Doctoral theses at NTNU, 2021:380

Lemei Zhang

Exploring Multifaced User Modelling in Textual Data Streams

Doctoral thesis

NTNU
Norwegian University of Science and Technology
Thesis for the Degree of
Philosophiae Doctor
Faculty of Information Technology and Electrical
Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

Lemei Zhang

Exploring Multifaced User Modelling in Textual Data Streams

Thesis for the Degree of Philosophiae Doctor

Trondheim, December 2021

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU
Norwegian University of Science and Technology

Thesis for the Degree of Philosophiae Doctor

Faculty of Information Technology and Electrical Engineering
Department of Computer Science

ISBN 978-82-326-5551-9 (printed ver.)
ISBN 978-82-326-6086-5 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (online ver.)

Doctoral theses at NTNU, 2021:380

Printed by NTNU Grafisk senter

Abstract

User modelling technologies play an important role in the success of many online applications such as recommender systems. However, it is far from enough to solve the cold-start issue and data sparsity problem commonly existing in the real-world dataset purely relying on user-item interactions. To this end, the objective of this doctoral thesis is to develop effective user modelling approaches to build high-quality user profiles for better mining users' intrinsic and potential interests while alleviating cold-start and data sparsity issues raised from traditional collaborative filtering methods. Specifically, we focus on analyzing and exploiting user/item related attributes and auxiliary information knowledge from online data streams to obtain users' needs or preferences.

To leverage attributes of users/items, such as time, location, news title and article content, we first proposed a neural time series forecasting model (NTSF) to draw users' interest patterns over time on Twitter which takes emerging topics, users' intrinsic interests, users' recent behaviors and cyclic patterns of users into consideration. To jointly capture sequential patterns in streams of clicks and various item semantic features, we further devise a Deep Joint Neural Network (DeepJoNN) which consists of two parts of deep neural networks (CNN and RNN) coupled together in a hierarchical way. Considering the uncertainty of user behaviors in textual data streams, we propose a dynamic attention-integrated neural network to integrate spatial-temporal, semantic, inter- and intra-session features in a unified framework for modelling complex dynamic user interests.

We also study the auxiliary information, especially knowledge bases or knowledge graph (KG), in the user of improving user profiles for effective recommendations. Specifically, we firstly investigate the recent research progress about recommending on graphs. To explore the influence of semantic features inferenced from KGs on user modelling and multiple relations in KGs in revealing user intents, we then propose a novel Relational Knowledge-aware Heterogeneous Graph Attention Network, ReKaH_GAT, which fuses item sequential information within sessions and path connectivity with relations in KGs to understand user intents and improve the interpretability of recommender systems.

Through extensive evaluation, we show that our proposed user-modelling approaches perform better than traditional methods in user behavior prediction and recommendation tasks.

Preface

This thesis is submitted to the Norwegian University of Science and Technology (NTNU) for partial fulfilment of the requirements for the degree of philosophiae doctor.

This doctoral work has been performed at the Department of Computer Science (IDI), NTNU, Trondheim, with Professor Jon Atle Gulla as main supervisor and with Professor Kjetil Nørvåg and Associate Professor Xiaomeng Su as co-supervisors.

The thesis has been part of the research project Recommendation Technologies (RecTech), supported by the Research Council of Norway, grant number 245469 under the BIA innovation research program.

Acknowledgements

First of all, I would like to express my sincere gratitude towards my supervisor Professor Jon Alte Gulla for his excellent supervision and great support during my PhD study. Without his patient guidance, encouragement, immense knowledge and advice, I would not be able to finish my PhD career. The knowledge and methodology I learned from him are invaluable not only to my research but to my life. I would like to thank Professor Kjetil Nørvåg, Professor Xiaomeng Su for their continuous support, insightful discussion and patient guidance.

I would like to express my deep gratitude to the partners of the RecTech project, and Adresseavisen in particular, for giving me the opportunity to pursue my PhD study. Many thanks to all of my co-authors: Dr. Özlem Özgöbek, Rolf Dyrnes Svendsen, Agnes Stenbom, and Jørgen Frøland. I appreciate all their efforts on the research work and I enjoy the collaboration with them all the time. I would also like to appreciate all my fellow friends at Web Intelligence and Semantics Lab for their constant companion and precious friendship: Dr. Cristina Sanchez Marco, Nils Barlaug, Yujie Xing, and Tu My Doan. Many thanks to Elena Volkova, Ana Rita Sousa and Arne Sund for their technical support and close collaboration. I am truly thankful for the help and support from the department administrations and technical staff.

Finally, I would like to sincerely thank my family for their love, encouragement and constant support. A special thanks to my husband, Peng being always with me, and my daughter, Xiaobei, who is like light coming in from a dark place and makes me want to be a better person.

Contents

List of Figures

List of Tables

Part I

Introduction and Literature Review

Chapter 1

Introduction

This chapter presents an overview of research work conducted during my PhD study. In Section 1.1, we present the motivation behind our research, while Section 1.2 briefly explains the research context. In Section 1.3, the research goals and questions are discussed, followed by our research contributions and approaches presented in Section 1.4 and 1.5 respectively. Then Section 1.6 summarizes our publications included in this thesis. Finally, we describe the structure of the rest of the thesis.

1.1 Motivation

With the proliferation of online user activities, users are generating a large volume of streams every day. Consequently, such an information-rich online world raises new opportunities in various applications and areas. One major research domain is how to customize or adapt systems according to users' specific needs, which falls into the category of user modelling discipline. As a subdivision of human-computer interaction research, the fundamental goal of user modelling is to provide users with experiences fitting their specific background knowledge and objectives [1], which is intuitively desirable and have significant implications for both individuals for improving the online experience as well as satisfaction, and industrial companies for increasing profits, improving societal reputation as well as developing potential customers. However, facing the information overload, where users are provided with a variety of information varied from forms and topics before they can isolate what they really need, challenges lie in saying the "right" thing at the "right" time in the "right" way [1] taking into account different aspects of objective and subjective factors.

Research effort has been undertaken to model users' preferences to help users find their interested items by analyzing their historical behaviors with barely user-item interactions using collaborative filtering. However, for some online platforms with textual data streams, collaborative filtering often suffers from cold-start and data-sparsity issues faced with newly registered users, inactive users, anonymous users, or items with few historical interaction records available, which may render the generation procedure of user personalization not necessarily indicating the users' intrinsic interests. As such, attributes of users/items can be a beneficial supplementary material for modelling user preferences while alleviating cold-start and data-sparsity issues when there are countable user-item interactions. Attribute features can be relevant information appended to each interaction the user towards an item [2]. For instance, some datasets contain time and location accompanied by a user's interaction with a certain item. Content-aware attributes such as news title and article content can be acquired together with user clicking events. Such interaction-relevant attributes not only provide a context for user interaction, but also affects user interaction decision to some extent, and may even determine the user's current tendency toward an item. Other terms may be used to indicate attributes interchangeably such as features, taxonomy, entities, contextual information, etc.

Based on their inherited characteristics, different attributes related to building user profiles for better-personalized systems and services in this thesis can be categorized into explicit and implicit attributes. Explicit attributes refer to attributes related to users/items that can be extracted by experts, such as category, entities, keywords of articles, or directly attached to user/item like timestamp, location etc. Implicit attributes are latent features that cannot be acquired directly but should be inferred through analyzing data streams such as topics, intra- and inter-session properties, or discovered in the form of representations learned by models e.g. from article content or user-generated tweets.

To incorporate explicit and implicit attributes for better modelling user preferences, several challenges need to be considered: (1) Different types of attributes. Attributes usually appear in different forms. For instance, categorical features and locations can be expressed using predefined proper nouns in text, the timestamp is usually a string of numbers, article titles and contents are written with formal expressions, while user-generated tweets are in the form of free texts full of abbreviations and misspellings. Then how to process such attributes presented in different forms and turn them into a machine-readable language need to be seriously considered. (2) Different effects of attributes on user profiling process. Obviously, different attributes normally play different roles in influencing a user's next decision. For instance, some people prefer to read news with different topics or categories at different time of the day. They may tend to learn about current affairs during the day but prefer to read some entertainment gossip during the night break. These users can be seen as time-driven which time factor probably plays a more important role in affecting user preferences than other factors. (3) Latent/implicit attributes distillation. There is no doubt that the content of a news article or a tweet is the key to whether a user wants to continue reading. However, implicit signals from content cannot be clearly depicted or explicitly expressed. Furthermore, reading or clicking of an item cannot fully reveal the personal interest or need of the users. They may click on an article they do not want to read due to mistakes or randomness. Thus how to effectively extract and mine such information from news articles or tweets that affect the user's decision needs to be carefully considered.

Apart from the user/item attributes that can be exploited to capture and mine user's preferences in different scenarios, auxiliary information especially knowledge bases or knowledge graph (KG) can also be valuable external resources to enrich the user profiles. It is generally regarded as an effective means to uncover the item relationships by providing heterogeneous information related to items, such as different entities and relations. Meanwhile, items and their relations with entities can be naturally formed into graphs that intuitively reveal the potential correlations among items via indirect intertwined links. Furthermore, graph architecture makes it easier for users to understand the recommendation results to some extent. To effectively integrate KG into recommender systems for better understanding and mining user interests, we need to consider the following challenges and issues: (1) How to model the heterogeneity of the KG? Compared with homogeneous networks where there are only one type of nodes and links, KGs consist of multi-typed nodes and links, such as Pep Guardiola$\xrightarrow{sport}$Football Player, Pep Guardiola$\xrightarrow{memberOf}$FC Barcelona. As can be seen that "Football Player" (occupation) and "FC Barcelona" (football team) connect "Pep Guardiola" with different relations, namely "sport" and "memberOf". Such heterogeneity usually carries various semantic information and can be beneficial for inferencing the subtle item relationships from different perspectives. Thence to encode such heterogeneous information in KG is a widely acknowledged yet non-trivial task for better user modelling. (2) How to design a novel data-driven user modelling approach that can capture the semantics of graphs and meanwhile take into account the different contributions of different nodes/paths related to a specific item? The candidate item contains entities being reached through one hop, two hops or more hops from starting node, resulting in different paths in KG. In other cases, different routes can lead to different end nodes from the same starting point within the same number of hops. These paths generally convey distinct semantic relations and typically are of different importance in characterizing user preferences over items or nodes. It is very likely that certain paths can better describe a user's inclination than others. Moreover, users may have different intents which drive users to consume different items. Relations in KG can be modelled as an intermedium to describe user intents. In the case of Pep Guardiola$\xrightarrow{memberOf}$FC Barcelona, the user may emphasize the organization (football team in this case) more, while in the case of Bernabé Martí$\xrightarrow{\text{hasOccupation}}$Opera singer, the user may emphasize occupation (e.g. opera singer) and related activities (e.g. opera) more than others. Therefore, how to fully exploit the differences of importance and contributions of related nodes, and mine user intent in-depth, is a desirable research problem for better understanding and extravagating user's preferences. (3) How to effectively train a recommendation model on the constructed large-scale graph with millions even billions of nodes and links? In such a big-data era, real graphs can easily have millions or billions of nodes and links, such as some well-known social networks and e-commerce networks [3], which leads to high costs in terms of both time and space for user modelling and recommendation. To this end, it is necessary to study more efficient and meanwhile accurate user modelling techniques that enable large-scale computations on graphs.

To sum up, the motivation of this thesis comes from the demands on exploring effective and efficient user modelling techniques to build high-quality user profiles for better mining users' intrinsic and potential interests, support decision-making systems while alleviating cold-start and data sparsity issues raised from traditional collaborative filtering methods. In this direction, the thesis especially focuses on analyzing and exploiting user/item related attributes and auxiliary information knowledge from online data streams to obtain users' needs or preferences. Furthermore, the research preliminarily investigates the explainability of deep user profiling methods in recommender systems.

1.2 Research Context

The research work in this PhD thesis has been carried out as a part of a four-year PhD program at the Department of Computer Science at Norwegian University of Science and Technology within the project Recommendation Technologies (RecTech). The RecTech project is funded by the Research Council of Norway under the BIA innovation research program with project number 245469. RecTech is performed in cooperation with Adresseavisen/ Polaris Media, Cxense in Oslo, NTNU in Trondheim and VTT in Finland.

The main objective of the RecTech project is to research and develop the next generation recommender systems for news as well as other social media streams. User profiling and deep content analysis are two main tasks, of which key technologies including computational linguistics, machine learning and big data mining play a central role in RecTech.

1.3 Research Questions

The general goal of this thesis can be summarized as ***understanding and modelling complex dynamic user interests from multiple aspects in textual data streams***. Specifically, we focus on two kinds of significant contextual resources, called attributes of users/items and auxiliary information, for efficient user modelling. To drive our research, we identified the following two main research questions:

- **[RQ1]** – *How can attribute features of users and items be learned and integrated for effective user modelling and recommendation?*

  In the first research question, we aim to explore the influence of users/items' explicit and implicit attribute features for dynamic user modelling, especially for cold-start scenarios. Furthermore, we also want to investigate a unified framework combining explicit and implicit attribute features to model complex dynamic user interests.

  This general research question can be divided into three concretized sub-questions:

  RQ1.1 What kinds of temporal patterns can be leveraged to predict dynamic user interests in textual data streams?

  RQ1.2 How can item-level semantic features (e.g. categories, keywords, titles) be applied to alleviate the cold start issue in session-based recommendation?

  RQ1.3 How can the spatial-temporal, semantic, inter- and intra-session features be integrated into a unified framework for modelling complex dynamic user interests and effective recommendations?

- **[RQ2]** – *How can auxiliary feature from knowledge graph be employed to understand user intents and improve the explainability of recommender systems?*

  In the second research question, we want to find the recent research progress about recommending on graphs which has attracted considerable interests in both research and industry communities, and meanwhile provide in-depth insights on graph-learning based recommendation approaches through systematically taxonomic analysis. Based on this, we would like to understand the influence of the knowledge graph on recommendation performances and how to integrate the knowledge graph into user modelling procedures.

  This general research question can be divided into two concretized sub-questions:

  RQ2.1 How to understand data-driven mechanisms behind graph-learning based recommendation approaches through taxonomic assessment on recent advances?

  RQ2.2 Does explicitly modelling of relations in KG help capture user intents for improving session-based recommendation performance and explainability?

1.4 Research Contributions

This section summarises five main research contributions in the thesis, in accordance with the research questions presented in Section 1.3.

A novel neural time series forecasting model for personalized time-aware user interests prediction [C1]

User's interests present dynamic time-aware patterns. To explore the interaction among these patterns in affecting the evolution of user's personalized interests, we propose a neural time series forecasting model which takes emerging topics, user's intrinsic interests, user's recent behaviors and cyclic patterns of users into consideration. Furthermore, Long Short-Term Memory Recurrent Neural Network (LSTM-RNN) is employed to differentiate and quantify various types of user's interest patterns automatically. Unlike existing research that only discusses a certain attribute of time presentation, we discuss the multi-dimensional time characteristics of users in Twitter. Our empirical analysis is performed on real-world Twitter datasets.

An efficient deep joint network for session-based recommendations with

contextual augmentation [C2]

To alleviate the cold start issue, we have proposed a Deep Joint Neural Network (DeepJoNN) which could jointly model the sequential pattern of session clicks and various item features such as ID, category, keywords and entities for session-based recommendation. DeepJoNN consists of two parts of deep neural networks (CNN and RNN) coupled together in a hierarchical way and thus could extract contextual patterns and process long and short-term dependencies simultaneously. At the same time, character-level embedding over input features is adopted to allow integrating different types of data and reduce engineering computation. The effectiveness of our proposed tensor-based CNN module is verified through experimental results on two real-world datasets.

A novel dynamic attention-integrated neural network to model user's interests over time in a unified framework for session-based recommendation [C3]

We propose a novel neural network framework, dynamic attention-integrated neural network to model user's dynamic interests over time in a unified framework for personalized session-based recommendation. The proposed model can jointly exploit users' long-term interests, user behavior sequence patterns, users' main purpose in the current session, as well as public behavior mining to model users' preferences. In order to improve the recommendation accuracy, dynamic topic modelling and convolutional neural network (CNN) sentence model are adopted to effectively learn the item semantic embedding. More importantly, to handle diverse variance of users' clicking behavior, we introduce a novel attention scheme that would dynamically assign influence factors on recent models based on the users' spatio-temporal reading characteristics. The fusion of various side information and the effectiveness of different fusion strategies in session-based news recommendation have been verified through empirical analysis with real-world datasets.

A taxonomic assessment on graph learning-based recommender system approaches [C4]

We propose a novel taxonomy to categorize various graphs in the Graph Learning-based Recommender Systems (GLRSs) and analyze their characteristics from a data-driven perspective. Then, we propose a novel taxonomy to classify existing graph-learning based recommendation approaches, which clearly demonstrates the evolution process of recent studies. Furthermore, the resources regarding GLRSs, including benchmark datasets and open-source knowledge graphs, are systematically summarized. Finally, we analyze the limitations of existing works and suggest a few future research directions of GLRSs such as dynamicity, interpretability, fairness and so on.

A relational knowledge-aware heterogeneous graph attention network for user intents modeling and session-based recommendation [C5]

We propose a novel Relational Knowledge-aware Heterogeneous Graph Attention Network (ReKaH_GAT) to model entity-relation interactions and user intents explicitly for session-based news recommendation. In ReKaH_GAT, we design an original transformation schema from traditional KG to Entity-Relation Interaction (ERI) graph where the complex graph structure, entity and relation semantics are embedded in a unified way. A novel heterogeneous graph attention network with a self-attentive layer is applied subsequently to learn the context and intent embeddings from a session-specific ERI. Meanwhile, the semantic session embedding learned from pre-trained multi-lingual BERT is combined with contextual and intentional session embeddings to achieve a robust news recommendation.

1.5 Publications

In this section, we present the list of scientific papers published during the PhD studies that cover the above contributions. For each paper, we refer to the corresponding chapter in which the content of the paper is included and point out the relevance of the aforementioned research questions.

P1. Zhang, Lemei, Peng Liu, and Jon Atle Gulla. *A neural time series forecasting model for user interests prediction on Twitter*. In Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization, pp. 397-398. 2017.

Summary: The content of this paper is included in Chapter 3 and is aimed at answering the research question **RQ1.1**.

P2. Zhang, Lemei, Peng Liu, and Jon Atle Gulla. *A deep joint network for session-based news recommendations with contextual augmentation*. In Proceedings of the 29th on Hypertext and Social Media, pp. 201-209. 2018.

Summary: The content of this paper is included in Chapter 4 and is aimed at answering the

Table 1.1: Relations between contributions and publications.

Papers	C1	C2	C3	C4	C5
P1	•				
P2		•			
P3		•			
P4			•		
P5				•	
P6					•

research question **RQ1.2**.

P3. Gulla, Jon Atle, Lemei Zhang, Peng Liu, Özlem Özgöbek, and Xiaomeng Su. *The adressa dataset for news recommendation*. In Proceedings of the International Conference on Web Intelligence, pp. 1042-1048. 2017.

Summary: The content of this paper is included in Chapter 4 and is aimed at answering the research question **RQ1.2**.

P4. Zhang, Lemei, Peng Liu, and Jon Atle Gulla. *Dynamic attention-integrated neural network for session-based news recommendation*. Machine Learning 108, no. 10 (2019): 1851-1875.

Summary: The content of this paper is included in Chapter 5 and is aimed at answering the research question **RQ1.3**.

P5. Zhang, Lemei, Peng Liu, and Jon Atle Gulla. *Recommending on graphs: a new perspective for recommender systems*. User Modeling and User-Adapted Interaction, 2nd round review.

Summary: The content of this paper is included in Chapter 6 and is aimed at answering the research question **RQ2.1**.

P6. Zhang, Lemei, Peng Liu, and Jon Atle Gulla. *Demystifying Knowledge-aware User Intents for Session-based News Recommendation*. In review with International Conference on Advanced Information Systems Engineeringing (CAiSE) 2022.

Summary: The content of this paper is included in Chapter 7 and is aimed at answering the research question **RQ2.2**.

As a summary, Table 1.1 presents the relations between the papers and our research contributions listed in section 1.4.

Additional Publications. In the course of this PhD, I also contributed to the following publications, but they are not included in this thesis because they are not directly connected to its research topic.

A1. Jon Atle Gulla, Rolf Dyrnes Svendsen, Lemei Zhang, Agnes Stenbom, Jørgen Frøland. *Recommender Systems in Online News Personalization*. Accepted by AI Magazine.

A2. Liu, Peng, Lemei Zhang, and Jon Atle Gulla. *Multilingual Review-aware Deep Recommender System via Aspect-based Sentiment Analysis*. ACM Transactions on Information Systems (TOIS) 39, no. 2 (2021): 1-33.

A3. Liu, Peng, Lemei Zhang, and Jon Atle Gulla. *Dynamic attention-based explainable recommendation with textual and visual fusion*. Information Processing & Management 57, no. 6 (2020): 102099.

A4. Liu, Peng, Lemei Zhang, and Jon Atle Gulla. *Real-time social recommendation based on graph embedding and temporal context*. International Journal of Human-Computer Studies 121 (2019): 58-72.

A5. Liu, Peng, Lemei Zhang, and Jon Atle Gulla. *Learning Multi-granularity Dynamic Network Representations for Social Recommendation*. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 691-708. 2018.

A6. Liu, Peng, Jon Atle Gulla, and Lemei Zhang. *Dynamic topic-based sentiment analysis of large-scale online news*. In International Conference on Web Information Systems Engineering, pp. 3-18. 2016.

1.6 Thesis Structure

The thesis is divided into four main parts. Part I gives an introduction to the main topics of the thesis and summarises the technical background in these areas. Then, Part II and Part III present our research on complex dynamic user interests modelling from multiple aspects. In particular, Part II focuses on exploiting attribute information of users/items, Part III focuses on the integration of auxiliary features from graph-structured data. Finally, we conclude the thesis and give an overview of future work in Part IV. A more detailed outline of the contents is given as follows:

Part I Introduction and Literature Review

Chapter 1 introduces the motivation and context of our research, presents the research questions studied and summarizes the contributions of the thesis.

