}_j, \tag{5.2}$$

where $\boldsymbol{w}_0$ is the offset term, β_i, and λ_{ij} are the attention weights for first-order and second-order feature interactions. They are computed as follows, given feature matrix $\mathbf{F} = \{\boldsymbol{f}_1, \ldots, \boldsymbol{f}_m\}$, $\boldsymbol{\beta} = \text{softmax}(\text{ReLU}(\boldsymbol{q}_1^T \mathbf{F}))$, $\lambda_{ij} = \text{softmax}(\boldsymbol{q}_2^T \text{ReLU}(\mathbf{W}_a(\boldsymbol{f}_i \odot \boldsymbol{f}_j) + \boldsymbol{b}))$. Here $\boldsymbol{q}_1, \boldsymbol{q}_2 \in \mathbb{R}^{d_a}$, $\mathbf{W}_a \in \mathbb{R}^{d_a \times d_a}$ and $\boldsymbol{b} \in \mathbb{R}^{d_a}$ denote learnable tensors to build attention weights. $\odot$ is the element-wise multiplication. Once more, we use the attention weights as the information source for interpretability.

Equation (5.2) contains an enumeration of the first-order features and second-order feature interactions which are incomprehensive compared with the exponential-sized feature interaction space. Studies have shown that the first two orders of features are already capable of contributing sufficient information for learning interactive features and adding higher-order features is only making a marginal information supplement. That being said, GEAPR still enjoys great compatibility with higher-order features interaction models [LCC20, BFU16].

5.2.5 POI Geographical Influence

In GEAPR, we model the geographical influence features from two aspects: learnable user geolocational interest and predefined POI area influence. As shown in Figure 5.2, the left figure shows three trainable user preference geographical distributions and the right one shows an example of pre-defined POI influence score measured by Manhattan distance. Specifically, we first divide the city map of POIs into grids with n_{lat} units on the latitude axis and n_{long} units on the longitude axis. We model the geographical influence of a POI in grid p to a target grid t using the influential score $g_{p,t}$ [LZX14] as

$$g_{p,t} = K\left(\frac{d_{\text{man}}(p,t)}{\sigma_g}\right)$$

where σ_g denotes the standard deviation of distances, $K(\cdot)$ denotes a standard normal distribution, and $d_{\text{man}}(a,b)$ measures the Manhattan distance from the grid a to grid b. Therefore, we can define the influential score vector $\boldsymbol{g}_p$ of POI p as $\boldsymbol{g}_p \in \mathbb{R}^{(n_{\text{long}} \cdot n_{\text{lat}})}$ which is essentially a flattened 2-dimensional influential score matrix. We are also curious about the geographical preference distribution of users which is defined as a learnable parameter representing user preference, $\boldsymbol{h}_g \in \mathbb{R}^{(n_{\text{long}} \cdot n_{\text{lat}})}$. Each user has one unique $\boldsymbol{h}_g$. We define the geographical influence correlation between users and POIs by taking the product $\boldsymbol{h}_g^T \boldsymbol{g}_p$. The overlap between user-preferred regions and POI influential regions can be selected and amplified by multiplication.

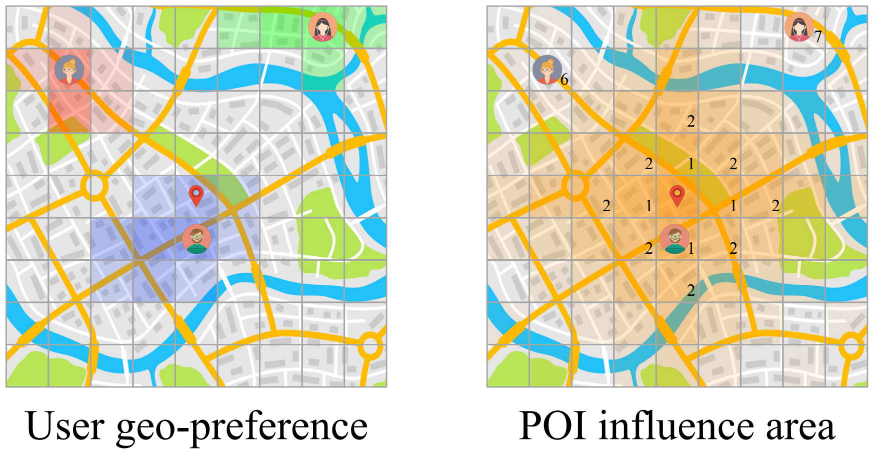


Figure 5.2: Geolocation encoding for user preference and POI influence.

5.2.6 Objective and Optimization

After showing the derivations of the representation of the four causing factors, we show how to make predictions for future check-ins. We first aggregate $\boldsymbol{h}_s$, $\boldsymbol{h}_n$, and $\boldsymbol{h}_a$ by an attention mechanism as Equation (5.3) and Equation (5.4) since they all encode users motivation.

$$\boldsymbol{h}_u = \pi_s \cdot \text{ReLU}(\boldsymbol{h}_s) + \pi_n \cdot \text{ReLU}(\boldsymbol{h}_n) + \pi_a \cdot \text{ReLU}(\boldsymbol{h}_a) \tag{5.3}$$

$$\pi_{x\in\{s,n,a\}} = \frac{\exp(\boldsymbol{w}^T \text{ReLU}(\boldsymbol{h}_x))}{\sum_{x'\in\{s,n,a\}} \exp(\boldsymbol{w}^T \text{ReLU}(\boldsymbol{h}_{x'}))} \tag{5.4}$$

Then Equation (5.5) computes the possibility of the potential check-in $s_{u,p}$ which is defined by the dot-product with motivation feature and geographical feature of users and POIs. If $\boldsymbol{r}_p$ represents the motivation-related POI semantics, then

$$s_{u,p} = [\boldsymbol{h}_u || \boldsymbol{h}_g] \cdot [\boldsymbol{r}_p || \boldsymbol{g}_p] = \boldsymbol{h}_u^T \boldsymbol{r}_p + \boldsymbol{h}_g^T \boldsymbol{g}_p. \tag{5.5}$$

The overall objective function is Equation (5.6) which sums a ranking loss L_{rank} and a regularization loss L_{reg} weighted by a hyper-parameter. In GEAPR, we use L_2 norm as the regularization term.

$$L = L_{\text{rank}}(\mathcal{D}, \mathcal{D}') + cL_{\text{reg}} \tag{5.6}$$

We use negative sampling to implement the ranking term that specifically penalize on the negative samples $\mathcal{D}'$ while optimizing the positive samples $\mathcal{D}$. There are two standard ways to implement the ranking loss, L_{rank}, namely *pair-wise* or *point-wise* ranking loss.

Point-wise (PO) Loss. Point-wise loss forces the positive instances to approach an indicator 1 and pushes the negative instances to indicator 0 via a cross-entropy loss of binary classification. $y = 1$ if $(u, p) \in \mathcal{D}$, $y = 0$ if $(u, p) \in \mathcal{D}'$, and $\sigma(\cdot)$ is the sigmoid function.

$$L_{\text{rank-po}} = -\sum_{\mathcal{D},\mathcal{D}'} \left(y \log(\sigma(s_{u,p})) + (1 - y) \log(1 - \sigma(s_{u,p}))\right).$$

Pair-wise (PA) Loss. Pair-wise loss tries to capture the partial order relationships in the training data and maintain that order between the scores of positive instances and negative instances. We follow the method in RankNet [BSR05] as the equation below with $(u, p) \in \mathcal{D}$, $(u, p') \in \mathcal{D}'$, and $\Delta_{u,p,p'} = s_{u,p} - s_{u,p'}$.

$$L_{\text{rank-pa}} = \sum_{\mathcal{D},\mathcal{D}'} -\Delta_{u,p,p'} + \log(1 + \exp(\Delta_{u,p,p'})).$$

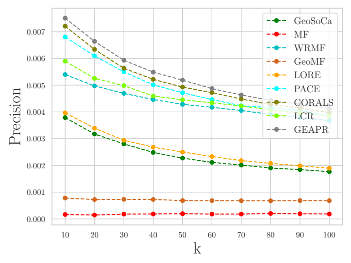
We use Adam [KB14] to optimize the parameters since all modules in GEAPR are continuous and differentiable.

Complexity We use $\delta(n)$ to denote the complexity of the multiplication of an n dimensional vector and an $n \times n$ dimensional matrix. We omit the detailed computation of the complexity and give the result as follows. Summing up complexity of the four modules and of the overall optimization, the forward pass complexity of GEAPR is $T = r\delta(n_u) + \delta(d) + n_u\delta(d_p) + m^2\delta(d_a) + O(n_{\text{long}} \times n_{\text{lat}})$. Further, considering that different dimensions are on the same scale, we rewrite T as

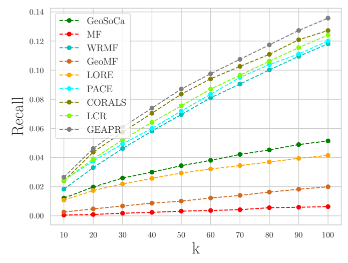
$$T = r\delta(n_u) + (n_u + m^2)\delta(d) + O(n_{\text{long}} \times n_{\text{lat}}).$$

If hyperparameters r, d, m, n_{long}, and n_{lat} are set, they can be viewed as constants. Then the complexity T of a training forward pass is proportional to $\delta(n_u)$, which equals to $O(n_u^2)$ and is common to graph neural networks [VCC17].

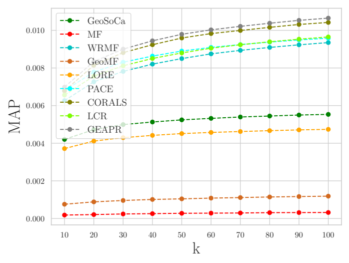
5.3 Experiments



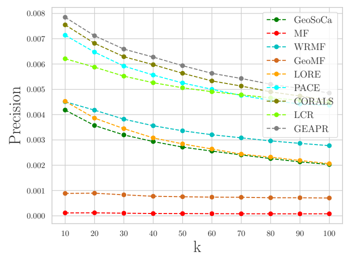
(a) Prec@k for Toronto



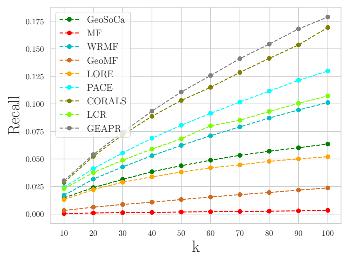
(b) Recall@k for Toronto



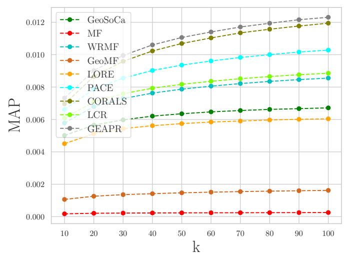
(c) MAP@k for Toronto



(d) Prec@k for Phoenix



(e) Recall@k for Phoenix



(f) MAP@k for Phoenix

Figure 5.3: Performance evaluation of GEAPR compared with baseline models.

This section reports the evaluation of GEAPR on effectiveness and interpretability by its performance on real-world datasets and case studies of interpretation.

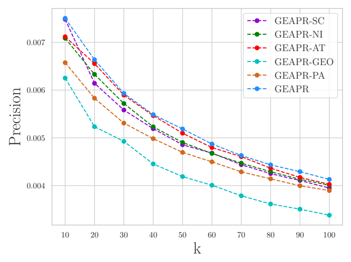
5.3.1 Experimental Setup

5.3.1.1 Dataset

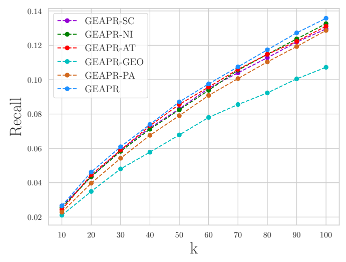
We use Yelp Challenge[1] Round 13 dataset for effectiveness and interpretability tests. We divide the reviews by cities and subsets of "Toronto" and "Phoenix" are used due to larger sizes. Yelp dataset contains comprehensive details of the reviews, user attributes, user friendships, and POI locations. The friendship network in the datasets serves as the information

[1]`https://www.yelp.com/dataset`

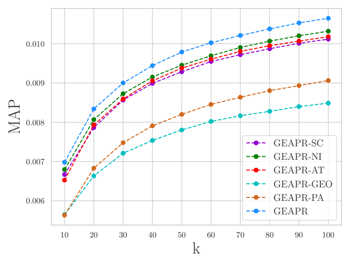
source of the neighbor impact encoder and the structural context encoder of GEAPR. The adjacency matrix is the input of the neighbor impact encoder and also helps to create the structural context. Yelp Round 13 dataset *does not* include the owners of the check-in records. We instead consider the abundant reviews to be equivalent to check-ins since intuitively each review corresponds with a past check-in. We sort all reviews grouped by users chronologically and filter out users with less than 10 reviews to avoid cold-start. We partition the review set of each individual into 9:1 where 90% data is used for training and the 10% rest for testing. Table 5.2 shows the statistics.



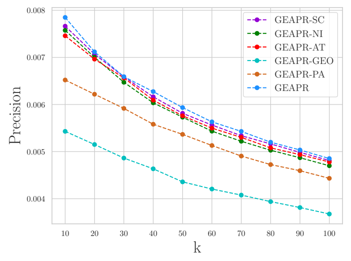
(a) Prec@k for Toronto



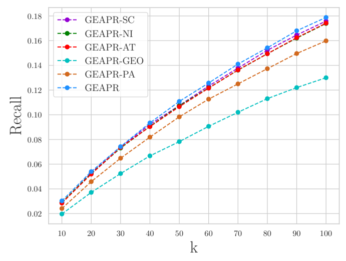
(b) Recall@k for Toronto



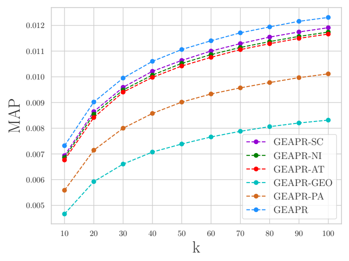
(c) MAP@k for Toronto



(d) Prec@k for Phoenix



(e) Recall@k for Phoenix



(f) MAP@k for Phoenix

Figure 5.4: Ablation study of GEAPR compared with its variants.

5.3.1.2 Metrics and Baseline Models

For effectiveness evaluation, we consider three metrics widely utilized in information retrieval and recommender systems, including Mean Average Precision at K (**MAP@*k***), Precision at k (**Prec@*k***), and Recall at k (**Recall@*k***). For the i-th test sample, given the list of the top

k POIs ranked according to the predicted scores $\boldsymbol{s}_i = (s_{i1}, s_{i2}, \ldots, s_{ik})$, and the m POIs that a particular user checked-in in the test set $T_i = \{t_{i1}, \ldots, t_{im}\}$, the three metrics are defined as follows where n is the number of test samples and $\boldsymbol{s}_{i,1-j}$ denotes the prefix sublist of $\boldsymbol{s}_i$ of length j. Prec@$k = \frac{1}{n}\sum_{i=1}^{n}\frac{|T_i \cap \boldsymbol{s}_i|}{k}$, Recall@$k = \frac{1}{n}\sum_{i=1}^{n}\frac{|T_i \cap \boldsymbol{s}_i|}{|T_i|}$, MAP@$k = \frac{1}{n}\sum_{i=1}^{n}\mathrm{AP}_i(T_i, \boldsymbol{s}_i)$, and $\mathrm{AP}_i(T_i, \boldsymbol{s}_i) = \frac{1}{k}\sum_{j=1}^{k}\frac{|T_i \cap \boldsymbol{s}_{i,1-j}|}{j}$. All three metrics take values in $[0, 1]$. Larger values represent better results. Prec@k and Recall@k measure how good the top-ranked POIs match with the ground truth and MAP@k signifies if the ground truth is ranked at higher positions. k is from 10 to 100.

Table 5.2: Statistics of the datasets for evaluation.[2]

Dataset	#.User	#.POI	#.Reviews	#.U-Cxn	%.Reviews	%.U-Cxn
Toronto	9582	9102	234388	104402	2.687×10^{-3}	1.139×10^{-3}
Phoenix	11289	9633	249029	163900	2.290×10^{-3}	1.286×10^{-3}

Eight baselines are used for performance comparison including **MF**, **GeoMF**, **WRMF**, **LORE**, **GeoSoCa**, **PACE**, **CORALS**, and **LCR**. They include matrix factorization based models and deep-learning-based models. We provide brief descriptions for the baseline models for comparisons with GEAPR[3]. The parameter settings follow the default values in the source code or in their original papers. **MF**, the Matrix factorization model that decomposes the user-POI check-in matrix and make predictions by reconstruction. **GeoMF** [LZX14], a geolocation-enhanced MF model that considers the geographical factors as constraints. **WRMF** [HKV08], Weight regularized MF, incorporates both implicit and explicit check-ins for future check-in predictions. **LORE** [ZCL14] builds a location-location transition graph to specifically model the sequential influence of POIs to user preferences. In **GeoSoCa** [ZC15],

[2] "#": *the count of*; "%": *the density of*; "U-Cxn": user friendship connections.

[3] The implementation of PACE: `https://github.com/yangji9181/PACE2017`; The implementation of the rest of baselines: `http://spatialkeyword.sce.ntu.edu.sg/eval-vldb17`.

Geolocation, social connections, and POI categories are all considered to extract diverse information. **PACE** [YBZ17] is a deep learning-based multi-task learning algorithm that predicts POI, user context, and spot context simultaneously for better accuracy and robustness. **CORALS** [LJJ19] incorporates the modeling of reputation prediction of POIs in location recommendations. **LCR** [LLW20], Local Collaborative Ranking, assumes that user-POI matrix is local low-rank rather than global low-rank to mitigate the data sparsity issue of POI recommendation.

Even so, our experiments are still able to provide coverage on both traditional methods and deep learning-based methods. In addition, sequential POI recommendation models [ZMZ19, ZZK17, YCG20, JXZ19, SQC20, WYC20] are excluded as well due to different use cases as discussed earlier in Section 2.1.3.

5.3.1.3 Reproduction

The prototype of GEAPR was implemented by Python (3.6.8) and TensorFlow (1.14.0) and run with a 16 GB Nvidia Tesla V100 GPU embedded in a Nvidia DGX-1 server. The code is publicly available on GitHub[4]. and a comprehensive end-to-end instruction on how to run the code is also provided.

The hyper-parameters that generate the results in Section 5.3 are listed in Table 5.3. To prevent overfitting, dropout is employed in the graph attention network module for structural context modeling and the attentional factorization machines module for attribute impact modeling. The dropout rates are also shown in Table 5.3.

5.3.2 Effectiveness

This section reports the comparisons of GEAPR and the baselines for performance analysis, and of GEAPR and its variants for ablation study. A parameter sensitivity study is also

[4]The source code is available here: `https://github.com/zyli93/GEAPR`

Table 5.3: Parameter settings for the experiments of GEAPR.

Parameters	Toronto	Phoenix
d_a, d_n, d_p, and d	$\{64,64,64,32\}$	
$n_{\text{long}}, n_{\text{lat}}$	$\{30, 30\}$	
R, γ, and c	$\{3, 0.05, 0.1\}$	
Loss function	Point-wise	
Regularization function	L_2, $c = 0.0001$	
Optimizer	Adam	
Learning rate	0.001	
Hidden layers of SC module	2 layers: $\{64,48\}$	
Negative sampling (P:N)	1:105	1:90
NI module dropout	0.3	0
AT module dropout	0.3	0.2

provided to cast light on the heuristics to parameter tuning. Please note that POI recommendation tries to identify all potential POIs from an enormous candidate base, which is essentially hard due to the unpredictability of users' minds. Users can receive multifaceted stimuli, a great proportion of which are implicit and difficult to capture based merely on the dataset. As a result, the numeric values of POI recommendation are **relatively small** for all state-of-the-art models.

5.3.2.1 Comparison with Baselines

The experimental results on effectiveness are shown in Figure 5.3. Point-wise (PO) loss is selected due to its superior performance and the results by pair-wise loss are shown in Section 5.3.2.2. It is demonstrated that GEAPR can achieve the state-of-the-art result by outperforming MF, GeoMF, WRMF, LORE, GeoSoCa, PACE, CORALS, and LCR on all three metrics. In other words, the ground truth of the test samples is effectively identified at high ranking positions. The advancement of deep learning has pushed the performance of this task to an extreme such that numerically small performance increases should also be considered **significant**. Therefore, GEAPR has made significant progress on accurate POI recommendation. As PACE is deep learning-based and requires larger input data density, its performance slightly drops under the settings of Table 5.2. CORALS requires dense review data for location reputation modeling and hence the density hurts its performance as well. The architecture that maintains interpretability requires GEAPR to *avoid unjustifiable* element-wise feature multiplication and aggregate information by weighted average pooling. Hence some complex or subtle information is missed and the learning power of GEAPR is undermined. That being said, GEAPR still achieves great performances.

5.3.2.2 Ablation Study

Ablation study illustrates in Figure 5.4 the contributions of the four factors and the performance difference of the two ranking losses. “GEAPR-X” refers to the variant that differs from GEAPR by “X”. X can be “SC” (no structural context), “NI” (no neighbor impact), “AT” (no user attributes), “GEO” (no geolocations), or “PA” (uses pair-wise loss). The curve of GEAPR is higher than other variants in all six subfigures. Four conclusions are drawn: (1) Attribute information provides performance gain indicating its usefulness. The reason of the relatively small contribution is the lack of the diversity of attribute information revealing user interests such as age and gender; (2) Removing the geolocation causes the largest performance deterioration. Therefore, the geographical information is the most influential factor as it is closely related to the POI task. It also shows that accurately mining geographical information is critical in accurate POI recommendation; (3) Taking away any factor will hurt the performance meaning that *all three non-geographical factors take effect and contribute uniquely* to the performance increase; (4) Pair-wise loss is not as useful as point-wise loss. Pair-wise loss relies on an advanced sampling strategy to sample data that is closer to the "decision hyperplane" with larger probabilities at the cost of extra computation. Instead, GEAPR draws negative samples uniformly with the goal of being generalizable to the diverse possible distributions of the data.

5.3.2.3 Parameter Sensitivity

We briefly introduce the parameter sensitivity study of GEAPR. None of the tuned parameters has a monotonic relationship with the evaluation results. For example, a greater negative sampling ratio will slow down the training and overly penalize on the unobserved check-ins so that the ground truth can also be mistakenly concealed, whereas a smaller ratio overfits the positive samples but underfits the negative, hurting the performance in another way. As for embedding size d, assigning d as 32, 64, or 128 only produces a little performance fluctuation but making it larger or smaller will deteriorate the recommendation accuracy. There is no theoretical guarantee on their optimality of the hyper-parameters used to derive the results in Figure 5.3. It is plausible to grid-search for other settings with better performances.

5.3.3 Interpretability

In this section, we demonstrate the interpretability of GEAPR by plotting the heat maps of $\boldsymbol{\pi} = \{\pi_s, \pi_n, \pi_a\}$, $\boldsymbol{\alpha}_u$ in the graph attention network module for neighbor impact, $\boldsymbol{\beta}$ and $\boldsymbol{\lambda}$ in the attribute influence module denoting the first- and second-order interaction importance, respectively, and $\boldsymbol{h}_g$ for geolocation feature. They are designed to probe the importance of features and generate interpretation accordingly.