Chapter 2 overviews the background knowledge and some state-of-the-art approaches related to this thesis.

Part II Mining Attribute Information for User Modelling

Chapter 3 investigates temporal attributes that affecting user's preference from multiple perspecitives.

Chapter 4 studies the problem of extracting and modelling semantic features for better understanding users' preferences in session settings.

Chapter 5 presents a framework for integrating the explicit and implicit user/item related attributes for session-based news recommendation, and provides the evaluation results of our proposed method.

Part III Exploring Graph Structured Data for User Modelling

Chapter 6 describes the recent advances and challenges of graph learning-based recommender systems.

Chapter 7 presents and evaluates a graph learning-based approach for predicting next item interacted by the user by modelling large-scale knowledge graph, and provides explaniations based on recommendation results.

Part IV Conclusions and Future Work

Chapter 8 concludes our research by recisiting the research questions addressed in this thesis, and provides an outlook on future research directions.

Chapter 2

Literature Review

In this chapter, we briefly describe fundamental knowledge in the research area of user modelling that can facilitate the understanding of the content of this thesis. We start with describing the preliminaries related to acquiring and constructing user profiles and modelling temporal features in Section 2.1. Then in Section 2.2, we review the modern applications and techniques for user modelling, followed by evaluation procedure and benchmark dataset used in our thesis in Section 2.3.

2.1 Technological Background

2.1.1 User Profile Acquisition and Construction

User profiling is an attempt to deal with information about internet users. It is a crucial and fundamental part of the rest user modelling process. In this section, we briefly introduce several methodologies attempting to build user profiles related to this thesis.

Term Frequence-Inverse Document Frequency (TF-IDF) [4]. *Term Frequence (TF)* is the number of occurrences of the word w in a document d, denoted as $TF(w, d)$. The higher the value of $TF(w, d)$ is, the more the word w is representative of document d. Accorddingly, *Document Frequence (DF)* is the number of documents in which the word w occurs, denoted as $DF(w)$. The Inverse Document Frequence (IDF) of the word w is given as follows:

$$IDF(w) = 1 + \log\left(\frac{|D|}{DF(w)}\right)$$

Where $|D|$ is the number of documents. Therefore, when $|D|$ is fixed, the more documents the word w occurs, the lower the value of the $IDF(w)$ is, which means that the term w is less representative of the document with lower $IDF(w)$ score. On the other hand, the higher $IDF(w)$ socre normally indicates the term w has a better ability to distinguish documents.

It is truly that we prefer higher TF and higher IDF to find the more representative and higher discriminative ability terms to build user profiles. Hence, incorporating such requirements into a single formula can be expressed as:

$$TF - IDF(w, d) = TF(w, d) \times IDF(w)$$

Higher TF-IDF score is desired for terms to be selected as representative features for constructing user profiles.

Topic model. The simplest topic model, known as Latent Dirichlet Allocation (LDA) [5] assumes that documents exhibit multiple topics, where a topic is defined to be a distribution over a fixed vocabulary of terms. It is a Bayesian network that generates a document using a mixture of topics. In its generative process, for each document d, a multinomial distribution θ_d over topics is randomly sampled from a Dirichlet with parameter α. To generate each word, a topic $z_{d,n}$ is chosen from this topic distribution, and a word $w_{d,n}$ is generated by randomly sampling from a topic-specific multinomial distribution $\beta_{z_{d,n}}$ parameterized by η. This can be illustrated as a directed graphical model in Figure 2.1. LDA assumes that words are exchangeable within each document, and documents are exchangeable within the corpus. However, for many corpora like news articles, the latter assumption is inappropriate since the documents reflect evolving content and topics change over time. Thus, a variant of LDA, Dynamic Topic Model (DTM) [6], which captures the evolution of topics in a sequentially organized corpus of documents, is more suitable for scenarios where topics change dynamically.

Specifically, in DTM, the documents are firstly divided by time slice, e.g., by week. Assuming there are K topics in a corpus, the documents of each slice can be modeled with a K-component topic model, where the topics associated with slice t evolve from the topics associated with slice $t - 1$. The generative process is given as follows:

1. Draw topics $\pi_t | \pi_{t-1} \sim \mathcal{N}(\pi_{t-1}, \sigma^2 I)$
2. For each document:
    a. Draw $\theta_d \sim Dir(\alpha)$
    b. For each word:
        i. Draw $Z \sim Multi(\theta_d)$
        ii. Draw $W_{t,d,n} \sim Multi\left(f\left(\pi_{t,z}\right)\right)$

Where $\mathcal{N}(\cdot)$ is Gaussian distribution and $\pi_{t,k}$ denotes a multivariate Gaussian random variable for topic k in slice t. $Dir(\cdot)$ denotes Dirichlet distribution, $Multi(\cdot)$ denotes Multinomial distribution, and $f(\pi_{t,z}) = \frac{\exp(\pi_{t,z,i})}{\sum_i \exp(\pi_{t,z,i})}$ is the function that maps the real-vector $\pi_{t,z}$ to the simplex. As illustrated with graphical represnetations of DTA in Figure 2.2, different from LDA, the topics are drawn from logistic normal rather than a Dirichlet. Besides, each time slice is a separate LDA model, where the k-th topic at slice t has smoothly evolved from the k-th topic at slice $t-1$. Rather than a single distribution over words generated by LDA, in DTA, a topic is a sequence of distributions over words representing the underlying changes of the theme of the corpus over time.

Fast Fourier Transform (FFT). Fourier Transform is a mathematical operation that changes the domain of a signal from time to frequency. It is a method for expressing a function as a sum of periodic components, and for recovering the signal from those components. When both functions and its Fourier Transform are replaced with discretized counterparts, it is called the discrete Fourier Transform (DTF). The reason why DFTs can be widely spread and applied in many domains is that they have a fast and effective computing algorithm, called the Fast Fourier Transform (FFT), which is known to Gauss as early as 1805 [7] and was brought to light in its current form by Cooley and Tukey [8]. Through FFT, the periodicities in input data and as well as the relative strengths of any periodic components and be revealed. Specifically, assuming the input sequence can be represented as a vector f_n in time domain, Then the FFT transformation process could be defined as

$$F_k = \sum_{n=1}^{N-1} f_n e^{-\frac{i2\pi kn}{N}}$$

Where F_k is the representation of f_n in frequency domain, N is the dimension of f_n. Then the peak values in F_k can be viewed as the cycles. One time series can probably have multiple cycles since the orginal input can be a combination of multile signals. In this thesis, FFT is leveraged to detect the periodic features in user preferences.

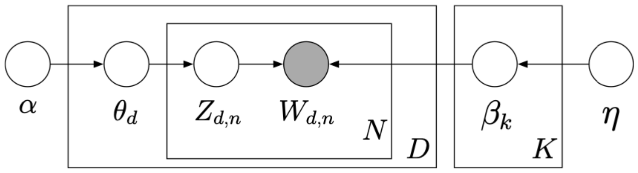


Figure 2.1: A graphical representation of LDA, where nodes represent random variables and edges represent dependence between random variables. Shaded node denotes observed variable otherwise is unobserved variables. The rectangular boxes denote replication.

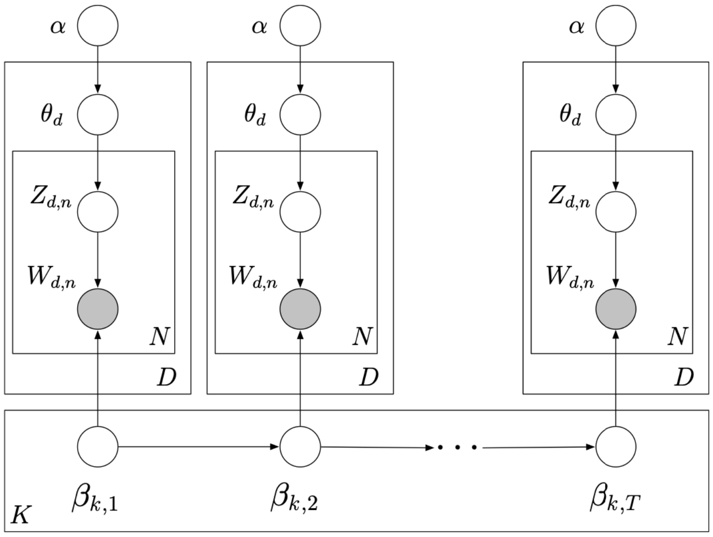


Figure 2.2: A graphical model representation of DTM, where each topic's parameter $\boldsymbol{\beta}_{t,k}$ evolve over time.

2.1.2 Temporal Feature Modelling

As is well-known that user interests are changing over time, temporal aspects play a crucial role in modelling user preferences. Two widely spread approaches for modelling user dynamic properties in conventional recommender systems include sliding window and time decay function.

Sliding Window. The sliding window method is adopted for sequences where order matters. For time-series data, the sliding window approach takes a series of data from previous time steps within a predefined window size as input and output the next time step(s) as prediction(s). Before modelling the dynamics of users' interests, the user interaction data or interested topics need to be arranged in chronological order, which is then divided according to the predefined window size. Then modelling operation can be performed on the data in the time window in turn. The window size can be defined by a fixed number of items, e.g. the recent 200 clicked items, or a period of time, e.g. one day or one week. For instance, the authors of [9] capture the dynamic level of user interests by building user profiles daily over the extracted topics from Twitter. Then weekly historical user profiles are adopted to learn user interests properties. Yin et al. [10] define a multi-granularity of time intervals to capture user-oriented and time-oriented topics from different levels, followed by a unified probabilistic model to model user behaviors for efficient recommendation.

Decay function. A time decay function is often leveraged on temporal and streaming data analysis to reduce the importance of older data, without eliminating their influence, on the results of the analysis, among which exponential time decay is commonly used in practice, compared with other decay functions e.g. polynomial decay. Different from sliding window methods that choose to consider within a limited range of historical data, the decay function tends to keep the whole historical records but assign fewer weights on the older ones compared with more recent user behaviors. The intuition behind the decay functions for modelling user interests is that recent interests should contribute more than old ones. A general time decay function can be defined as

$$y = A_0 x^{-\lambda(t-t_0)}$$

Where A_0 denotes the value at time zero, λ is a positive constant that determines the rate/percentage of decay. t_0 represents the time zero and t represents the current time.

Many researches adopt the decay function in the process of modelling user preferences with respect to historical user-generated content i.e. on long-term user interest profiles. For instance, Abel et al. [11][12] have observed that a user's interests change over time and have modelled user interests in specific time frames as a set of weighted topics. The weights are calculated based on a proposed time-sensitive interest decay function according to the temporal distance between the topics occurrence time and current time. Similarly, Amr et al. [13] have leveraged an exponential time decay function to weigh user historical behaviours to provide a more complete picture of user interests and more accurate profiles.

2.2 Applications for User Modelling

2.2.1 User Interest Prediction

User behavior has been widely studied directly and indirectly in many areas according to time [14, 15, 16, 17, 18]. In [14], the authors introduce TUMS, a Twitter-based User Modeling Service to infer semantic user profiles from the messages people post on Twitter. Other researches such as [15] model tie-strength between two users to do recommendations for social streams. Recently, many researches have been conducted using time series technologies to forecast some aspects of users' online behavior, especially in recommendation systems. Radinsky et al. [16] proposed a dynamics model learner (DML) which is based on a space-state model and considers trend, periodicity, noise, surprise and seasonality detection extracted from a user searching logs on the web. Preum et al. [17] explored the activity patterns of temporal user behavior using a multi-scale adaptive personalized (MAPer) model to forecast user activity linearly. Besides, the Fourier-assisted Auto-Regressive Integrated Moving Average (FARIMA) process is proposed by [18] to tackle the year-long seasonal period of purchasing data to help product recommendation. However, some of the aforementioned models do not consider personalized multi-features modelling, and others do not take into account the dynamic changes of users' interests in the online environment.

2.2.2 News Recommendations

News recommendation aims to recommend to users the news that matches their personal interests best [19]. As a popular service and an important way to retain users, the industry puts much effort into news recommendation researches [20]. Several adaptive news recommending systems, such as Google News and Yahoo! News provide personalized news recommendation services for a substantial amount of online users. Conventional news recommendations can be roughly categorized into three groups: collaborative filtering, content-based filtering and hybrid methods. The first one makes use of news ratings by users to provide recommendation services, and they are content-free. In practice, most collaborative filtering systems are constructed based on users' past rating behaviors, either using a group of users "similar" to the given user to predict news ratings [21] or modelling users' behaviors in a probabilistic way [22]. However, collaborative filtering is ineffective for the cold-start problem. Content-based methods try to sequentially find newly-published articles similar to the user's reading history in terms of content. Generally speaking, news content is often represented using vector space model (e.g., TF-IDF) [23], or topic distributions obtained by language models (e.g., PLSI and LDA), and specific similarity measurements are adopted to evaluate the relatedness between news articles. However, in some scenarios, simply representing the user's profile information by a bag of words is insufficient to capture the exact reading interest of the user. Hybrid solutions combine two or more recommendation methods to gain better performance. Representative examples include Rao et al. [24], in which the inability of collaborative filtering to recommend news items is alleviated by combining it with content-based filtering.

With the surge of deep learning techniques, neural network-based recommender systems have attracted increasing attention and achieved superior performance than previous models due to their ability to capture complex nonlinear user-item relationships [25]–[28]. For instance, in [28], the authors adopt reinforcement learning to model future rewards and consider user feedback as a supplement. A duelling bandit gradient decent strategy is incorporated to explore new attractive news for users. More recently, the attention mechanism is introduced from the machine translation domain to capture the important parts from candidates [29]–[31]. The work of [29] adopt multi-head self-attention to capture the relatedness between the news, and additive attention to learn important words. Another attention mechanism is proposed to learn the importance of user profiles from different views in [30]. To enrich user/item profiles with extra sources, recent studies begin to incorporate KGs into news recommendations. DKN proposed in [32], attempts to learn entity embeddings with CNNs in KGs while using an attention module to match candidate news articles.

Other important factors related to news recommendations include news recency issues and data sparsity problems. Many online users read limited news stories compared with the entire repository, and hence the access matrix is very sparse. Cold start problems caused by newly registered users will also lead to a sparse problem. To address this problem, model-based collaborative (i.e. matrix factorization, probabilistic matrix factorization) is most commonly adopted to reduce dimensions and consequently reduce the level of sparsity [33, 34]. To alleviate the recency issue in news recommendation, Amr et al. [35] report take a list of articles as input, which have been selected in advance by several criteria including recency. Das et al. [20] choose to re-build the recommender models every hour in order to present the freshest information to the users. In our work, we adopt the time-decay function to reduce the weight of the historical news articles, and character-level encoding to alleviate the sparsity problem.

2.2.3 Session-based Recommendations

Classical content-based methods and collaborative filtering do not work well in the session-based setting when no user profile can be constructed from past user behavior. A natural solution to this problem is the item-to-item recommendation approach [21], in which two items are deemed to be similar if they are frequently clicked together in the same sessions. It is a simple but effective method. However, a drawback of the item-to-item recommendation is that it does not consider click order and generates predictions based only on the last click. Figueiredo et al. [36] propose a Bayesian generative model to model click sequences. Shani et al. [37] present a Markov decision process (MDP), which incorporates the transition probability between items, to provide recommendations in a session-based manner. Learning item embeddings is another approach suitable for session-based recommendations. The authors of [38] leverage item metadata to regularize item embeddings, which makes it relevant to content-based approaches.

Recently, several studies have been done to use neural network-based models including deep learning techniques for recommendation tasks. Hidasi et al. [39] propose to use recurrent neural networks (RNN) with Gated Recurrent Units (GRU) for session-based recommendation. The model considers the first item clicked by a user as the initial input of RNN, and generates recommendations based on it. Then the user might click one of the recommendations, which is fed into RNN next, and the successive recommendations are produced based on the whole previous clicks. Tan et al. [40] further improve this RNN-based model by utilizing two crucial techniques, i.e., a method to account for shifts in the input data distribution and data augmentation. In a later work, Hidasi et al. [41] extend their previous work by combining rich features of clicked items such as item IDs, textual descriptions, and images. They use different RNNs to represent different types of features and train those networks in a parallel fashion. Jannach and Ludewig [42] combine KNN with a session-based RNN [39] demonstrating further performance gains. However, the combination scheme is a fixed weighting hyperparameter and lacks a nonlinear interaction to capture more complex relations. Li et al. [43] explored a hybrid encoder with an attention model to capture both the user’s sequential behavior and main purpose in the current session. Liu et al. [44] propose a short-term attention/memory priority model for session-based recommendation, which is capable of capturing users' general interests from the long-term memory of a session context, whilst taking into account users' current interests from the short-term memory of the last-clicks. To incorporate the user's long-term preference, Quadrana et al. [45] provide a seamless way of transferring the knowledge acquired on the long-term dynamics of the user interest to session-level and devise a Hierarchical RNN to model the user activity across and within sessions. Specifically, they involved an item-level attention mechanism that allows the decoder to dynamically select and linearly combine different parts of the input sequence.

Nowadays, Graph Neural Network (GNN) which learns the representation of graph-structured data, is broadly applied for session-based recommendation. Wu et al. [46] propose a novel session-based recommendation with GNN to model separated session sequences into graph-structured data and use GNN to capture complex item transitions. Xu et al. [47] propose a graph contextualized self-attention model, which utilizes both GNN and self-attention mechanisms to learn local dependencies and long-range dependencies respectively, for session-based recommendation. Yu et al. [48] propose a novel target attentive graph neural network (TAGNN) model for session-based recommendation. By incorporating graph modelling and a target-aware attention module, TAGNN jointly considers user interests given a certain target item as well as complex item transitions in sessions.

2.2.4 Knowledge Graph-based Recommendations

Recent studies [49, 50] have witnessed the successes of knowledge graph (KG) in mitigating data sparsity and cold start problems in recommendation due to the rich semantic information related to entities and entity relations encoded in the KG. A knowledge graph is a type of directed heterogeneous graph, typically consisting of entity-relation-entity triples (h, r, t). There are a lot of graph-based methods proposed to make use of KG in the recommendation. László et al. [49] introduce an adaptive rating estimation method, which is capable to incorporate heterogeneous information sources and improving the recommendation quality. By applying the spreading activation technique [50] on KG, this approach could provide lower rating estimation error and higher coverage for recommendation compared to those collaborative filtering methods only using user-item interactions. Later, Catherine et al. [51] propose a recommendation approach based on a general-purpose probabilistic logic system called ProPPR (Programming with Personalized

PageRank), to perform knowledge graph-based recommendations. The authors leverage the link structure of the knowledge graph as well as type information about the entities to improve predictions. Chaudhair et al. [52] present the Relation of Entities Recommendation Agent (RERA), a new content-based system that adopts a novel normalized version of Personalized Page Rank to rank candidate items for recommendation. Nevertheless, these graph-based methods only make use of the topological structure of KG without considering to model the semantics carried by KG.

In order to take advantage of the semantics of entities and entity relations in KG, state-of-the-art recommendation methods adopt meta path, which predefines the specific format and length of the paths connecting two entities in KG, to build feature space and then manually extract features from KG for a better recommendation. Yu et al. [53] propose a recommendation model (HeteRec) with meta-path-based latent features to capture the different types of relationships between entities and learn the importance of each relationship type in KG. Luo et al. [54] investigate a social network based recommendation algorithm on KG named HeteCF to model the relationships of user-item, user-user and item-item by meta-path based similarity and propose a leveraging method to evaluate the weight of different relations. Despite their success for recommendations, all existing path-based methods heavily depend on the handcrafted features.

While embedding-based approaches learn entity and relation embeddings in the same space such that items sharing similar contextual knowledge should be projected closer in embedding space. Zhang et al. [55] propose a unified framework, called Collaborative Knowledge Base Embedding (CKE), to jointly learn the item latent representations in collaborative filtering as well as items' semantic representations from the knowledge base. The empirical study demonstrates the superiority of CKE against graph and meta path-based methods. After that, Huang et al. [56] propose a Knowledge-enhanced Sequential Recommender (KSR) method which integrates the RNN-based networks with the Key-Value Memory Network (KV-MN) and incorporate KG information to enhance the semantic representation of KV-MN. Wang et al. [32] propose a deep knowledge-aware network (DKN) that incorporates knowledge graph representation into news recommendation. DKN utilizes TransE to generate the entity embedding and context embedding, then feeds them into a CNN framework to recommend. However, they sacrifice the intuitiveness and effectiveness in characterizing inter-item relations and lack the reasoning ability for recommendation results, especially when multi-hop relations occur in

KG. Recent works try to combine path-based and embedding-based methods for recommendation[57]-[59]. In the work of [57], KPRN extract paths with a predefined max length and then entities, entity properties as well as relations embeddings are learned through LSTM layer. Finally, candidate items are generated according to the predicted score on a set of paths. However, KPRN requires enumerating all the possible paths between user-item pairs, which can be impractical for large-scale KGs. Besides, it fails to give explanations on interlinks of the path or show the discriminative properties within the path. Besides, none of the aforementioned studies tries to model KG to explain user intent in a finer-grained way. Recently, Wang et al. [60] explore intents of a user by modelling different aggregations of KG relations for better recommendations and interpretability, but they fail to model entity-relation correlations and take into account the sequential patterns.

2.3 Evaluation

2.3.1 Evaluation Metrics

As the fundamental aspect of user modelling, evaluation of the quality of the inferred user interest models can be generally from two perspectives, namely qualitative analysis and quantitative analysis. Different evaluation metrics are adopted according to different evaluation manners (e.g., either from qualitative or from quantitative perspective) as well as different applications or tasks, e.g., recommendation. In this thesis, we mainly focus on two application scenarios, user interest prediction and recommendation.