5.3.3.1 User Motivation Study

We plot three examples on this topic shown in Figure 5.5, 5.6, and 5.7. The blue bars show the motivation breakdown $\boldsymbol{\pi}$; the green bars show the attributes importance $\boldsymbol{\beta}$; the variable-length dark orange bars show the friends count of a user and the weights learned from the

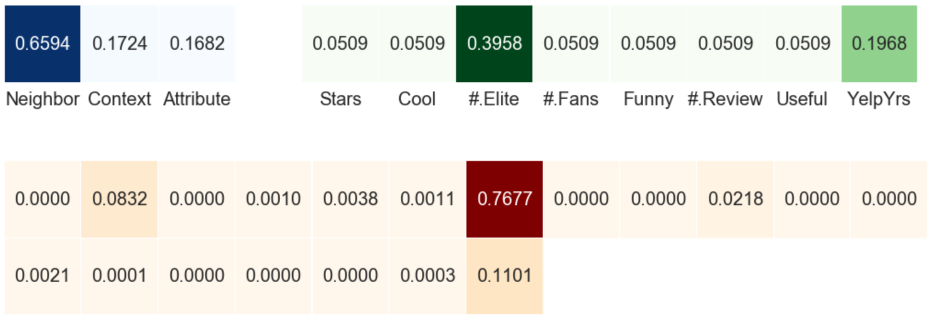


Figure 5.5: Example with significant neighbor impact.

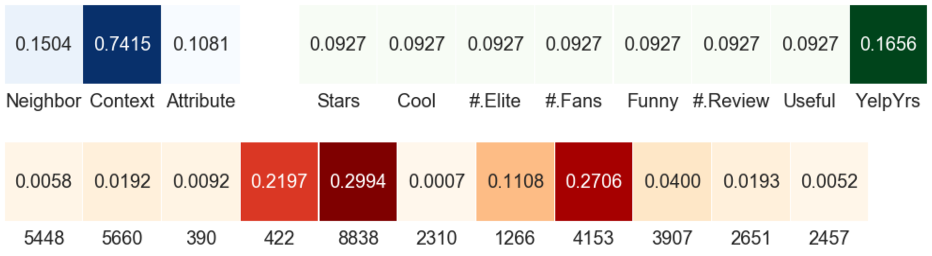


Figure 5.6: Example with significant structural context.

neighbor impact module ($\boldsymbol{\alpha}_u$). The $\boldsymbol{\lambda}$ is not plotted since it is observed that the values in $\boldsymbol{\lambda}$ are almost identical, showing that the second-order features are orthogonal to the formation of user motivation. This fact is consistent with our perception since features provided by the dataset are mostly counts or scores (e.g., review counts, funny score, cool score, etc.) and are independent of each other by intuition.

Figure 5.5 shows an example with important *neighbor feature* since its weight is the

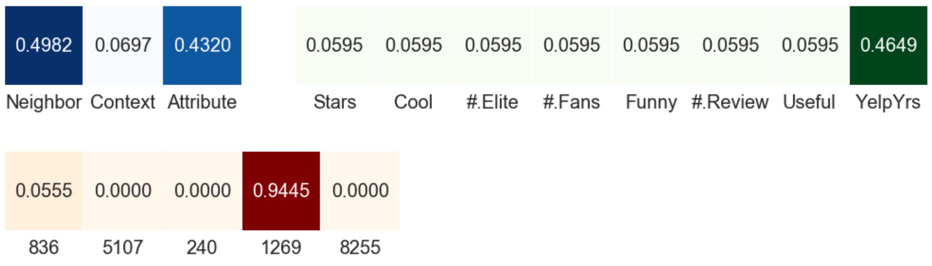


Figure 5.7: Example with significant user attribute.

largest in the blue heat map. From the 19 neighbors of the user, only one user has a huge impact on causing the user's motivation. The single peak of the user neighbor impact boosts of direct neighbor importance and decreases of the context importance. And the strong neighbor and context combined depress the weight of attribute since all weights add up to 1.

Figure 5.6 shows an example with important *structural context* features. The actual structural context is hard to depict since it is a dense vector with dimension size identical to the number of users. But we can still observe that the friends of this user are contributing more evenly impacts compared with Figure 5.5. That is, the user's visit preference is actually influenced by many neighbors and potentially the further neighbors, i.e., the users in the same structural context or community.

Figure 5.7 exemplify a case with both important *attribute* information and important *neighbor impact*. The lack of friendship connections (only 5 friends) and the single-peaked neighbor impact push the model to learn motivation from attributes. Therefore a heavy-weight is put on `YelpYrs` that stands for the number of years the user had been on Yelp. It turns out that the user's "Yelping years" is 8 years, a time long enough to form a user's visiting habits. From the figures, we notice a common property of attribute weights (green bars) that `#.Elite` and `YelpYrs` are usually highlighted as opposed to other user attributes. `#.Elite` denotes how many times the user had been awarded as "Yelp elite". We suppose that these two features have tight bonds with user activity and are understood by the model as a signal of a generally stronger motivation.

In addition, we acknowledge the insufficiency of attribute information of the Yelp dataset even though other POI datasets do *not* provide attributes at all. Without informative features, the models can not produce convincing rationale but complex statistical signals from the data.

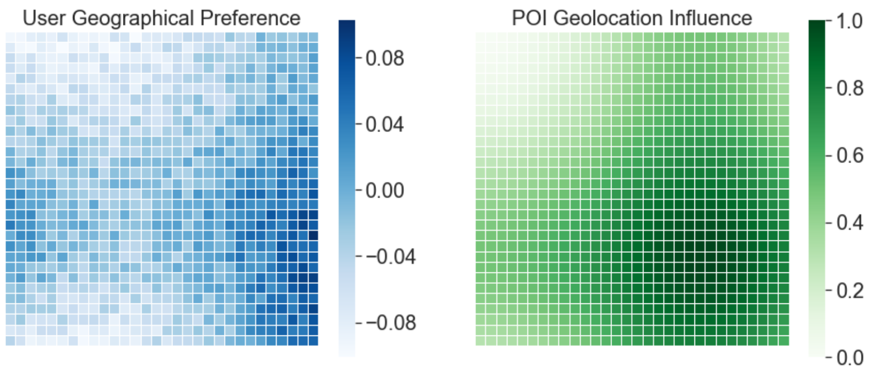


Figure 5.8: User geographical preference *v.s.* single POI influence distribution

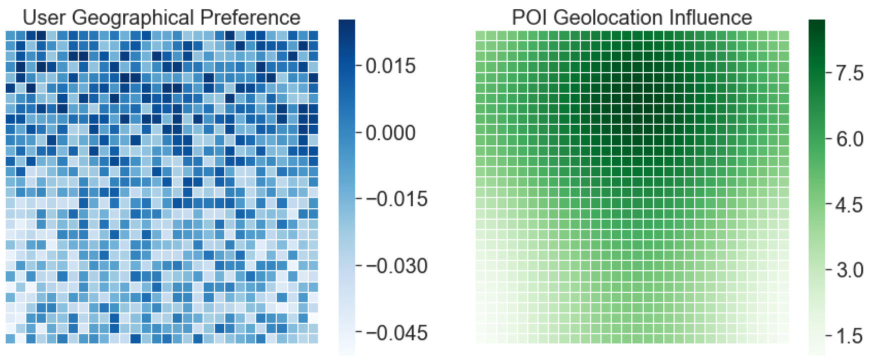


Figure 5.9: User geographical preference *v.s.* sum of POI influence distribution

5.3.3.2 Geolocation Preference

As shown in Equation (5.5), the geographical correlation with user and POI is modeled by a linear dot-product. That is, a greater preference value represents a stronger inclination to that zone. We exemplify the results by showing the learned user preference against (1) the predefined influence distribution of a single visited POI and (2) the normalized summation of the influence distributions of all visited POIs in Figure 5.8 and Figure 5.9. Each figure contains 30×30 grids each representing a grid in the real-world map of POIs.

In Figure 5.8, we observe that the user preference values concentrate at the bottom right corner which agrees with the influential center of the POI. In addition, the top left corner is not favored by the user and the POI influence figure also shows the same pattern of low influence. In Figure 5.9, the heat zones are generally aligned between the two figures. The upper halves of both user preference and POI influence are heated and the bottom parts are relatively inactive. This demonstrates that GEAPR is capable of capturing the geographical preference of a user and understand which parts the more favored than other parts.

5.4 Summary

In this chapter, we propose GEAPR, a graph-enhanced POI recommendation algorithm that incorporates user friendship network information in addition to user attributes and geolocation features. Specifically, GEAPR decomposes the motivation of user check-ins into four different aspects: social structural context, neighborhood impact, user attribute, and geolocation, and quantifies the importance of each feature. In addition, GEAPR employs the attention mechanism to generate interpretations that reveal the salient motivating factors, influential neighbors, informative attribute interactions, and heated geographical areas, etc. Experimental results demonstrate the effectiveness and interpretability of GEAPR.

We list the following potential improvements as future work: (1) It may be helpful to also build a POI graph by semantics and apply graph mining algorithms; (2) A better way to preserve non-linear geolocational features is needed to learn complex information.

5.5 Supplementary Materials

5.5.1 Complexity of GEAPR

Here we provide detailed analysis of computational complexity T for the forward pass of GEAPR. We use $\delta(n)$ to denote the complexity of the multiplication of an n dimensional vector and an $n \times n$ dimensional matrix.

Structural Context. The complexity of obtaining the RWR vector $\boldsymbol{p}^{(r)}$ is $T(\boldsymbol{p}^{(r)}) = r(O(n_u) + \delta(n_u)) = r\delta(n_u)$. The complexity from $\boldsymbol{p}^{(r)}$ to $\boldsymbol{h}'_s$ and then to $\boldsymbol{h}_s$ is $T(\boldsymbol{h}_s) = O(rn_u) + \delta(n_u) + \delta(d)$. Overall, its complexity is $T(\boldsymbol{p}^{(r)}) + T(\boldsymbol{h}_s) = r\delta(n_u) + \delta(n_u) + \delta(d) = r\delta(n_u) + \delta(d)$.

Neighbor Impact. The complexity of neighbor impact consists of computing α_{uj} and computing $\boldsymbol{h}_n$. $T(\alpha_{uj})$ is dominated by the denominator of Equation (5.1) as $T(\alpha_{uj}) = n_u(\delta(2d_p) + O(d)) = n_u\delta(d_p)$. $T(\boldsymbol{h}_n) = n_u\delta(d_p) + O(n_u) = n_u\delta(d_p)$ since we can pre-compute and save all α_{uj}. Therefore, its complexity is $T(\alpha_{uj}) + T(\boldsymbol{h}_n) = n_u\delta(d_p)$.

Attribute Interactions. We compute its complexity as the summation of computing first-order and second-order terms. $T(\text{first-order}) = m\delta(d_a)$ and $T(\text{second-order}) = m^2(\delta(d_a) + O(d_a) + O(d_a)) = m^2\delta(d_a)$. Overall, its complexity is $T(\text{first-order}) + T(\text{second-order}) = m\delta(d_a) + m^2\delta(d_a) = m^2\delta(d_a)$.

Geographical Influence. The complexity comes from the dot product with $n_{\text{long}} \times n_{\text{lat}}$ dimensions. Therefore, its complexity is $O(n_{\text{long}} \times n_{\text{lat}})$.

Overall. The computational complexity of the prediction terms in Equation (5.6) can be merged into the previous terms. Therefore, summing up complexity of the four modules and of the overall optimization, the forward pass complexity of GEAPR is $T = r\delta(n_u) + \delta(d) + n_u\delta(d_p) + m^2\delta(d_a) + O(n_{\text{long}} \times n_{\text{lat}})$. Further, considering that different dimensions are on the same scale, we rewrite T as

$$T = r\delta(n_u) + (n_u + m^2)\delta(d) + O(n_{\text{long}} \times n_{\text{lat}}).$$

CHAPTER 6

Understanding the Recommendation with Aspect-Sentiment Co-Extraction

Compliments and concerns in reviews are valuable for understanding users' shopping interests and their opinions with respect to specific aspects of certain items. Existing review-based recommenders favor large and complex language encoders that can only learn latent and uninterpretable text representations. They lack explicit user-attention and item-property modeling, which however could provide valuable information beyond the ability to recommend items. Therefore, we propose a tightly coupled two-stage approach, including an Aspect-Sentiment Pair Extractor (ASPE) and an Attention-Property-aware Rating Estimator (APRE). Unsupervised ASPE mines Aspect-Sentiment pairs (AS-pairs) and APRE predicts ratings using AS-pairs as concrete aspect-level evidences. Extensive experiments on seven real-world Amazon Review Datasets demonstrate that ASPE can effectively extract AS-pairs which enable APRE to deliver superior accuracy over the leading baselines.

6.1 Motivation and Background

Reviews and ratings are valuable assets for the recommender systems of e-commerce websites since they immediately describe the users' subjective feelings about the purchases. Learning user preferences from such feedback is straightforward and efficacious. Previous research on review-based recommendation has been fruitful [CZJ18, CZL18, BLT17, LLD19]. Cutting-edge natural language processing (NLP) techniques are applied to extract the latent user sentiments, item properties, and the complicated interactions between the two components.

However, existing approaches have disadvantages bearing room for improvement. Firstly, they dismiss the phenomenon that users may hold different *attentions* toward various *properties* of the merchandise. An item property is the combination of an aspect of the item and the characteristic associated with it. Users may show strong attentions to certain properties but indifference to others. The attended advantageous or disadvantageous properties can dominate the *attitude* of users and consequently, decide their generosity in rating.

Reviews	**Microphone**	**Comfort**	**Sound**
R1 [5 stars]: *Comfortable. Very high quality sound. . . . Mic is good too. There is an switch to mute your mic. . . I wear glasses and these are comfortable with my glasses on. . . .*	*good* (satisfied)	*comfortable*	*high quality* (praising)
R2 [3 stars]: *I love the comfort, sound, and style but the mic is complete junk!*	*complete junk* (angry)	*love*	*love*
R3 [5 stars]: *. . . But this one feels like a pillow, there's nothing wrong with the audio and it does the job. . . . con is that the included microphone is pretty bad.*	*pretty bad* (unsatisfied)	*like a pillow* (enjoyable)	*nothing wrong*

Table 6.1: Example reviews of a headset with three aspects, microphone quality, comfort level, and sound quality, as well as the sentiments.

Table 6.1 exemplifies the impact of the user attitude using three real reviews for a headset. Three aspects are covered: microphone quality, comfortableness, and sound quality. Comparing R1 and R2, we find that different users react differently (microphone quality) to the same item due to distinct personal attentions and, consequently, give divergent ratings. Comparing R1 and R3, we find that a user can still rate highly of an item due to special attention on particular aspects (comfort level) regardless of certain unsatisfactory or indifferent properties (microphone and sound qualities). The microphone quality is controversial. R2 and R3 criticize it but R1 praises it. The sole disagreement between R1 and R2 is on microphone, which is the major concern of R2, results in the divergence of ratings (5 stars vs. 3 stars). However, R3 neglects that disadvantage and grades highly (5 stars) for its superior comfortableness indicated by the metaphor of "pillow".

Secondly, understanding user motivations in granular item properties provides valuable information beyond the ability to recommend items. It requires aspect-based NLP techniques to extract explicit and definitive aspects. However, existing aspect-based models mainly use *latent* or *implicit* aspects [CZJ18] whose real semantics are unjustifiable. Similar to Latent Dirichlet Allocation (LDA, [BNJ03]), the semantics of the derived aspects (topics) are mutually overlapped [HMG20]. These models undermine the resultant aspect distinctiveness and lead to uninterpretable and sometimes counterintuitive results. The root of the problem is the lack of large review corpora with aspect and sentiment annotations. The existing ones are either too small or too domain-specific [WP18] to be applied to general use cases. Progress on sentiment term extraction [DS19, TGX20, CLW20] takes advantage of neural networks and linguistic knowledge and partially makes it possible to use unsupervised term annotation to tackle the lack-of-huge-corpus issue.

In this chapter, we seek to understand how reviews and ratings are affected by users' perception of item properties in a fine-grained way and discuss how to utilize these findings transparently and effectively in rating prediction. We propose a two-stage recommender with an **unsupervised** Aspect-Sentiment Pair Extractor (ASPE) and an Attention-Property-aware Rating Estimator (APRE). ASPE extracts `(aspect, sentiment)` pairs (AS-pairs) from reviews. The pairs are fed into APRE as explicit user attention and item property carriers indicating both frequencies and sentiments of aspect mentions. APRE encodes the text by a contextualized encoder and processes implicit text features and the annotated AS-pairs by a dual-channel rating regressor. ASPE and APRE jointly extract explicit aspect-based attentions and properties and solve the rating prediction with a great performance.

Aspect-level user attitude differs from user preference. The user attitudes produced by the interactions of user attentions and item properties are sophisticated and granular sentiments and rationales for interpretation (see Section 6.3.4 and 6.5.3). Preferences, on the contrary, are coarse sentiments such as like, dislike, or neutral. Preference-based models may infer that R1 and R3 are written by headset lovers because of the high ratings. Instead, attitude-based methods further understand that it is the comfortableness that matters to R3 rather than the item being a headset. Aspect-level attitude modeling is more accurate, informative, and personalized than preference modeling.

Note. Due to the page limits, some supportive materials, marked by "†", are presented in the **Supplementary Materials**. We strongly recommend readers check out these materials. The source code of our work is available on GitHub at `https://github.com/zyli93/ASPE-APRE`.

6.2 ASPE and APRE

6.2.1 Problem Formulation

Review-based rating prediction involves two major entities: users and items. A user u writes a review $r_{u,t}$ for an item t and rates a score $s_{u,t}$. Let R^u denote all reviews given by u and R^t denote all reviews received by t. A rating regressor takes in a tuple of a review-and-rate event (u, t) and review sets R^u and R^t to estimate the rating score $s_{u,t}$.

6.2.2 Unsupervised ASPE

We combine three separate methods to label AS-pairs without the need for supervision, namely PMI-based, neural network-based (NN-based), and language knowledge- or lexicon-based methods. The framework is visualized in Figure 6.1.

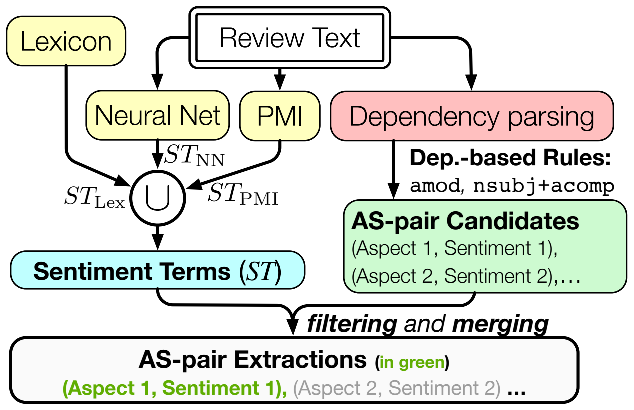


Figure 6.1: Pipeline of Aspect-Sentiment Pair (AS-pair) Extractor (ASPE).

6.2.2.1 Sentiment Terms Extraction

PMI-based method Pointwise Mutual Information (PMI) originates from Information Theory and is adapted into NLP [ZLS19, TGX20] to measure statistical word associations in corpora. It determines the sentiment polarities of words using a small number of carefully selected positive and negative seeds (s^+ and s^-) [TGX20]. It first extracts candidate sentiment terms satisfying the part-of-speech patterns by [Tur02] and then measures the polarity of each candidate term w by

$$\mathrm{Pol}(w) = \sum_{s^+} \mathrm{PMI}(w, s^+) - \sum_{s^-} \mathrm{PMI}(w, s^-). \tag{6.1}$$

Given a sliding window-based context sampler ctx, the $\mathrm{PMI}(\cdot, \cdot)$ between words is defined by

$$\mathrm{PMI}(w_1, w_2) = \log \frac{p(w_1, w_2)}{p(w_1)p(w_2)}, \tag{6.2}$$

where $p(\cdot)$, the probability estimated by token counts, is defined by $p(w_1, w_2) = \frac{|\{ctx|w_1,w_2 \in ctx\}|}{\text{total } \#ctx}$ and $p(w_1) = \frac{|\{ctx|w_1 \in ctx\}|}{\text{total } \#ctx}$. Afterward, we collect the top-q sentiment tokens with strong polarities, both positive and negative, as ST_{PMI}.

NN-based method As discussed in Section 2.1.5, co-extraction models [DS19] can accurately label AS-pairs only in the training domain. For sentiment terms with consistent semantics in different domains such as *good* and *great*, NN methods can still provide a robust extraction recall. In this work, we take a pretrained SDRN [CLW20] as the NN-based method to generate ST_{NN}. The pretrained SDRN is considered an off-the-shelf tool similar to the pretrained BERT which is irrelevant to our rating prediction data. Therefore, we argue ASPE is unsupervised for open domain rating prediction.

Knowledge-based method PMI- and NN-based methods have shortcomings. The PMI-based method depends on the seed selection. The accuracy of the NN-based method deteriorates when the applied domain is distant from the training data. As compensation, we integrate a sentiment lexicon ST_{Lex} summarized by linguists since expert knowledge is widely used in unsupervised learning. Examples of linguistic lexicons include SentiWordNet [BES10] and Opinion Lexicon [HL04]. The latter one is used in this work.

Building sentiment term set The three sentiment term subsets are joined to build an overall sentiment set used in AS-pair generation: $ST = ST_{\text{PMI}} \cup ST_{\text{NN}} \cup ST_{\text{Lex}}$. The three sets compensate for the discrepancies of other methods and expand the coverage of terms shown in Table 6.5.

6.2.2.2 Syntactic AS-pairs Extraction

To extract AS-pairs, we first label AS-pair *candidates* using dependency parsing and then filter out non-sentiment-carrying candidates using (ST). This procedure is explained in detail by pseudocode of Algorithm 1.

Algorithm 1: AS-pairs Generation

```
   Input: Sentiment terms $ST$, dependency parser DepParser, threshold $c$.
   Output: AS-pairs
   Data: Review-rating corpus $R$; WordNet with synsets.
   /* Initialize AS-pair candidate and AS-pair sets                        */
1  $AS\text{-}cand$, $AS\text{-}pairs \leftarrow \emptyset, \emptyset$
   /* Extract AS-pair candidates.                                          */
2  foreach review $r \in R$ do
3      dep-graph$_r$ $\leftarrow$DepParser($r$)
4      foreach dependency relation $r_{\text{dep}}$ in dep-graph$_r$ do
5          if $r_{\text{dep}}$ is nsubj+acomp or $r_{\text{dep}}$ is amod then
6              Add corresponding ($noun$, $adj.$) tuple to $AS\text{-}cand$ (Figure 6.2)
   /* Merge synonym aspects                                                */
7  foreach ($noun$,$adj.$) tuple $\in AS\text{-}cand$ do
8      MergeSynAspect(synsets, $noun$)
   /* Filter out non-AS-pairs by $ST$ and frequency threshold $c$.         */
9  foreach ($noun$, $adj.$) tuple $\in AS\text{-}cand$ do
10     if $adj. \in ST$ and Freq[$noun$] $> c$ then
11         Add ($noun$, $adj.$) to $AS\text{-}pairs$
12 return $AS\text{-}pairs$
```

Dependency parsing extracts the syntactic relations between the words. Some nouns are considered potential aspects and are modified by adjectives with two types of dependency relations shown in Figure 6.2: `amod` and `nsubj+acomp`. The pairs of nouns and the modifying adjectives compose the AS-pair candidates. Similar techniques are widely used in unsupervised aspect extraction models [TC20, DS19]. AS-pair candidates are noisy since not all adjectives in it bear sentiment inclination. ST comes into use to **filter** out non-sentiment-carrying AS-pair candidates whose adjective is not in ST. The left candidates form the AS-pair set. Admittedly, the dependency-based extraction for (noun, adj.) pairs is *suboptimal* and causes missing aspect or sentiment terms. An implicit module is designed to remedy this issue. Open domain AS-pair co-extraction is blocked by the lacking of public labeled data and is left for future work.