Qualitative analysis is a way to measure the quality of proposed user modelling techniques from the subjective aspect. A common way of evaluating the quality of the inferred user interest is by user study, which is performed by analyzing the collected explicit feedbacks from target users on the inferred interests. A good user study needs to set clear objectives and recruit the right participants from the target audience. Another way for qualitative analysis is a case study which is to perform an in-depth and detailed examination of a particular case. Compared with user study which may have difficulty enrolling real target users and set comprehensive and targeted questionnaires, a case study does not necessarily require the participation of real users according to different needs. For instance, a case study on error samples aims to analyze the causes and reasons behind the model output. The case study of some samples aims to give examples to explain the ability of the proposed model for specific tasks or additional tasks. Therefore, a case study will be one of the main qualitative analysis methods running through our experiments and explaining the final representation of model capabilities.

Quantitative analysis focuses on quantifying the collection and analysis of data. To evaluate the quality of the inferred user interests in a quantitative way, the user interests are considered as inputs, and the differences between the model output and the groud-truth annotation can be used to measure the performance of the user modelling techniques. In different studies, the evaluation criteria used by different applications may differ. For recommendation purposes, the evaluation can be performed in either an online or offline environment. Online evaluations require interaction with real users which is not always applicable for most researchers. Thus, offline evaluation in which the participation of the target users is not usually required has become a universally applicable evaluation method.

In this thesis, the evaluation of the quality of the user modelling process will be based on the effectiveness of the recommendation and user interest prediction tasks in offline mode. Specifically, the following evaluation metrics are adopted:

Pearson Correlation Coefficient. It is the covariance of the two variables divided by the product of their standard deviations. In our thesis, pearson correlation metric is adopted to measure the differences between the predicted user interest topics and the ground-truth user interests. Given the input variables of X and Y represent the predicted and groud-truth user interest, pearson correlation can be calculated as:

$$\rho = \frac{cov(X,Y)}{\sigma_X \sigma_Y} = \frac{\mathbb{E}[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}$$

Where $\mathbb{E}(\cdot)$ is the expectation, $\sigma_X = \sqrt{\mathbb{E}[(X-\mathbb{E}[X])^2]}$ is the standard diviation of X. Similarly, $\sigma_Y = \sqrt{\mathbb{E}[(X-\mathbb{E}[X])^2]}$ is the standard diviatio of Y. Lower value of ρ is required for better prediction performance.

Mean Square Error (MSE). MSE is among the most widely used evaluation metrics for assessing the quality of a predictor or an estimator. Given vectors X and Y representing the predicted output and ground-truth data, the MSE is computed as

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - X_i)^2$$

Where n is the number of dimension of X and Y. Lower value of MSE is desired for the better performance of the proposed model.

Precision and Recall [61]. They are widely used for evaluating the recommendation performance on the ranked list of top k items output from the recommendation model, which are denoted as Precision@k and Recall@k. Precision@k is the proportion of recommended items in the top-k set that are relevant:

$$Precision@k = \frac{number\ of\ relevant\ recomemnded\ items}{total\ number\ of\ recommended\ items}$$

Recall@k is the proportion of relevant items found in the top-k recommendations:

$$Recall@k = \frac{number\ of\ recommend\ relevant\ items}{total\ number\ of\ relevant\ items}$$

F1 Score. It is the harmonic mean between recall and precision values and can be denoted as

$$F1 = \frac{2 \times Precision@k \times Recall@k}{Precision@k + Recall@k}$$

The abovementioned metrics are used for evaluating the accuracy of the recommendation results. To evaluate the ranking quality of the recommendation list generated by recommender system, one can adopt Mean Reciprocal Rank.

Mean Reciprocal Rank (MRR) [62]. It is defined as the average of the reciprocal ranks of the desired items. The rank is set to zero if it is above k. MRR@k can be defined as

$$MRR@k = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{rank_i}$$

Where $rank_i$ refers to the rank position of the first relevant item for the target user.

2.3.2 Benchmark Datasets

The investigation and diffusion of user modelling can benefit from the publicly available datasets with real users, which empower the evaluation of user modelling by enabling the researchers to perform experiments on the same benchmark datasets. Many real user data, due to privacy issues, are accessible to a restricted number of researchers. Despite these limitations, efforts were still made by some institutes and organizations to create and publish benchmark datasets for academic use. Though some companies cannot provide published datasets due to privacy or commercial reasons, they provide researchers with

APIs used to obtain user data under certain conditions, such as Twitter and Last.fm. Next, we will introduce the datasets we employ in our study.

Twitter Dataset. The first dataset was acquired from Twitter official API. Twitter is clearly one of the most popular services on the Social Web. People could post their feelings, ideas and topics of interest with a limit of 140 characters before 2017. With millions of active users generating micro posts, user-generated content on Twitter can be used for analyzing users' preferences and exploring temporal user modelling. To obtain the data, we first selected a few users as "seeds". Then we crawled the friends of the seed users and the friends of their friends by requesting Twitter API. It is like a water ripple with the seed users as the centre, diverging to the surroundings to find more users. After generating enough users, we crawled the users' posts in the past year. Meanwhile, we performed a preliminary screen of users by filtering users with less than 100 posts. The users' initial posts were a series of free-form texts, which could contain noise, e.g., in the form of typos or repeated characters. As such, standard NLP-based preprocessing techniques were employed. Specifically, we removed stopwords, repeated characters, emoticons, special characters and punctuations. Finally, the dataset spans from July 2015 to July 2016 containing 2,834,910 tweets posted by 1948 users after filtering. The number of Twitter messages posted per user follows a power-law distribution. The majority of users published less than 100 messages during our observation period while only a small fraction of users wrote more than 10,000 Twitter messages and one user-produced even slightly more than 20,000 tweets (no spam).

Adressa Dataset [63]. The second dataset was provided by Adresseavisen, a local newspaper company in Norway. The Cxense platform for news recommendation and monitoring was used to extract the dataset. The Adressa dataset was prepared as part of the RecTech project on recommendation technology. After removing the users' sensitive information, we released two versions of the datasets, containing one week of user interactions from 1 January to 7 January 2017, and ten weeks of user interaction data spanning from 1 January to 31 March 2017, respectively. The datasets are publically available on the website. The article contents are not directly available online but can be achieved by request due to some articles behind paywall which cannot be freely available online. Article content is saved separately in files and the files are named using the article id which can be aligned with ids in the datasets. All news articles in Adressa dataset are in Norwegian. Basic statistics on 10 weeks dataset are listed in Table 2.2. More statistics on the dataset can be found in [63]. We explored different attributes of Adressa dataset in our studies, which were proven to have different effects on users' clicking behavior. Specifically, we especially explored the article title, category, keywords in Chapter 4. User location, clicking timestamp and topics were studied in Chapter 5. In Chapter 6, we focused on the article content and entities for alleviating cold-start and data sparsity issues. Due to the low ratio of subscribers, our researches mainly focus on modelling user behaviors and providing recommendations based on the characteristics of users' clicking patterns in the same session. SessionStart and SessionStop attributes can be used in combination to identify the session information. The number of news articles interacted per user follows a power-law distribution as seen in Figure 2.3.

Last.fm Dataset [64]. Last.fm is an online radio station and music recommender service that recommends tracks to users based on their listening habits. It collects implicit feedback about the tracks played by a user and released datasets with more than one billion listening events, intended to be used for music retrieval and recommendation purpose. The dataset used for our study called LFM-1b contains 8 GB of data including artists, albums, tracks, users, and listening events spanning from 1 January 2013 to 11 March 2013 10 weeks. Last.fm also provides extensive tools and APIs which can be used by researchers and developers to download user or item related information with limits. Therefore, to achieve more detailed information on artists, we leveraged artist.getInfo API to download artists descriptions to enrich user profiles for recommendation purposes. As data preprocessing, several standard NLP-based steps were adopted. First, we filtered out punctuation, numbers, special characters and stopwords. Then stemmer was adopted to unify word forms. Finally, the preprocessed vocabulary can be used as input for user modelling.

Weibo-Net-Tweet Dataset. This dataset is provided by Jing et al. [65]. It was collected from Weibo.com, a popular social network in China including in total 1.7 million users and 300 thousand microblogs by 2017. Just like Twitter, Weibo enables users to share their feelings, ideas in the form of tweets, and leave comments to the public. Therefore, the user-generated content includes user interested topics contributing to the mining of user interests. The dataset was originally used for studying social influence locality. It contains six parts mainly including friends network, tweets and rewets, user profiles, and tweet content. In our study, we mainly used its user profiles, tweet and retweet content for user modelling. Since the tweets and retweets were written with free-form texts, several preprocessing steps need to perform. Specifically, we first deleted non-textual information

Table 2.1: Some attributes in Adressa dataset.

Attribute	type	Description
eventId	int	The identifier used to differentiate distinct events from the same user.
activeTime	int	The active time on a page in seconds, if known.
os	string	Operating system that the user used when log in.
referrerUrl	sting	The URL of the referrer page.
deviceType	string	The type of the device.
sessionStart	boolean	Indicates whether the event is considered as the first event in session.
sessionStop	boolean	Indicates whether the event is considered as the last event in session.
userId	string	The cross-site user identifier which can be used to differentiate devices/browsers, or identify different subscription users by the user id.
category	sting	The category of the news article.
city	string	The city name inferred from the IP address.
country	string	The country code inferred from the user's IP address.
region	string	The region code inferred from the IP address.
time	int	The time of event, measured in Unix time.
canonicalUrl	sting	Canonical URL as calculated based on incoming events and the fetched page content.
documentId	string	The document id. This will be the same for different URLs that are considered equivalent according to Cxense's normalization algorithm. E.g., http://www.example.com/, http://www.example.com and http://example.com will all have the same id value.
title	string	The title of the article.
keywords	list	The keywords of the article.
namedEntities	list	The named entities of the article, including their types, counts and weights.
author	string	The author of the article.
publishTime	string	The publish time of the article.
profile	Array of object	A set of items which are extracted or generated from the page content. Usually a string or keywords or Named Entities from the page. The possible types and weight of items are given.
item	sting	Item extracted or generated from the page content. Usually a string or keyword extracted from the page.

Table 2.2: Basic statistics of Adressa dataset

Data Statistics	
Items	Values
Number of articles	48486
Number of entries	27223576
Number of total users	3083438
Number of subscriber users	380527
Ratio of subscriber users to total users	0.1234

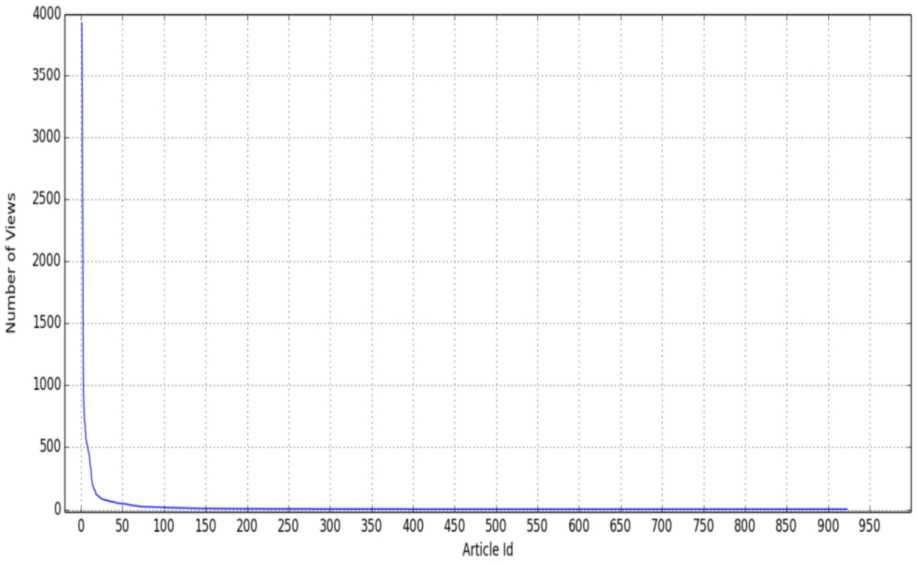


Figure 2.3: Number of article views per article.

such as messy codes, emoticons and tags by scanning the content of the original microblogs. Then we performed words segmentation using the words segmentation tool “jieba”. After that, stopwords based on the stop words list we’ve collected were removed as a necessary step. Finally, we dropped the tweets less than 5 words.

ClefNewsReel Dataset. It is also a news dataset released by Plista GmbH and TU Berlin [66], which contains user logs from eight German news publisher in February 2016. This dataset is originally published for offline evaluation of an online news recommendation campaign. The dataset includes many types of events caused by readers’ actions. For instance, readers may access news articles or click on recommendations. In this thesis, we select session sequences in which the articles were clicked by the users for our experimental dataset. We also leveraged news content as auxiliary information for user modelling. We also extract named entities (NEs) from news titles and content using Spacy, a natural language processing tool for multi-lingual texts. The extracted NEs can then be aligned with an external knowledge graph for further use.

Part II

Mining Attribute Information for User Modelling

Chapter 3

Temporal User Interest in Social Media

Time series forecasting methods have been used for many areas, such as foreign exchange rate and stock price. However, they are rarely used for predicting user behavior in social networks. One important reason is the sparsely distributed feature matrics which make it difficult to model a personalized time pattern. In this chapter, we make use of a large volume of tweets on Twitter with the help of Mediawiki api to extract and enrich the feature matrices, and thus solve the sparsity problem. Then according to the existing researches, which show that users' interests on social networks are mainly affected by two factors, short-term and long-term influence, a neural time series forecasting model (NTSF) is introduced to fit and predict user preference trend. In this model, we integrate emerging/hot topic detection to deal with the short-term aspects, and use Fast Fourier Transformation (FFT) to differentiate cyclic behavior of users. Considering the nonstationary and nonlinear characteristics appearing through user interest patterns, Long Short-Term Memory Recurrent Neural Network (LSTM-RNN) is employed to balance the influence of short and long term aspects, as well as adjust model parameters according to historical results. Our experimental results with extensive Twitter datasets verify the effectiveness of our approach.

3.1 Introduction

With the frequently changing of user preferences on social networks, quantifying and modelling user interests have become more and more important in many aspects, such as user interesting topics recommendation and target population advertising. Time plays a crucial role in influencing and understanding users' changeable preferences among various factors which directly or indirectly result in users' interesting behavior, like geo-location, celebrity effect and social circle.

Fortunately, extensive current work has been done to study temporal user behavior patterns and model their time-aware properties. Because it not only facilitates tracking the evolution of user preferences [67], but also is a key input for many context-aware recommender systems [68, 69], which may lead to significant improvements on recommending accuracy [33]. According to the observation, user preference patterns are influenced mainly by two aspects, users' intrinsic interests and the attention of the general public, also called users' long-term and short-term interests separately.

Users' long-term interests are relatively stable compared with constantly varied public attention. Considering this characteristic, a lot of efforts have been made in building and predicting users' behavior. Michelson at el. use topic profile to discover the main topics of interest for users in microblogs [70]. Stoyanovich at el. model users' interest in terms of the tags which are used to annotate content [71]. Social neighbours analysis is also leveraged in prediction in [72]. Based on the abovementioned studies, many articles make a step further, they combine tendency of the public and users' intrinsic preference by using random walk-based methods, machine learning algorithms, matrix factorization and topic model-based algorithms, Bayesian inference and so on [73-77]. However, time series prediction is seldom used in temporal personalized prediction. One of the most important reasons is that the feature matrix is too sparse to be presented within time dimension [18]. In this chapter, we screen active users from Twitter, and automatically extract valuable topics with the help of Mediawiki api from their tweets, retweets and comments. Then we use Neural Time Series to do prediction.

Time series forecasting means the use of a model to predict future values based on past records. It has been widely used in many areas for dynamic system modelling, such as engineering, science, sociology and economics [78]. Users' interests present dynamic time-aware patterns. In Figure 3.1, we illustrate the interest tendencies of three topics from daily tweets of a specific user within November 2012 and November 2013. As shown in this figure, users' attentions of some specific topics are comparatively stable and could be considered as the static interests, such as umemployment in our example. In addition, it also illustrates that some topics the user interested in have certain lags which can be regarded as the recent interests, for instance congress which the user often posted for several consecutive days. Besides, some users' interest topics show with certain time intervals which can be seen as cyclic patterns as economy in this instance. Many present approaches only consider users' static interests or recent interests but ignoring cyclic interests of the users. Furthermore, according to [79], users are easily attracted by emerging

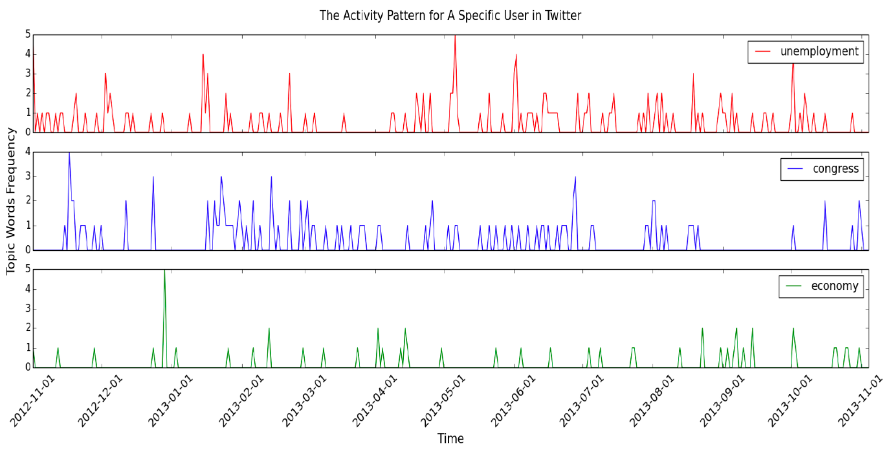


Figure 3.1: Illustration of three topics trends from November 2012 to November 2013 of a specific Twitter user. X axis denotes time and y axis denotes the topic words frequency from the user's daily tweets.

/hot news in social network (e.g. Twitter). Thus in this work, our proposed model incorporates the aforementioned four aspects from users, and for the first time, we explore the interaction among these patterns in affecting the evolution of users' personalized interests. Considering nonstationary and nonlinear characteristcs appearing through users' preference trends, a time series mixed with LSTM-RNN algorithm, which incorporates a five-layer network with nonlinear recurrent projection in each layer is proposed to model these multiple temporal features.

The key contributions described in this chapter are:

1) We enrich the topic-based feature matrics with the help of Mediawiki api related with users' tweets content.

2) We proposed a neural time series forecasting model to draw users' interest patterns over time on Twitter which takes emerging topics, users' intrinsic interests, users' recent behaviors and cyclic patterns of users into consideration.

3) We use LSTM-RNN model to differentiate and quantify various types (lag, seasonal, cyclic, random pattern etc.) of users' interests automatically.

4) We conduct extensive experiments from Twitter datasets to evaluate the performance of the proposed model.

The rest of this chapter is organized as follows: we abstract our problem in Section 3.2. In Section 3.3, we briefly introduce our NTSF model for personalized user prediction. Our experimental setup and results are shown in Section 3.4. Finally, we present the conclusions in Section 3.5.

3.2 Problem Definition

Nowadays, social networks become more and more popular since they are available in multiple languages and enable users to connect with friends or people no matter where they are. Among this, Twitter is one of the most popular applications for users connecting around which has 100 million users login daily[1], and thus is a good way for researches study users' behavior through their active logs.

For a specific user μ_i, the input feature series is $\mathcal{F}_t = \{f_j \mid j \in Z\}$ at time epoch t based on tweets from Twitter and related information from Wikipedia. Supposing in time epoch $t + 1$, user μ_i's interests output could be represented as y_{t+1}. Then we need to find the appropriate learning function $X(\cdot)$ which could get the output of next time epoch by giving previous feature series from previous epochs. So we have:

$$y_{t+1} = X(\mathcal{F}_{t-n}), n = 0, 1, 2, \dots \tag{1}$$

The input features series $\mathcal{F}_t$ is composed with multiple feature matrix, including user recent features S_t, emerging topic extraction ε_t, user cyclic features C_t, and user static features B_t.

3.3 Neural Time Series Forecasting Model

In this section we present incrementally the formation of our NTSF model. Since our model includes multiple features as input, we will first describe how we extract various features from users' historical tweets.

3.3.1 Multi-Scale Feature Extraction

We extract various kinds of features including historical static features, cyclical features, recent features as well as emerging/hot topics. Before we begin extracting features, we should first sort tweets according to users' incremental time stamps.

[1] https://dev.twitter.com/streaming/overview

User Recent Feature Extraction. For the purpose of predicting users' recent interests, the best way is to extract from users' recent activity records. A most common way is only to get topic words from their tweets. In this chapter, after we extract topics from users' daily words, we get more information with the help of Mediawiki api, and enrich the users' interest matrix by adding useful topics from Wikipedia. At time epoch t, user's recent features matrix S_t could be presented as $S_t = \{s_{i,j}\}$, where $i \in [0, t-1], j > 0$. Each factor $s_{i,j}$ represents the frequency of topic j at time epoch i.

Emerging/Hot Topics Feature Extraction. To extract emerging/hot topics from tweets, we adapt the twevent mentioned in [80]. The process can be separated as three components: tweet segmentation, event segment detection and event segment clustering.

Given a tweet $d \in \mathcal{J}$, where $\mathcal{J}$ denotes Twitter dataset, and split d into non-overlapping segments as words or phrase, $d =< s_1,\ s_2,\ \dots,\ s_m >$, s_i represent each segment. In this chapter, we use NLTK Toolkit[2] to extract tokens instead of n-gram method used in original segmentation component in Twevent.