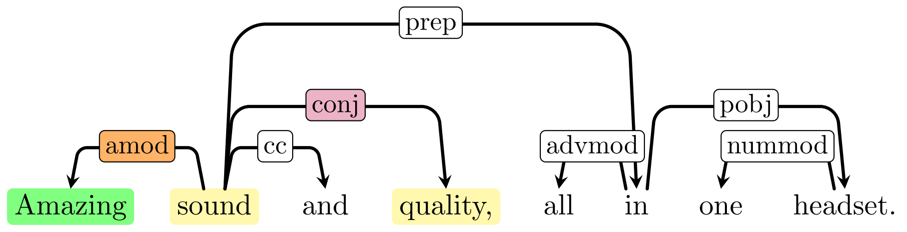
We introduce `ItemTok` as a special aspect token of the `nsubj+acomp` rule where `nsubj` is a pronoun of the item such as *it* and *they*. Infrequent aspect terms with less than c occurrences are ignored to reduce sparsity. We use WordNet synsets [Mil95] to **merge** the synonym aspects. The aspect with the most synonyms is selected as the representative of that aspect set.

Discussion ASPE is different from *Aspect Extraction* (AE) [TC20, LLL19, WHZ20, MLW19, AL18, XLS18, SXL17, HLN17] which extracts aspects only and infers sentiment polarities in {`pos`, `neg`, (`neu`)}. AS-pair co-extraction, however, offers more diversified emotional signals than the bipolar sentiment measurement of AE.

6.2.3 APRE

APRE, depicted in Figure 6.3, predicts ratings given reviews and the corresponding AS-pairs. It includes a user review encoder in the orange dashed box and an item review encoder in the top blue box, each containing an implicit channel (left) and an aspect-based explicit channel (right).

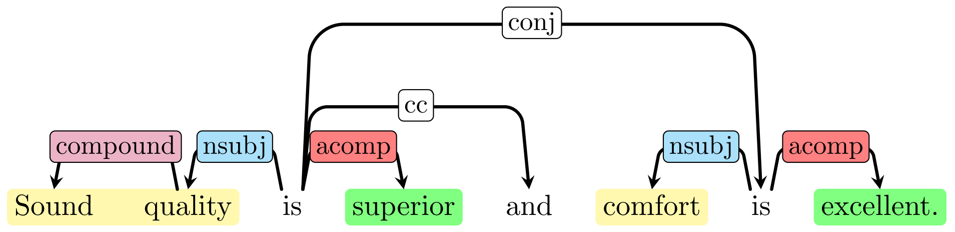
APRE first encodes language into embeddings, then learns explicit and implicit features, and finally computes the score regression. One distinctive feature of APRE is that it explicitly models the aspect information by incorporating a d_a-dimensional aspect representation $\boldsymbol{a}_i \in \mathbb{R}^{d_a}$ in each side of the substructures for review encoding. Let $\mathbf{A}^{(u)} = \{\boldsymbol{a}_1^{(u)}, \ldots, \boldsymbol{a}_k^{(u)}\}$ denotes the k aspect embeddings for users and $\mathbf{A}^{(t)}$ for items. k is decided by the number `amod` **dependency relation:**



Extracted AS-pair candidates:

`(sound, amazing)`, `(quality, amazing)`

`nsubj+acomp` **dependency relation:**



Extracted AS-pair candidates:

`(Sound quality, superior)`, `(comfort, excellent)`

Figure 6.2: Two dependency-based rules for AS-pair candidates extraction. Effective dependency relations and aspects and sentiments *candidates* are highlighted.

of unique aspects in the AS-pair set.

Language encoding The reviews are encoded into low-dimensional token embedding sequences by a fixed pre-trained BERT [DCL19], a powerful transformer-based contextualized language encoder. For each review r in R^u or R^t, the resulting encoding $\mathbf{H}^0 \in \mathbb{R}^{(|r|+2)\times d_e}$ consists of $(|r|+2)$ d_e-dimensional contextualized vectors: $\mathbf{H}^0 = \{\boldsymbol{h}^0_{\text{[CLS]}}, \boldsymbol{h}^0_1, \ldots, \boldsymbol{h}^0_{|r|}, \boldsymbol{h}^0_{\text{[SEP]}}\}$. `[CLS]` and `[SEP]` are two special tokens indicating starts and separators of sentences. We use a trainable linear transformation, $\boldsymbol{h}^1_i = \mathbf{W}^T_{\text{ad}}\boldsymbol{h}^0_i + \boldsymbol{b}_{\text{ad}}$, to adapt the BERT output representation $\mathbf{H}^0$ to our task as $\mathbf{H}^1$ where $\mathbf{W}_{\text{ad}} \in \mathbb{R}^{d_e\times d_f}$, $\boldsymbol{b}_{\text{ad}} \in \mathbb{R}^{d_f}$, and d_f is the transformed dimension of internal features. BERT encodes the token semantics based upon the context which resolves the polysemy of certain sentiment terms, e.g., "cheap" is positive for *price* but negative for *quality*. This step transforms the sentiment encoding to attention-property modeling.

Explicit aspect-level attitude modeling For aspect a in the k total aspects, we pull out all the contextualized representations of the sentiment words[1] that modify a, and aggregate their representations to a single embedding of aspect a in r as

$$\boldsymbol{h}_{u,r}^{(a)} = \sum \boldsymbol{h}_j^1, w_j \in ST \cap r \text{ and } w_j \text{ modifies } a.$$

An observation by [CSW20] suggests that users tend to use semantically consistent words for the same aspect in reviews. Therefore, sum-pooling can nicely handle both *sentiments* and *frequencies* of term mentions. Aspects that are not mentioned by r will have $\boldsymbol{h}_{u,r}^{(a)} = \mathbf{0}$. To completely picture user u's attentions to all aspects, we aggregate all reviews from u, i.e. R^u, using *review-wise aggregation* weighted by $\alpha_{u,r}^{(a)}$ given in the equation below. $\alpha_{u,r}^{(a)}$ indicates the significance of each review's contribution to the overall understanding of u's attention to aspect a

$$\alpha_{u,r}^{(a)} = \frac{\exp(\tanh(\boldsymbol{w}_{\text{ex}}^T[\boldsymbol{h}_{u,r}^{(a)}; \boldsymbol{a}^{(u)}]))}{\sum_{r' \in R^u} \exp(\tanh(\boldsymbol{w}_{\text{ex}}^T[\boldsymbol{h}_{u,r'}^{(a)}; \boldsymbol{a}^{(u)}]))},$$

where $[\cdot;\cdot]$ denotes the concatenation of tensors. $\boldsymbol{w}_{\text{ex}} \in \mathbb{R}^{(d_f+d_a)}$ is a trainable weight. With the usefulness distribution of $\alpha_{u,r}^{(a)}$, we aggregate the $\boldsymbol{h}_{u,r}^{(a)}$ of $r \in R^u$ by weighted average pooling:

$$\boldsymbol{g}_u^{(a)} = \sum_{r \in R^u} \alpha_{u,r}^{(a)} \boldsymbol{h}_{u,r}^{(a)}.$$

Now we obtain the user attention representation for aspect a, $\boldsymbol{g}_u^{(a)} \in \mathbb{R}^{d_f}$. We use $\mathbf{G}_u \in \mathbb{R}^{d_f \times k}$ to denote the matrix of $\boldsymbol{g}_u^{(a)}$. The item-tower architecture is omitted in Figure 6.3 since the **item property modeling** shares the identical computing procedure. It generates

[1]BERT uses WordPiece tokenizer that can break an out-of-vocabulary word into shorter word pieces. If a sentiment word is broken into word pieces, we use the representation of the first word piece produced.

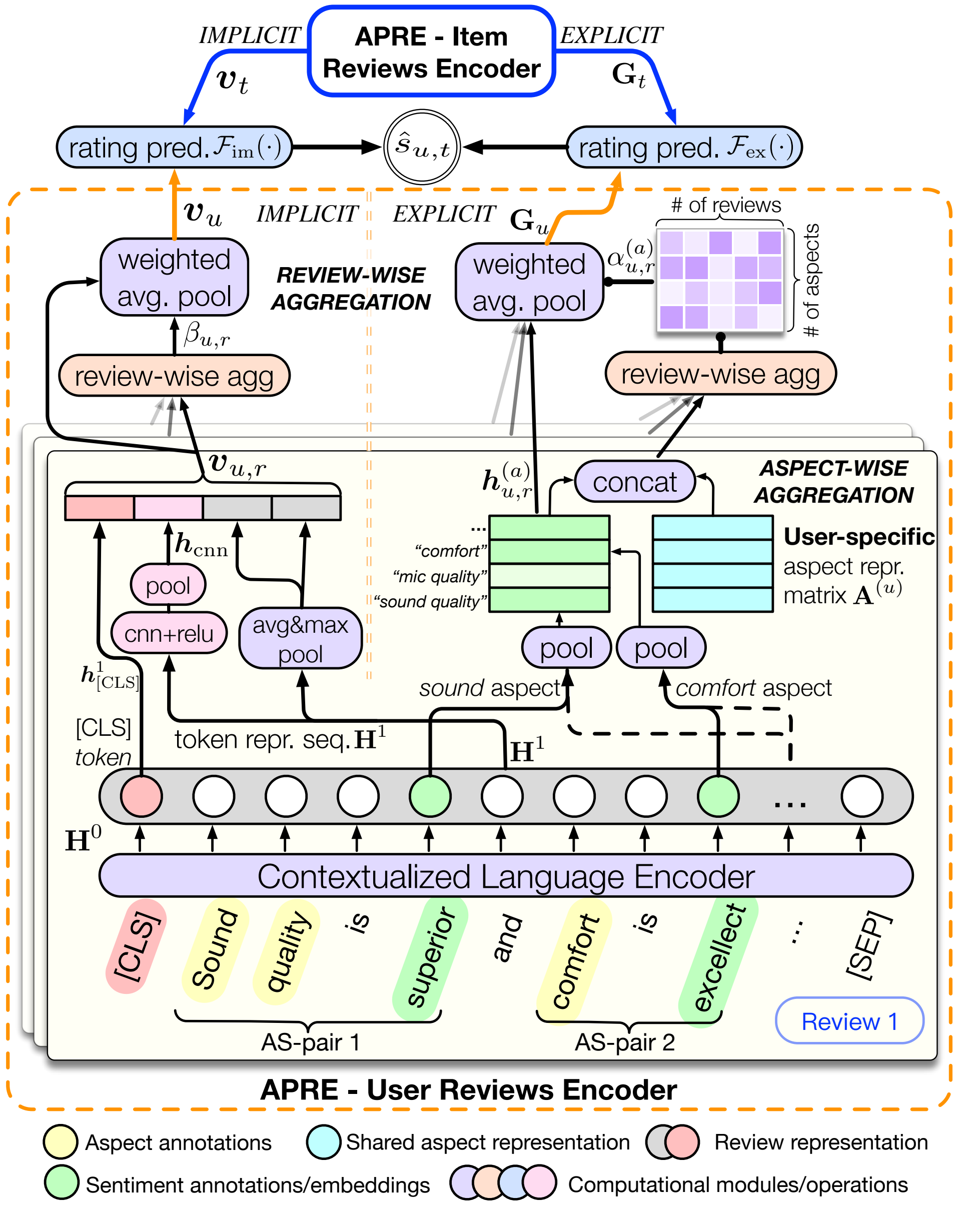


Figure 6.3: Pipeline of Attention-Property-aware Rating Estimator (APRE). Item encoder details are omitted since it is identical to the user encoder.

the item property representations $\boldsymbol{g}_t^{(a)}$ of $\mathbf{G}_t$. Mutual attention [LLD19, TLH18, DNC20] is *not* utilized since the generation of user attention encodings $\mathbf{G}_u$ is independent to the item properties and vice versa.

Implicit review representation It is acknowledged by existing works shown in Section 2.1.6 and Section 2.1.7 that implicit semantic modeling is critical because some emotions are conveyed without explicit sentiment word mentions. For example, "*But this one feels like a pillow ...*" in R3 of Table 6.1 does not contain any sentiment tokens but expresses a strong satisfaction of the *comfortableness*, which will be missed by the extractive annotation-based ASPE.

In APRE, we combine a global feature $\boldsymbol{h}^1_{\text{[CLS]}}$, a local context feature $\boldsymbol{h}_{\text{cnn}} \in \mathbb{R}^{n_c}$ learned by a convolutional neural network (CNN) of output channel size n_c and kernel size n_k with max pooling, and two token-level features, average and max pooling of $\mathbf{H}^1$ to build a comprehensive multi-granularity review representation $\boldsymbol{v}_{u,r}$:

$$
\begin{aligned}
\boldsymbol{v}_{u,r} &= \left[\boldsymbol{h}^1_{\text{[CLS]}}; \boldsymbol{h}_{\text{cnn}}; \text{MaxPool}(\mathbf{H}^1); \text{AvgPool}(\mathbf{H}^1)\right], \\
\boldsymbol{h}_{\text{cnn}} &= \text{MaxPool}(\text{ReLU}(\text{ConvNN\_1D}(\mathbf{H}^1))).
\end{aligned}
$$

We apply review-wise aggregation without aspects for latent review embedding $\boldsymbol{v}_u$

$$
\begin{aligned}
\beta_{u,r} &= \frac{\exp(\tanh(\boldsymbol{w}_{\text{im}}^T \boldsymbol{v}_{u,r}))}{\sum_{r' \in R^u} \exp(\tanh(\boldsymbol{w}_{\text{im}}^T \boldsymbol{v}_{u,r'}))}, \\
\boldsymbol{v}_u &= \sum_{r \in R^u} \beta_{u,r} \boldsymbol{v}_{u,r},
\end{aligned}
$$

where $\beta_{u,r}$ is the counterpart of $\alpha^{(a)}_{u,r}$ in the implicit channel, $\boldsymbol{w}_{\text{im}} \in \mathbb{R}^{d_{im}}$ is a trainable parameter, and $d_{im} = 3d_f + n_c$. Using similar steps, we can also obtain $\boldsymbol{v}_t$ for the item implicit embeddings.

Rating regression and optimization Implicit features $\boldsymbol{v}_u$ and $\boldsymbol{v}_t$ and explicit features $\mathbf{G}_u$ and $\mathbf{G}_t$ compose the input to the rating predictor to estimate the score $s_{u,t}$ by

$$\hat{s}_{u,t} = \underbrace{b_u + b_t}_{\text{biases}} + \underbrace{\mathcal{F}_{\text{im}}([\boldsymbol{v}_u; \boldsymbol{v}_t])}_{\text{implicit feature}} + \underbrace{\langle \boldsymbol{\gamma}, \mathcal{F}_{\text{ex}}([\mathbf{G}_u; \mathbf{G}_t]) \rangle}_{\text{explicit feature}}.$$

$\mathcal{F}_{\text{im}} : \mathbb{R}^{2d_{im}} \to \mathbb{R}$ and $\mathcal{F}_{\text{ex}} : \mathbb{R}^{2d_f \times k} \to \mathbb{R}^k$ are multi-layer fully-connected neural networks with ReLU activation and dropout to avoid overfitting. They model user attention and item property interactions in explicit and implicit channels, respectively. $\langle \cdot, \cdot \rangle$ denotes inner-product. $\boldsymbol{\gamma} \in \mathbb{R}^k$ and $\{b_u, b_t\} \in \mathbb{R}$ are trainable parameters. The optimization function of the trainable parameter set Θ with an L_2 regularization weighted by λ is

$$J(\Theta) = \sum_{r_{u,t} \in R} (s_{u,t} - \hat{s}_{u,t})^2 + L_2\text{-reg}(\lambda).$$

$J(\Theta)$ is optimized by back-propagation learning methods such as Adam [KB14].

6.3 Experiments

6.3.1 Experimental Setup

Datasets We use seven datasets from Amazon Review Datasets [HM16][2] including AutoMotive (AM), Digital Music (DM), Musical Instruments (MI), Pet Supplies (PS), Sport and Outdoors (SO), Toys and Games (TG), and Tools and Home improvement (TH). Their statistics are shown in Table 6.2.

We use 8:1:1 as the train, validation, and test ratio for all experiments. Users and items with less than 5 reviews and reviews with less than 5 words are removed to reduce data sparsity.

Baseline models Thirteen baselines in *traditional* and *deep learning* categories are compared with the proposed framework. The pre-deep learning traditional approaches predict

[2]`https://jmcauley.ucsd.edu/data/amazon`

Dataset	Abbr.	#Reviews	#Users	#Items	Density	Ttl. #W	#R/U	#R/T	#W/R
AutoMotive	AM	20,413	2,928	1,835	3.419×10^{-3}	1.77M	6.274	10.011	96.583
Digital Music	DM	111,323	14,138	11,707	6.053×10^{-4}	5.69M	7.087	8.558	56.828
Musical Instruments	MI	10,226	1,429	900	$\underline{7.156\times10^{-3}}$	0.96M	6.440	10.226	103.958
Pet Supplies	PS	157,376	19,854	8,510	8.383×10^{-4}	14.23M	7.134	<u>16.644</u>	100.469
Sports and Outdoors	SO	<u>295,434</u>	<u>35,590</u>	<u>18,357</u>	4.070×10^{-4}	<u>26.38M</u>	7.471	14.484	99.199
Toys and Games	TG	167,155	19,409	11,924	6.500×10^{-4}	17.16M	<u>7.751</u>	12.616	114.047
Tools and Home improv.	TH	134,129	16,633	10,217	7.103×10^{-4}	15.02M	7.258	11.815	<u>124.429</u>

Table 6.2: The statistics of the seven real-world datasets. (W: Words; U: Users; T: iTems; R: Reviews.)

ratings solely based upon the entity IDs. Table 6.3 introduces their basic profiles which are extended in Section 6.5.2†. Specially, **AHN-B** refers to AHN using pretrained BERT as the input embedding encoder. It is included to test the impact of the input encoders.

Evaluation metric We use Mean Square Error (MSE) for performance evaluation. Given a test set R_{test}, the MSE is defined by

$$\text{MSE} = \frac{1}{|R_{\text{test}}|} \sum_{(u,r)\in R_{\text{test}}} (\hat{s}_{u,r} - s_{u,r})^2.$$

Reproducibility We provide instructions to reproduce AS-pair extraction of ASPE and rating prediction of baselines and APRE in Section 6.5.1†. The source code of our models is publicly available on GitHub[3].

6.3.2 AS-pair Extraction of ASPE

We present the extraction performance of unsupervised ASPE. The distributions of the frequencies of extracted AS-pairs in Figure 6.5 follow the trend of Zipf's Law with a deviation

[3] `https://github.com/zyli93/ASPE-APRE`

Model	Reference	Cat.	U/T ID	Review
MF	-	Trad.	✓	
WRMF	[HKV08]	Trad.	✓	
FM	[Ren10]	Trad.	✓	
ConvMF	[KPO16]	Deep	✓	✓
NeuMF	[HC17]	Deep	✓	
DeepCoNN	[ZNY17]	Deep		✓
D-Attn	[SHY17]	Deep		✓
NARRE	[CZL18]	Deep	✓	✓
ANR	[CZJ18]	Deep		✓
MPCN	[TLH18]	Deep	✓	✓
DAML	[LLD19]	Deep		✓
AHN	[DNC20]	Deep	✓	✓
AHN-B	Same as **AHN**	Deep	✓	✓

Table 6.3: Compared baselines with inputs marked by "✓". "U" and "T" denote Users and iTems.

common to natural languages [Li92], meaning that ASPE performs consistently across domains. We show the qualitative results of term extraction separately.

Sentiment terms Generally, the AS-pair statistics given in Table 6.4 on different datasets are quantitatively consistent with the data statistics in Table 6.2 regardless of domain. Figure 6.4 is a Venn diagram showing the sources of the sentiment terms extracted by ASPE from AM. All three methods are efficacious and contribute uniquely, which can also be verified by Table 6.5.

We provide Table 6.5 ancillary to the Venn diagram in Figure 6.4 and the corresponding

Data	c	#AS-pairs/R	#A/U	#A/T	#A	#S
AM	50	3.076	12.681	16.284	291	8,572
DM	100	1.973	5.792	8.380	296	9,781
MI	50	3.358	12.521	16.323	167	8,143
PS	150	3.445	14.886	23.893	529	12,563
SO	250	4.078	19.401	28.314	747	17,195
TG	150	4.482	19.053	26.657	680	13,972
TH	150	5.235	22.833	29.816	659	14,145

Table 6.4: Statistics of unsupervised AS-pair extraction. c denotes frequency threshold.

conclusion in Section 6.3.2. We use P (PMI), N (Neural network), and L (Lexicon) to denote the produced sentiment term sets of the three methods, respectively. Operator $\setminus$ denotes set minus, e.g., $P \cap L \setminus N$ refers to the set of terms that are in both P and L but not in N. All sets contain commonly-used sentimental adjectives that can modify automotive items. Table 6.5 illustrates the contributions of the three distinct sentiment term extraction methods discussed in Section 6.2.2, namely PMI-based method, neural network-based method, and lexicon-based method. All three methods can extract useful sentiment-carrying words in the domain of Automotive. Their contributions cannot overwhelm each other, which strongly explains the necessity of the unsupervised methods for term extraction in the domain-general usage scenario. Altogether they provide comprehensive coverage of sentiment terms in AM.

Aspect terms Table 6.6 presents the most frequent aspect terms of all datasets. *ItemTok* is ranked top as users tend to describe overall feelings about items. Domain-specific terms (e.g., *car* in AM) and general terms (e.g., *price*, *quality*, and *size*) are intermingled illustrating the comprehensive coverage and the high accuracy of the result of ASPE.

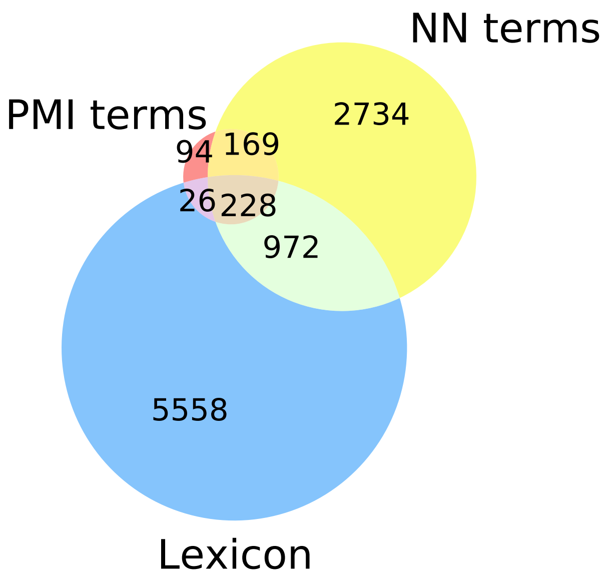


Figure 6.4: Sources of sentiment terms from AM.

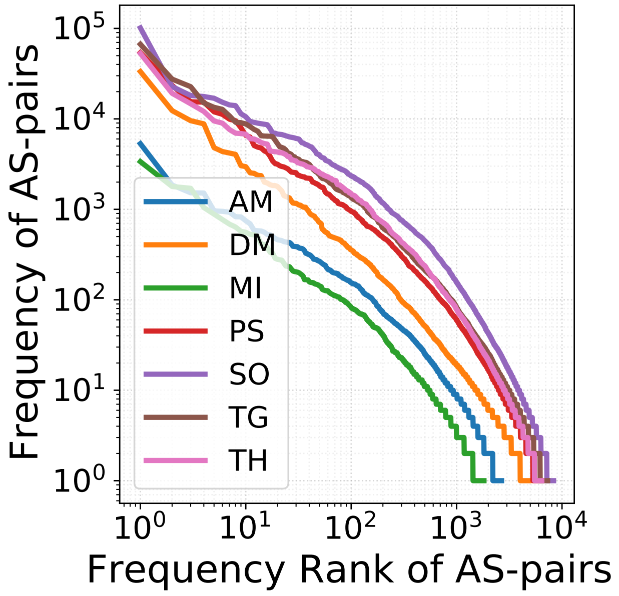


Figure 6.5: Freq. rank vs. frequency of AS-pairs

P only	N only	L only	$P \cap N \backslash L$	$P \cap L \backslash N$	$N \cap L \backslash P$	$P \cap N \cap L$
countless	therapeutic	fateful	ultimate	uplifting	dazzling	amazing
dreamy	vital	poorest	new	concerned	costly	beautiful
edgy	uncanny	tedious	rhythmic	joyful	devastated	classic
entire	adept	unwell	generic	bombastic	faster	delightful
forgettable	fulfilling	joyous	atmospheric	unforgettable	graceful	enjoyable
melodious	attracted	illegal	greater	phenomenal	affordable	fantastic
moral	celestial	noxious	supernatural	inventive	supreme	gorgeous
propulsive	harmonic	lovable	contemporary	classy	robust	horrible
tasteful	newest	crappy	surprising	insightful	useless	inexpensive
uninspired	enduring	arduous	tremendous	masterful	unpredictable	magnificent

Table 6.5: Example sentiment terms of each part of the Venn diagram (Figure 6.4) from AM dataset.

AM	DM	MI	PS	SO	TG	TH
ItemTok	song	*ItemTok*	*ItemTok*	*ItemTok*	*ItemTok*	*ItemTok*
product	*ItemTok*	sound	dog	knife	toy	light
time	album	guitar	food	quality	game	tool
car	music	string	cat	product	piece	quality
look	time	quality	toy	size	quality	price
price	sound	tone	time	price	child	product
quality	voice	price	product	look	color	bulb
light	track	pedal	price	bag	part	battery
oil	lyric	tuner	treat	fit	fun	size
battery	version	cable	water	light	size	flashlight

Table 6.6: High frequency aspects of the corpora.

6.3.3 Rating Prediction of APRE

Comparisons with baselines For the task of review-based rating prediction, a percentage increase **above** 1% in performance is considered significant [CZJ18, TLH18]. According to Table 6.7[4], our model outperforms all baseline models including the AHN-B on all datasets by a minimum of 1.337% on MI and a maximum of 4.061% on TG, which are significant improvements. It demonstrates (1) the superior capability of our model to make accurate rating predictions in different domains (Ours vs. the rest); (2) the performance improvement is NOT because of the use of BERT (Ours vs. AHN-B). AHN-B underperforms the original word2vec-based AHN[5] because the weights of word2vec vectors are trainable while the BERT embeddings are fixed, which reduces the parameter capacity. Within baseline models, deep-learning-based models are generally stronger than entity ID-based traditional methods and

[4]All reported improvements over the best baselines are statistically significant with p-value < 0.01.

[5]The authors of AHN also confirmed this observation.

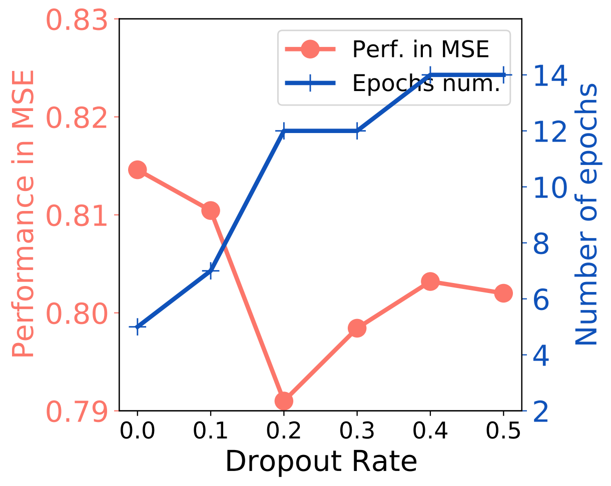
recent ones tend to perform better.

Ablation study Ablation studies answer the question of which channel, explicit or implicit, contributes to the superior performance and to what extent? We measure their contributions by rows of *w/o EX* and *w/o IM* in Table 6.7. *w/o EX* presents the best MSEs of an APRE variant *without explicit* features under the default settings. The impact of AS-pairs is nullified. *w/o IM*, in contrast, shows the best MSEs of an APRE variant only leveraging the explicit channel while removing the implicit one (*without implicit*). We observe that the optimal performances of the single-channel variants all fall behind those of the dual-channel model, which reflects positive contributions from both channels. *w/o IM* has lower MSEs than *w/o EX* on several datasets showing that the explicit channel can supply comparatively more performance improvement than the implicit channel. It also suggests that the costly latent review encoding can be less effective than the aspect-sentiment level user and item profiling, which is a useful finding.

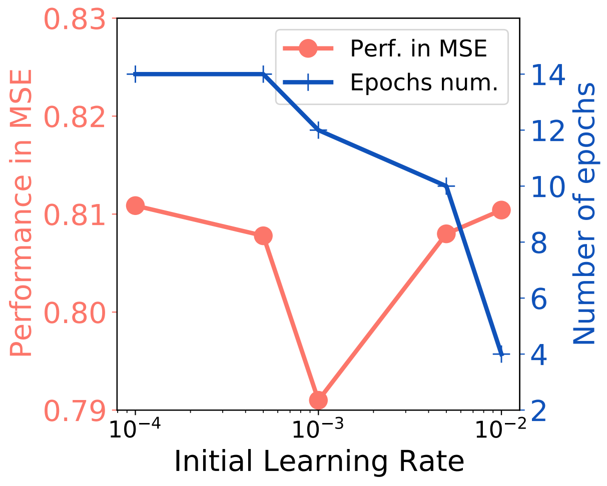
Hyper-parameter sensitivity A number of hyper-parameter settings are of interest, e.g., dropout, learning rate (LR), internal feature dimensions (d_a, d_f, n_c, and n_k), and regularization weight λ of the L_2-reg in $J(\Theta)$. We run each set of experiments on sensitivity search 10 times and report the average performances.

We tune dropout rate in $[0, 0.1, 0.2, 0.3, 0.4, 0.5]$ and LR[6] in $[0.0001, 0.0005, 0.001, 0.005, 0.01]$ with other hyper-parameters set to default, and report in Figure 6.6 the minimum MSEs and the epoch numbers (Ep.) on AM. For dropout, we find the balance of its effects on avoiding overfitting and reducing active parameters at 0.2. Larger dropouts need more training epochs. For LR, we also target a balance between training instability of large LRs and overfitting concern of small LRs, thus 0.001 is selected. Larger LRs plateau earlier with fewer

[6]The reported LRs are initial since Adam and a LR scheduler adjust it dynamically along the training.



(a) Dropout vs. MSE and Ep.



(b) Init. LR vs. MSE and Ep.

Figure 6.6: Hyper-parameter searching and sensitivity - Part 1.

epochs while smaller LRs later with more.

Figure 6.7 analyzes hyper-parameter sensitivities to changes on internal feature dimensions (d_a, d_f, and n_c), CNN kernel size n_k, and λ of L_2-reg weight. We always set $d_f = d_a = n_c$ for the consistency of internal feature dimensions. For (d_f, d_a, n_c) in Figure 6.7a, we choose values from $[50, 100, 150, 200]$ since the output dimension of the BERT encoder is 256. The best performance occurs at 200. The training time spent is stable across different values. CNN kernel size n_k in Figure 6.7b varies in $[4, 6, 8, 10]$. We observe that generally larger kernel sizes may in turn hurt the performance as the local features are fused with larger sequential contexts in natural language. The epoch numbers are stable as well. Figure 6.7c demonstrates how λ affects the performance. As λ becomes larger, the "resistance" against the loss minimization increases so that the training epoch number increases. However, there are no clear trends of performance fluctuation meaning that the sensitivity to L_2-reg weight is insignificant.

Finally, we evaluate the effect of adding non-linearity to embedding adaptation function (EAF) mentioned in Section 6.2.3 which transforms $\mathbf{H}^0$ to $\mathbf{H}^1$ by $\boldsymbol{h}_i^1 = \sigma\left(\mathbf{W}_{\text{ad}}^T \boldsymbol{h}_i^0 + \boldsymbol{b}_{\text{ad}}\right)$.

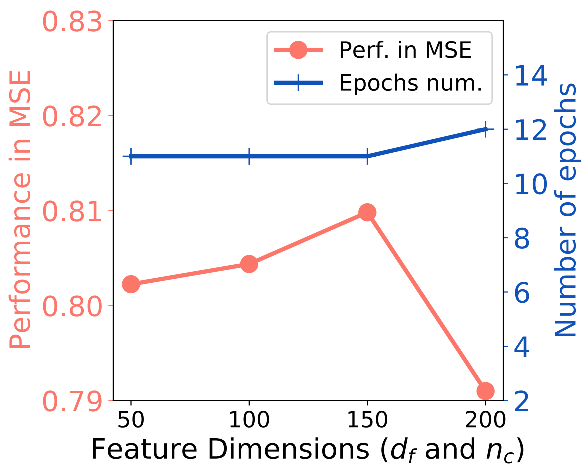
We try LeakyReLU, tanh, and identity functions for $\sigma(\cdot)$ and report the performances in Figure 6.7d. Without non-linear layers, APRE is able to achieve the best results whereas non-linearity speeds up the training.

Efficiency A brief run time analysis of APRE is given in Table 6.8. The model can run fast with all data in GPU memory such as AM and MI denoted by "*", which demonstrates the efficiency of our model and the room for improvement on the run time of datasets that cannot fit in the GPU memory. Other datasets are run on CPU memory. The efficiency of ASPE is less critical since it only runs once for each dataset.

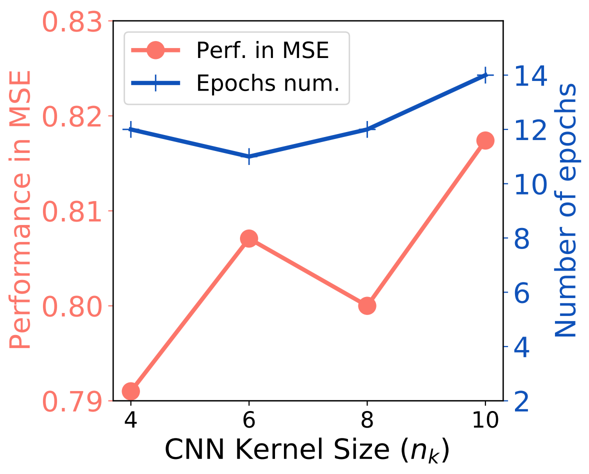
6.3.4 Case Study for Interpretation

Finally, we showcase an interpretation procedure of the rating estimation for an instance in AM: how does APRE predict u_*'s rating for a smart driving assistant t_* using the output AS-pairs of ASPE? We select seven example aspect categories with all review snippets mentioning those categories. Each category is a set of similar aspect terms, e.g., {*look*, *design*} and {*beep*, *sound*}. Without loss of generality, we refer to the categories as aspects. Table 6.9 presents the aspects and review snippets given by u_* and received by t_* with AS-pairs annotations. Three aspects, {*battery*, *install*, *look*}, are shared (yellow rows). Each side has two unique aspects never mentioned by the reviews of the other side: {*materials*, *smell*} of u_* (green rows) and {*price*, *sound*} of t_* (blue rows).

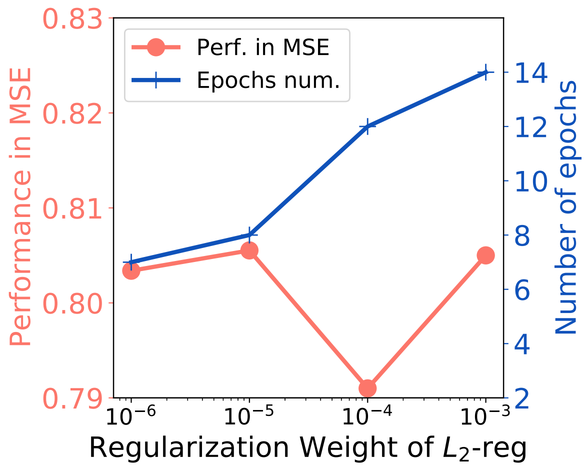
APRE measures the aspect-level contributions of user-attention and item-property interactions by the last term of $s_{u,t}$ prediction, i.e., $\langle \boldsymbol{\gamma}, \mathcal{F}_{\text{ex}}([\mathbf{G}_u; \mathbf{G}_t])\rangle$. The contribution on the ith aspect is calculated by the ith dimension of γ times the ith value of $\mathcal{F}_{\text{ex}}([\mathbf{G}_u; \mathbf{G}_t])$ which is shown in Table 6.10. The top two rows summarize the attentions of u_* and the properties of t_*. *Inferred Impact* states the interactional effects of user attentions and item properties based on our assumption that attended aspects bear stronger impacts to the final prediction. On the overlapping aspects, the inferior property of *battery* produces the only negative



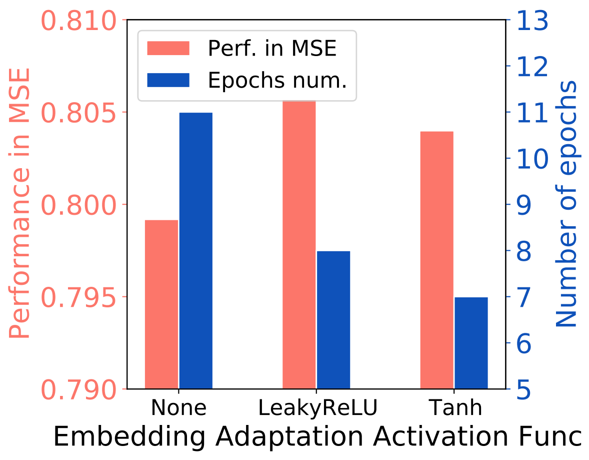
(a) Dims vs. MSE and Ep.



(b) n_k vs. MSE and Ep.



(c) L_2-reg vs. MSE and Ep.



(d) EAF vs. MSE and Ep.

Figure 6.7: Hyper-parameter sensitivity and searching - Part 2.

score (-0.008) whereas the advantages on *install* and *look* create positive scores (0.019 and 0.015), which is consistent with the inferred impact. Other aspects, either *unknown* to user attentions or to item properties, contribute relatively less: t_*'s unappealing *price* accounts for the small score 0.009 and the mixture property of *sound* accounts for the 0.006.

This case study demonstrates the usefulness of the numbers that add up to $\hat{s}_{u,t}$. Although small in scale, they carry significant information of valued or disliked aspects in u_*'s perception of t_*. This process of decomposition is a great way to interpret model prediction on an aspect-level granularity, which is a capacity that other baseline models do not enjoy.

In Section 6.5.3†, another case study indicates that a certain imperfect item property without user attentions only inconsiderably affects the rating although the aspect is mentioned by the user's reviews.

6.4 Summary

In this chapter, we propose a tightly coupled two-stage review-based rating predictor, consisting of an Aspect-Sentiment Pair Extractor (ASPE) and an Attention-Property-aware Rating Estimator (APRE). ASPE extracts aspect-sentiment pairs (AS-pairs) from reviews and APRE learns explicit user attentions and item properties as well as implicit sentence semantics to predict the rating. Extensive quantitative and qualitative experimental results demonstrate that ASPE accurately and comprehensively extracts AS-pairs without using domain-specific training data and APRE outperforms the state-of-the-art recommender frameworks and explains the prediction results taking advantage of the extracted AS-pairs.

Several challenges are left open such as fully or weakly supervised open domain AS-pair extraction and end-to-end design for AS-pair extraction and rating prediction. We leave these problems for future work.

6.5 Supplementary Materials

This section exhibits additional content regarding the experiments such as a detailed experimental setup, the instructions to reproduce the baselines and our model, and another case study.

6.5.1 Reproducibility of ASPE and APRE

ASPE+APRE is implemented in Python (3.6.8) with PyTorch (1.5.0) and run with a single 12GB Nvidia Titan Xp GPU. The code is available on GitHub[7] and comprehensive instructions on how to reproduce our model are also provided. The default hyper-parameter settings for the results in Section 6.3.3 are as follows:

ASPE In the AS-pair extraction stage, we set the size of *ctx* to 5 and the PMI term quota q to 400 for both polarities. The counting thresholds c for different datasets are given in Table 6.4. SDRN [CLW20] utilized for term extraction is trained under the default settings in the source code[8] with the SemEval 14/15 datasets mentioned in Section 2.1.5. `spaCy`[9], a Python package specialized in NLP algorithms, provides the dependency parsing pipeline.

APRE In the rating prediction stage, we use a pre-trained BERT model with 4 layers, 4 heads, and 256 hidden dimensions ("BERT-mini") for manageable GPU memory consumption. The BERT parameters (or weights) are fixed. The BERT tokenizer and model are loaded from the Hugging Face model repository[10]. The initial learning rate is set to 0.001 with two adjusting mechanisms: (1) the Adam optimizer $(\beta_1, \beta_2) = (0.9, 0.999)$ (the default

[7]`https://github.com/zyli93/ASPE-APRE`

[8]`https://github.com/chenshaowei57/SDRN`

[9]`https://spacy.io`

[10]`https://huggingface.co/google/bert_uncased_L-4_H-256_A-4`

setting in PyTorch); (2) a learning rate scheduler, `StepLR`, with step size as 3 and `gamma` as 0.8. Dropout is set to 0.2 for both towers. d_f, d_a, and n_c are all set to 200 for consistency. The CNN kernel size is 4. The L_2-reg weight, λ, is set globally to 0.0001. We use a clamp function to constrain the predictions in the interval $(1.0, 5.0)$.

6.5.2 Baseline information of APRE

We introduce baseline models mentioned in Table 6.3 including the source code of the software and the key parameter settings. For the fairness of comparison, we only compare the models that have **open-source** implementations.

MF, WRMF, FM, and NeuMF[11] Matrix factorization views user-item ratings as a matrix with missing values. By factorizing the matrix with the known values, it recovers the missing values as predictions. Weighted Regularized MF [HKV08] assigns different weights to the values in the matrix. Factorization machines [Ren10] consider additional second-order feature interactions of users and items. Neural MF [HLZ17] is a combination of generalized MF (GMF) and a multilayer perceptron (MLP). Hyper-parameter settings: The number of factors is 200. Regularization weight is 0.0001. We run for 50 epochs with a learning rate of 0.01 with the exception of MI that uses a learning rate of 0.02 for MF and FM. The dropout of NeuMF is set to 0.2.

ConvMF A CNN-based model proposed by [KPO16][12] that utilizes a convolutional neural network (CNN) for feature encoding of text embeddings. Hyper-parameter settings: The regularization factor is 10 for the user model and 100 for the item model. We used a dropout rate of 0.2.

ANR Aspect-based Neural Recommender [CZJ18][13] first proposes aspect-level represen-

[11]Source code of MF, WRMF, FM, and NeuMF is available in `DaisyRec`, an open-source Python Toolkit: `https://github.com/AmazingDD/daisyRec`.

[12]`https://github.com/cartopy/ConvMF`.

[13]`https://github.com/almightyGOSU/ANR`.

tations of reviews but its aspects are completely latent without constraints or definitions on the semantics. Hyper-parameter settings: L_2 regularization is 1×10^{-6}. Learning rate is 0.002. Dropout rate is 0.5. We used 300-dimensional pretrained Google News word embeddings.

DeepCoNN DeepCoNN [ZNY17][14] separately encodes user reviews and item reviews by complex neural networks. Hyper-parameter settings: Learning rate is 0.002 and dropout rate is 0.5. Word embedding is the same as ANR.

NARRE A model similar to DeepCoNN enhanced by attention mechanism [CZL18]. Attentional weights are assigned to each review to measure its importance. Hyper-parameter settings: L_2 regularization weight is 0.001 Learning rate is 0.002. Dropout rate is 0.5. We used the same word embeddings as described for ANR.

D-Attn[15] Dual attention-based model [SHY17] utilizes CNN as text encoders and builds local- and global-attention (dual attention) for user and item reviews. Hyper-parameter settings: In accordance with the paper, we used 100-dimensional word embedding. The factor number is 200. Dropout rate is 0.5. Learning rate and regularization weight are both 0.001.

MPCN Multi-Pointer Co-Attention Network [TLH18] selects a useful subset of reviews by pointer networks to build the user profile for the current item. Hyper-parameter settings are the same as D-Attn except that the dropout is 0.2.

DAML DAML [LLD19] forces encoders of the user and item reviews to interchange information in the fusion layer with local- and mutual- attention so that the encoders can mutually guide the representation generation. Hyper-parameter settings are the same as MPCN.

[14]Source code of DeepCoNN and NARRE: `https://github.com/chenchongthu`.

[15]Source code of D-Attn, MPCN, and DAML: `https://github.com/ShomyLiu/Neu-Review-Rec`

AHN Asymmetrical Hierarchical Networks [DNC20][16] that guide the user representation generation using item side asymmetric attentive modules so that only relevant targets are significant. Experiments are reproduced following the settings in the paper.

6.5.3 Case Study II for Interpretation

Finally, we show another case study from AM dataset using the same attention-property-score visualization schema as Section 6.3.4. In this case, our model is predicting the score user u_* will give to a color and clarity compound for vehicle surface t_*. The mentioned aspects of u_* and the properties of t_* are given in Table 6.11 including three overlapping aspects (*quality*, *look*, *cleaning*) and one unique aspect of each side (*size* of u_* and *smell* of t_*). A summarization table, Table 6.12, shows the summarized attentions and properties, the inferred impacts, and the corresponding score components of $\langle \boldsymbol{\gamma}, \mathcal{F}_{\text{ex}}([\mathbf{G}_u; \mathbf{G}_t]) \rangle$.