In a certain period of time t, bursty segments in terms of frequency will be potentially related to some hot topics talked and shared by Twitter users. Let N_t be the number of tweets within t, $f_{s,t}$ be the number of tweets within t containing segment s. Then the probability of observing frequency $f_{s,t}$ can be modeled by a binomial distribution.

$$\mathrm{P}(f_{s,t}) = \binom{N_t}{f_{s,t}} p_s^{f_{s,t}} (1-p_s)^{N_t - f_{s,t}} \tag{2}$$

where p_s is the expected probability of segment s within time period t. Given a large N_t, this binomial distribution can be approximated with Gaussian distribution.

$$\mathrm{P}(f_{s,t}) \sim \mathcal{N}(N_t p_s, N_t p_s (1-p_s)) \tag{3}$$

Thus, the expected number of segment s within time period t can be represent as $E[s|t] = N_t p_s$. The more the number of s within t is, the burstier the segment s will be. The standard deviation could be represented as $\sigma[s|t] = \sqrt{N_t p_s (1-p_s)}$. Then, we use the following equation to calculate bursy probability of segment s.

$$P_b(\mathrm{s}, \mathrm{t}) = \delta\left(\alpha \frac{f_{s,t} - (E(s|t) + \sigma[s|t])}{\sigma[s|t]}\right) \tag{4}$$

[2] http://www.nltk.org/

where $\delta(\cdot)$ is the sigmoid function and α is the parameter learned through experiment. In our case, we set $\alpha = 10$. We also take users' influence into consideration. Let $u_{s,t}$ be the user frequency, the number of tweets which contain segment s within t. Then we give each segment s a weight $w_b(s,t) = P_b(s,t)\log(u_{s,t})$ as the measure of their emerging/hot segment probability. After ordering all segments according to their weights, we take top-K bursty segments with biggest weights as most potential emerging/hot events related topics. In our case, we set $K = \sqrt{N_t}$.

After extracting all emerging/hot topics, we form feature matrix as $\varepsilon_t = \{e_{i,j}\}$, where $i \in [0, t-1]$, $j > 0$. Each $e_{i,j}$ represents the bursty probability of topic j at time epoch i.

User Cyclic Feature Extraction. Users' interesting topics sometimes show cyclic patterns, such as president election in the US, the cycle is nearly every four years. In order to capture such kind of cyclic features through user historical records. In this chapter, we user Fast Fourier Transform (FFT) to find the most likely cycles from discrete sampling points from user behavior. Assuming that the feature sampling vector are f_n in time domain, then the transformation process could be represented as

$$F_k = \sum_{n=1}^{N-1} f_n e^{-\frac{i2\pi kn}{N}} \tag{5}$$

where F_k is the representation of f_n in frequency domain, $e^{-\frac{i2\pi kn}{N}}$ is an N-th primitive root of unity, N is the total sampling number, and $\frac{1}{N}$ could be seen as sampling frequency, n in this equation is every sampling point. In this way, in frequency domain we could choose the highest N_c's frequencies as cycle features.

After FFT transformation of the original feature vectors, we could get users' cyclic matrix as $C_t = \{c_{i,j}\}$, where $i \in [0, t-1]$, $j > 0$. Each factor $c_{i,j}$ represents if there exists cyclic patterns of topic j at time epoch i.

User Static Feature Extraction. If we have unlimited memory, then we could save the whole history patterns of the user. But it is impossible to set the model training batch size as long as users' log in days in Twitter. Then the model can only take the most recent few days to learn the user patterns. However, users' long-term patterns are relatively stable, which means they are unlikely to change in a short period of time. In such case, we extract users' static features by averaging the frequency of user daily reading topics $f_{i,j}$.

$$b_{i,j} = \sum_{i=1}^{\|T\|-1} f_{i,j} \|T\| \tag{6}$$

where factor $b_{i,j}$ represents the static property of topic j at time epoch i. $\|T\|$ denotes the length of the total time period in our datasets. Due to the stable properties of this kind of feature, in a small period of time t, the static feature matrics B_t should be consisted of identical vectors $b_{i,j}$, where $i \in [0, t-1]$, $j > 0$.

User Feature Input Formation. After extracting users' static, cyclic, emerging features and recent features, the input of our model should be the series connection of these four features.

$$F_t = [S_t;\ \varepsilon_t;\ C_t;\ B_t] \tag{7}$$

3.3.2 LSTM-based Recurrent Neural Network

Since neural network has drawn more and more attention recent years, especially in the areas of language modelling and generating text, machine translation, sentence legitimacy check, POS tagging etc. In this chapter, we present out continuous LSTM-based RNN model and how it works to deal with users' interest topics forecasting problems.

Basic RNN. RNN (Recurrent Neural Network) is a kind of Neural Network that could process sequences of inputs. It can potentially learn from the past data and predict the new one. Figure 3.2 shows the structure of basic RNN. Let x be the input sequence of the network and y be the output of the prediction value of next time epoch, a denotes the hidden layer processing. We also assume that w_{in}, w_{out} and w_a denote input matrix, output matrix and transit matrix linking to the next layer respectively. Supposing we have three states at time $t-1, t$ and $t+1$. a_i $(t-1 \le i \le t+1)$ is the value of hidden layer at states i which could be represented based on previous value a_{i-1}

$$a_i = \mathrm{f}(w_{in}x_i + w_a a_{i-1}) \tag{8}$$

where f is a non-linear learning function. In our case, various kinds of features have different impacts on determining the next stage of predicting state, which is the probability of specific topics. Thus, in order to model such elements into learning procedure, the representation of transition matrics can be calculated as

$$a_i = \mathrm{f}\left(\sum_{f_i \in F} w_{f_i} x_i + w_a a_{i-1}\right) \tag{9}$$

The output could be represented as

$$y_{out} = \mathrm{g}(w_{out} a_i) \tag{10}$$

where g is the output gate to adjust and map the value into different areas.

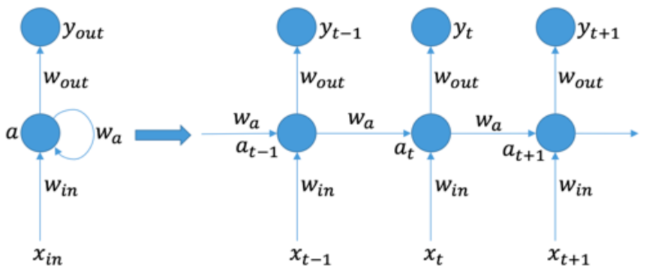


Figure 3.2: Recurrent Neutral Network.

LSTM-based RNN. Though RNN could connect previous information to the present tasks. However, if we want to predict next state based on not just the last few states, but more previous states or patterns, as the information we need extend, basic RNN become unable to learn to connect the information. In such case, the LSTM memory block instead of traditional hidden layer are introduced which could learn more previous, long-term dependencies, for instance users' previous behaviour representation sequences could lead to users' next move. Furthermore, "forget gate" are also introduced into traditional LSTM memory cells, which could automatically reset memory blocks once their contents are out of date and become useless.

The structure of LSTM memory block with one cell is shown in Figure 3.3. In this figure, y_t^i and y_t^o denote the input and output of the memory cell. w_i, w_{in}, w_f and w_{out} represent the weight matrices of the input cell, input gate, forget gate and output gate of forward process. u_i, u_{in}, u_f and u_{out} represent the weight matrices of the input cell, input gate, forget gate and output gate of backward process. b_{in}, b_f and b_{out}represent the bias of the input gate, forget gate and output gate.

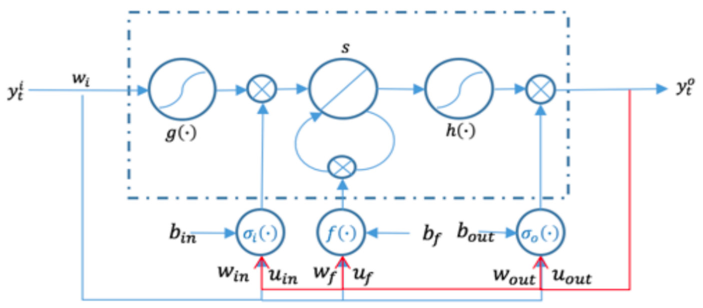


Figure 3.3: LSTM memory block with one cell.

3.3.3 Neural Time Series Forecasting Model

The architecture of NTSF is shown in Figure 3.4. We use LSTM-based RNN to read the input sequence, one time stamp at a time, to obtain users' historical features. Though LSTM could handle long-term and short-term dependencies, when the long-term dependencies are separated by too many intervals or the lags are too long, LSTM needs to keep the useful features for a quite long period of time. Some important features could be lost in transmission process. As the result of this concern, we introduce cyclic and static features of the users by analyzing historical traces. The key difference with previous time series model is that we aim at modelling users' interests by adopting various features. Thus in our case, given the input sequence $x_{in,t}$ at time slide t, the objective of our model is to minimize the distance between the predicted output $\hat{y}_{out,t}$ and the actual output $y_{out,t}$. The distance can be defined as:

$$\cos\left(y_{out,t}, \hat{y}_{out,t}\right) = \frac{\sum_i y_{out,t,i}\hat{y}_{out,t.i}}{\sqrt{\sum_i y_{out,t,i}^2}\sqrt{\sum_i \hat{y}_{out,t,i}^2}} \tag{11}$$

However, in order to calculate the users' attention score (Section 3.4.3), we need to map this output into range of $[0, 1]$. Therefore, the posterior probability of a predicted output given the user's historical features and public's attention as input is estimated as:

$$\mathrm{P}(y_{out}|x_{in}, \mathrm{t}) = \mathrm{g}(\cos\,(y_{out,t}, \hat{y}_{out,t})) \tag{12}$$

where $\hat{y}_{out,t}$ is the function of x, and $g(\cdot)$ is a softmax function which is selected as $g(x) = 1/(1 + e^{-x})$. Through training process, the model parameters are estimated to maximize the likelihood of the actual interesting texts given the historical and public inputs, which could be incorporated in the objective function:

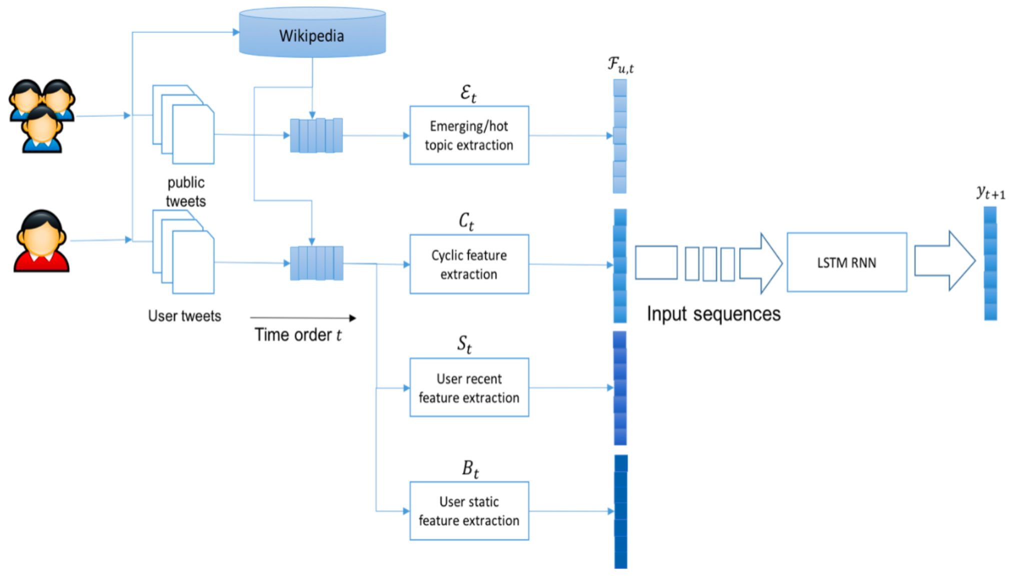


Figure 3.4: The architecture of the proposed NTSF model.

$$J = -\sum \ln P(y_{out,t} | x_{in,t}) + \frac{\lambda}{2} \|\theta\|^2 \tag{13}$$

$$= -\sum \ln(1 + e^{-\cos(y_{out,t}, \hat{y}_{out,t})}) + \frac{\lambda}{2} \|\theta\|^2 \tag{14}$$

where $\theta = \{w_{in}, w_{out}, w_f, b_{out}, b_{in}, b_f\}$ denotes all the parameters that need to be estimated, λ is the coefficient of regularization term. The partial derivatives of J with respect to all the parameters that need to be estimated could be learnt using Stochastic Gradient Decent (SGD) with Back Propagation Through Time (BPTT) algorithm [81]. To accelerate training, we use full-batch training during BPTT due to only thousands of datasets for each users. In Algorithm 1, we sketch the training process from high level of NTSF model.

3.4 Experiments

In this section, we conduct extensive experiments to compare the performance of our model with the state-of-art predicting methods and evaluate the influence of various features extraction, as well as personalized users' interest topics prediction. We also test the performance of convergence rate with different settings of NTFS model and RNN only model.

Algorithm 1 Training NTSF.

```
Input:  $\mathcal{F}_u$ = user input feature matrix;
        $\alpha$ = fixed step size;
        $u$ = momentum parameter;
        $nEpoch$ = maximum number of Epochs;
Output: trained model;
1:  Initialization: Set all parameters in $\theta$ to small random numbers,
    $i = 0, k = 1$;
2:  Procedure $LSTM - RNN(\theta)$
3:  while $i \leq nEpoch$ do
4:      for j=1 → n do
5:          Compute $\nabla J(\theta_k)$;
6:          Update $\triangle\theta_k \leftarrow -\alpha\nabla J(\theta_k) + u\triangle\theta_{k-1}$;
7:          k ← k+1;
8:      end for
9:      i ← i+1;
10: end while
```

3.4.1 Experimental Setup

In this paper, we use Twitter dataset collected from July 2015 to July 2016 by using Twitter api[3]. We filter users who login Twitter nearly every day and post tweets especially containing long stories, news articles (post behavior includes post, repost and comments). The dataset covers all the reviews of a whole year. These reviews were first stemmed and split into unigram features, then we filtered stopwords and performed feature dimension reduction to keep only the most promising features according to TF-IDF score. The dataset consists of 185,071 users in total and more than 20 million time-stamped tweets. However, it does not include any follow relationships. We sample the dataset to obtain the users who posted more than 1000 tweets a year, and who have Twitter posts spanning over at least 300 days. In total we obtain 2,834,910 tweets from 1948 users for evaluating our NTSF model. We compare the probabilities of interested topics predicted from our NTSF model with that from actual tweets for the next time stamp of a specific user. The default settings of the NTSF model are listed in Table 3.1. During training process, some of those parameters may be updated to achieve a better performance.

[3] https://dev.twitter.com/docs

Table 3.1: Experiment Settings

Parameter	Defaul Value
Number of Hidden Layers	3
Activation Function	ReLU
Initial Learning Rate	0.1
Optimizer	sgd
Regularizer	L2(l = 0.01)
Weight initialization function	uniform

3.4.2 Baseline Methods

We compare our proposed NTSF model with several other state-of-art predicting algorithms to make the results more convincing. These include the classic time series prediction model incorporating seasonal factors which is Seasonal Time Series Autoregressive Integrated Moving Average (SARIMA), and other representative machine learning algorithm Support Vector Regression (SVR), and basic RNN. In addition, the most hot topics in the training set as prediction for each users (Popular-based) are also as the comparison in our experiments.

For the SARIMA model, we use a function that conducts a search over all possible models and returns the best results according to Akaike Information Criterion (AIC) value. We update parameters according to monthly training datasets by using sliding window approach, which is better than daily and weekly updates cycle. For instance, if we want to predict the tendency at t epoch, our training datasets will be within $(t-m) \sim (t-1)$ time period, where m represents the length of the sliding window. Then we use the model with these parameters to predict various interests.

For SVR method, we tried three different kinds of kernels to test the model, including linear, polynomial and Radial Basis Function (RBF). We found that the RBF kernel performed the best with the lowest train and test MSE, which is $k(x_i, x_j) = exp(-\|x_i - x_j\|^2 / 2\gamma^2)$, where$\gamma$ is the parameter of RBF kernel and can be selected through cross-validation. The weights of the features are initialised using uniform distribution.

The RNN model is the NTSF model without LSTM memory cells, and initialization of other parameters and training process are the same as NTSF.

3.4.3 Evaluation Metrics

In this chapter, we adopt 3 kinds of methodologies to evaluate the performance of our model. The collected datasets T_u for a specific user u are divided into two parts. The first 10 months datasets are used for model training, and the rest 2 months datasets are used for prediction. First, we define Normalized Attention Score (NAS) to quantify the focus of the user to a certain topic. Assuming topic i contains a series of representative topic words, $T_i = w_{i,1}, w_{i,2}, \dots, w_{i,M}$, where M is the number of topic words. Let $TF_{u,t,w_{i,j}}$ be the term frequency of the topic word j in topic i for user u at time epoch t , which is the number of word j appearing in user's daily documents. Then $NAS_{u,t,i}$ which represents the focus towards topic i of user u at time epoch t, can be defined as

$$NAS_{u,t,i} = \frac{\sum_{w_{i,j} \in T_i} TF_{u,t,w_{i,j}}}{\sum_{w_{i,j} \in T_i} TF_{u,t,w_{i,j}} + \sum_{w_{i,j} \notin T_i} TF_{u,t,w_{i,j}}} \tag{15}$$

$NAS_{u,t,i}$ can be interpreted as the proportion of the targeted topic i in reading documents of user u at t time period. Besides, we also employed the widely accepted metrics for evaluation of the model performance in regression settings, and these include two standard evaluation metrics: mean square error (MSE) and Pearson correlation (rho) between the actual $y_{out,t}$ and the predicted output $\hat{y}_{out,t}$ at each testing time epoch t. Their calculation can be defined as:

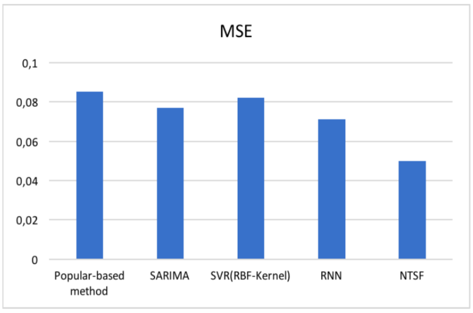
$$\text{MSE} = \frac{1}{N} \sum_{i=1}^{N} (\hat{y}_{u,t,i} - y_{u,t,i})^2 \tag{16}$$

$$\rho_{\hat{y},y} = \frac{\sum_{i \in \hat{y}_{u,t} \cap y_{u,t}} (\hat{y}_{u,t,i} - \hat{\mu}_{u,t}) \cdot (y_{u,t,i} - \mu_{u,t})}{\sqrt{\sum_{i \in \hat{y}_{u,t} \cap y_{u,t}} (\hat{y}_{u,t,i} - \hat{\mu}_{u,t})^2} \cdot \sqrt{\sum_{i \in \hat{y}_{u,t} \cap y_{u,t}} (y_{u,t,i} - \mu_{u,t})^2}} \tag{17}$$

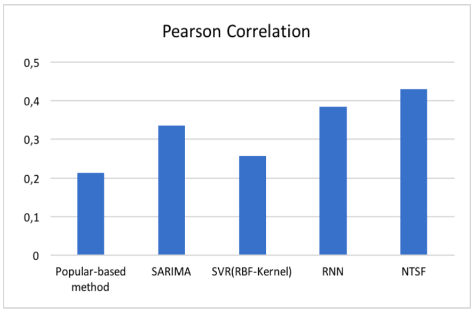
3.4.4 Experimental Results

In this section, we conducted empirical experiments to demonstrate the effectiveness of NTSF model. For twitter dataset, we adopt the training set for model learning based on Algorithm 1, and predict the tendency of time series with the fellowing 20% in testing set.

Comparing with Baseline Methods. We evaluate the performance of NTSF comparing with other three baseline methods (Section 3.4.2). Referring to Figure 3.5, NTSF outperforms all baseline methods in terms of MSE and Pearson Correlation. We see that the popular-only baseline performs very poorly which confirms that user may not be interested in hot/popular topic, and many users tend to have their own interest patterns. It is also shown that RNN model without LSTM cells is not good enough for this task due to the error explosion or vanishing problems. The SVR model performs only a little better than popular-only baseline which shows that the prediction ability of SVR related to that whether the patterns of data appear the similar distribution as their kernel function. SARIMA model performs less effective than RNN and NTSF model for that users' behavior patterns in Twitter datasets show both habitual and dynamic, and NTSF is designed to capture this kind of dynamics as well as habitual behavior according to temporal contexts.



(a) Prediction results in terms of mean MSE

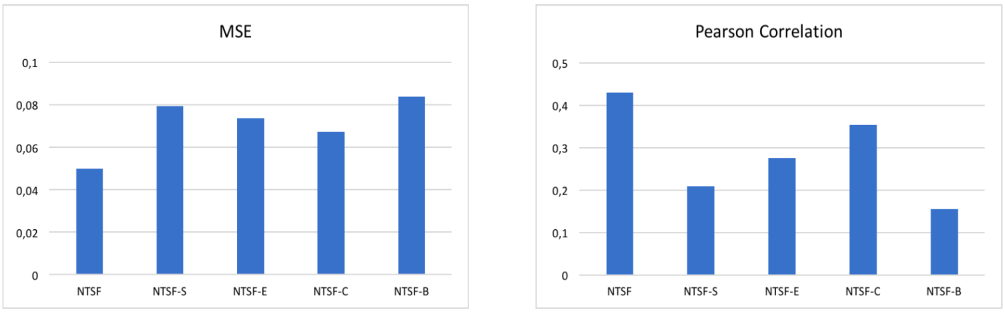


(b) Prediction results in terms of mean Pearson Correlation

Figure 3.5: Comparing different baseline methods in terms of mean MSE and mean Pearson Correlation. NTSF outperforms all baseline methods with Twitter datasets.

Effect of Different Kinds of Features. We measure the effectiveness of using multiple adaptive features by adopting control variable method. Specifically, we compare performance of NTSF model with the performance of NTSF without hot/emerging features (NTSF-E), NTSF model without recent tweets histories (NTSF-S), NTSF model without cyclic features (NTSF-C) and NTSF model without users' static interests extractions (NTSF-B). The results are shown in Figure 3.6. Sub-figures (a) and (b) represent the performance of different models in terms of mean MSE and mean Pearson Correlation. NTSF model results in better performance than the others as it has lower MSE value and higher Pearson Correlation value. It is shown from both figures that users' static interests plays the most important role in influencing users' daily interests trends for that both MSE and Pearson Correlation value drop the most among the others. This importance of features are followed by users' recent behaviors for the drop value is ranked in the second place.

Though users' cyclic patterns are also crucial, but the importance is not as much as the other features through the analysis of our Twitter datasets.