In this case study, we can observe the interesting phenomenon also exemplified in Table 6.1 by the contrast between R1 and R3 that the aspect *look*, which has been mentioned by u_* and reviewed negatively as a property of t_* (*"strange yellow color"*), only produces an inconsiderable bad effect (-0.002) on the final score prediction. This indicates that the imperfect look (or color) of the item, although also mentioned by u_* in his/her reviews, receives little attention from u_* and thus poses a tiny negative impact on the predicted rating decision of the user. The other two overlapping aspects show intuitive correlations between their inferred impacts and the scores. The unique aspects, *size* and *smell*, have relatively small influences on the prediction because they are either not attended aspects or not mentioned properties.

It is also notable that some sentences that carry strong emotions may contain few explicit sentiment mentions, e.g., "*But for an all in one cleaner and wax I think this outperforms most.*" It backs the design of APRE which carefully takes implicit sentiment signals into

[16]`https://github.com/Moonet/AHN`

consideration, and also calls for an advanced way for aspect-based sentiment modeling beyond term level. Different proportions of such sentences in different datasets may account for the inconsistency of better performances between the two variants of the ablation study.

Models	AM	DM	MI	PS	SO	TG	TH
TRADITIONAL MODELS							
MF	1.986	1.715	2.085	2.048	2.084	1.471	1.631
WRMF	1.327	0.537	1.358	1.629	1.371	1.068	1.216
FM	1.082	0.436	1.146	1.458	1.212	0.922	1.050
DEEP LEARNING-BASED MODELS							
ConvMF	1.046	0.407	1.075	1.458	1.026	0.986	1.104
NeuMF	0.901	0.396	0.903	1.294	0.893	0.841	1.072
D-Attn	0.816	0.403	0.835	1.264	0.897	0.887	0.980
D-CNN	0.809	0.390	0.861	1.250	0.894	0.835	0.975
NARRE	0.826	0.374	0.837	1.425	0.990	0.908	<u>0.958</u>
MPCN	0.815	0.447	0.842	1.300	0.929	0.898	0.969
ANR	0.806	0.381	0.845	1.327	0.906	0.844	0.981
DAML	0.829	<u>0.372</u>	0.837	<u>1.247</u>	0.893	<u>0.820</u>	0.962
AHN-B	0.810	0.385	0.840	1.270	0.896	0.829	0.976
AHN	<u>0.802</u>	0.376	<u>0.834</u>	1.252	<u>0.887</u>	0.822	0.967
OUR MODELS AND PERCENTAGE IMPROVEMENTS							
Ours	**0.791**	**0.359**	**0.823**	**1.218**	**0.863**	**0.788**	**0.936**
Δ(%)	1.390	3.621	1.337	2.381	2.784	4.061	2.350
Val.	0.790	0.362	0.821	1.216	0.860	0.790	0.933
ABLATION STUDIES							
w/o EX	0.814	0.379	0.833	1.244.	0.882	0.796	0.965
w/o IM	0.798	0.374	0.863	1.226	0.873	0.798	0.956

Table 6.7: MSE of baselines, our model (**Ours** for test and **Val.** for validation), and variants. The row of Δ calculates the percentage improvements over the <u>best baselines</u>.

AM	DM	MI	PS	SO	TG	TH
127s*	31min	90s*	36min	90min	51min	35min

Table 6.8: Per epoch run time of APRE on the seven datasets.

From reviews given by user u_*. All aspects attended (✓).	
battery	[**To** t_1] After leaving this attached to my car for two days of non-use I have a ***dead*** *battery*. Never had a ***dead*** *battery* . . . , so *I am blaming this device.*
install	[**To** t_2] *This* was ***unbelievably easy*** to install. I have done . . . . The real key . . . the *installation* is so ***easy***. [**To** t_3] There were ***many*** *installation options*, but once . . . , *they clicked on easily.*
look	[**To** t_3] It was *not perfect and not shiny*, but it did *look* ***better***. [**To** t_4] It takes some ***elbow*** *grease*, but the *results are remarkable.*
material	[**To** t_5] *The plastic* however is ***very thin*** and *the cap* is ***pretty cheap***. [**To** t_6] Great value. . . . . They are ***very hard*** *plastic*, so they don't mark up panels.
smell	[**To** t_7] This has a ***terrible*** *smell* that really lingers awhile. It goes on green. . . .
From reviews received by item t_*.	
battery	[**From** u_1] The reason this won't work on an iPhone 4 or . . . because it uses ***low*** *power* Bluetooth, ... (✗)
install	[**From** u_2] Your mileage and gas mileage and cost of fuel is tabulated for each trip-*Installation* is ***pretty simple*** - but it . . . . (✓)
look	[**From** u_3] Driving habits, fuel efficiency, and engine health are ***nice*** *features*. The *overall design* is ***nice*** and *easy to navigate.* (✓)
price	[**From** u_4] In fact, there are similar products to this available at a ***much lower*** *price* that do work with . . . (✗)
sound	[**From** u_5] The Link device makes an ***audible*** *sound* when you go over 70 mpg, brake hard, or accelerate too fast. (✓) [**From** u_6] Also, the *beep* the link device makes . . . sounds ***really cheapy***. (✗)

Table 6.9: Examples of reviews given by u_* and received by t_* for Case Study I with *Aspect*-***Sentiment*** pair mentions as well as *other sentiment evidences* on seven example aspects.

Aspects	material	smell	battery	install	look	price	sound
Attn. of u_*	✓	✓	✓	✓	✓	n/a	n/a
Prop. of t_*	n/a	n/a	✗	✓	✓	✗	✓/✗
Inferred Impact	*Unk.*	*Unk.*	***Neg.***	***Pos.***	***Pos.***	*Unk.*	*Unk.*
$\gamma_i \mathcal{F}_{\text{ex}}(\cdot)_i$ $(\times 10^{-2})$	1.0	0.8	-0.8	1.9	1.5	0.9	0.6

Table 6.10: Attentions and properties summaries, inferred impacts, and the learned aspect-level contributions for Case Study I.

	From reviews given by user u_*.
quality	[**To** t_1] As soon as I poured it into the bucket and started getting ready, I can tell the product was already ***better*** *quality* than my previous washing liquid.
look	[**To** t_4] I bought [this item] because I had neglected my paint job for too long. . . . it made my black *paint job* look ***dull***.
cleaning	[**To** t_2] . . . *I was able to dry my car in record time and not have any water marks left on the paint.* I just slide the towel over any parts with water and *it left no trace of water* and a ***clean*** *shine* to my car. [**To** t_3] I had completely neglected these areas, except for *minor* ***cleaning*** and protection. *Once I applied it, the difference was night and day!*
size	[**To** t_6] The *size* was ***great*** as well, *allowing me to get larger areas in an easier amount of time* so that I could wash my car quicker than I have in the past.
	From reviews received by item t_*.
quality	[**From** u_1] Adding too little soap will increase the tendency . . . This thick, ***high*** *quality* soap helps prevent against that. (✓) [**From** u_2] . . . Cons: A bit pricey, but *quality matters*, and this product absolutely has it. *Worth every cent for sure! (✓)*
look	[**From** u_3] I was a bit ***disappointed***. It is a ***strange yellow*** *color* and it is thick and I personally did not care for the smell. (✗)
cleaning	[**From** u_4] As far as *cleaning power* it does ***fairly good***, . . . The best cleaning of a car is in steps, but *for an all in one cleaner and wax I think this outperforms most.* (✓)
smell	[**From** u_5] Just giving some useful feedback about the truth behind the product . . . that it *smells* ***good***. [**From** u_6] *I believe this preserves the wax layer longer* . . . This is much thicker than the [some brand] soap, and has a very *pleasant* ***smell*** to it. (✓)

Table 6.11: Examples of reviews given by u_* and received by t_* for Case Study II with *Aspect*-***Sentiment*** pair mentions as well as *other sentiment evidences* on five example aspects.

Aspects	size	quality	look	cleaning	smell
Attn. of u_*	✓	✓	–	✓	n/a
Prop. of t_*	n/a	✓	✗	✓	✓
Inferred Impact	*Unk.*	***Pos.***	***Neg.***	***Pos.***	*Unk.*
$\gamma_i \mathcal{F}_{\text{ex}}(\cdot)_i$ $(\times 10^{-2})$	0.5	2.9	-0.2	1.4	0.3

Table 6.12: Attentions and properties summaries, inferred impacts, and the learned aspect-level contributions on the score prediction for Case Study II.

CHAPTER 7

Classifying Comparative Statements in Reviews

In addition to the descriptive statements in reviews which can be processed by the ASPE-APRE framework, comparative statements are left unattended. We study Comparative Preference Classification (CPC) which aims at predicting whether a preference comparison exists between two entities in a given sentence and, if so, which entity is preferred over the other. High-quality CPC models can significantly benefit applications such as comparative question answering and review-based recommendation. Among the existing approaches, non-deep learning methods suffer from inferior performances. The state-of-the-art graph neural network-based ED-GAT [MMW20] only considers syntactic information while ignoring the critical semantic relations and the sentiments to the compared entities. We propose Sentiment Analysis Enhanced COmparative Network (SAECON) which improves CPC accuracy with a sentiment analyzer that learns sentiments to individual entities via domain adaptive knowledge transfer. Experiments on the CompSent-19 [PBF19] dataset present a significant improvement on the F1 scores over the best existing CPC approaches.

7.1 Motivation and Background

Comparative preference classification (CPC) is a natural language processing (NLP) task that predicts whether a preference comparison exists between two entities in a sentence and, if so, which entity wins the game. For example, given the sentence: *Python is better suited for data analysis than MATLAB due to the many available deep learning libraries*, a decisive comparison exists between *Python* and *MATLAB* and comparatively *Python* is preferred over *MATLAB* in the context.

The CPC task can profoundly impact various real-world application scenarios. Search engine users may query not only factual questions but also comparative ones to meet their specific information needs [GMR17]. Recommendation providers can analyze product reviews with comparative statements to understand the advantages and disadvantages of the product comparing with similar ones.

Several models have been proposed to solve this problem. [PBF19] first formalize the CPC problem, build and publish the CompSent-19 dataset, and experiment with numerous general machine learning models such as Support Vector Machine (SVM), representation-based classification, and XGBoost. However, these attempts consider CPC as a sentence classification while ignoring the semantics and the contexts of the entities [MMW20].

ED-GAT [MMW20] marks the first entity-aware CPC approach that captures long-distance *syntactic* relations between the entities of interest by applying graph attention networks (GAT) to dependency parsing graphs. However, we argue that the disadvantages of such an approach are clear. Firstly, ED-GAT replaces the entity names with "entityA" and "entityB" for simplicity and hence deprives their *semantics*. Secondly, ED-GAT has a deep architecture with ten stacking GAT layers to tackle the long-distance issue between compared entities. However, more GAT layers result in a heavier computational workload and reduced training stability. Thirdly, although the competing entities are typically connected via multiple hops of dependency relations, the unordered tokens along the connection path cannot capture either global or local high-quality semantic context features.

In this work, we propose a Sentiment Analysis Enhanced COmparative classification Network (SAECON), a CPC approach that considers not only syntactic but also semantic features of the entities. The semantic features here refer to the context of the entities from which a sentiment analysis model can infer the sentiments toward the entities. Specifically, the encoded sentence and entities are fed into a dual-channel context feature extractor to learn the global and local context. In addition, an auxiliary Aspect-Based Sentiment Analysis (ABSA) module is integrated to learn the *sentiments* towards individual entities which are greatly beneficial to the comparison classification.

ABSA aims to detect the specific emotional inclination toward an aspect within a sentence [MPC18, HPH19, PO20, CQ20, WSY20]. For example, the sentence *I liked the service and the staff but not the food* suggests positive sentiments toward service and staff but a negative one toward food. These *aspect* entities, such as service, staff, and food, are studied individually.

The well-studied ABSA approaches can be beneficial to CPC when the compared entities in a CPC sentence are considered as the aspects in ABSA. Incorporating the individual sentiments learned by ABSA methods into CPC has several advantages. Firstly, for a comparison to hold, the preferred entity usually receives a positive sentiment while its rival gets a relatively negative one. These sentiments can be easily extracted by the strong ABSA models. The contrast between the sentiments assigned to the compared entities provides a vital clue for an accurate CPC. Secondly, the ABSA models are designed to target the sentiments toward phrases, which bypasses the complicated and noisy syntactic relation path. Thirdly, considering the scarcity of the data resource of CPC, the abundant annotated data of ABSA can provide sufficient supervision signal to improve the accuracy of CPC.

There is one challenge that blocks the knowledge transfer of sentiment analysis from the ABSA data to the CPC task: *domain shift*. Existing ABSA datasets are centered around specific topics such as restaurants and laptops, while the CPC data has mixed topics [PBF19] that are all distant from restaurants. In other words, sentences of ABSA and CPC datasets are drawn from different distributions, also known as *domains*. The difference in the distributions is referred to as a domain shift [GL15, HLN18] and it is harmful to an accurate knowledge transfer. To mitigate the domain shift, we design a domain adaptive layer to remove the domain-specific feature such as topics and preserve the domain-invariant feature such as sentiments of the text so that the sentiment analyzer can smoothly transfer knowledge from sentiment analysis to comparative classification.

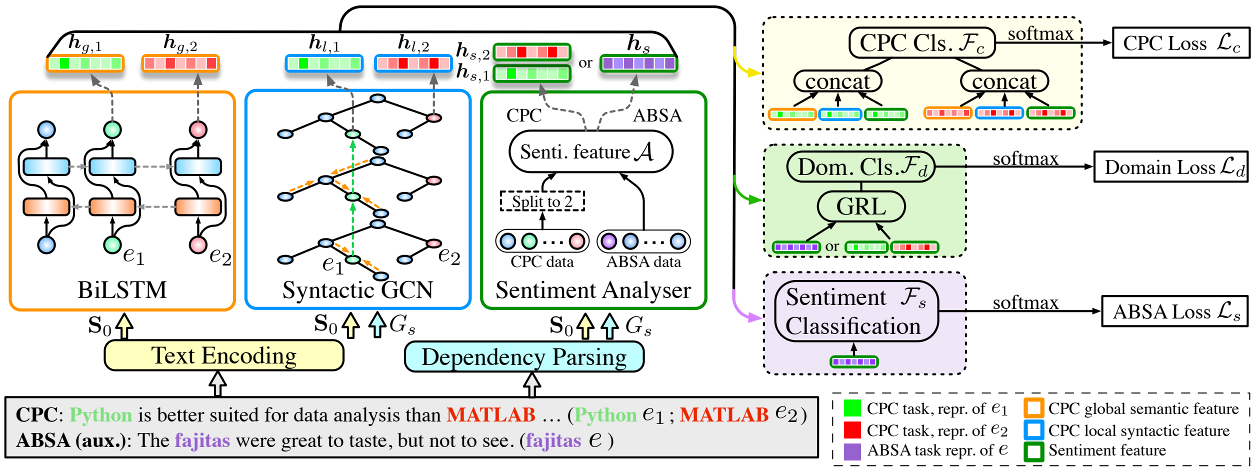


Figure 7.1: Pipeline of SAECON. "Cls." is short for classifier.

7.2 SAECON

In this section, we first formalize the problem and then explain SAECON in detail. The pipeline of SAECON is depicted in Figure 7.1 with essential notations. Sentences from two domains shown in the gray box are fed into text encoder and dependency parser. The resultant representations of the entities from three substructures are discriminated by different colors shown in the legend.

7.2.1 Problem Statement

CPC Given a sentence s from the CPC corpus D_c with n tokens and two entities e_1 and e_2, a CPC model predicts whether there exists a preference comparison between e_1 and e_2 in s and if so, which entity is preferred over the other. Potential results can be `Better` (e_1 wins), `Worse` (e_2 wins), or `None` (no comparison exists).

ABSA Given a sentence s' from the ABSA corpus D_s with m tokens and one entity e', ABSA identifies the sentiment (positive, negative, or neutral) associated with e'.

We denote the source domains of the CPC and ABSA datasets by $\mathcal{D}_c$ and $\mathcal{D}_s$. D_c and D_s contain samples that are drawn from $\mathcal{D}_c$ and $\mathcal{D}_s$, respectively. $\mathcal{D}_c$ and $\mathcal{D}_s$ are similar but different in topics which produces a domain shift. We use s to denote sentences in $D_c \cup D_s$ and E to denote the entity sets for simplicity in later discussion. $|E| = 2$ if $s \in D_c$ and $|E| = 1$ otherwise.

7.2.2 Text Feature Representations

A sentence is encoded by its word representations via a text encoder and parsed into a dependency graph via a dependency parser [CM14]. Text encoder, such as GloVe [PSM14] and BERT [DCL19], maps a word w into a low dimensional embedding $\boldsymbol{w} \in \mathbb{R}^{d_0}$. GloVe assigns a fixed vector while BERT computes a token[1] representation by its textual context. The encoding output of s is denoted by $\mathbf{S}_0 = \{\boldsymbol{w}_1, \ldots, \boldsymbol{e}_1, \ldots, \boldsymbol{e}_2, \ldots, \boldsymbol{w}_n\}$ where $\boldsymbol{e}_i$ denotes the embedding of entity i, $\boldsymbol{w}_i$ denotes the embedding of a non-entity word, and $\boldsymbol{w}_i, \boldsymbol{e}_j \in \mathbb{R}^{d_0}$.

The dependency graph of s, denoted by G_s, is obtained by applying a dependency parser to s such as Stanford Parser [CM14] or spaCy[2]. G_s is a syntactic view of s [MT17, LTB16]

[1]BERT generates representations of wordpieces which can be substrings of words. If a word is broken into wordpieces by BERT tokenizer, the average of the wordpiece representations is taken as the word representation. The representations of the special tokens of BERT, `[CLS]` and `[SEP]`, are not used.

[2]`https://spacy.io`

that is composed of vertices of words and directed edges of dependency relations. Advantageously, complex syntactic relations between distant words in the sentence can be easily detected with a small number of hops over dependency edges [MMW20].

7.2.3 Contextual Features for CPC

Global Semantic Context To model more extended context of the entities, we use a bi-directional LSTM (BiLSTM) to encode the entire sentence in both directions. Bi-directional recurrent neural network is widely used in extracting semantics [LJS19]. Given the indices of e_1 and e_2 in s, the global context representations $\boldsymbol{h}_{g,1}$ and $\boldsymbol{h}_{g,2}$ are computed by averaging the hidden outputs from both directions.

$$\begin{aligned}\overrightarrow{\boldsymbol{h}_{g,i}}, \overleftarrow{\boldsymbol{h}_{g,i}} &= \text{BiLSTM}(\mathbf{S}_0)[e_i.\text{index}], \quad i = 1, 2 \\ \boldsymbol{h}_{g,i} &= \frac{1}{2}\left(\overrightarrow{\boldsymbol{h}_{g,i}} + \overleftarrow{\boldsymbol{h}_{g,i}}\right), \boldsymbol{h}_{g,i} \in \mathbb{R}^{d_g}.\end{aligned}$$

Local Syntactic Context In SAECON, we use a dependency graph to capture the syntactically neighboring context of entities that contains words or phrases modifying the entities and indicates comparative preferences. We apply a Syntactic Graph Convolutional Network (SGCN) [BTA17, MT17] to G_s to compute the local context feature $\boldsymbol{h}_{l,1}$ and $\boldsymbol{h}_{l,2}$ for e_1 and e_2, respectively. SGCN operates on directed dependency graphs with three major adjustments compared with GCN [KW17]: considering the directionality of edges, separating parameters for different dependency labels[3], and applying edge-wise gating to message passing.

GCN is a multilayer message propagation-based graph neural network. Given a vertex v in G_s and its neighbors $\mathcal{N}(v)$, the vertex representation of v on the $(j+1)$th layer is given as

$$\boldsymbol{h}_v^{(j+1)} = \rho\left(\sum_{u \in \mathcal{N}(v)} \mathbf{W}^{(j)} \boldsymbol{h}_u^{(j)} + \boldsymbol{b}^{(j)}\right),$$

[3]Labels are defined as the combinations of directions and dependency types. For example, edge $((u, v)$, `nsubj`$)$ and edge $((v, u)$, $\texttt{nsubj}^{-1})$ have different labels.

where $\rho(\cdot)$ denotes an aggregation function such as mean and sum, $\mathbf{W}^{(j)} \in \mathbb{R}^{d^{(j+1)} \times d^{(j)}}$ and $\boldsymbol{b}^{(j)} \in \mathbb{R}^{d^{(j+1)}}$ are trainable parameters, and $d^{(j+1)}$ and $d^{(j)}$ denote latent feature dimensions of the $(j+1)$th and the jth layers, respectively.

SGCN improves GCN by considering different edge directions and diverse edge types, and assigns different parameters to different directions or labels. However, there is one caveat: the directionality-based method cannot accommodate the rich edge type information; the label-based method causes combinatorial over-parameterization, increased risk of overfitting, and reduced efficiency. Therefore, we naturally arrive at a trade-off of using direction-specific weights and label-specific biases.

The edge-wise gating can select impactful neighbors by controlling the gates for message propagation through edges. The gate on the jth layer of an edge between vertices u and v is defined as

$$g_{uv}^{(j)} = \sigma\left(\boldsymbol{h}_u^{(j)} \cdot \boldsymbol{\beta}_{d_{uv}}^{(j)} + \gamma_{l_{uv}}^{(j)}\right), \quad g_{uv}^{(j)} \in \mathbb{R},$$

where d_{uv} and l_{uv} denote the direction and label of edge (u, v), $\boldsymbol{\beta}_{d_{uv}}^{(j)}$ and $\gamma_{l_{uv}}^{(j)}$ are trainable parameters, and $\sigma(\cdot)$ denotes the sigmoid function.

Summing up the aforementioned adjustments on GCN, the final vertex representation learning is

$$\boldsymbol{h}_v^{(j+1)} = \rho\left(\sum_{u \in \mathcal{N}(v)} g_{uv}^{(j)} \left(\mathbf{W}_{d_{uv}}^{(j)} \boldsymbol{h}_u^{(j)} + \boldsymbol{b}_{l_{uv}}^{(j)}\right)\right).$$

Vectors of $\mathbf{S}_0$ serve as the input representations $\boldsymbol{h}_v^{(0)}$ to the first SGCN layer. The representations corresponding to e_1 and e_2 are the output $\{\boldsymbol{h}_{l,1}, \boldsymbol{h}_{l,2}\}$ with dimension d_l.

7.2.4 Sentiment Analysis with Knowledge Transfer from ABSA

We have discussed in Section 7.1 that ABSA inherently correlates with the CPC task. Therefore, it is natural to incorporate a sentiment analyzer into SAECON as an auxiliary task to take advantage of the abundant training resources of ABSA to boost the performance on CPC. There are two paradigms for auxiliary tasks: (1) incorporating fixed parameters that are pretrained solely with the auxiliary dataset; (2) incorporating the architecture only with untrained parameters and jointly optimizing them from scratch with the main task simultaneously [LWZ18, HLN18, WP18].