(a) Prediction results in terms of mean MSE (b) Prediction results in terms of mean Pearson Correlation

Figure 3.6: Comparing the influence factors of various features in terms of mean MSE and mean Pearson Correlation.

Personalized Prediction of interests topics. We illustrate how NAS can be used as a metrics for tendency prediction of users' interests in Figure 3.7 in term of emerging topics. After getting the prediction vector, we order the output dimensions according to their values. Then we take out the top-5 values to calculate NAS and draw the plot according to the daily NAS values for specific user. Figure 3.7 shows that our NTSF model can capture the tendency of users' interests variations as the real situation but existing a tolerable time delay.

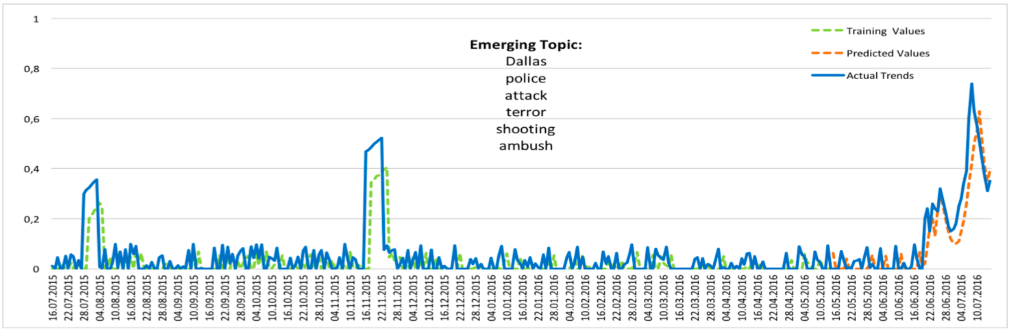


Figure 3.7: An example of personalized prediction of users' interested topics in terms of NAS metrics.

Convergence Rate. We also evaluate the convergence rate with different parameter settings between NTSF model and RNN only model and the result is shown as a boxplot in Figure 3.8. According to Figure 3.8, we can conclude that NTSF model converges faster than RNN only model under the same parameter settings generally. Besides, model with 0.05 initial learning rate converges slowest whereas model with 0.2 initial learning rate convergence fastest.

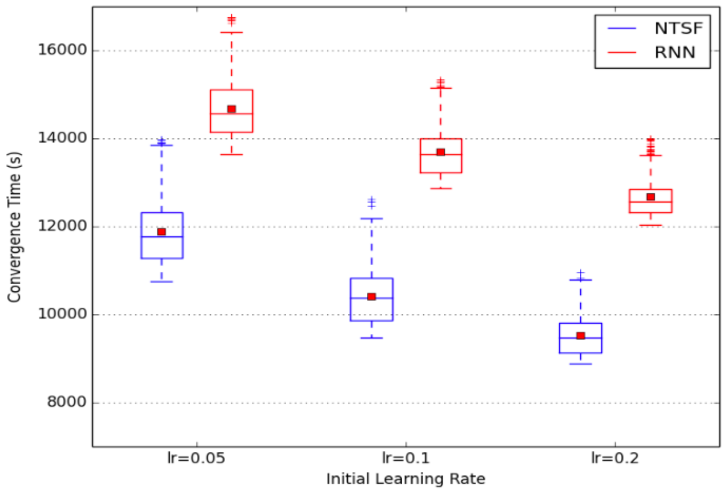


Figure 3.8: Comparing convergence rate with different initial learning rate (lr) of NTSF model and RNN only model. Boxes show the 25th and the 75th percentile of the data, while the whiskers indicate the whole range of the convergence time. The red dots represent mean values, and the lines in the boxes represent median values. The cross marks denote the outliers.

3.5 Discussion

Our experimental results with Twitter datasets suggest that by combination of multiple input features: static features, cyclic features, recent features and extracting hot/emerging topics as features, NTSF improves the prediction performance compared with baseline algorithms. We also verify the different effectiveness of various input features on the final performance of our model. From an extensive convergence analysis, we discovered that 1) the convergence rate of LSTM-RNN is much higher than that of RNN only because LSTM-RNN could deal with error explosion and vanishing problems during training process. 2) Convergence rate is mainly correlated with the learning rate in both RNN model and NTSF model, but it is not directly proportional to the initial learning rate. 3) Larger learning rate normally causes less learning time when the model reaches convergence. However, model will not converge with too large learning rate. In addition, the precision of NTSF model have a great relationship with parameter selection and optimization. Though training with full-batch dataset will usually get better performance, the learning process will need more memory from the server for caching and loading the datasets and results during training procedure with the increasing number of datasets. Thus, when dealing with gigantic amount of training data, we need to use other optimization methods, such as mini-batch and distributed system to improve the training process as well as prediction performance.

3.6 Conclusion

In this chapter, we propose to use Recurrent Neural Network for personalized time aware user interests prediction based on Twitter datasets. To overcome the data sparsity problem, we enrich our user behavior matrices with the help of Mediawiki api, and we also leverage multiple aspects of users' activities, including tweets, retweets and comments. For the first time, our Neural Time Series Forecasting Model (NTSF) extracts the common interests patterns and predicts their interested topics for a specific user by introducing various features.

Compared with existing work, the uniqueness of our approach is that we have explored the multi-dimensional attributes of time, applied distinct methods to model the variable preferences and temporal characteristics of users, and devised an algorithm to infuse all features from multiple aspects of time. In this way, the designed framework can capture the evolutional trends of users' interested topics with tolerable time decay.

Experimental results on real datasets show that NTSF outperforms the classic and state-of-art methods on prediction problems. The different MSE and Pearson Correlation decay on the variations of the NTSF model with respect to removing different time modelling modules give us a good indication of the significance of all the components. The illustration of an arbitrary user's interests prediction instance shows that this approach has good potential in modelling users' dynamic preferences in Twitter.

Chapter 4

Semantic Feature Mining for Session-based Recommendations

Session-based recommendations have drawn more and more attention in many recommendation settings of modern online services. Unlike many other domains such as books and music, news recommendations suffer from new challenges of fast updating rate and recency issues of news articles and lack of user profiles. In this chapter, we proposed a method that combines user click events within session and news contextual features to predict the next click behavior of a user. The model consists of two different kinds of hierarchical neural networks to learn article contextual properties and temporal sequential patterns in streams of clicks. Character-level embedding over input features is adopted to allow integrating different types of data and reduce engineering computation. Besides, we also introduced a time-decay method to compute the freshness of news articles within a time slide. Experimental results on two real-world datasets show significant improvements over several baselines and state-of-the-art methods on session-based neural networks.

4.1 Introduction

News recommender systems have become popular and are employed by many multimedia companies in recent years, as a natural consequence of the increasing complexity and scale of web services and e-commerce platforms. They are able to cope with the information overload and to assist users in finding information matching their individual preferences. Unlike other recommender system domains like books, music and movies, news recommender systems must address additional challenges [82]. For example, large publishers release hundreds of news daily, implying that they must deal with fast-growing numbers of items that get quickly outdated and irrelevant to most readers. User interests change much faster compared with other domains. News articles have recency issues which make users tend to read recent news, not the old ones. In addition, the news domains suffer from extreme levels of sparsity.

Generally, traditional news recommender systems always produce relevant items based on news content and user input profiles [82]. However, problems arise when it goes to unregistered users, since there are no profiles for them, while on the other hand unregistered users occupy a large proportion of the total news readers. According to the statistics performed on Cxense platform[4], the subscribers only take up about 20% of all users in Adresseavisen company[5], a news portal in Norway. Providing accurate recommendations to non-subscribers is also very necessary in real-life applications.

Under such a situation, a typical solution is to base recommendations on session data, e.g. session clicks. These data have two important characteristics. First, session clicks are sequential in nature and the order of clicks may contain information of user intent. Second, clicked items are often associated with metadata such as names, categories, and descriptions, which provide additional information about user taste. Most recent works have focused on exploiting these two characteristics to draw more information from data when designing recommendation strategies. Hidasi et al. for the first time investigated the use of recurrent neural networks (RNN) for session-based next-item recommendation [39]. RNNs are a natural choice for this problem and have been successfully explored for other sequence-based prediction problems in the past [83, 84]. An experimental evaluation on two datasets indicated that their GRU4REC method significantly outperforms item-based k-nearest-neighbor (kNN) methods by 15% to 30% in terms of ranking metrics. Massimo et al. further improved the work in [39] by proposing a model based Hierarchical RNN, that extends previous RNN-based session modelling with one additional GRU level that models the user activity across sessions and the evolution of his interests over time [45]. Despite these positive results, some questions regarding the effectiveness of the session-based recommendation method remain open. First, from our exploratory analysis using standard recurrent architectures for session modelling, we find that they model sequential patterns in streams of clicks and consider all past events to improve recommendation performance. But they fail to integrate different feature types and jointly model their interactions. Second, in the news domain, recency issues of news articles is another predominant challenge that they ignore. Third, the lack of scalable models applicable to

[4] https://www.cxense.com/

[5] http://www.adressa.no/

deal with the large amounts of noisy data seriously restrains the application of recommendation services in the real world.

In this chapter, we propose a neural network (NN) based model, named Deep Joint Neural Networks (DeepJoNN), to address these three problems. The proposed model uses a recurrent neural network (RNN) to capture sequential patterns in streams of clicks and associated features. A convolutional neural network (CNN) is exploited to consider item IDs and all content features including hierarchical categories, keywords and entities as texts and represent the resulting textual data with a character-level model [85]. DeepJoNN provides an effective way to jointly model temporal and content patterns that are indicative of readers' intention with two types of hierarchical neural networks (CNN and RNN). At the same time, representing all features at the character level frees us from the need for time-consuming feature engineering and can be applied for different types of features. We applied our model to several real data sets and the experimental results demonstrate the promising and reasonable performance of our approach.

In summary, our contributions are as follows:

1) The proposed Deep Joint Neural Networks (DeepJoNN) jointly model sequential patterns of session clicks and different content features of items for session-based recommendation. To the best of our knowledge, DeepJoNN is the first one that jointly models both user and item from news streams using two different kinds of hierarchical neural networks.

2) We propose to use character-level representation for all types of features, which frees us from feature engineering steps and reduces the number of model parameters. Experiments demonstrate the effectiveness of the proposed method.

3) We extend the traditional Convolutional Neural Network to multi-dimensional level processing by integrating tensor-based feature representation method.

4) We conduct extensive experiments to evaluate the performance and generality of our model on two real large-scale datasets. The results show the advantages of our method for session-based recommendation in comparison with state-of-the-art techniques.

The remainder of the chapter is organized as follows. In section 4.2, we present our DeepJoNN model in detail. We describe the data sets, experimental settings and the state-of-the-art methods we use in section 4.3, as well as experimental results and analysis.

Finally, we present the conclusions in Section 4.4.

4.2 The DeepJoNN Model

In this section, we describe the proposed Deep Joint Neural Networks (DeepJoNN) for session-based news recommendation.

In the session-based recommender system, there is a set of items that a user can interact with; note that the term "item" is used in a broad sense here. We experiment with the proposed models using two different datasets, where the possible recommendations are news articles and artists respectively. The datasets are described in Section 4.3.

Let M be the set of items in the system, and $M_v \in \mathbb{R}^{m_v \times n}$ is the embedding representation of item v. The embedding representation approach will be introduced in the next section. Let $S = [S_1, S_2, \ldots, S_{n_s}]$ be a set of sessions, and $S_i = [U_{i,1}, U_{i,2}, \ldots, U_{i,n_u}]$ be the set of events grouped by users and then ordered by timestamp within session S_i, where n_s and n_u denote the number of sessions in training data and the number of users within a session respectively. $U_{i,u} = [s_{i,u,1}, s_{i,u,2}, \ldots, s_{i,u,q}]$ where $s_{i,u,q}$ denotes the q-th interaction event of user u within session i. Our DeepJoNN model retrieves the corresponding embedding representation of $s_{i,u,q}$ for each interaction event q of user u in session i, and feed those into the CNN layer of the model. The common task for all the recommendation models we experiment with is to predict each consecutive item for user u in a session S_i. That is, for a sub-session $[s_{i,u,1}, s_{i,u,2}, \ldots, s_{i,u,j}]$ for user u of S_i, the system is to predict $s_{i,u,j+1}$. A recommendation C_i, is an ordered list of k recommended items, where we would want to see the next item as close to the top as possible.

4.2.1 Character-level Representation

Character-level representation as input for especially Convolutional Neutral Network (CNN) has been widely used in NLP domain, and presents competitive results compared with traditional models with fewer parameters and computational cost [85, 86]. The only problem appears when processing misspelling and informal words, which seldom happens in news domain. Inspired by this idea, we encode the input features such as keywords, categories and entities into character-level but still different from the existing approaches, which will be described in this section.

Let Ψ denote the vocabulary of characters of size $|\Psi|$. Suppose an item feature f is given as a sequence of characters $[\psi_1, \psi_2, \ldots, \psi_p]$, where p is the length of f. Then the character-level encoding of feature f is given by vector $e^f \in R^{L_f}$, where L_f is the length of feature f in the dataset, and the i-th element will be set as the index number of the i-th character in Ψ, and set as 0 otherwise. Note that the index number of character in Ψ starts from 1. In this work, we use vocabulary Ψ of 74, including all lower case characters and upper case characters from Norwegian alphabet, 10 digit characters, and several other characters, which are shown below:

ABCDEFGHIJKLMNOPQRSTUVWXYZÆØÅabcdefghijklmnopqrstuvwxyzæøå12345
67890 .-/|:@

For each clicked event, we represent the associated features as follows:

- Item ID and User ID. A sequence of hash code is encoded according to their digits and alphabets and represented as vectors.
- Keywords and Entities. Each word or phrase such as "været" and "Anne-Grete", in keywords and entities is encoded as one vector, and all encoded words and phrases are stacked vertically to form as a matrix $K^{m_k \times n_k}$, where m_k is the maximum number of words and phrases in keywords and entities which is set as 408 in our case, and n_k is the maximum length of characters in each unit which is set as 100 in our experiment.
- Category. Categories are usually organized in a hierarchy by the website owner. To utilize the information encoded in the hierarchy, we concatenate the current category with all its ancestors up to the root and use the resulting sequence of characters as category feature, for instance "nyheter|moreromsdal". Category is encoded as vector.

Given character-level encoded vectors and matrices for each type of features, we stack the vector and matrices on top of each other to form to the final matrix of $M_{s,j}^u \in \mathbb{R}^{m \times n}$ for the j-th event in session s of user u, where $m = m_k + 3$ and n is the longest features which is the same as n_k in our experiments. If the length of the feature is shorter than its maximum length, then we fill empty positions with all zeros to get a matrix with equal size for each event within the dataset.

4.2.2 DeepJoNN Architecture

In this section, we describe the proposed Deep Joint Neural Networks (DeepJoNN) for session-based recommendation. The architecture of our model, shown in Figure 4.1 is straightforward. After character-level embedding into matrix as input, a 2-layer convolutional neural network is deployed to learn the input feature patterns, and then its output will be transferred as input to RNN layer.

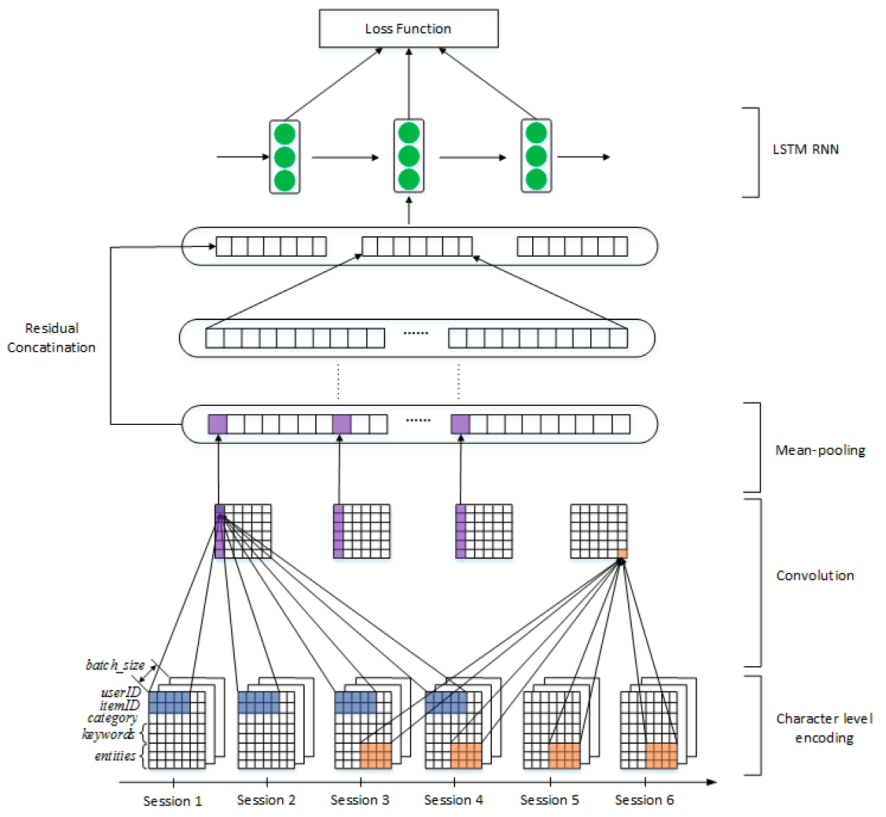


Figure 4.1: The architecture of the proposed model.

CNN with Multiple Dimensional Input. Many researchers have found that CNNs are useful in extracting information from raw data, ranging from computer vision to speech recognition and several NLP tasks. By applying convolution operations at different level of granularity, a CNN can extract features that are useful for learning tasks and reduce the need of manual feature engineering [87]. This characteristic is adopted in our task to extract useful patterns of separate features or their combinations, from streams of click events within sessions.

Recall that $S = [S_1, S_2, \dots, S_{n_s}]$ is a set of sessions, and $S_i = [U_{i,1}, U_{i,2}, \dots, U_{i,n_u}]$ denotes the set of events within session S_i. $s_{i,u,q}$ in $U_{i,u} = [s_{i,u,1}, s_{i,u,2}, \dots, s_{i,u,q}]$ denotes the q-th interaction event of user u within session i. For the sake of efficiency in training, we adopt the session-parallel mini-batch mechanism described in [41] to consider user identifiers during training as shown in Figure 4.2. To begin with, the first events of the first session of the first mini-batch with size b of users constitute as input matrix $M \in \mathbb{R}^{m \times n \times b}$ to the model. Meanwhile the next items of the first session of the same users mentioned above form as the target output from the model. Then the next events of the first session of the same users used as the input for the next iteration, and so on. When a mini-batch end, if there are other users in the first session that haven't been trained, then these users will be regarded as input for next mini-batches. With session-parallel mini-batch integrated into the training process, the model can be trained efficiently over different users having different number of sessions and different number of events within sessions.

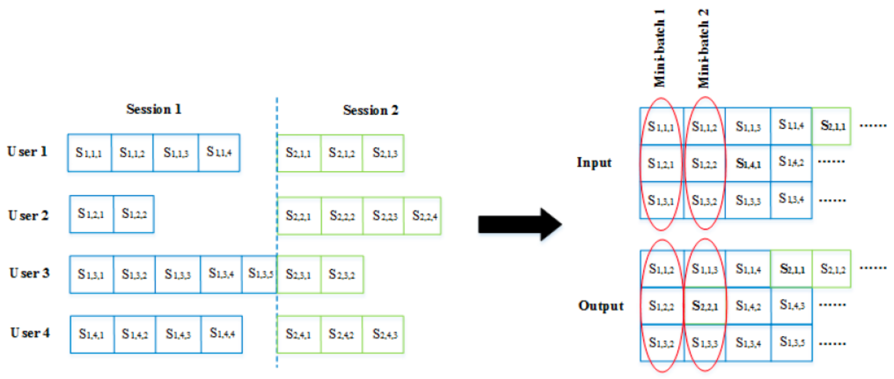


Figure 4.2: User-parallel mini-batches for mini-batch size 3.

Inspired by the work [88] which uses low-rank n-gram tensors to directly exploit interactions between words already at the convolution stage, we extend it to learn tensor based feature mapping from multi-dimensional contextual input features. Recall that $M \in \mathbb{R}^{m \times n \times b}$ represents the character-level input matrix with m number of features, n of character embedding length and b of minibatch size. For each matrix $M_{s,j}^{u} \in \mathbb{R}^{m \times n}$ within batch, $e_i \in \mathbb{R}^n$ is a vector denoting the i-th feature. The consecutive c vectors ending at position j is obtained by concatenating the corresponding vectors

$$v_j = [e_{j-c+1}; e_{j-c+2}; \dots; e_j] \tag{1}$$

which can be seen as the combination of consecutive features, e.g. category and keywords, or different keywords. Out-of-index position are simply set to all zeros.

Thus, the filter for v_j can be denoted as $K_j \in \mathbb{R}^{cn \times h}$ which can be thought as c smaller filter applied to each vector in v_j. The operator maps each v_j in the input matrix to $K_j^T v_j \in \mathbb{R}^h$ so that the input features is transformed into a sequence of combined feature representation,

$$V_j = [K_j^T v_1, \dots, K_j^T v_L] \in \mathbb{R}^{L \times h} \tag{2}$$

In order to capture relevant information in the combined features in a more direct way, the outer product is conducted for $v_j = [e_{j-c+1}; e_{j-c+2}; \dots; e_j]$. Suppose for $c = 3$, the value of result from outer product at position (i, j, k) can be denoted as

$$(e_1 \otimes e_2 \otimes e_3)_{ijk} = e_{1i} \cdot e_{2j} \cdot e_{3k} \tag{3}$$

Therefore, the filters for the concatenate matrix should also be maintained as high-order tensors. In other words, the filters are linear mappings over the higher dimensional interaction terms and can be denoted as T. Let $z \in \mathbb{R}^h$ denotes the resulting h-dimensional feature representation for $c = 3$ combined feature vectors, and it can be achieved through multiplying the filter T and the combined features vectors. The l-th coordinate of z is given by

$$\begin{aligned} z_l &= \sum_{i,j,k} T_{ijkl} \cdot (e_1 \otimes e_2 \otimes e_3)_{ijk} \\ &= \sum_{i,j,k} T_{ijkl} \cdot e_{1i} \cdot e_{2j} \cdot e_{3k} \end{aligned} \tag{4}$$

However, directly maintaining the filter T as full tensor can lead to parametric explosion. To solve this problem, a low-rank-factorization of the tensor T is introduced. Specifically, T is decomposed into a sum of h rank-1 tensors

$$T = \sum_{i=1}^{h} P_i \otimes Q_i \otimes R_i \otimes O_i \tag{5}$$

where $P, Q, R \in \mathbb{R}^{h \times c}$ and $O \in \mathbb{R}^{h \times h}$ are four smaller parameter matrices. For simplicity, we assume that the number of rank-1 components in the decomposition is equal to the feature dimension h. Plugging the low-rank factorization into Eq.(1), the feature mapping can be rewritten in a vector form as

$$z = O^T (Pe_1 \odot Qe_2 \odot Re_3) \tag{6}$$

where $\odot$ is the element-wise product. Note that while Pe_1 is a linear mapping from each feature e_i into a h dimensional feature space, higher order terms arise from the element-wise products (the same as Qe_2 and Re_3).

After z has been derived, we put z into an activation function $f\ (\cdot)$. In the proposed model, we use Rectified Linear Units (ReLUs). Deep convolutional neural networks with ReLUs train several times faster than their equivalents with tanh units. Then we apply Eq. 7, a mean pooling operation, over the feature map and take the average value as the feature corresponding to this particular filter.

$$f_j = mean\{f(z_1), f(z_2), \dots, f(z_L)\} \tag{7}$$

We have described the process by which one feature is extracted from one filter. The model uses multiple filters to obtain various features and the output vector of the convolutional layer is as Eq. 8.

$$F = \{f_1, f_2, \dots, f_J\} \tag{8}$$

where J denotes the number of filters in the convolutional layer. The results from the mean pooling layer are passed to a fully connected layer with weight matrix W. As shown bellow:

$$x_i = f(W \times F + b) \tag{9}$$

The output of the fully connected layer $x_i \in \mathbb{R}^{n_2 \times b}$ is considered as the feature for input events, where n_2 represents the number of neutrons of the last CNN layer.

Session-based Recurrent Neutral Network. A recurrent neutral network is a type of neutral network particularly suited for modelling sequential data. The main difference between RNNs and conventional feedforward deep models is the existence of an internal hidden state in the units that compose the network. Such hidden state summarizes all historical information up to current training timestamp , and meanwhile an additional memory cell is designed to alleviate gradients vanishing/exploding issues. Thus, RNN is suitable for our tasks to learn inter-session and intra-session patterns. In session s_i, an RNN takes the input matrix $x_i \in \mathbb{R}^{n_2 \times b}$ which is the output from CNN layers, and the hidden state vector $h_{i-1} \in \mathbb{R}^{n_o \times b}$ and produce the next hidden state h_i by applying the following recursive operation

$$h_i = f(W \cdot x_i + U \cdot h_{i-1} + b) \tag{10}$$

Here $W \in \mathbb{R}^{n_o \times n_2}$, $U \in \mathbb{R}^{n_o \times n_o}$, $b \in \mathbb{R}^{n_o}$ are parameters of an affine transformation and f is an element-wise nonlinearity. n_o and n_2 represent the number of items and the number of output units from CNN layer. Long short-term memory (LSTM) optimizes the traditional RNN by integrating a memory cell vector $c_i \in \mathbb{R}^{n_2 \times n_o}$ during each session in our tasks and thus addresses the problem of exploding/vanishing gradient when learning long-term dependencies. Concretely, one step of an LSTM takes as input x_i, h_{i-1}, c_{i-1} and produces h_i, c_i via the following intermediate calculations

$$\begin{aligned} i_i &= \sigma(W^i \cdot x_i + U^i \cdot h_{i-1} + b^i) \\ f_i &= \sigma(W^f \cdot x_i + U^f \cdot h_{i-1} + b^f) \\ o_i &= \sigma(W^o \cdot x_i + U^o \cdot h_{i-1} + b^o) \\ g_i &= \tanh(W^g \cdot x_i + U^g \cdot h_{i-1} + b^g) \\ c_i &= f_i \odot c_{i-1} + i_i \odot g_i \\ h_i &= o_i \odot tanh(c_i) \end{aligned} \tag{11}$$

Where $\sigma(\cdot)$ and $\tanh(\cdot)$ are the element-wise sigmoid and hyperbolic tangent functions, $\odot$ is the element-wise multiplication operator, and i_i, f_i, o_i are referred to as input, forget and output gate. For the first session with subscript $i = 1$, h_0 and c_0 are initialized to zero matrix. Parameters that need to be tuned for LSTM are W^p, U^p, b^p for $p \in \{i, f, o, g\}$. However, in our experiments, sometimes gradient exploding is still an issue, but can be alleviated by using optimization strategies such as gradient clipping.

Ranking Loss Design. Ranking is the core of a Recommender System, and it consists pointwise, pairwise and listwise ways. Pointwise ranking estimates the score or the rank of items independently of each other, whereas pairwise ranking compares the score or the rank of pairs of a positive and a negative one. Listwise ranking uses the scores and ranks of all items and compares them to the perfect ordering. However, listwise ranking is not used so often because of the expensive computational cost for sorting. As a part of our model training procedure, we adopt both pointwise ranking (BPR) and pairwise ranking (TOP1). Their definitions are defined as follows:

- BPR: Byesian Personalized Ranking [89] is a matrix factorization method that uses pairwise ranking loss. It compares the score of a positive and a sampled negative item. In our experiment, we compare the score of the positive item with several sampled items and use their average as the loss. The loss at a given point in one session is defined as

$$L_i = -1/N_i \cdot \sum_{j=1}^{N_i} \log\left(\sigma(\hat{r}_{i,k} - \hat{r}_{i,j})\right) \tag{12}$$

where N_i is the item size, $\hat{r}_{i,q}$ is the score on item q at a given point of session i, k is the target item and j are items with score less then 0.5.

- TOP1: This ranking loss criteria proposed by [39] is defined as the regularized approximation of the relative rank of the relevant item. In our task, the TOP1 loss can be defined as

$$L_i = 1/N_i \cdot \sum_{j=1}^{N_i} \sigma\left(\hat{r}_{i,j} - \hat{r}_{i,k}\right) + \sigma(\hat{r}_{i,j}^2) \tag{13}$$

The square part in this equation is the regularization term to avoid exploding score when certain positive items act as negative examples.

- CROSS ENTROPY: Besides, cross entropy is also a widely used loss function in recommendation area which can be formulated in our tasks as

$$L_i = -1/N_i \cdot \sum_{j=1}^{N_i} r_{i,j} \log \hat{r}_{i,j} + (1 - r_{i,j}) \log(1 - \hat{r}_{i,j}) \tag{14}$$

where N_i is the item size, $\hat{r}_{i,q}$ is the score on item q at a given point of session i, k is the target item and j are items with score less then 0.5.

Recency Issues of News. News articles always suffer from recency issues which describes the phenomenon that the freshness of a newly published article in news domain to users can only lasts 2 to 3 days on average. Beyond this time period, this news article will hardly intrigue interests for users. To solve this problem, we integrate a time-decay function into our model which is defined as below.

$$R_{decay} = e^{-\lambda \cdot (t - t_0)} \tag{15}$$

where λ is the parameter that needs to be tuned during training, and controls the decay rate for the news. t and t_0 represent the predicated time at one point within session and the publication time of the news. The decay rates are multiplied by the output values from LSTM RNN layer to form the final outputs.

4.3 Experiments

4.3.1 Datasets

We used two datasets for our experiments. The first is the Adressa 16G dataset[6] which contains 93,948 news articles, 398,545 readers, and about 113 million events over a 90-

[6] https://www.ntnu.no/wiki/display/smartmedia/SmartMedia+Program

days period [63]. Each of these events represent that a user read a particular news article. As preprocessing, we filtered top 50 active users and keeped sessions with more than 2 events for a user. Besides, we removed the records that users visited the news front page for there are no articles related information within the events. In order to evaluate our model's generality, we adopted Last.fm provided by Schedl [64], which contains 10 weeks of log data between 1/1/2013 and 11/3/2013. To enrich the content information for the dataset, we also used Last.fm API [7] to collect artist information to improve the recommendation accuracy. The characteristics of the datasets are summarized in Table 4.1.

Table 4.1: Main properties of the experimental datasets.

Dataset	*Adressa*	*Last.fm*
#sessions	2,215	73,273
#users	50	2,501
#events	62,908	580,393
#items	6,765	7,899
#sessions_train	2,146	70,772
#events_train	41,746	560,976
#sessions_test	50	2,501
#events_test	5,650	19,417

To split users' historical logs into sessions, for both datasets, following Zheleva et al. [90] and Baur et al.[91], we use the time gap approach to generate sessions. If the gap between two reading/play items is less than 30 minutes for user u, they belong to the same session. Otherwise, they will be separated into two sessions. However, especially for Addressa dataset, users only read one articles for most of the time when setting time gap as 30 minutes, which may not have enough samples for training. Thus, we also test different time gaps that influence recommendation performance.

The testing set is build with the last session of each user. The remaining sessions form the training set. Besides, we also leave the last session of each user from training set as a validation set, which is used for hyper-parameter selection during each iteration in training procedure. To help the reproducibility of our experiments, we report the hyper-parameters of our model in Table 4.2.

[7] https://www.last.fm/api/show/artist.getInfo

Table 4.2: Main hyper-parameters of DeepJoNN.

Parameter	*Value*	*Parameter*	*Value*
Batch-size	20	Loss	TOP1
Learning rate	0.01	#Hidden units	100,100,100
Optimization	adagrad	#Parameters	2170100
L2	0.0001	Drop out	0.1/0.1/0
#Layer_CNN/RNN	2/1	Hidden activation	ReLU, tanh

4.3.2 Evaluation Metrics

Based on temporally ordered lists of read/played items, our objective is to correctly predict the next item a target user will likely read/play. The ground truth at a particular time step is therefore represented by a single user-item tuple. To present the user with adequate recommendations, the target item should be among the top few recommended items. Since we are interested in measuring top-k recommendation instead of rating prediction, we measure the quality by looking at the $Recall@K$ and $MRR@K$, which are widely used for evaluating top-k recommender systems.

- $MRR@k$ (Mean Reciprocal Rank) is defined as the average of the reciprocal ranks of the desired items [62]. The rank is set to zero if it is above k.
- $Recall@k$ is defined as the fraction of cases where the item actually consumed in the next event is among the top k items recommended [61].

We set $k = 20$, as it appears desirable from a user's perspective to expect the target among the first 20 items [39].

4.3.3 Baselines

To validate the effectiveness of DeepJoNN, we compared our model with the following session-based recommendation methods.

- Popular-based Method (POP): This method recommends items with the largest number of interactions by the users.
- Item KNN: Item KNN is a simple, yet effective method, which is widely deployed in practice. In this method, two item are considered similar if they co-occur frequently in different sessions. In our situation, we recommend items based on cosine similarity

between different sessions.

- BPR-MF[8]: It is one of the commonly used matrix factorization methods, but cannot directly apply to session-based recommendations for the new session do not have feature vectors precomputed. Instead, we use the average value of item feature vectors that had occurred in the session before the predicting point, as the user feature vector [89].
- Hierarchical RNN (HRNN)[9]: Proposed by [45], the model is a personalized RNN model with cross-session information transfer in a seamless way. HRNN relays end evolveds latent hidden states of the RNNs across user sessions.

4.3.4 Performance Evaluation

The performance of DeepJoNN and the baselines are reported in terms of $Recall@k$ and $MRR@k$ on two kinds of datasets in Figure 4.3. In addition to the baseline models, we also compare our DeepJoNN model with and without integrating time-decay factors, which are represented as DeepJoNN and $\mathrm{DeepJoNN} - \mathrm{t}$ respectively.

The figure shows that all models perform better on Last.fm dataset than Adressa. It is mainly because the sparsity and unbalance characteristics appearing in Adressa dataset. As described in [63], only small amount of news are read by a lot of users, and thus, in our experiments, these kinds of news have a better chance to get higher scores than the others. The deep learning models consistently outperform the other models on both datasets in terms of Recall and MRR, despite the fact that Item KNN is a very competitive baseline. Beside, all DeepJoNN models outperform HRNN model on both evaluation metrics which we believe is mainly because the integration of contextual features. Furthermore, the integration of tensor-based feature mapping method in CNN can also be a possible reason that explain the higher performance. The DeepJoNN model using time-decay factors consistently outperforms other experimented models in terms of all evaluation metrics, on both datasets.

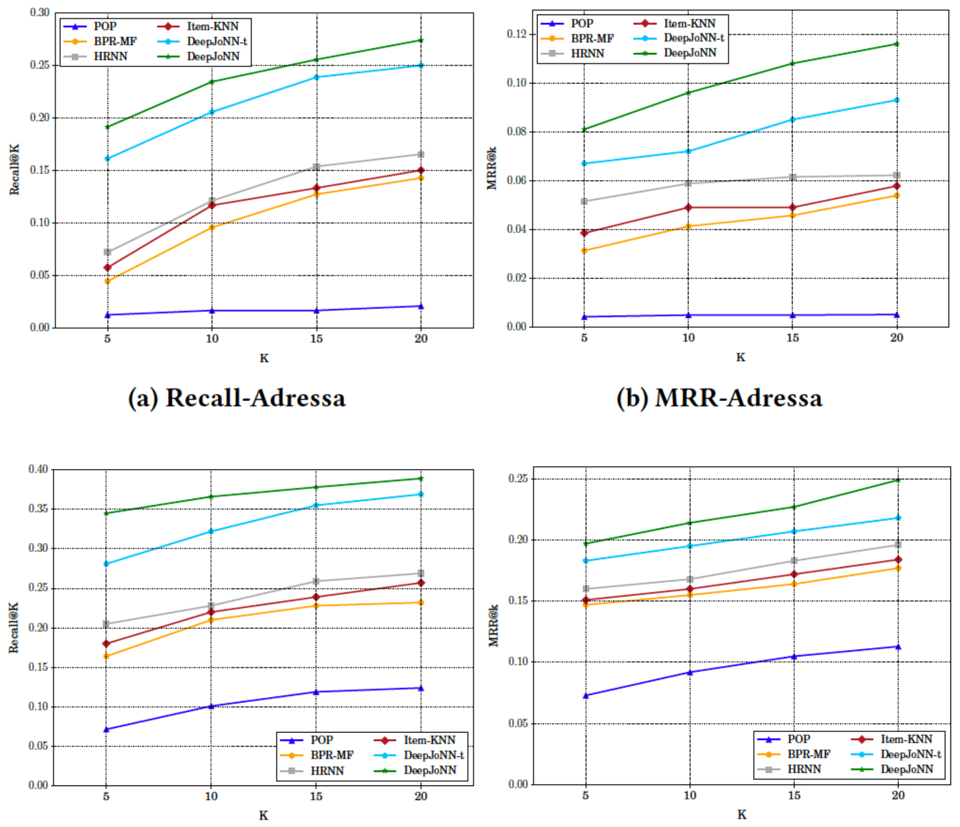
4.3.5 Model Parameter Analysis

In this section, we analyse the influence of the gap length of the session and the value of h

[8] https://github.com/bbc/theano-bpr

[9] https://github.com/mquad/hgru4rec

parameters in CNN filters to the performance of our model. In our experiments, parameter



(a) Recall-Adressa

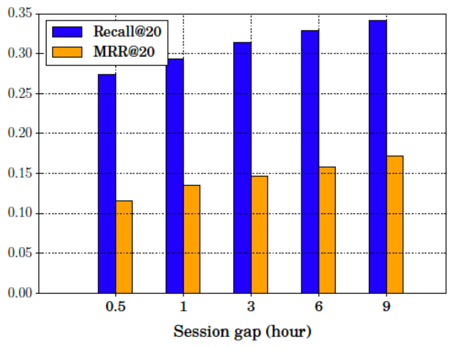
(b) MRR-Adressa

(c) Recall-Last.fm

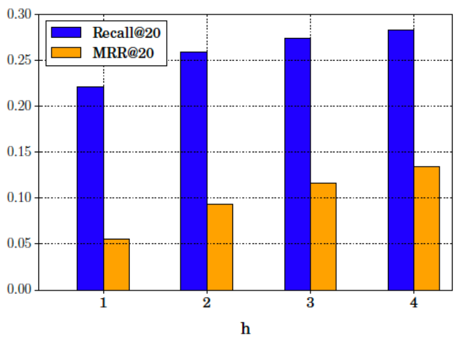
(d) MRR-Last.fm

Figure 4.3: Performance comparison w.r.t. top@k rank scores in terms of *Recall@k* and *MRR@k* on Adressa and Last.fm datasets. k ranges from 5 to 20 and h fixed at 3 for DeepJoNN and DeepJoNN-t.

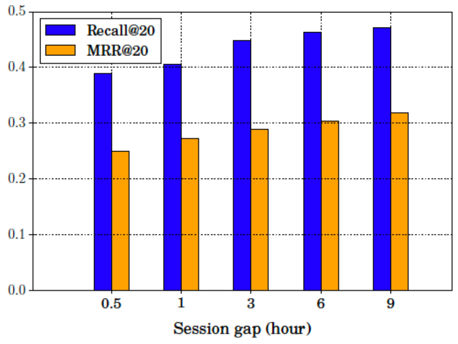
h is varied from 1 to 4, and session length is set to 0.5, 1, 3, 6, 9 hour(s). The results are shown in Figure 4.4. Just as we expected, the performance continually goes up along with the higher value setting to h for that the intra-relationships between features, especially between words and phrases, for one event are also important for recommendation performance. However, when h is more than 3, more computational resources are needed to process more parameters and tensor operations without recommendation performance obviously increasing. Thus, we believe 3 is a suitable order for balance the computational cost and model performance in our tasks. Besides, with smaller gap length of sessions, there are fewer users and events within a session and thus little diversities are incorporated in our dataset for one session. Furthermore, frequently switchings of users during training may also cause fluctuation effect of the parameter tuning and will lead to slow convergence.



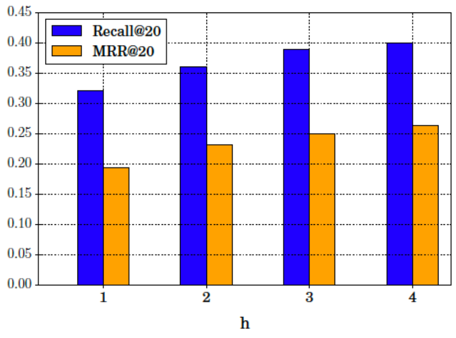
(a) Session gap-Adressa



(b) h value-Adressa



(c) Session gap-Last.fm



(d) h value-Last.fm

Figure 4.4: Performance of DeepJoNN on recommendation tasks with varied session lengths and *h* values.

4.3.6 Comparison of Loss Functions

In this section, we measure the performance gain of the proposed improvements over the 3 different loss functions, TOP1, Bayesian Personalized Ranking(BPR) [89] and Cross Entropy. In our tasks, we conducted 2 groups of experiments with 300 hidden units (2-layers CNN with 100 units in each layer, and 1-layer RNN with 100 units in each layer), and 600 hidden units (2-layers CNN with 200 units in each layer, and 2-layer RNN with

100 units in each layer) respectively. Within each group of experiments, we run 10 times with different random seeds for each loss function in order to get different performance in terms of Recall and MRR value. Our results are shown in Table 4.3, and the numbers in brackets represent the fluctuation towards positive direction.

Table 4.3: Recall@10 and MRR@10 for different types of configuration of DeepJoNN with different losses. Best results per dataset are highlighted.

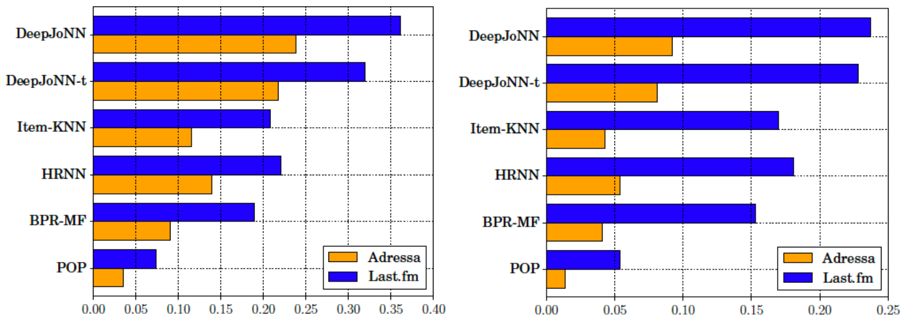
Loss / #Units	**Adressa Dataset**		**Last.fm Dataset**	
	Recall@20	*MRR@20*	*Recall@20*	*MRR@20*
TOP1 300	0.271 (+3.4%)	0.115 (+1.2%)	0.389 (+2.6%)	0.249 (+1.6%)
BPR 300	0.249 (+3.0%)	0.106 (+2.0%)	0.360 (+2.4%)	0.227 (+1.4%)
Cross-entropy 300	0.261 (+4.1%)	0.109 (+1.4%)	0.392 (+3.2%)	0.258 (+1.7%)
TOP1 600	0.296 (+2.5%)	**0.157 (+1.8%)**	**0.407 (+3.6%)**	**0.271 (+2.1%)**
BPR 600	**0.321 (+5.0%)**	0.119 (+1.3%)	0.367 (+2.1%)	0.241 (+1.2%)
Cross-entropy 600	0.263 (+2.6%)	0.103 (+1.5%)	0.374 (+1.7%)	0.238 (+1.3%)

From the listed results, it can be observed that cross entropy achieved better than the other two functions in terms of Last.fm, whereas TOP1 scheme get the best result for Adressa dataset with 300 hidden units. Therefore, for different recommendation tasks, different loss functions may result in slightly different recommendation performance. However, cross entropy is relatively unstable compared with other schemes for the number in brackets are slightly bigger under the same settings, which from another side proves the observation in research [39], that pointwise ranking based losses were usually unstable, even with regularization. Additionally, it does not always happen that deep networks with bigger and deeper configuration can achieve better performance which makes parameter tuning an important task in training deep networks.