Option (1) ignores the domain shift between D_c and D_s, which degrades the quality of the learned sentiment features since the domain identity information is noisy and unrelated to the CPC task. SAECON uses option (2). For a smooth and efficient knowledge transfer from $\mathcal{D}_s$ to $\mathcal{D}_c$ under the setting of option (2), the ideal sentiment analyzer only extracts the textual feature that is contingent on sentimental information but orthogonal to the identity of the source domain. In other words, the learned sentiment features are expected to be *discriminative* on sentiment analysis but *invariant* with respect to the domain shift. Therefore, the sentiment features are more aligned with the CPC domain $\mathcal{D}_c$ with reduced noise from domain shift.

In SAECON, we use a *gradient reversal layer* (GRL) and a *domain classifier* (DC) [GL15] for the domain adaptive sentiment feature learning that maintains the discriminativeness and the domain-invariance. GRL+DC is a straightforward, generic, and effective modification to neural networks for domain adaptation [KGC19, GFL19, BPS19, LWZ18]. It can effectively close the shift between complex distributions [GL15] such as $\mathcal{D}_c$ and $\mathcal{D}_s$.

Let $\mathcal{A}$ denote the sentiment analyzer which alternatively learns sentiment information from D_s and provides sentimental clues to the compared entities in D_c. Specifically, each CPC instance is split into two ABSA samples with the same text before being fed into $\mathcal{A}$ (see the "Split to 2" in Figure 7.1). One takes e_1 as the queried aspect the other takes e_2.

$$\mathcal{A}(\mathbf{S}_0, G_s, E) = \begin{cases} \boldsymbol{h}_{s,1}, \boldsymbol{h}_{s,2} & \text{if } s \in D_c, \\ \boldsymbol{h}_s & \text{if } s \in D_s. \end{cases}$$

$\boldsymbol{h}_{s,1}$, $\boldsymbol{h}_{s,2}$, and $\boldsymbol{h}_s \in \mathbb{R}^{d_s}$. These outputs are later sent through a GRL to not only the CPC and ABSA predictors shown in Figure 7.1 but also the DC to predict the source domain y_d

of s where $y_d = 1$ if $s \in D_s$ otherwise 0. GRL, trainable by backpropagation, is transparent in the forward pass ($\mathrm{GRL}_\alpha(\boldsymbol{x}) = \boldsymbol{x}$). It reverses the gradients in the backward pass as

$$\frac{\partial \mathrm{GRL}_\alpha}{\partial \boldsymbol{x}} = -\alpha \mathbf{I}.$$

Here $\boldsymbol{x}$ is the input to GRL, α is a hyperparameter, and $\mathbf{I}$ is an identity matrix. During training, the reversed gradients maximize the domain loss, forcing $\mathcal{A}$ to forget the domain identity via the backpropagation and mitigating the domain shift. Therefore, the outputs of $\mathcal{A}$ stay invariant to the domain shift. But as the outputs of $\mathcal{A}$ are also optimized for ABSA predictions, the distinctiveness with respect to sentiment classification is retained.

Finally, the selection of $\mathcal{A}$ is flexible as it is architecture-agnostic. In this work, we use the LCF-ASC aspect-based sentiment analyzer proposed by [PO20] in which two scales of representations are concatenated to learn the sentiments to the entities of interest.

7.2.5 Objective and Optimization

SAECON optimizes three classification errors overall for CPC, ABSA, and domain classification. For CPC task, features for local context, global context, and sentiment are concatenated: $\boldsymbol{h}_{e_i} = [\boldsymbol{h}_{g,i}; \boldsymbol{h}_{l,i}; \boldsymbol{h}_{s,i}]$, $i \in \{1, 2\}$, and $\boldsymbol{h}_{e_i} \in \mathbb{R}^{d_s+d_g+d_l}$. Given $\mathcal{F}_c$, $\mathcal{F}_s$, $\mathcal{F}_d$, and $\mathcal{F}$ below denoting fully-connected neural networks with non-linear activation layers, CPC, ABSA, domain predictions are obtained by

$$\begin{aligned} \hat{y}_c &= \delta(\mathcal{F}_c([\mathcal{F}(\boldsymbol{h}_{e_1}); \mathcal{F}(\boldsymbol{h}_{e_2})])) && \text{(CPC only)}, \\ \hat{y}_s &= \delta(\mathcal{F}_s(\boldsymbol{h}_s)) && \text{(ABSA only)}, \\ \hat{y}_d &= \delta(\mathcal{F}_d(\mathrm{GRL}(\mathcal{A}(\mathbf{S}_0, G_s, E)))) && \text{(Both tasks)}, \end{aligned}$$

where δ denotes the softmax function. With the predictions, SAECON computes the cross entropy losses for the three tasks as $\mathcal{L}_c$, $\mathcal{L}_s$, and $\mathcal{L}_d$, respectively. The label of $\mathcal{L}_d$ is y_d. The computations of the losses are omitted due to the space limit.

In summary, the objective function of the proposed model SAECON is given as follows,

$$\mathcal{L} = \mathcal{L}_c + \lambda_s \mathcal{L}_s + \lambda_d \mathcal{L}_d + \lambda \text{reg}(L2),$$

where λ_s and λ_d are two weights of the losses, and λ is the weight of an $L2$ regularization. We denote $\boldsymbol{\lambda} = \{\lambda_s, \lambda_d, \lambda\}$. In the actual training, we separate the iterations of CPC data and ABSA data and input batches from the two domains alternatively. Alternative inputs ensure that the DC receives batches with different labels evenly and avoid overfitting to either domain label. A stochastic gradient descent based optimizer, Adam [KB15], is leveraged to optimize the parameters of SAECON. Algorithm 2 explains the alternative training paradigm in detail.

7.3 Experiments

7.3.1 Experimental Settings

Dataset CompSent-19 is the first public dataset for the CPC task released by [PBF19]. It contains sentences with entity annotations. The ground truth is obtained by comparing the entity that appears earlier (e_1) in the sentence with the one that appears later (e_2). The dataset is split by convention [PBF19, MMW20]: 80% for training and 20% for testing. During training, 20% of the training data of each label composes the development set for model selection. The detailed statistics are given in Table 7.1.

Three datasets of restaurants released in SemEval 2014, 2015, and 2016 [PGP14, PGP15, XTP16] are utilized for the ABSA task. We join their training sets and randomly sample instances into batches to optimize the auxiliary objective $\mathcal{L}_s$. The proportions of `POS`, `NEU`, and `NEG` instances are 65.8%, 11.0%, and 23.2%.

Note. The rigorous definition of `None` in CompSent-19 is that the sentence *does not contain* a comparison between the entities rather than that entities are both preferred or disliked. Although the two definitions are not mutually exclusive, we would like to provide

Algorithm 2: Optimization of SAECON with two alternative tasks

```
Input: Loss weights $\boldsymbol{\lambda}$; Learning rate $\eta$
Data: $D_c$ and $D_s$.
1  while not converge do
2      $\{s, E\}$, task ← getAltSample($D_c$, $D_s$)
3      $\mathbf{S}_0 \leftarrow$ TextEncode($s$)
4      $G_s \leftarrow$ DepParse($s$)
5      if task is CPC then
6          $\{\boldsymbol{h}_{g,i}\}, \{\boldsymbol{h}_{l,i}\}$ $(i = 1, 2) \leftarrow$ methods in Section 7.2.3.
7          $\boldsymbol{h}_{s,1}, \boldsymbol{h}_{s,2} \leftarrow \mathcal{A}(\mathbf{S}_0, G_s, E)$
8          $\mathcal{L}_c, \mathcal{L}_d \leftarrow$ methods in Section 7.2.5
9          optimize($\{\mathcal{L}_c, \lambda_d \mathcal{L}_d, \lambda \text{reg}(L2)\}, \eta$)
10     else
           // sentiment analysis
11         $\boldsymbol{h}_s \leftarrow \mathcal{A}(\mathbf{S}_0, G_s, E)$
12         $\mathcal{L}_s, \mathcal{L}_d \leftarrow$ methods in Section 7.2.5
13         optimize($\{\lambda_s \mathcal{L}_s, \lambda_d \mathcal{L}_d, \lambda \text{reg}(L2)\}, \eta$)
```

a clearer background of the CPC problem.

Imbalanced Data CompSent-19 is badly imbalanced (see Table 7.1). `None` instances dominate in the dataset. The other two labels combined only account for 27%. This critical issue can impair the model performance. Three methods to alleviate the imbalance are tested. *Flipping labels*: Consider the order of the entities, an original `Better` instance will become a `Worse` one and vice versa if querying (e_2, e_1) instead. We interchange the e_1 and e_2 of all `Better` and `Worse` samples so that they have the same amount. *Upsampling*: We upsample `Better` and `Worse` instances with duplication to the same amount of `None`. *Weighted loss*: We upweight the underpopulated labels `Better` and `Worse` when computing

Dataset	`Better`	`Worse`	`None`	**Total**
Train	872 (19%)	379 (8%)	3,355 (73%)	4,606
Development	219 (19%)	95 (8%)	839 (73%)	1,153
Test	273 (19%)	119 (8%)	1,048 (73%)	1,440
Total	1,346 (19%)	593 (8%)	5,242 (73%)	7,199
Flipping labels	1,251 (21%)	1,251 (21%)	3,355 (58%)	5,857
Upsampling	3,355 (33%)	3,355 (33%)	3,355 (33%)	10,065

Table 7.1: Statistics of CompSent-19. The rows of *Flipping labels* and *Upsampling* show the numbers of the augmented datasets to mitigate label imbalance.

the classification loss. Their effects are discussed in Section 7.3.2.

Evaluation Metric The F1 score of each label and the micro-averaging F1 score are reported for comparison. We use F1(B), F1(W), F1(N), and micro-F1 to denote them. The micro-F1 scores on the development set are used as the criteria to pick the best model over training epochs and the corresponding test performances are reported.

Reproducibility The implementation of SAECON is publicly available on GitHub[4]. Details for reproduction are given in Section 7.5.1.

Baseline Models Seven models experimented in [PBF19] and the state-of-the-art ED-GAT [MMW20] are considered for performance comparison and described in Section 7.5.4. **Fixed** BERT embeddings are used in our experiments same as ED-GAT for comparison fairness.

[4]`https://github.com/zyli93/SAECON`

7.3.2 Performances on CPC

Comparing with Baselines We report the best performances of baselines and SAECON in Table 7.2[5]. "-B" and "-G" denote different versions of the model using BERT [DCL19] and GloVe [PSM14] as the input embeddings, respectively. SAECON with BERT embeddings achieves the highest F1 scores comparing with all baselines, which demonstrates the superior ability of SAECON to accurately classify entity comparisons. The F1 scores for `None`, i.e., F1(N), are consistently the highest in all rows due to the data imbalance where `None` accounts for the largest percentage. `Worse` data is the smallest and thus is the hardest to predict precisely. This also explains why models with higher micro-F1 discussed later usually achieve larger F1(W) given that their accuracy values on the majority class (`None`) are almost identical. BERT-based models outperform GloVe-based ones, indicating the advantage of contextualized embeddings.

In later discussion, the reported performances of SAECON and its variants are based on the BERT version. The performances of the GloVe-based SAECON demonstrate similar trends.

Ablation Studies Ablation studies demonstrate the unique contribution of each part of the proposed model. Here we verify the contributions of the following modules: (1) The bi-directional global context extractor (BiLSTM); (2) The syntactic local context extractor (SGCN); (3) The domain adaptation modules of $\mathcal{A}$ (GRL); (4) The entire auxiliary sentiment analyzer, including its dependent GRL+DC ($\mathcal{A}$+GRL for short). The results are presented in Table 7.3. In Table 7.3, row "−X" denotes a variant without module X. Removing $\mathcal{A}$ will also remove GRL+DC.

Four points worth noting. Firstly, the SAECON with all modules achieves the best performance on three out of four metrics, demonstrating the effectiveness of all modules (SAECON vs. the rest); Secondly, the synergy of $\mathcal{A}$ and GRL improves the performance (−($\mathcal{A}$+GRL)

[5]All reported improvements over the best baselines are statistically significant with p-value < 0.01.

Model	Micro.	F1(B)	F1(W)	F1(N)
Majority	68.95	0.0	0.0	81.62
SE-Lin	79.31	62.71	37.61	88.42
SE-XGB	85.00	<u>75.00</u>	43.00	92.00
SVM-Tree	68.12	53.35	13.90	78.13
BERT-CLS	83.12	69.62	<u>50.37</u>	89.84
AvgWE-G	76.32	48.28	20.12	86.34
AvgWE-B	77.64	53.94	26.88	87.47
ED-GAT-G	82.73	70.23	43.30	89.84
ED-GAT-B	<u>85.42</u>	71.65	47.29	<u>92.34</u>
SAECON-G	83.78	71.06	45.90	91.05
SAECON-B	**86.74**	**77.10**	**54.08**	**92.64**

Table 7.2: Performance comparisons between SAECON and baselines on F1 scores (%).

vs. SAECON) whereas the $\mathcal{A}$ without domain adaptation hurts the classification accuracy instead ($-$GRL vs. SAECON), which indicates that the auxiliary sentiment analyzer is beneficial to CPC accuracy only with the assistance of GRL+DC modules; Thirdly, removing the global context causes the largest performance deterioration ($-$BiLSTM vs. SAECON), showing the significance of long-term information. This observation is consistent with the findings of [MMW20] in which eight to ten stacking GAT layers are used for global feature learning; Finally, the performances also drop after removing the SGCN ($-$SGCN vs. SAECON) but the drop is less than removing the BiLSTM. Therefore, local context plays a less important role than the global context ($-$SGCN vs. $-$BiLSTM).

Hyperparameter Searching We demonstrate the influences of several key hyperparameters. Such hyperparameters include the initial learning rate (LR, η), feature dimensions ($\boldsymbol{d} = \{d_g, d_l, d_s\}$), regularization weight λ, and the configurations of SGCN such as directionality, gating, and layer numbers.

Variants	**Micro.**	**F1(B)**	**F1(W)**	**F1(N)**
SAECON	**86.74**	**77.10**	**54.08**	92.64
−BiLSTM	85.21	72.94	43.86	92.63
−SGCN	86.53	76.22	51.38	92.24
−GRL	86.53	76.16	49.77	**92.93**
−($\mathcal{A}$+GRL)	85.97	74.82	52.44	92.45

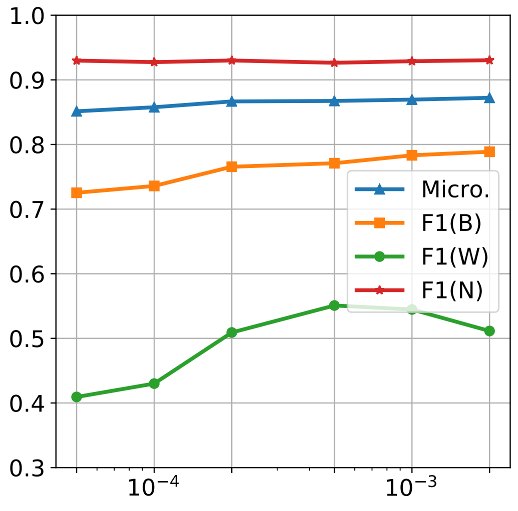
Table 7.3: Ablation studies between SAECON and its variants with modules disabled.

For LR, $\boldsymbol{d}$, and λ in Figures 7.2a, 7.2b, and 7.2c, we can observe a single peak for F1(W) (green curves) and fluctuating F1 scores for other labels and the micro-F1 (blue curves). In addition, the peaks of micro-F1 occur at the same positions of F1(W). This indicates that the performance on `Worse` is the **most** influential factor to the micro-F1. These observations help us locate the optimal settings and also show the strong learning stability of SAECON.

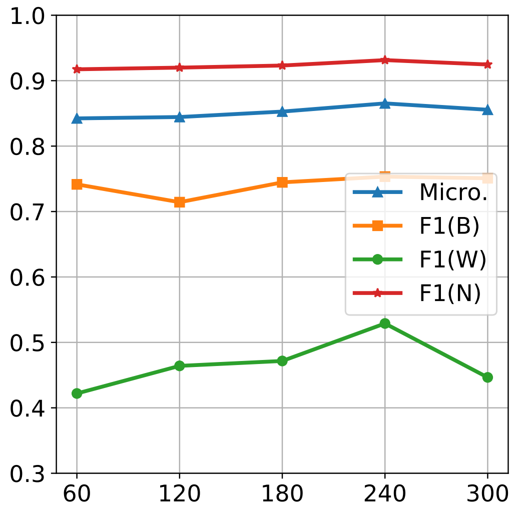
Figure 7.2d focuses on the effect of SGCN layer numbers. We observe clear oscillations on F1(W) and find the best scores at two layers. More layers of GCN result in oversmoothing [KW17] and hugely downgrade the accuracy, which is eased but not entirely fixed by the gating mechanism. Therefore, the performances slightly drop on larger layer numbers.

Table 7.4 shows the impact of directionality and gating. Turning off either the directionality or the gating mechanism ("✗✓" or "✓✗") leads to degraded F1 scores. SGCN without modifications ("✗✗") drops to the poorest micro-F1 and F1(W). Although its F1(N) is the highest, we hardly consider it a good sign. Overall, the benefits of the directionality and gating are verified.

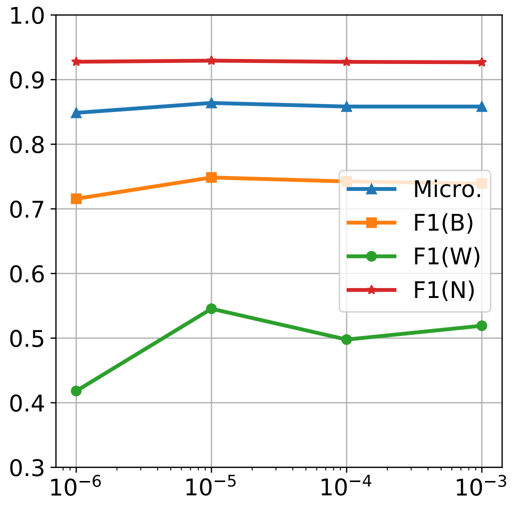
Alleviating Data Imbalance The label imbalance severely impairs the model performance, especially on the most underpopulated label `Worse`. The aforementioned imbalance alleviation methods are tested in Table 7.5. The *Original* (OR) row is a control experiment using the raw CompSent-19 without any weighting or augmentation.



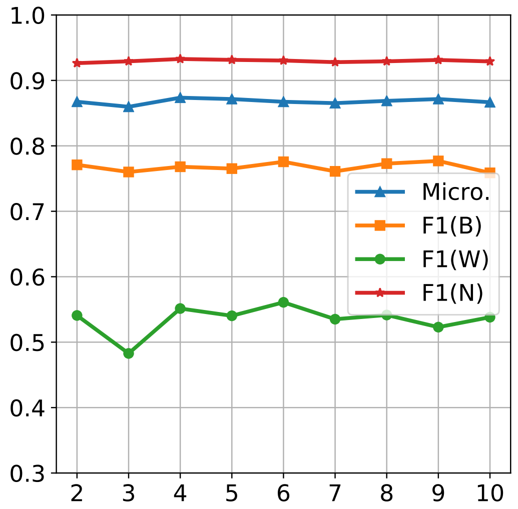
(a) F1 vs. Learning rate (η)



(b) F1 vs. Feature dims. ($\boldsymbol{d}$)



(c) F1 vs. Reg. weight (λ)



(d) F1 vs. SGCN layers

Figure 7.2: Searching and sensitivity for four key hyperparameters of SAECON in F1 scores.

The optimal solution is the weighted loss (WL vs. the rest). One interesting observation is that data augmentation such as flipping labels and upsampling cannot provide a performance gain (OR vs. FL and OR vs. UP). Weighted loss performs a bit worse on F1(N) but consistently better on the other metrics, especially on `Worse`, indicating that it effectively

Directed	Gating	Micro.	F1(B)	F1(W)	F1(N)
✓	✓	**86.74**	**77.10**	**54.08**	92.64
✗	✓	86.18	75.72	49.78	92.40
✓	✗	85.35	74.03	43.27	92.34
✗	✗	85.35	73.39	35.78	**93.04**

Table 7.4: Searching and sensitivity for the directionality and gating of SGCN by F1 scores (%).

alleviates the imbalance issue. In practice, static weights found via grid search are assigned to different labels when computing the cross entropy loss. We leave the exploration of dynamic weighting methods such as the Focal Loss [LGG17] for future work.

Methods	Micro.	F1(B)	F1(W)	F1(N)
Weighted loss (WL)	**86.74**	**77.10**	**54.08**	92.64
Original (OR)	85.97	73.80	46.15	92.90
Flipping labels (FL)	84.93	73.07	42.45	91.99
Upsampling (UP)	85.83	73.11	46.36	**92.95**

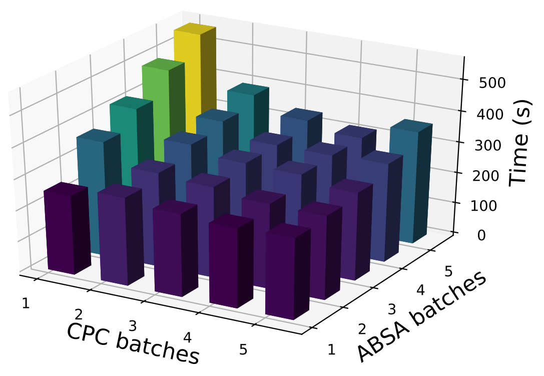
Table 7.5: Performance analysis for mitigating data imbalance with F1 scores (%).

Alternative Training One novelty of SAECON is the alternative training that allows the sentiment analyzer to learn both tasks across domains. Here we analyze the impacts of different batch ratios (BR) and different domain shift handling methods during the training. BR controls the number of ratio of batches of the two alternative tasks in each training cycle. For example, a BR of 2 : 3 sends 2 CPC batches followed by 3 ABSA batches in each iteration.

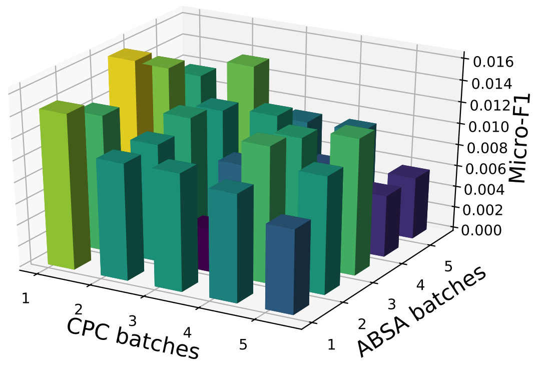
Figure 7.3a presents the entire training time for ten epochs with different BR. A larger BR takes shorter time. For example, a BR of 1:1 (the leftmost bar) takes a shorter time than

CPC sentences with sentiment predictions by $\mathcal{A}$	Label	Δ
S1: This is all done via the gigabit [**Ethernet**:`POS`] interface, rather than the much slower [**USB**:`NEG`] interface.	Better	+2
S2: Also, [**Bash**:`NEG`] may not be the best language to do arithmetic heavy operations in something like [**Python**:`NEU`] might be a better choice.	Worse	−1
S3: It shows how [**JavaScript**:`POS`] and [**PHP**:`POS`] can be used in tandem to make a user's experience faster and more pleasant.	None	0
S4: He broke his hand against [**Georgia Tech**:`NEU`] and made it worse playing against [**Virginia Tech**:`NEU`].	None	0

Table 7.6: Case studies for the effect of the sentiment analyzer $\mathcal{A}$.



(a) BR vs. Training time (s)



(b) BR vs. Micro-F1

Figure 7.3: Analyses for batch ration (BR). The values of micro-F1 are actual numbers minus 0.85 for the convenience of visualization.

1:5 (the yellow bar). Figure 7.3b presents the micro-F1 scores for different BR. We observe two points: (1) The reported performances differ slightly; (2) Generally, the performance is better when the CPC batches are less than ABSA ones. Overall, the hyperparameter selection tries to find a "sweet spot" for effectiveness and efficiency, which points to the BR of 1:1.

Figure 7.4 depicts the performance comparisons of SAECON (green bars), SAECON−GRL (the "−GRL" in Figure 7.3, orange bars), and SAECON with pretrained and fixed parameters of $\mathcal{A}$ (the option (1) mentioned in Section 7.2.4, blue bars). They represent different levels of domain shift mitigation: The pretrained and fixed $\mathcal{A}$ *does NOT* handle the domain shift at all; The variant −GRL only attempts to implicitly handle the shift by alternative training with different tasks to converge in the middle although the domain difference can be harmful to both objectives; SAECON, instead, explicitly uses GRL+DC to mitigate the domain shift between $\mathcal{D}_s$ and $\mathcal{D}_c$ during training.

As a result, SAECON achieves the best performance especially on F1(W), −GRL gets the second, and the "option (1)" gets the worst. These demonstrate that (1) the alternative training (blue vs. green) for an effective domain adaptation is necessary and (2) there exists a positive correlation between the level of domain shift mitigation and the model performance, especially on F1(W) and F1(B). A better domain adaptation produces higher F1 scores in the scenarios where datasets in the domain of interest, i.e., CPC, is unavailable.

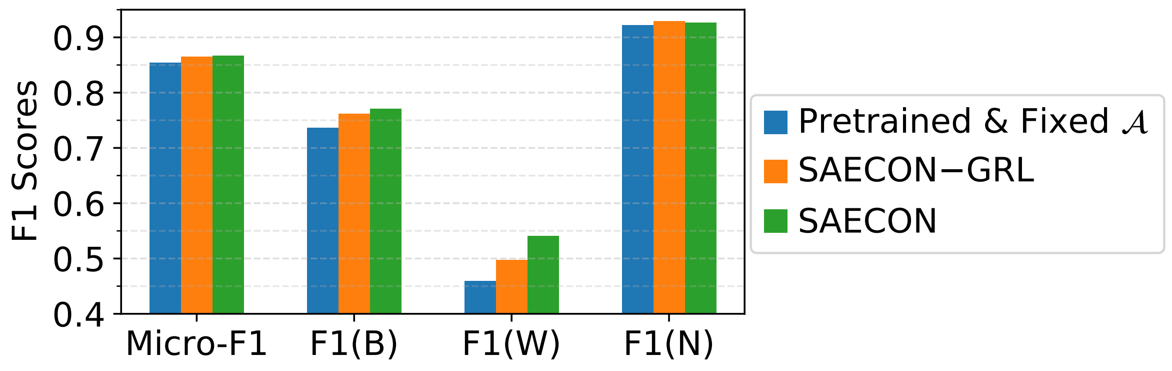


Figure 7.4: Visualization of the domain shift mitigation.

7.3.3 Case Study

In this section, we qualitatively exemplify the contribution of the sentiment analyzer $\mathcal{A}$. Table 7.6 reports four example sentences from the test set of CompSent-19. The entities

e_1 and e_2 are highlighted together with the corresponding sentiment predicted by $\mathcal{A}$. The column "Label" shows the ground truth of CPC. The "Δ" column computes the sentiment distances between the entities. We assign $+1$, 0, and -1 to sentiment polarities of `POS`, `NEU`, and `NEG`, respectively. Δ is computed by the sentiment polarity of e_1 minus that of e_2. Therefore, a positive distance suggests that e_1 receives a more positive sentiment from $\mathcal{A}$ than e_2 and vice versa. In **S1**, sentiments to *Ethernet* and *USB* are predicted positive and negative, respectively, which can correctly imply the comparative label as `Better`. **S2** is a `Worse` sentence with *Bash* predicted negative, *Python* predicted neutral, and a resultant negative sentiment distance -1. For **S3** and **S4**, the entities are assigned the same polarities. Therefore, the sentiment distances are both zeros. We can easily tell that preference comparisons do not exist, which is consistent with the ground truth labels. Due to the limited space, more interesting case studies are presented in Section 7.5.5.

7.4 Summary

This chapter proposes SAECON, a CPC model that incorporates a sentiment analyzer to transfer knowledge from ABSA corpora. Specifically, SAECON utilizes a BiLSTM to learn global comparative features, a syntactic GCN to learn local syntactic information, and a domain adaptive auxiliary sentiment analyzer that jointly learns from ABSA corpora and CPC corpora for a smooth knowledge transfer. An alternative joint training scheme enables the efficient and effective information transfer. Qualitative and quantitative experiments verified the superior performance of SAECON. For future work, we will focus on a deeper understanding of CPC data augmentation and an exploration of weighting loss methods for data imbalance.

7.5 Supplementary Materials

This section contains the supplementary materials. Here we provide additional supporting information in four aspects, including additional description for the training, the reproducibility details of SAECON, brief introductions of baselines, and additional case studies.

7.5.1 Implementation of SAECON

The proposed SAECON is implemented in Python (3.6.8) with PyTorch (1.5.0) and run with a single 16GB Nvidia V100 GPU. The source code of SAECON is publicly available on GitHub[6] and comprehensive instructions on how to reproduce our model are also provided.

The implementation of SGCN is based on PyTorch Geometric[7]. The implementation of our sentiment analyzer $\mathcal{A}$ is adapted from the official source code of LCF-ASC [PO20][8]. The dependency parser used in SAECON is from spaCy [9]. The pretrained embedding vectors of GloVe are downloaded from the office site[10]. The pretrained BERT model is obtained from the Hugging Face model repository[11]. The implementation of the gradient reversal package is available on GitHub[12]. We would like to appreciate the authors of these packages for their precious contributions.

[6]`https://github.com/zyli93/SAECON`

[7]`https://github.com/rusty1s/pytorch_geometric`

[8]`https://github.com/HieuPhan33/LCFS-BERT`

[9]`https://spacy.io/`

[10]`https://nlp.stanford.edu/projects/glove/`

[11]`https://huggingface.co/bert-base-uncased`

[12]`https://github.com/janfreyberg/pytorch-revgrad`

7.5.2 Default Hyperparameters

The default hyperparameter settings for the results reported in Section 7.3.2 are given in Table 7.7.

Hyperparameter	Setting
GloVe embeddings	pretrained, 100 dims (d_0)
BERT version	`bert-base-uncased`
BERT numeric config.	12 heads, 12 layers, 768 dims (d_0)
BERT parameters	pretrained by Hugging Face
Dependency parser	spaCy, pretrained model
Batch config.	size = 16, batch ratio = 1 : 1
Init. learning rate (η)	5×10^{-4}
CPC loss weight	2 : 4 : 1 (B:W:N)
$\boldsymbol{\lambda}$ ($\{\lambda, \lambda_s, \lambda_d\}$)	$\{1 \times 10^{-4}, 1, 1\}$
Activation function	ReLU ($f(x) = \max(0, x)$)
$\boldsymbol{d}$ ($\{d_g, d_l, d_s\}$)	$\{240, 240, 240\}$
Optimizer	Adam ($\beta_1 = 0.9, \beta_2 = 0.999$)
Learning rate scheduler	`StepLR` (steps = 3, $\gamma = 0.8$)
GRL config.	$\alpha = 1.0$ (Default setting)
SGCN numeric config.	2 layers (768→256→240)
SGCN architectural config..	Directed, Gated
Data augmentation	Off (weighted loss only)

Table 7.7: Default hyperparameter settings for SAECON.

7.5.3 Reproduction of ED-GAT

We briefly introduce the reproduction of the state-of-the-art baseline, ED-GAT, in both GloVe and BERT versions. We implement ED-GAT with the same software packages as SAECON such as PyTorch-Geometric, spaCy, and PyTorch, and run it within the same machine environment. The parameters all follow the original ED-GAT setting [MMW20] except the dimension of GloVe. It is set to 300 in the original paper but 100 in our experiments for the fairness of comparison. The number of layers is select as 8 and the hidden size is set to 300 for each layer with 6 attention heads. We trained the model for 15 epochs with Adam optimizer with a batch size of 32.

7.5.4 Baseline models

We briefly introduce the compared models in Section 7.3.2.

Majority-Class A simple model which chooses the majority label in the training set as the prediction of each test instance.

SE Sentence Embedding encodes the sentences into low-dimensional sentence representations using pretrained language encoders [CKS17, BAP15] and then feeds them into a classifier for comparative preference prediction. **SE** has two versions [PBF19] with different classifiers, namely **SE-Lin** with a linear classifier and **SE-XGB** with an XGBoost classifier.

SVM-Tree This method [TL15] applies convolutional kernel methods to CSI task. We follow the experimental settings of [MMW20].

AvgWE A word embedding-based method that averages the word embeddings of the sentence as the sentence representation and then feeds this representation into a classifier. The input embeddings have several options, such as GloVe [PSM14] and BERT [DCL19]. These variants are denoted by **AvgWE-G** and **AvgWE-B** separately.

BERT-CLS Using the representation of the token "[CLS]" generated by BERT [DCL19] as the sentence embedding and a linear classifier to conduct comparative preference classification.

ED-GAT Entity-aware Dependency-based Graph Attention Network [MMW20] is the first entity-aware model that analyzes the entity relations via the dependency graph and multi-layer graph attention layer.

7.5.5 Additional Case Studies

In this section, we present four supplementary examples for case study in Table 7.8 which have different sentiments compared with their counterparts in Table 7.6. **S1** shows a `NEU` versus `NEG` comparison which results in a sentiment distance +1 and a CPC prediction `Better`. "Ruby" is not praised in this sentence so it has `NEU`. But "Perl" is assigned a negative emotion through a simple inference. **S2** shows a stronger contrast between the entities. "Mid Missouri" is said "much worse" while the "South Georgia" is "much warmer", which clearly indicates the sentiments and the comparative classification results.

S3 and **S4** are two sentences both with two parallel negative entities. The sport equipment in **S3** is sold "poorly" and the drinks in **S4** are "ten times worse" both indicating negative sentiments. Therefore, the labels are `None`.

Supplementary CPC sentences with sentiment predictions by $\mathcal{A}$	Label	Δ
S1: [**Ruby**:`NEU`] wasn't designed to "exemplify best practices", it was to be a better [**Perl**:`NEG`].	Better	$+1$
S2: And from my experience the ticks are much worse in [**Mid Missouri**:`NEG`] than they are in [**South Georgia**:`POS`] which is much warmer year round.	Worse	-2
S3: As an industry rule, [**hockey**:`NEG`] and [**basketball**:`NEG`] sell comparatively poorly everywhere.	None	0
S4: [**Milk**:`NEG`], [**juice**:`NEG`] and soda make it ten times worse.	None	0

Table 7.8: Additional case studies for the effect of sentiment analyzer $\mathcal{A}$ (see Section 7.5.5).

CHAPTER 8

Conclusion

We conclude the dissertation in this chapter. In previous chapters, we introduced the motivation of incorporating multimodal data into preference analysis and recommendation and past endeavors on a variety of topics related to utilizing different data modalities. Then we demonstrated five projects as examples for utilizing network structures, user and item attribute information, geographical information, and review text to improve recommendation performance. The benefit of incorporating these data types into recommendations has been demonstrated by the diverse evaluation methods. Therefore, the following conclusions are drawn.

Firstly, recommender systems inherently carry different data modalities such as graph structures, attribute information, and natural language. Incorporating different types of information properly can boost the performance of recommendation tasks such as CTR prediction, POI recommendation, and rating prediction. For example, question routing performance is enhanced with the help of the graph and text.

Secondly, different data modalities make possible the interpretation of decision-making. In this dissertation, we explored the following three different methods for prediction explanation: explaining by finding salient features and their interactions in InterHAt, explaining by decomposing motivation in GEAPR, and explaining by profiling user attention and item property and capturing their overlap.

Finally, we noticed that upgrades on modules and architecture provide a stronger performance improvement. For example, heterogeneous embedding in NeRank achieves better performance than homogeneous methods. As the deep learning toolkit grows, the recommendation with multimodality data will continue to be optimized.

We propose a few directions for future work: (1) In this dissertation, several modalities are utilized including network, attribute, and text. However, there are other modalities that can be considered as well such as image and video. Some existing works explored integrating image representation into preference learning which can be further incorporated into recommender systems; (2) For the explanation of the recommendation, this dissertation covers two ways: explaining via salient features and explaining via salient aspects. Another possible technical direction is to recommend via automatic explanation generation. However, the technical difficulty is the lack of training data; (3) New social media emerged recently such as AR/VR-based platforms. Certain adjustments are in need for these new media to recommend items in these scenarios. How to deal with the new challenges remains to be explored.

REFERENCES

[AG20] Cem Rifki Aydin and Tunga Güngör. "Combination of Recursive and Recurrent Neural Networks for Aspect-Based Sentiment Analysis Using Inter-Aspect Relations." *IEEE Access*, 2020.

[AL18] Stefanos Angelidis and Mirella Lapata. "Summarizing Opinions: Aspect Extraction Meets Sentiment Prediction and They Are Both Weakly Supervised." In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[BAP15] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. "A large annotated corpus for learning natural language inference." In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015.

[BCB14] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." *arXiv preprint arXiv:1409.0473*, 2014.

[BCK15] Linas Baltrunas, Karen Church, Alexandros Karatzoglou, and Nuria Oliver. "Frappe: Understanding the Usage and Perception of Mobile App Recommendations In-The-Wild." *arXiv preprint arXiv:1505.03014*, 2015.

[BES10] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. "Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining." In *Proceedings of the Seventh International Conference on Language Resources and Evaluation*, 2010.

[BFU16] Mathieu Blondel, Akinori Fujino, Naonori Ueda, and Masakazu Ishihata. "Higher-order factorization machines." In *Advances in Neural Information Processing Systems*, pp. 3351–3359, 2016.

[BLB19] Lingxian Bao, Patrik Lambert, and Toni Badia. "Attention and Lexicon Regularized LSTM for Aspect-based Sentiment Analysis." In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, 2019.

[BLT17] Konstantin Bauman, Bing Liu, and Alexander Tuzhilin. "Aspect based recommendations: Recommending items with the most valuable aspects based on user reviews." In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017.

[BNJ03] David M Blei, Andrew Y Ng, and Michael I Jordan. "Latent dirichlet allocation." *Journal of machine Learning research*, 2003.

[BPS19] Yonatan Belinkov, Adam Poliak, Stuart Shieber, Benjamin Van Durme, and Alexander Rush. "On Adversarial Removal of Hypothesis-only Bias in Natural Language Inference." In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics*, 2019.

[BSR05] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. "Learning to rank using gradient descent." In *ICML*, 2005.

[BTA17] Jasmijn Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an. "Graph Convolutional Encoders for Syntax-aware Neural Machine Translation." In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017.

[CBS16] Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart. "Retain: An interpretable predictive model for healthcare using reverse time attention mechanism." In *NIPS*, pp. 3504–3512, 2016.

[CGC20] Anirban Chakraborty, Debasis Ganguly, and Owen Conlan. "Relevance Models for Multi-Contextual Appropriateness in Point-of-Interest Recommendation." In *SIGIR*, 2020.

[CKH16] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. "Wide & deep learning for recommender systems." In *Workshop on DLRS*, pp. 7–10. ACM, 2016.

[CKS17] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. "Supervised Learning of Universal Sentence Representations from Natural Language Inference Data." In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017.

[CLW20] Shaowei Chen, Jie Liu, Yu Wang, Wenzheng Zhang, and Ziming Chi. "Synchronous Double-channel Recurrent Network for Aspect-Opinion Pair Extraction." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[CLZ20] Chaochao Chen, Ziqi Liu, Peilin Zhao, Jun Zhou, and Xiaolong Li. "Privacy preserving point-of-interest recommendation using decentralized matrix factorization." *arXiv preprint arXiv:2003.05610*, 2020.

[CM14] Danqi Chen and Christopher D Manning. "A fast and accurate dependency parser using neural networks." In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, 2014.

[CP13] Shuo Chang and Aditya Pal. "Routing questions for collaborative answering in community question answering." In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 494–501. ACM, 2013.

[CQ20] Zhuang Chen and Tieyun Qian. "Relation-Aware Collaborative Learning for Unified Aspect-Based Sentiment Analysis." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020*, 2020.

[CS17] Ting Chen and Yizhou Sun. "Task-guided and path-augmented heterogeneous network embedding for author identification." In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pp. 295–304. ACM, 2017.

[CSW20] Xiao Chen, Changlong Sun, Jingjing Wang, Shoushan Li, Luo Si, Min Zhang, and Guodong Zhou. "Aspect sentiment classification with document-level sentiment preference modeling." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[CZH17] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. "Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention." In *SIGIR*, 2017.

[CZJ18] Jin Yao Chin, Kaiqi Zhao, Shafiq Joty, and Gao Cong. "ANR: Aspect-based neural recommender." In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018.

[CZL18] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. "Neural attentional rating regression with review-level explanations." In *Proceedings of the 2018 World Wide Web Conference*, 2018.

[DCL18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding." *arXiv preprint arXiv:1810.04805*, 2018.

[DCL19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT*, 2019.

[DCS17] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. "metapath2vec: Scalable representation learning for heterogeneous networks." In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 135–144. ACM, 2017.

[DNC20] Xin Dong, Jingchao Ni, Wei Cheng, Zhengzhang Chen, Bo Zong, Dongjin Song, Yanchi Liu, Haifeng Chen, and Gerard de Melo. "Asymmetrical hierarchical networks with attentive interactions for interpretable review-based recommendation." In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[DS19] Hongliang Dai and Yangqiu Song. "Neural Aspect and Opinion Term Extraction with Mined Rules as Weak Supervision." In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

[DSW20] Chunning Du, Haifeng Sun, Jingyu Wang, Qi Qi, and Jianxin Liao. "Adversarial and domain-aware bert for cross-domain sentiment analysis." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[DZT15] Yuxiao Dong, Jing Zhang, Jie Tang, Nitesh V Chawla, and Bai Wang. "Coupledlp: Link prediction in coupled networks." In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 199–208. ACM, 2015.

[GBY18] Leilani H Gilpin, David Bau, Ben Z Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. "Explaining explanations: An overview of interpretability of machine learning." In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*, pp. 80–89. IEEE, 2018.

[GCH19] Xinyu Guan, Zhiyong Cheng, Xiangnan He, Yongfeng Zhang, Zhibo Zhu, Qinke Peng, and Tat-Seng Chua. "Attentive aspect modeling for review-aware recommendation." *ACM Transactions on Information Systems (TOIS)*, **37**(3):1–27, 2019.

[GFL19] Shuhao Gu, Yang Feng, and Qun Liu. "Improving Domain Adaptation Translation with Domain Invariant and Specific Information." In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.

[GL15] Yaroslav Ganin and Victor Lempitsky. "Unsupervised domain adaptation by backpropagation." In *International conference on machine learning*. PMLR, 2015.

[GL16] Aditya Grover and Jure Leskovec. "node2vec: Scalable feature learning for networks." In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 855–864. ACM, 2016.

[GMR17] Samir Gupta, A. S. M. Ashique Mahmood, Karen Ross, Cathy H. Wu, and K. Vijay-Shanker. "Identifying Comparative Structures in Biomedical Text." In *BioNLP 2017*, 2017.

[GNH11] Rainer Gemulla, Erik Nijkamp, Peter J Haas, and Yannis Sismanis. "Large-scale matrix factorization with distributed stochastic gradient descent." In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 69–77. ACM, 2011.

[GPM14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets." In *Advances in neural information processing systems*, 2014.

[GTY17] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. "DeepFM: A Factorization-Machine based Neural Network for CTR Prediction." In *IJCAI*, pp. 1725–1731, 2017.

[GZL17] Kun Gai, Xiaoqiang Zhu, Han Li, Kai Liu, and Zhe Wang. "Learning Piece-wise Linear Models from Large Scale Data for Ad Click Prediction." *arXiv preprint arXiv:1704.05194*, 2017.

[HAA18] Dong Huk Park, Lisa Anne Hendricks, Zeynep Akata, Anna Rohrbach, Bernt Schiele, Trevor Darrell, and Marcus Rohrbach. "Multimodal explanations: Justifying decisions and pointing to the evidence." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8779–8788, 2018.

[HC17] Xiangnan He and Tat-Seng Chua. "Neural factorization machines for sparse predictive analytics." In *SIGIR*, pp. 355–364, 2017.

[HDO98] Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. "Support vector machines." *IEEE Intelligent Systems and their applications*, **13**(4):18–28, 1998.

[HJW20] Chunli Huang, Wenjun Jiang, Jie Wu, and Guojun Wang. "Personalized review recommendation based on users' aspect sentiment." *ACM Transactions on Internet Technology (TOIT)*, **20**(4):1–26, 2020.

[HK16] F Maxwell Harper and Joseph A Konstan. "The movielens datasets: History and context." *Acm transactions on interactive intelligent systems (tiis)*, **5**(4):19, 2016.

[HKV08] Yifan Hu, Yehuda Koren, and Chris Volinsky. "Collaborative filtering for implicit feedback datasets." In *2008 Eighth IEEE International Conference on Data Mining*, 2008.

[HL04] Minqing Hu and Bing Liu. "Mining and summarizing customer reviews." In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2004.

[HLN17] Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. "An unsupervised neural attention model for aspect extraction." In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017.

[HLN18] Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. "Adaptive Semi-supervised Learning for Cross-domain Sentiment Classification." In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018.

[HLZ17] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. "Neural collaborative filtering." In *Proceedings of the 26th International Conference on World Wide Web*, 2017.

[HM16] Ruining He and Julian McAuley. "Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering." In *Proceedings of the 25th International Conference on World Wide Web*, 2016.

[HMG20] Jiaxin Huang, Yu Meng, Fang Guo, Heng Ji, and Jiawei Han. "Weakly-Supervised Aspect-Based Sentiment Analysis via Joint Aspect-Sentiment Topic Embedding." In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

[HPH19] Minghao Hu, Yuxing Peng, Zhen Huang, Dongsheng Li, and Yiwei Lv. "Open-Domain Targeted Sentiment Analysis via Span-Based Extraction and Classification." In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

[HPJ14] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, et al. "Practical lessons from predicting clicks on ads at facebook." In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*, pp. 1–9. ACM, 2014.

[HPN18] Mengyue Hang, Ian Pytlarz, and Jennifer Neville. "Exploring student check-in behavior for improved point-of-interest prediction." In *KDD*, 2018.

[HYW19] Yunfeng Hou, Ning Yang, Yi Wu, and S Yu Philip. "Explainable recommendation with fusion of aspect information." *World Wide Web*, **22**(1):221–240, 2019.

[HZR16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[ITW18] Maximilian Ilse, Jakub M Tomczak, and Max Welling. "Attention-based Deep Multiple Instance Learning." *ICML*, 2018.

[JL06] Nitin Jindal and Bing Liu. "Identifying comparative sentences in text documents." In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2006.