4.3.7 Cold-Start Problem

Additionally, we also conducted experiments to study the effectiveness of different recommendation algorithms in addressing cold-start issues on the two kinds of datasets. Cold-start is normally refered to cold-start users and cold-start items two perspectives. The former indicates users with few historical interactions could be used, while the latter indicates items with few interactions with users. In this chapter, we only focus on cold-start users and leave the cold-start item issue to the future research direction. As preprocession, we removed users who have less then 3 events during sessions and less than 2 sessions in training set. The users who have more than 20 sessions are also removed from dataset. Besides, we also filtered users in testing set who were not contained in training dataset.

The experimental results are shown in Figure 4.5, from which we have the following observations: 1) our proposed DeepJoNN and DeepJoNN-t still performs best consistently in recommending coldstart cases; 2) by comparing the recommendation results in Figure 4.3, the Recall and MRR value of all recommendation algorithms kept nearly the same effectiveness. For instance, the Recall value of HRNN in cold-start settings is 13.90%, MRR values is 5.36%, and meanwhile it achieved nearly the same performance with value of 16.56% and 6.22% of Recall and MRR when set *k* to 20 on Adressa dataset. Similar phenomenon can also be found from Last.fm dataset. From training process, we also discovered that our DeepJoNN and HRNN converged faster than traditional RNN with better performance.



(a) Recall@20 (b) MRR@20

Figure 4.5: Recommendations for Cold-start Cases.

4.4 Conclusion

In this chapter, we presented a Deep Joint Network (DeepJoNN) for session-based recommendations. The proposed model allows combining various item features such as ID, category, keywords and entities, which then are transformed into character-level input matrix to the model. DeepJoNN consists of two parts of deep neural networks coupled together in a hierarchical way and thus could extract contextual patterns and process long and short-term dependencies simultaneously. In the comparison of the state-of-the-art baselines, DeepJoNN achieved nearly 11% and 12% improvements on datasets of Adressa and Last.fm respectively w.r.t. Recall value. Additionally, we also explored the influence of different parameter settings and conducted experiments on different loss functions. Our model also performed competitively on cold start users without user profiles.

Compared with existing methods, the innovation of our proposed approach lies in the way we integrated the tensor-based feature representation method into traditional convolutional operation and successfully applied it to encoding textual content for session-based recommendation purposes. The use of the outer product is conducted to explore the potential impact of combined features before convolution operation such that the explicit information can further be processed by CNN module. The improvements of recommendation accuracy verify the potential value of our approach in mining the content efficacy in affecting user preferences in session settings.

Chapter 5

Session-based Recommendations with Multi-level Side Information

Online news recommendation aims to continuously select a pool of candidate articles that meet the temporal dynamics of user preferences. Most of the existing methods assume that all user-item interactions are equally important for recommendations, which is not always applied in real-world scenarios since the user-item interactions are sometimes full of stochasticity and contingency. In addition, previous work on session-based algorithms only considers user sequence behaviors within the current session without incorporating users' historical interests or pointing out users' main purposes within such session. In this chapter, we propose a novel neural network framework, a dynamic attention-integrated neural network, to tackle the problems. Specifically, we propose a dynamic neural network to model users' dynamic interests over time in a unified framework for personalized news recommendations. News article semantic embedding, user interests modelling, session-based public behavior mining and an attention scheme that is used to learn the attention score of user and item interaction within sessions are four key factors for online sequences mining and recommendation strategy. Experimental results on three real-world datasets show significant improvements over several baselines and state-of-the-art methods on session-based neural networks.

5.1 Introduction

With the rapid development of web services and e-commerce platforms, news recommender systems have become popular and are employed by many multimedia companies in recent years. They are able to cope with the information overload and to assist users in finding information matching their individual profiles learned from historical user-item interactions. However, in many real-life recommendation settings, user profiles and past activities are not available, which renders traditional recommendation methods [33, 92, 93, 94] less useful. To a large extent, unprofiled users occupy a greater proportion of the total news readers, because many news websites allow users to read articles without authentication (not registered). To tackle this problem, session-based recommendation [95] is proposed to predict the next item that the user is probably interested in based solely on implicit feedback, i.e., user clicks, in the current session.

To have a better understanding of user interests modelling in the session-based recommendation, we show the user clicking patterns on three topics: winter sports (vintersport), culture (kultur) and local news (nordtrondelag) over three months in the experimental dataset in Figure 5.1. As can be seen from the sessions within the hour of 10 and 19 in two small graphs with the x-axis representing item id that the user clicked and the y-axis representing user id, the clicked item sets of different users are extremely similar in their respective sessions within a time period (1 min in our paper). It means that different users across different sessions (we refer to neighbourhood sessions in the following parts) have similar interests, and they tend to focus on the most popular/emerging topics within some time period. In two small graphs, the user id and item id are consistent. Thus, we can also find that the clicked item sets of the same user across different sessions at different time slices change over time, meaning that the user interests drift at different times in a day or on different days. Besides, from the line chart in Figure 5.1, we can find from the long-term click frequency of different topics that, there exist a number of periodic user interests e.g. culture topic, and continuous user interests e.g. local news and seasonal user interests e.g. winter sport, which can be valuable to recommend items. Therefore, it is critical to incorporate the aforementioned factors when modelling user interest for session-based news recommendation.

Recently, Hidasi et al. [39] apply recurrent neural networks (RNN) with Gated Recurrent Units (GRU) for session-based recommendation. The model considers the first item clicked by a user as the initial input of RNN, and generates recommendations based on it. Then the user might click one of the recommendations, which is fed into RNN next, and the successive recommendations are produced based on the whole previous clicks. Tan et al. [40] further improve this RNN-based model by utilizing two crucial techniques, i.e., a

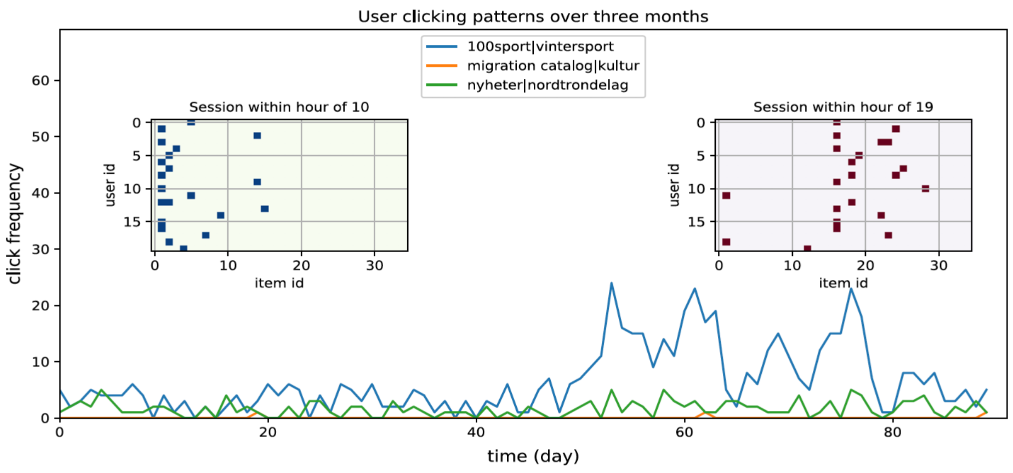


Figure 5.1: User clicking patterns over three months on different topics and user clicking patterns in the neighbourhood sessions of different users within 1 min.

method to account for shifts in the input data distribution and data augmentation. Despite these positive results, some problems regarding the effectiveness of the session-based recommendation method remain open: (1) they only take into account the user's sequential behavior in the current session, whereas the user's main purpose within that session is not emphasized. In other words, these methods cannot automatically select important interaction records in the user-item interaction history when recommending items. This greatly limits their application in real-world scenarios where a user accidentally clicks on wrong items or s/he is attracted by some unrelated items due to curiosity. (2) they do not incorporate the knowledge acquired on the long-term dynamics of the user interest in the session-based algorithm when user profiles are available. In such cases, it is reasonable to assume that the user behavior in past sessions might provide valuable information for providing recommendations in the next session.

In this chapter, we propose a novel dynamic attention-integrated neural network (DAINN) to tackle the aforementioned problems for the personalized recommendation task. Specifically, DAINN models the users' dynamic interests over time by jointly incorporating users' long-term interests, user behavior sequence patterns, users' main purpose in the current session, as well as public behavior mining into a unified framework. In order to improve the recommendation accuracy, dynamic topic modelling [6] and convolutional neural network (CNN) sentence model [96] are adopted to effectively learn the item semantic embedding. More importantly, to handle diverse variance of users' clicking behavior, we introduce a novel attention scheme that would dynamically assign influence factors on recent models based on the users' spatio-temporal reading characteristics. We applied our model to several real data sets and the experimental results demonstrate the promising and reasonable performance of our approach.

In summary, our contributions are as follows:

1) We propose a dynamic attention-integrated neural network (DAINN) to model users' dynamic interests over time in a unified framework for personalized session-based news recommendation.

2) The proposed model can jointly exploit users' long-term interests, user behavior sequence patterns, users' main purpose in the current session, as well as public behavior mining to model users' preferences. In addition, item semantic embedding learned from CNN sentence model is adopted to further improve the recommendation accuracy.

3) To handle the diverse variance of users' clicking behavior, a novel attention scheme is proposed, which considers the spatio-temporal reading characteristics of users.

4) We apply DAINN to three real-world datasets with extensive experiments. The results show that DAINN achieves substantive gains over state-of-the-art deep learning-based methods for recommendation. Specifically, DAINN outperforms baselines by 3 to 5% on F1 score and 2 to 5% on MRR.

The remainder of this chapter is organized as follows. In Section 5.2, we formally define our problem and present DAINN model. We describe the data sets, experiment settings and the prior information we use in Section 5.3. Section 5.4 shows a comprehensive experiment evaluation. Finally, we present the conclusions in Section 5.5.

5.2 Methodology

In this section, we propose a novel dynamic attention-integrated neural network (DAINN) for session-based news recommendation. Firstly, the problem is defined, including the relevant general terms and notations. Then we give the details about the unified recommendation framework, which includes user long-term interest modelling, temporal context mining, session-based public behavior mining, dynamic attention learning. As shown in Figure 5.2, the proposed DAINN model can be regarded as an interest network by considering the four factors jointly for learning users' dynamic preferences.

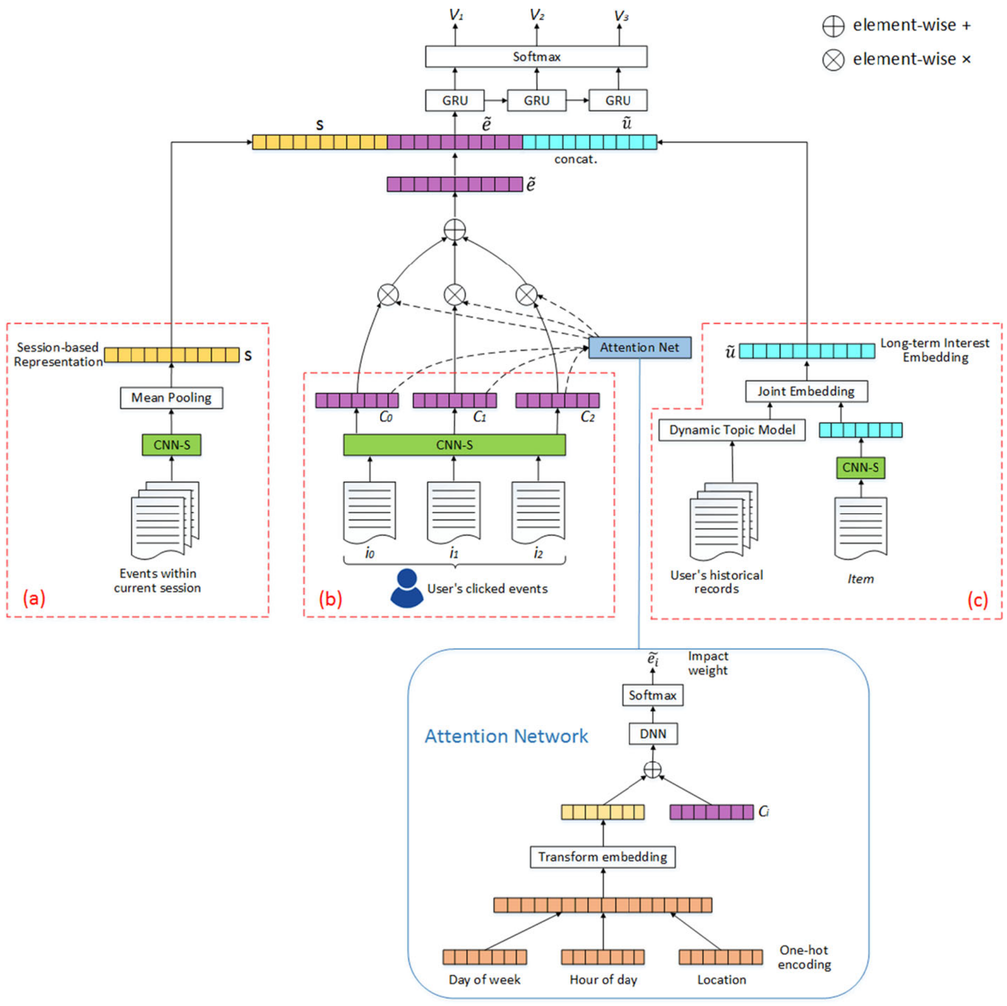


Figure 5.2: The unified framework for the personalized news recommendation via dynamic attention-integrated neural network.

5.2.1 Problem Definition

Notation. Throughout this paper, all vectors are column vectors and are denoted by bold lower case letters (e.g., $\boldsymbol{x}$ and $\boldsymbol{y}$), while matrices are represented by bold upper case letters (e.g., $\boldsymbol{X}$ and $\boldsymbol{M}$). The ith row of a matrix $\boldsymbol{X}$ is given by $\boldsymbol{X}_{i\cdot}$, while $\boldsymbol{X}_{\cdot j}$ represents the jth column. We use calligraphic letters to represent sets (e.g., $\mathcal{V}$ and $\mathcal{E}$). Table 5.1 summarizes the notations of frequently used variables.

Table 5.1: Notations used in this chapter.

Symbol	Description
$\boldsymbol{x}_i$	Representation of one clicked item in a session
n_s	The number of events in one session
m	The number of candidate items
n	The number of words in the input sentence
$\mathcal{V}$	The set of word vocabulary
N_w	The total number of words in vocabulary $\mathcal{V}$
d	Dimension of word embedding
$\boldsymbol{c}$	Representation of semantic embedding
w	Sliding window
n_u	The total number of clicked events within one session for user u
$\boldsymbol{s}$	Session-based representation
θ	Parameters of DAINN framework
$\boldsymbol{T}_p$	Parameter matrix of the softmax layer in GRU
N_u	The number of user's historical interested topics
λ	Time decay parameter
$\boldsymbol{T}_c$, $\boldsymbol{T}_u$	Transformation matrix in user long-term modelling
D_e, D_c	Dimensionality of the learned user topics representation and the textual embedding
N_c	Normalization parameter in user long-term modelling
D_d, D_h, D_l	The number of days in a week, the number of hours in a day and the number of locations in dataset
$\tilde{\boldsymbol{e}}_t$	The embedding after attention network of user u at timestamp t
$\tilde{\boldsymbol{u}}$	The embedding after user long-term modelling of user u
$\boldsymbol{v}$	The word distribution

Session-based recommendation. Session-based recommendation is the task of predicting what a user would like to click next when his/her current sequential transaction data is given. Here we give a formulation of the session-based recommendation problem.

Let $[\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_{n_s-1}, \boldsymbol{x}_{n_s}]$ be a click session, where $\boldsymbol{x}_i \in I$ $(1 \le i \le n_s)$ is the representation of one clicked item out of a total number of m candidate items. We build a model $\boldsymbol{\mathcal{F}}$ so that for any given prefix of the click sequence in the session, $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_{t-1}, \boldsymbol{x}_t]$, $1 \le t \le n_s$, we get the output $\boldsymbol{y} = \boldsymbol{\mathcal{F}}(\boldsymbol{X})$, where $\boldsymbol{y} =$

$[y_1, y_2, \ldots, y_{m-1}, y_m]$. We view $\boldsymbol{y}$ as a ranking list over all the next items that can occur in that session, where y_j $(1 \leq j \leq m)$ corresponds to the recommendation score of item j. Since a recommender typically needs to make more than one recommendations for the user, thus the top-k $(1 \leq k \leq m)$ items in $\boldsymbol{y}$ are recommended.

5.2.2 Dynamic Attention-integrated Neural Network

Overview. To improve the recommendation performance in news domain and address session-based recommendation problems, we proposed a novel dynamic attention-integrated neural network (DAINN). The basic idea of our model is to build a unified represent- ation of the current user, and then generate predictions on the user's next possible event with it. The representation should take into account various potential factors that influence user's next decision. As shown in Figure 5.2, the input of GRU is a joint output from three components, namely session-based public behavior mining, dynamic attention learning and user long-term interest modelling represented as (a), (b) and (c) respectively. The basic input is the sequence $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n]$ where $\boldsymbol{x}_i$ denotes the word-level representation of item i. Component (a) transfers the input sequence $\boldsymbol{X}_p = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{n_p}]$ collected from users within a predefined sliding window ω except current user u, into the representation $\boldsymbol{s}$ of public behavior pattern. Component (c) learns from user u's historical records and outputs the representation $\tilde{\boldsymbol{e}}$ of user's long-term interest pattern. Meanwhile component (b) converts the input sequence $\boldsymbol{X}_s = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{n_s}]$ of user u's current session into the high dimensional representation $\tilde{\boldsymbol{u}}$ with user's current purpose, along with the attention weight at time t (represented as s_t). Finally, the concatenation of the three representations is fed into GRU to generate top-k items with the highest possibilities that user u will click next. One should be clarified that CNN for semantic embedding models, denoted as CNN-S in Figure 5.2, share parameters in three components. CNN-S is adopted to extract semantic information from simple word-level representations of inputs, and its output is denoted as $\boldsymbol{c}$ in our paper.

In the following part of this section, we first describe the CNN-S model for semantic embedding used in each component. Then we introduce the session-based public behavior mining which is used to extract neibourhood session patterns from public users of component (a), user's long-term interest modelling of component (c) and dynamic attention learning which is used to extract user's main purpose within current session of component (b). The learning objective is introduced next, and finally, the top-k items generation process is explained.

CNN for semantic embedding. To model the textual content of the document, traditional methods including bag-of-words features [97, 98], e.g. TF-IDF feature or Naive Bayes and unsupervised learning objective [99, 100], e.g. topic models, are based on counting statistics which ignore word orders and suffer from sparsity and poor generalization performance. A more effective way to model the text is to represent each sentence in a given corpus as a distributed low-dimensional vector. Recently, inspired by the success of applying convolutional neural networks (CNN) in the field of computer vision [101], researchers have proposed many CNN-based models for semantic embedding [85, 96][10]. In this subsection, we introduce a typical type of CNN architecture, namely Kim CNN [96].

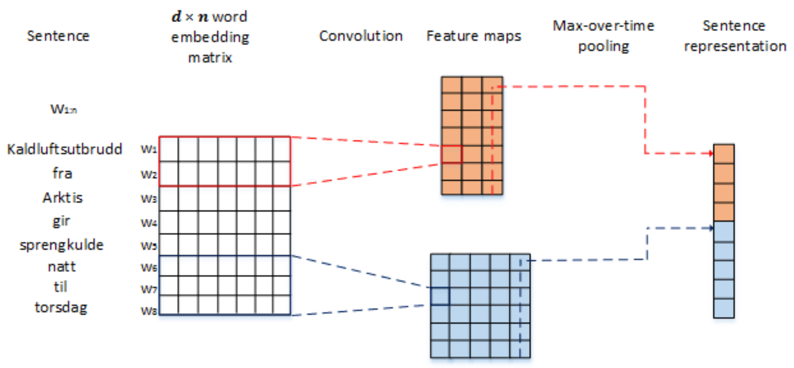


Figure 5.3: A typical architecture of CNN for semantic embedding (Kim 2014).

Figure 5.3 illustrates the architecture of Kim CNN. In Figure 5.2, Kim CNN are denoted as CNN-S. Let $W_{1:n}$ be the raw input of a sentence of length n, and $\boldsymbol{x} = [x_1, x_2, \dots, x_n] \in \mathbb{R}^{1\times n}$ be the word-level representation vector of the input sentence, where $x_i \in \mathbb{R}$ is the index of the ith word in vocabulary $\mathcal{V}$ in the sentence. We can get the word embedding of the ith word through word2vec pre-trained model $\boldsymbol{w}_i = \boldsymbol{\mathcal{H}}(x_i;\, \mathcal{V}), \boldsymbol{w}_i \in \mathbb{R}^{d\times 1}$, where d is

[10] Researchers have also proposed other types of neural network models for semantic embedding such as recurrent neural networks [102], recursive neural networks [103], and hybrid models [104]. However, CNN-based models are empirically proven to be superior than others [105] since they can detect and extract specific local patterns from sentences due to the convolution operation. To keep our presentation focused, we only discuss CNN-based models in this paper.

the dimension of word embeddings. Thus, we can get $\boldsymbol{W}_{1:n} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_n] \in \mathbb{R}^{d\times n}$, the word embedding matrix of the input sentence. A convolution operation with filter $\boldsymbol{h} \in \mathbb{R}^{d\times l}$ is then applied to the word embedding matrix $\boldsymbol{W}_{1:n}$, where l ($l \leq n$) is the window size of the filter. Specifically, a feature $\boldsymbol{d}_i$ is generated from a sub-matrix $\boldsymbol{W}_{i:i+l-1}$ by

$$\boldsymbol{d}_i = f(\boldsymbol{h} * \boldsymbol{W}_{i:i+l-1} + b) \quad (1)$$

where f is a non-linear transformation function such as the hyperbolic tangent (tanh) f (z) = (exp(z)−exp(−z))/(exp(z)+exp(−z)), $*$ is the convolution operator, and $b \in \mathbb{R}$ is a bias. After applying the filter to every possible position in the word embedding matrix, a feature map

$$\boldsymbol{D} = [\boldsymbol{d}_1, \boldsymbol{d}_2, \dots, \boldsymbol{d}_{n-l+1}] \quad (2)$$

is obtained, then a max-over-time pooling operation is used on feature map $\boldsymbol{D}$ to identify the most significant feature:

$$\boldsymbol{c} = \max\{\boldsymbol{D}\} = max[\boldsymbol{d}_1, \boldsymbol{d}_2, \dots, \boldsymbol{d}_{n-l+1}] \quad (3)$$

One can use multiple filters (with varying window sizes) to obtain multiple features, and these features are concatenated together to form the representation of the textual content.