[JW13] Zongcheng Ji and Bin Wang. "Learning to rank for question routing in community question answering." In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pp. 2363–2368. ACM, 2013.

[JW19] Sarthak Jain and Byron C. Wallace. "Attention is not Explanation." In *NAACL*, 2019.

[JXZ19] Xu Jiao, Yingyuan Xiao, Wenguang Zheng, Hongya Wang, and Youzhi Jin. "R2SIGTP: A novel real-time recommendation system with integration of geography and temporal preference for next point-of-interest." In *The World Wide Web Conference*, 2019.

[JZC16] Yuchin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. "Field-aware factorization machines for CTR prediction." In *Proceedings of the 10th ACM Conference on Recommender Systems*, 2016.

[KB14] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980*, 2014.

[KB15] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization." In *3rd International Conference on Learning Representations*, 2015.

[KGC19] Anush Kamath, Sparsh Gupta, and Vitor Carvalho. "Reversing Gradients in Adversarial Domain Adaptation for Question Deduplication and Textual Entailment Tasks." In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

[Kim14] Yoon Kim. "Convolutional neural networks for sentence classification." *arXiv preprint arXiv:1408.5882*, 2014.

[KPO16] Donghyun Kim, Chanyoung Park, Jinoh Oh, Sungyoung Lee, and Hwanjo Yu. "Convolutional matrix factorization for document context-aware recommendation." In *Proceedings of the 10th ACM Conference on Recommender Systems*, 2016.

[KW17] Thomas N. Kipf and Max Welling. "Semi-Supervised Classification with Graph Convolutional Networks." In *5th International Conference on Learning Representations*, 2017.

[KZC14] Svetlana Kiritchenko, Xiaodan Zhu, Colin Cherry, and Saif Mohammad. "NRC-Canada-2014: Detecting Aspects and Sentiment in Customer Reviews." In *Proceedings of the 8th International Workshop on Semantic Evaluation*, 2014.

[LCC20] Zeyu Li, Wei Cheng, Yang Chen, Haifeng Chen, and Wei Wang. "Interpretable Click-Through Rate Prediction through Hierarchical Attention." In *WSDM*, 2020.

[LCL15] Xutao Li, Gao Cong, Xiao-Li Li, Tuan-Anh Nguyen Pham, and Shonali Krishnaswamy. "Rank-geofm: A ranking based geographical factorization method for point of interest recommendation." In *SIGIR*, 2015.

[LCZ20] Guohui Li, Qi Chen, Bolong Zheng, Hongzhi Yin, Quoc Viet Hung Nguyen, and Xiaofang Zhou. "Group-Based Recurrent Neural Networks for POI Recommendation." *ACM Transactions on Data Science*, 2020.

[LGG17] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. "Focal loss for dense object detection." In *Proceedings of the IEEE international conference on computer vision*, 2017.

[LGH16] Huayu Li, Yong Ge, Richang Hong, and Hengshu Zhu. "Point-of-interest recommendations: Learning potential check-ins from friends." In *KDD*, 2016.

[Li92] Wentian Li. "Random texts exhibit Zipf's-law-like word frequency distribution." *IEEE Transactions on information theory*, **38**(6), 1992.

[LJJ19] Ruirui Li, Jyun-Yu Jiang, Chelsea J-T Ju, and Wei Wang. "CORALS: Who Are My Potential New Customers? Tapping into the Wisdom of Customers' Decisions." In *WSDM*, 2019.

[LJS19] Zeyu Li, Jyun-Yu Jiang, Yizhou Sun, and Wei Wang. "Personalized Question Routing via Heterogeneous Network Embedding." In *AAAI*, volume 33, pp. 192–199, 2019.

[LLD19] Donghua Liu, Jing Li, Bo Du, Jun Chang, and Rong Gao. "Daml: Dual attention mutual learning between ratings and reviews for item recommendation." In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

[LLL19] Huaishao Luo, Tianrui Li, Bing Liu, and Junbo Zhang. "DOER: Dual Cross-Shared RNN for Aspect Term-Polarity Co-Extraction." In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

[LLS16] Mu Li, Ziqi Liu, Alexander J. Smola, and Yu-Xiang Wang. "DiFacto: Distributed Factorization Machines." In *WSDM*, pp. 377–386, 2016.

[LLW20] Wei Liu, Hanjiang Lai, Jing Wang, Geyang Ke, Weiwei Yang, and Jian Yin. "Mix geographical information into local collaborative ranking for POI recommendation." *World Wide Web*, 2020.

[LPC17] Yiding Liu, Tuan-Anh Pham, Gao Cong, and Quan Yuan. "An Experimental Evaluation of Point-of-interest Recommendation in Location-based Social Networks." *PVLDB*, pp. 1010–1021, 2017.

[LTB16] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard S. Zemel. "Gated Graph Sequence Neural Networks." In *4th International Conference on Learning Representations*, 2016.

[LWS14] Yong Liu, Wei Wei, Aixin Sun, and Chunyan Miao. "Exploiting geographical neighborhood characteristics for location recommendation." In *CIKM*, 2014.

[LWW16] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. "Predicting the next location: A recurrent model with spatial and temporal contexts." In *AAAI*, 2016.

[LWZ18] Zheng Li, Ying Wei, Yu Zhang, and Qiang Yang. "Hierarchical Attention Transfer Network for Cross-Domain Sentiment Classification." In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[LXM13] Haifeng Liu, Guotong Xie, Jing Mei, Weijia Shen, Wen Sun, and Xiang Li. "An efficacy driven approach for medication recommendation in type 2 diabetes treatment using data mining techniques." *Studies in health technology and informatics*, **192**:1071–1071, 2013.

[LYB16] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. "Hierarchical question-image co-attention for visual question answering." In *Advances In Neural Information Processing Systems*, pp. 289–297, 2016.

[LZX14] Defu Lian, Cong Zhao, Xing Xie, Guangzhong Sun, Enhong Chen, and Yong Rui. "GeoMF: joint geographical modeling and matrix factorization for point-of-interest recommendation." In *KDD*, 2014.

[LZX20] Shuangli Li, Jingbo Zhou, Tong Xu, Hao Liu, Xinjiang Lu, and Hui Xiong. "Competitive Analysis for Points of Interest." In *KDD*, 2020.

[LZZ18] Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. "xDeepFM: Combining Explicit and Implicit Feature Interactions for Recommender Systems." In *KDD*, pp. 1754–1763, 2018.

[MCC13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781*, 2013.

[Mil95] George A. Miller. "WordNet: A Lexical Database for English." *Commun. ACM*, 1995.

[MLW19] Dehong Ma, Sujian Li, Fangzhao Wu, Xing Xie, and Houfeng Wang. "Exploring sequence-to-sequence learning in aspect term extraction." In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

[MMW20] Nianzu Ma, Sahisnu Mazumder, Hao Wang, and Bing Liu. "Entity-Aware Dependency-Based Deep Graph Attention Network for Comparative Preference Classification." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[MPC18] Yukun Ma, Haiyun Peng, and Erik Cambria. "Targeted Aspect-Based Sentiment Analysis via Embedding Commonsense Knowledge into an Attentive LSTM." In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[MSC13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. "Distributed representations of words and phrases and their compositionality." In *Advances in neural information processing systems*, pp. 3111–3119, 2013.

[MT17] Diego Marcheggiani and Ivan Titov. "Encoding Sentences with Graph Convolutional Networks for Semantic Role Labeling." In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017.

[MZW18] Chen Ma, Yingxue Zhang, Qinglong Wang, and Xue Liu. "Point-of-interest recommendation: Exploiting self-attentive autoencoders with neighbor-aware influence." In *CIKM*, 2018.

[NCL18] Jingchao Ni, Shiyu Chang, Xiao Liu, Wei Cheng, Haifeng Chen, Dongkuan Xu, and Xiang Zhang. "Co-regularized deep multi-network embedding." In *WWW*, pp. 469–478, 2018.

[NS15] Thien Hai Nguyen and Kiyoaki Shirai. "PhraseRNN: Phrase Recursive Neural Network for Aspect-based Sentiment Analysis." In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015.

[PAS14] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. "Deepwalk: Online learning of social representations." In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 701–710. ACM, 2014.

[PBF19] Alexander Panchenko, Alexander Bondarenko, Mirco Franzek, Matthias Hagen, and Chris Biemann. "Categorizing Comparative Sentences." In *Proceedings of the 6th Workshop on Argument Mining*, 2019.

[PGP14] Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. "SemEval-2014 Task 4: Aspect Based Sentiment Analysis." In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, 2014.

[PGP15] Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. "SemEval-2015 Task 12: Aspect Based Sentiment Analysis." In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, 2015.

[PND20] Amir Pouran Ben Veyseh, Nasim Nouri, Franck Dernoncourt, Quan Hung Tran, Dejing Dou, and Thien Huu Nguyen. "Improving Aspect-based Sentiment Analysis with Gated Graph Convolutional Networks and Syntax-based Regulation." In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020.

[PO20] Minh Hieu Phan and Philip O Ogunbona. "Modelling context and syntactical features for aspect-based sentiment analysis." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[PP17] Nikolaos Pappas and Andrei Popescu-Belis. "Multilingual hierarchical attention networks for document classification." *arXiv preprint arXiv:1707.00896*, 2017.

[PSM14] Jeffrey Pennington, Richard Socher, and Christopher D Manning. "Glove: Global vectors for word representation." In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

[QCR16] Yanru Qu, Han Cai, Kan Ren, Weinan Zhang, Yong Yu, Ying Wen, and Jun Wang. "Product-based neural networks for user response prediction." In *ICDM*, pp. 1149–1154. IEEE, 2016.

[RAB20] Hossein A Rahmani, Mohammad Aliannejadi, Mitra Baratchi, and Fabio Crestani. "Joint Geographical and Temporal Modeling based on Matrix Factorization for Point-of-Interest Recommendation." In *European Conference on Information Retrieval*. Springer, 2020.

[RDR07] Matthew Richardson, Ewa Dominowska, and Robert Ragno. "Predicting clicks: estimating the click-through rate for new ads." In *WWW*, pp. 521–530, 2007.

[Ren10] Steffen Rendle. "Factorization machines." In *2010 IEEE International Conference on Data Mining*, 2010.

[SH12] Yizhou Sun and Jiawei Han. "Mining heterogeneous information networks: principles and methodologies." *Synthesis Lectures on Data Mining and Knowledge Discovery*, **3**(2):1–159, 2012.

[SHJ16] Ying Shan, T. Ryan Hoens, Jian Jiao, Haijing Wang, Dong Yu, and J. C. Mao. "Deep Crossing: Web-Scale Modeling without Manually Crafted Combinatorial Features." In *KDD*, pp. 255–262, 2016.

[SHQ19] Chi Sun, Luyao Huang, and Xipeng Qiu. "Utilizing BERT for Aspect-Based Sentiment Analysis via Constructing Auxiliary Sentence." In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.

[SHY11] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. "Pathsim: Meta path-based top-k similarity search in heterogeneous information networks." *Proceedings of the VLDB Endowment*, **4**(11):992–1003, 2011.

[SHY17] Sungyong Seo, Jing Huang, Hao Yang, and Yan Liu. "Interpretable convolutional neural networks with dual local and global attention for review rating prediction." In *Proceedings of the 11th ACM Conference on Recommender Systems*, 2017.

[SHZ18] Chuan Shi, Binbin Hu, Xin Zhao, and Philip Yu. "Heterogeneous Information Network Embedding for Recommendation." *IEEE Transactions on Knowledge and Data Engineering*, 2018.

[SNH13] Yizhou Sun, Brandon Norick, Jiawei Han, Xifeng Yan, Philip S Yu, and Xiao Yu. "Pathselclus: Integrating meta-path selection with user-guided object clustering in heterogeneous information networks." *ACM Transactions on Knowledge Discovery from Data (TKDD)*, **7**(3):11, 2013.

[SQC20] Ke Sun, Tieyun Qian, Tong Chen, Yile Liang, Quoc Viet Hung Nguyen, and Hongzhi Yin. "Where to Go Next: Modeling Long-and Short-Term User Preferences for Point-of-Interest Recommendation." In *AAAI*, 2020.

[SSX18] Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. "AutoInt: Automatic Feature Interaction Learning via Self-Attentive Neural Networks." *arXiv preprint arXiv:1810.11921*, 2018.

[SXL17] Lei Shu, Hu Xu, and Bing Liu. "Lifelong Learning CRF for Supervised Aspect Extraction." In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017.

[TC18] Brian Tubay and Marta R Costa-jussà. "Neural machine translation with the transformer and multi-source romance languages for the biomedical WMT 2018 task." In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pp. 667–670, 2018.

[TC20] Stéphan Tulkens and Andreas van Cranenburgh. "Embarrassingly Simple Unsupervised Aspect Extraction." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[TGX20] Hao Tian, Can Gao, Xinyan Xiao, Hao Liu, Bolei He, Hua Wu, Haifeng Wang, and Feng Wu. "SKEP: Sentiment Knowledge Enhanced Pre-training for Sentiment Analysis." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[TJL20] Hao Tang, Donghong Ji, Chenliang Li, and Qiji Zhou. "Dependency graph enhanced dual-transformer structure for aspect-based sentiment classification." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[TL15] Maksim Tkachenko and Hady Lauw. "A Convolution Kernel Approach to Identifying Comparisons in Text." In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, 2015.

[TLH18] Yi Tay, Anh Tuan Luu, and Siu Cheung Hui. "Multi-pointer co-attention networks for recommendation." In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2018.

[Tob70] Waldo R Tobler. "A computer movie simulating urban growth in the Detroit region." *Economic geography*, **46**(sup1):234–240, 1970.

[TQF16] Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. "Effective LSTMs for Target-Dependent Sentiment Classification." In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics*, 2016.

[TQW15] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. "Line: Large-scale information network embedding." In *Proceedings of the 24th International Conference on World Wide Web*, pp. 1067–1077. International World Wide Web Conferences Steering Committee, 2015.

[Tur02] Peter D. Turney. "Thumbs up or Thumbs down? Semantic Orientation Applied to Unsupervised Classification of Reviews." In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, 2002.

[VCC17] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. "Graph attention networks." *arXiv preprint arXiv:1710.10903*, 2017.

[VSP17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." In *NIPS*, pp. 5998–6008, 2017.

[WAC14] Joachim Wagner, Piyush Arora, Santiago Cortes, Utsab Barman, Dasha Bogdanova, Jennifer Foster, and Lamia Tounsi. "DCU: Aspect-based Polarity Classification for SemEval Task 4." In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, 2014.

[WFF17] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. "Deep & Cross Network for Ad Click Predictions." In *ADKDD*, pp. 12:1–12:7. ACM, 2017.

[WHZ16] Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. "Attention-based LSTM for Aspect-level Sentiment Classification." In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016.

[WHZ20] Zhenkai Wei, Yu Hong, Bowei Zou, Meng Cheng, and YAO Jianmin. "Don't eclipse your arts due to small discrepancies: Boundary repositioning with a pointer network for aspect extraction." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[WML18] Shuai Wang, Sahisnu Mazumder, Bing Liu, Mianwei Zhou, and Yi Chang. "Target-sensitive memory networks for aspect sentiment classification." In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 2018.

[WP18] Wenya Wang and Sinno Jialin Pan. "Recursive neural structural correspondence network for cross-domain aspect and opinion co-extraction." In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 2018.

[WSY20] Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. "Relational Graph Attention Network for Aspect-based Sentiment Analysis." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[WWT17] Suhang Wang, Yilin Wang, Jiliang Tang, Kai Shu, Suhas Ranganath, and Huan Liu. "What your images reveal: Exploiting visual contents for point-of-interest recommendation." In *Proceedings of The Web Conference*, 2017.

[WYC20] Qinyong Wang, Hongzhi Yin, Tong Chen, Zi Huang, Hao Wang, Yanchang Zhao, and Nguyen Quoc Viet Hung. "Next Point-of-Interest Recommendation on Resource-Constrained Mobile Devices." In *Proceedings of The Web Conference*, 2020.

[WZX18] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. "DKN: Deep Knowledge-Aware Network for News Recommendation." In *WWW*, pp. 1835–1844, 2018.

[XLS18] Hu Xu, Bing Liu, Lei Shu, and Philip 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*, 2018.

[XLS20] Hu Xu, Bing Liu, Lei Shu, and Philip Yu. "DomBERT: Domain-oriented Language Model for Aspect-based Sentiment Analysis." In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

[XTP16] Dionysios Xenos, Panagiotis Theodorakakos, John Pavlopoulos, Prodromos Malakasiotis, and Ion Androutsopoulos. "AUEB-ABSA at SemEval-2016 Task 5: Ensembles of Classifiers and Embeddings for Aspect Based Sentiment Analysis." In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, 2016.

[XXY15] Tianjun Xiao, Yichong Xu, Kuiyuan Yang, Jiaxing Zhang, Yuxin Peng, and Zheng Zhang. "The application of two-level attention models in deep convolutional neural network for fine-grained image classification." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 842–850, 2015.

[XYH17] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. "Attentional Factorization Machines: Learning the Weight of Feature Interactions via Attention Networks." In *IJCAI*, pp. 3119–3125, 2017.

[XZL20] Kuanhong Xu, Hui Zhao, and Tianwen Liu. "Aspect-Specific Heterogeneous Graph Convolutional Network for Aspect-Based Sentiment Classification." *IEEE Access*, 2020.

[YBZ17] Carl Yang, Lanxiao Bai, Chao Zhang, Quan Yuan, and Jiawei Han. "Bridging collaborative filtering and semi-supervised learning: a neural approach for poi recommendation." In *KDD*, 2017.

[YCG20] Fuqiang Yu, Lizhen Cui, Wei Guo, Xudong Lu, Qingzhong Li, and Hua Lu. "A Category-Aware Deep Model for Successive POI Recommendation on Sparse Check-in Data." In *Proceedings of The Web Conference*, 2020.

[YWW17] Hongzhi Yin, Weiqing Wang, Hao Wang, Ling Chen, and Xiaofang Zhou. "Spatial-aware hierarchical collaborative deep learning for POI recommendation." *IEEE Transactions on Knowledge and Data Engineering*, 2017.

[YYD16] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. "Hierarchical attention networks for document classification." In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pp. 1480–1489, 2016.

[YYL11] Mao Ye, Peifeng Yin, Wang-Chien Lee, and Dik-Lun Lee. "Exploiting geographical influence for collaborative point-of-interest recommendation." In *SIGIR*, 2011.

[YZC16] Hongzhi Yin, Xiaofang Zhou, Bin Cui, Hao Wang, Kai Zheng, and Quoc Viet Hung Nguyen. "Adapting to user interest drift for poi recommendation." *IEEE TKDE*, 2016.

[YZZ18] Haochao Ying, Fuzhen Zhuang, Fuzheng Zhang, Yanchi Liu, Guandong Xu, Xing Xie, Hui Xiong, and Jian Wu. "Sequential recommender system based on hierarchical attention network." In *IJCAI*, 2018.

[ZC13] Jia-Dong Zhang and Chi-Yin Chow. "iGSLR: personalized geo-social location recommendation: a kernel density estimation approach." In *SIGSPATIAL*, 2013.

[ZC15] Jia-Dong Zhang and Chi-Yin Chow. "GeoSoCa: Exploiting geographical, social and categorical correlations for point-of-interest recommendations." In *SIGIR*, pp. 443–452, 2015.

[ZCL14] Jia-Dong Zhang, Chi-Yin Chow, and Yanhua Li. "Lore: Exploiting sequential influence for location recommendations." In *SIGSPATIAL*, 2014.

[ZGH19] Jingbo Zhou, Shan Gou, Renjun Hu, Dongxiang Zhang, Jin Xu, Airong Jiang, Ying Li, and Hui Xiong. "A collaborative learning framework to tag refinement for points of interest." In *KDD*, 2019.

[Zib16] Dávid Zibriczky12. "Recommender systems meet finance: a literature review." In *Proc. 2nd Int. Workshop Personalization Recommender Syst*, pp. 1–10, 2016.

[ZLK12] Tom Chao Zhou, Michael R Lyu, and Irwin King. "A classification-based approach to question routing in community question answering." In *Proceedings of the 21st International Conference on World Wide Web*, pp. 783–790. ACM, 2012.

[ZLS19] Chen Zhang, Qiuchi Li, and Dawei Song. "Aspect-based Sentiment Classification with Aspect-specific Graph Convolutional Networks." In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019.

[ZLZ17] Zhou Zhao, Hanqing Lu, Vincent W Zheng, Deng Cai, Xiaofei He, and Yueting Zhuang. "Community-Based Question Answering via Asymmetric Multi-Faceted Ranking Network Learning." In *AAAI*, pp. 3532–3539, 2017.

[ZMF18] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. "Deep Interest Evolution Network for Click-Through Rate Prediction." *arXiv preprint arXiv:1809.03672*, 2018.

[ZMZ19] Xiao Zhou, Cecilia Mascolo, and Zhongxiang Zhao. "Topic-enhanced memory networks for personalised point-of-interest recommendation." In *KDD*, 2019.

[ZNY17] Lei Zheng, Vahid Noroozi, and Philip S Yu. "Joint deep modeling of users and items using reviews for recommendation." In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, 2017.

[ZSL15] Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis Lau. "A C-LSTM neural network for text classification." *arXiv preprint arXiv:1511.08630*, 2015.

[ZYL17] Zhou Zhao, Qifan Yang, Hanqing Lu, Min Yang, Jun Xiao, Fei Wu, and Yueting Zhuang. "Learning max-margin geoSocial multimedia network representations for point-of-interest suggestion." In *SIGIR*, 2017.

[ZYL18] Zhou Zhao, Qifan Yang, Hanqing Lu, Tim Weninger, Deng Cai, Xiaofei He, and Yueting Zhuang. "Social-Aware Movie Recommendation via Multimodal Network Learning." *IEEE Trans. Multimedia*, 2018.

[ZYZ19] Fan Zhou, Ruiyang Yin, Kunpeng Zhang, Goce Trajcevski, Ting Zhong, and Jin Wu. "Adversarial point-of-interest recommendation." In *The World Wide Web Conference*, 2019.

[ZZH15] Zhou Zhao, Lijun Zhang, Xiaofei He, and Wilfred Ng. "Expert finding for question answering via graph regularized matrix completion." *IEEE Transactions on Knowledge and Data Engineering*, **27**(4):993–1004, 2015.

[ZZK17] Shenglin Zhao, Tong Zhao, Irwin King, and Michael R Lyu. "Geo-teaser: Geo-temporal sequential embedding rank for point-of-interest recommendation." In *Proceedings of The Web Conference*, 2017.

[ZZL14] Fangxi Zhang, Zhihua Zhang, and Man Lan. "ECNU: A Combination Method and Multiple Features for Aspect Extraction and Sentiment Polarity Classification." In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, 2014.

[ZZS18] Guorui Zhou, Xiaoqiang Zhu, Chengru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. "Deep Interest Network for Click-Through Rate Prediction." In *KDD*, pp. 1059–1068, 2018.

[ZZZ18] Guanjie Zheng, Fuzheng Zhang, Zihan Zheng, Yang Xiang, Nicholas Jing Yuan, Xing Xie, and Zhenhui Li. "Drn: A deep reinforcement learning framework for news recommendation." In *WWW*, pp. 167–176, 2018.