Session-based public behavior mining. As described in Figure 5.1, user clicking patterns across neighbourhood sessions with different users within some time periods are extremely similar. Besides, many recent works [39, 106] have proved the efficiency of adopting session-based methods on especially non-profile users. According to the statistics on our experimental dataset, users with historical records (also known as subscribers) take up less than 20% of total number of users. Thus, inter- and intra-session information is essential for recommendations.

Assuming that $\boldsymbol{X}_p = [\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_{n_p}]$ is a sequence of events that clicked within a predefined sliding window ω which is the time period before the current time t. n_p denotes the number of items within window ω. Each $\boldsymbol{x}_i$ represents the word-level item representation of the user excluding the current user u. As illustrated in Figure 5.2a, $\boldsymbol{X}_p$ is firstly put into CNN-S model to obtain the textual semantic embedding of these items according to Eq. (3) denoted as $\boldsymbol{C} = [\boldsymbol{c}_1, \boldsymbol{c}_2, \dots, \boldsymbol{c}_{n_p}]$. Then we use mean-pooling through horizontal axis as user u's session representations

$$\boldsymbol{s} = \frac{1}{n}\sum_{j=1}^{n} \boldsymbol{c}_j \quad (4)$$

If we do not consider the attention network and user u is a newly arrived user which has no historical record, we only consider item $\boldsymbol{x}_t$ that user u clicks at current time t. Then the semantic representation denoted as $\boldsymbol{c}_1$, of $\boldsymbol{x}_t$ can be acquired through CNN-S model. After that, the concatenation of embedded session-based representation $\boldsymbol{s}$ and semantic representation $\boldsymbol{c}_1$, $\boldsymbol{s} \oplus \boldsymbol{c}_1$, is sent through one or multiple layers of the Gated Recurrent Unit (GRU) [107] which is the simplified version of Long Short-Term Memory (LSTM) networks but still maintains all their properties. In GRU unit, the activation h_t at time t is a linear interpolation between the previous activation h_{t-1} and the candidate activation $\tilde{h}_t$:

$$\boldsymbol{h}_t = (1 - z_t)\boldsymbol{h}_{t-1} + z_t\tilde{\boldsymbol{h}}_t \tag{5}$$

where an update gate z_t decides how much the unit updates its activation, or content. The update gate is computed by

$$z_t = \sigma(\boldsymbol{W}_z\tilde{\boldsymbol{x}}_t + \boldsymbol{U}_z\boldsymbol{h}_{t-1}) \tag{6}$$

This procedure of taking a linear sum between the existing state and the newly computed state is similar to the LSTM unit. The GRU, however, does not have any mechanism to control the degree to which its state is exposed, but exposes the whole state each time. The candidate activation $\tilde{h}_t$ is computed similarly to that of the traditional recurrent unit but slightly different from Cho et al. [107]

$$\tilde{\boldsymbol{h}}_t = tanh(\boldsymbol{W}\tilde{\boldsymbol{x}}_t + \boldsymbol{U}(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1})) \tag{7}$$

where $\boldsymbol{r}_t$ is a set of reset gate and $\odot$ is an element-wise multiplication, and $\tilde{\boldsymbol{x}}$ is the output from previous layer or $\boldsymbol{s} \oplus \boldsymbol{c}$. We experimented on both formulations to compute $\tilde{\boldsymbol{h}}_t$ and they performed as well as each other. When $\boldsymbol{r}_t$ is close to 0, the reset gate effectively makes the unit act as if it is reading the first symbol of an input sequence, allowing it to forget the previously computed state. The reset gate can be computed as

$$\boldsymbol{r}_t = \sigma(\boldsymbol{W}\tilde{\boldsymbol{x}}_t + \boldsymbol{U}_r\boldsymbol{h}_{t-1}) \tag{8}$$

The output of GRU at timestamp t can be denoted as $\boldsymbol{o}_t = \boldsymbol{h}_t$. Inspired by the work of Pan et al. [108], we formulate our recommendation problem as a coherence loss, where the log probability of the recommendation is given by the sum of log probabilities over the clicked items as shown below

$$\mathcal{L}_{rec}(\boldsymbol{X}, \boldsymbol{v}) = -\log P(\boldsymbol{v}|\boldsymbol{X}) = \textstyle\sum_{t=1}^{n_s} -\log P(\boldsymbol{v}_t|\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, \dots, \tilde{\boldsymbol{x}}_{t-1}; \theta) \tag{9}$$

where $\{\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, \dots, \tilde{\boldsymbol{x}}_{n_s}\}$ is the sequentially predicted items. Here, $\tilde{\boldsymbol{x}}_i$ is corresponding to the representation of item i. $\boldsymbol{v}_t$ is the words distribution of next possible item. θ are the parameters of our framework, including parameters of CNN-S model and GRU model. By minimizing the above loss, the user's interests within and across sessions can be described dynamically. Here, a softmax layer is applied after GRU layer to produce a probability distribution over all the N_w words in the vocabulary as

$$P(\boldsymbol{v}_t|\widetilde{\boldsymbol{x}}_1, \widetilde{\boldsymbol{x}}_2, \dots, \widetilde{\boldsymbol{x}}_{t-1}; \theta) = \frac{\exp\{\boldsymbol{T}_p \boldsymbol{h}_t\}}{\sum_{j=1}^{N_w} \exp\{\boldsymbol{T}_p \boldsymbol{h}_j\}} \tag{10}$$

where $\boldsymbol{T}_p$ is the parameter matrix of the softmax layer in GRU.

User long-term interest modelling. Although texts have semantic information, they cannot reflect users' broad interest directly [109]. To represent texts and users' interest in a common space, as shown in Figure 5.2c, we jointly learn the relevance between text semantic embedding and user interested topics. Specifically, we first conduct some online topic modelling approach on all the users' historical behavior streams (e.g., news clicking streams or song listening streams) to build a shared user topic space and learn the topical distribution for each user. Then we aggregate the topic distributions of each user's real-time behavior streams to derive the representation of user interested topics at the current time, where a time decay [110] is used to weight the behavior streams. Therefore, the user's interested topics can be defined as in Eq. (11).

$$\boldsymbol{u} = \frac{1}{N_u} \sum_{i \in \mathcal{B}_u} \boldsymbol{m}_i \cdot e^{-\lambda|t-t_i|} \tag{11}$$

where $\boldsymbol{m}_i$ denotes representation of a user's interested topics of the ith behavior, $\mathcal{B}_u$ is the user's historical behaviors, $|t - t_i|$ indicates the time difference between the current time and the post time of user behavior i. N_u is the number of user's historical interested topics and λ is the time decay parameter. In this chapter, topics are extracted by Dynamic Topic Model introduced in Greene and Cross [111] and $\boldsymbol{m}_i$ is the word-level representation of topic i.

To project the textual semantic embedding and user interested topics into a common space, we adopt two transformation matrices, $\boldsymbol{T_c} \in \mathbb{R}^{D_e \times D_c}$ and $\boldsymbol{T_u} \in \mathbb{R}^{D_e \times D_u}$, where D_u and D_c is the dimensionality of the learned user topics representation and textual embedding respectively. To measure the relevance between textual semantic embedding and the user interested topics, one direct way is to calculate the distance between them. We integrate the textual semantic embedding of a user's clicking list in Eq. (12), and the distance loss is defined in Eq. (13):

$$\tilde{\boldsymbol{u}} = \frac{1}{N_c} \sum_{c \in \mathcal{T}_u} \boldsymbol{c} \cdot e^{-\lambda |t - t_c|} \tag{12}$$

$$\mathcal{L}_{long}(\boldsymbol{\mathcal{U}}, \tilde{\boldsymbol{\mathcal{U}}}) = \sum_{\boldsymbol{u} \in \boldsymbol{\mathcal{U}}, \tilde{\boldsymbol{u}} \in \tilde{\boldsymbol{\mathcal{U}}}} \| \boldsymbol{T_u} \cdot \boldsymbol{u} - \boldsymbol{T_c} \cdot \tilde{\boldsymbol{u}} \|_F^2 \tag{13}$$

where $\mathcal{T}_u$ is the textual semantic embedding vectors of the clicked news for user u. $|t - t_c|$ indicates the time difference between the current time and the post time when the user clicks the specific news/song. N_c is a normalization parameter. One need to be noticed that, if user long-term interests can be achieved, for personalized recommendation tasks, CNN-S model in Figure 2a–c parts will share parameters for user u.

Dynamic attention learning. Given user u with clicked items $\{i_1, i_2, \ldots, i_{n_s}\}$ within session s , and his/her learned contextual representation after CNN-S can be defined as $\{\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_{n_s}\}$. To represent user u's attention at timestamp t, one can simply average all the embeddings of his/her clicked items:

$$\tilde{\boldsymbol{c}}_t = \frac{1}{n_s} \sum_{k=1}^{n_s} \boldsymbol{c}_k \tag{14}$$

However, user's interests are full of stochasticity and contingency, and user's clicked items supposed to have different impacts on the next possible clicking item. Specifically, our attention measurement scheme is mainly constructed based on threefold factors:

- Day of Week: Users read different topics of news at different week days, for example, during a working day or at the weekend, while relaxing.
- Hour of day: As illustrated in Fig. 1, user's interested topics may vary over time across day. For instance, a user may tend to read more financial news in the morning than in the afternoon, while s/he reads more sports or entertainment news at night.
- Location: According to the analysis of address a dataset, we find that users incline to read news happening around them. For example, a user from Oslo reads more news occurred in Oslo than news occurred in other regions.

In order to incorporate these three aspects, we first use the one-hot representation to denote the three factors. Specifically, we take binary vectors $\boldsymbol{r}_d \in R^{D_d}, \boldsymbol{r}_h \in R^{D_h}$ and $\boldsymbol{r}_l \in R^{D_l}$, where only the value of the column corresponding to the presented day, hour and location are set as 1 and the values for other columns are 0. D_d, D_h and D_l represent the number of days in a week, the number of hours in a day and the number of locations in dataset respectively. Then, the three vectors are concatenated as $\boldsymbol{r}_t = [\boldsymbol{r}_d; \boldsymbol{r}_h; \boldsymbol{r}_l]$. To learn the three factors and item's representation $\boldsymbol{c}_i$ together, one ordinary way is the simple concatenation strategy as $\boldsymbol{e}_t = [\boldsymbol{c}_t; \boldsymbol{r}_t]$. However, we argue that factor embedding and item embedding are learned by different methods, which means they are in different representation space. Thus, we introduce the transformed embeddings

$$\tilde{\boldsymbol{r}}_t = g(\boldsymbol{r}_t) \tag{15}$$

where $g(\cdot)$ is the transformation function, and can be either linear

$$g(\boldsymbol{r}_t) = \boldsymbol{T}\boldsymbol{r}_t \tag{16}$$

or non-linear

$$g(\boldsymbol{r}_t) = sigmoid(\boldsymbol{T}\boldsymbol{r}_t + \boldsymbol{b}) \tag{17}$$

where $\boldsymbol{T} \in R^{D_{\tilde{r}} \times D_r}$ is the trainable transformation matrix and $\boldsymbol{b} \in R^{D_{\tilde{r}} \times 1}$ is the trainable bias. Since the transformation is continuous, it can map factor embeddings to item space while preserving their original spacial relationship. We therefore can concatenate these two embeddings as $\boldsymbol{e}_t = [\boldsymbol{c}_t; \tilde{\boldsymbol{r}}_t]$ at timestamp t.

Inspired by the work in Wang et al. [32], we use an attention network to model the different impacts of user's clicked news $\boldsymbol{c}_t$. The attention network is illustrated in the bottom part of Figure 5.2. Different from Wang et al. [32], we not only consider the clicking patterns within current session, but also integrate various influential factors into the attention model. Specifically, for user u's clicked news representation $\boldsymbol{c}_t$ at timestamp t and factor representation $\tilde{\boldsymbol{r}}_t$, after concatenation of their embeddings, we apply a DNN $\mathcal{G}$ as the attention network and the softmax function to calculate the normalized impact weight:

$$s_t^i = softmax\left(\mathcal{G}(\boldsymbol{e}_t^i)\right) = \frac{\exp(\mathcal{G}(e_t^i))}{\sum_{k=1}^{N_u} \exp(\mathcal{G}(e_t^k))} \tag{18}$$

The attention network $\mathcal{G}$ receives concatenation embeddings as input and outputs the impact weight. Then the embedding of user u at timestamp t can be calculated as the weighted sum of his clicked news embeddings:

$$\tilde{\boldsymbol{e}}_t = \sum_{i=1}^{N_u} s_t^i \, \boldsymbol{e}_t^i \tag{16}$$

We will demonstrate the efficacy of the attention network in the experiment section.

Unified recommendation framework. Recall that in Sect. 5.2.2.1, we formulate our recommendation problem as a coherence loss in Eq. (9) with respect to the input item representation and the output words distribution. If we also consider user's historical records as described in Sect. 5.2.2.3, given a user's current interested topics u, we can formulate our recommendation problem as

$$\mathcal{L}_{rec}(\boldsymbol{u},\boldsymbol{v}) = -\log P(\boldsymbol{v}|\boldsymbol{u},\boldsymbol{X}) = \sum_{t=1}^{n_s} -\log P(\boldsymbol{v}_t|\boldsymbol{u},\boldsymbol{x}_1,\dots,\boldsymbol{x}_{t-1};\theta) \tag{20}$$

where θ represents not only the parameters of GRU and session-based CNN sentence network, but also $\boldsymbol{T_u}, \boldsymbol{T_c}$ of user long-term interest model. The input of GRU layer is the concatenation of session-based representation s, output of attention model $\tilde{\boldsymbol{e}}$ and the user long-term interest embedding $\tilde{\boldsymbol{u}}$, denoted as $\boldsymbol{s} \oplus \tilde{\boldsymbol{e}} \oplus \tilde{\boldsymbol{u}}$. By minimizing the above loss, the user interest evolvement can be described dynamically, which makes the recommendation more coherent and reasonable. Finally, we can obtain the objective function below

$$\mathcal{L} = \sum_{u\in\mathcal{U}} \mathcal{L}_{rec}(\boldsymbol{u},\boldsymbol{v}) + \lambda_1\mathcal{L}_{long}(\boldsymbol{\mathcal{U}},\tilde{\boldsymbol{\mathcal{U}}}) + \lambda_2\|\theta\|_2^2 \tag{21}$$

where λ_1 is the trade-off parameter for these objectives, and λ_2 is the coefficient of the weight decay term. By optimizing the above overall loss function in a unified framework, our proposed method achieves dynamic news recommendation with considering inter- and intra-session modelling, user interest modelling, as well as dynamic attention learning.

Recommending top-K items. Given a target user u with the request time t, in order to recommend top-K items that user u would like to choose, we compute the ranking score with respect to the predicting words distribution and item word-level distribution as in Eq. (22)

$$S(\boldsymbol{v}_i,\boldsymbol{v}_j,t) = \boldsymbol{v}_i\cdot\boldsymbol{v}_j = \sum_{k=1}^{N_w} v_{ik}\cdot v_{jk} \tag{22}$$

where $\boldsymbol{v}_i = [v_{i1},\dots,v_{iN_w}]$, N_w is the number of vocabulary. Since the effectiveness of news articles are very short (usually less than 7 days), the candidate items are limited for target users when performing Eq. (22). In other words, we can filter candidate items according to their publication time before calculate items' ranking score, and thus avoiding computing all possible items in database.

5.3 Experimental Setup

5.3.1 Datasets

We used three datasets from different areas for our experiments, namely Adressa, Last.fm and Weibo-Net-Tweet. The first is the Adressa 16G dataset[11] which contains 93,948 news articles, 398,545 readers, and about 113 million events over a 90-days period [96]. Each of these events represent that a user read a particular news article. As preprocessing, we filtered sessions with less than 3 events for a user. Besides, we removed the records that users visited the news front page for there are no articles related information within the events. In order to evaluate our model's generality, we adopted Last.fm provided by Schedl [64], which contains 10 weeks of log data between 1/1/2013 and 11/3/2013[12]. To enrich the content information for the dataset,we also used Last.fm API[13] to collect artist information to improve the recommendation accuracy. The third dataset is provided by Jing et al. [112] from Sina Weibo.com[14], which includes in total 1.7 million users and 300 thousand microblogs. We perform the similar preprocessing procedure on the other two types of datasets, including getting rid of the session with less than 3 events and removing the duplicate records. The characteristics of the datasets are summarized in Table 5.2.

To split users' historical logs into sessions, for Adressa dataset, it contains tags to represent the start and end of a session. As for Last.fm and Weibo-Net-Tweet datasets, following Zheleva et al. [90] and Baur et al. [91], we use the time gap approach to generate sessions. If the gap between two post items is less than 30 min for user u, they belong to the same session. Otherwise, they will be separated into two sessions.

The testing set is build with the last event of each session of each user. The remaining events form the training set. Besides, we also leave the last event of each session of each user from training set as a validation set, which is used for hyper-parameter selection during each iteration in training procedure. In order to test the user's long-term interests influence on our recommendation approach, we also selected users with historical records, namely user profile, from these three datasets to do the evaluation. The rest users without user profile are experimented as cold start problem in the following section.

[11] http://reclab.idi.ntnu.no/dataset/

[12] http://www.cp.jku.at/datasets/LFM-1b/

[13] https://www.last.fm/api/show/artist.getInfo

[14] https://www.aminer.cn/influencelocality

Table 5.2: Some statistics of the datasets.

Dataset	Adressa	Last.fm	Weibo-Net-Tweet
#sessions	9,211,140	73,273	2,126,697
#users	4,805,071	2501	1,776,950
#events	113,579,695	580,393	23,755,810
#items	48,486	7899	300,000
#events train	104,368,555	507,120	21,269,113
#events test	9,211,140	73,273	2,126,697
#events per session	12	8	11

5.3.2 Evaluation metrics

Based on temporally ordered lists of read/played items, our objective is to correctly predict the next item a target user will likely read/play. The ground truth at a particular time step is therefore represented by a single user-item tuple. To present the user with adequate recommendations, the target item should be among the top few recommended items. Since we are interested in measuring top-k recommendation instead of rating prediction, we measure the performance by looking at the Recall@k, Precision@k, F1 score and MRR@k, which are widely used for evaluating top-k recommender systems.

- MRR@k (Mean Reciprocal Rank) is defined as the average of the reciprocal ranks of the desired items [62]. The rank is set to zero if it is above k.
- Precision@k is defined as the proportion of recommended items in the top-k set that are actually consumed in the next event.
- Recall@k is defined as the proportion of the items actually consumed in the next event among the top k items recommended.
- F1 score is the harmonic mean between recall and precision values and can be denoted as F1 = 2 $*$ Precision@k $*$ Recall@k/(Precision@k + Recall@k) [61].

In recommendation performance experiment, we vary k to 5, 10, 20 to test top-k recommendation efficiency. In other experiments, we set $k = 20$, as it appears desirable from a user's perspective to expect the target among the first 20 items [39].

5.3.3 Baselines

To validate the effectiveness of DAINN, we compared our model with the following session-based recommendation methods.

- *Popular-based Method (POP): This method recommends items with the largest number of interactions by the users.*
- Item KNN: Item KNN is a simple, yet effective method, which is widely deployed in practice. In this method, two item are considered similar if they co-occur frequently in different sessions. In our situation, we recommend items based on cosine similarity between different sessions.
- BPR-MF[15]: It is one of the commonly used matrix factorization methods, but cannot directly apply to session-based recommendations for the new session do not have feature vectors precomputed. Instead, we use the average value of item feature vectors that had occurred in the session before the predicting point, as the user feature vector [89].
- Hierarchical RNN (HRNN)[16]: Proposed by Quadrana et al. [45], the model is a personalized RNN model with cross-session information transfer in a seamless way. HRNN relays end evolveds latent hidden states of the RNNs across user sessions.
- Neural Attentive Recommendation Machine (NARM)[17]: The model incorporates an itemlevel attention mechanism into RNN for capturing both the user's sequential behavior and main purpose in the current session [43].

5.3.4 Parameter Settings

For user long-term interest modelling, we resort to the standard perplexity [5] and choose the topic number that leads to small perplexity and fast convergence. Therefore, we obtain the topic numbers $N_u^A = 70, N_u^L = 100$ and $N_u^W 100$ for Adressa, Last.fm and Weibo-Net-Tweet, respectively. The embedding dimension D_e is set to 300. For time decay rate λ, we set it to 0.2 for Adressa dataset, but a relatively slow decay $\lambda = 0.1$ for the other two datasets. The sliding window size w is set as 150 in our experiments for the simplicity, which means we adopt 150 neighbourhood events of other users in public behavior mining procedure. In model training phase, the trad-off parameter λ_1 is set to 0.4 by grid-search over $\{0.2, 0.4, 0.6, 0.8\}$ and cross validation. The coefficient λ_2 of weight decayterm is set to $1e - 4$. We leverage stochastic gradient descent to optimize our model, and the learning rate is set to 0.001. Besides, we adopt one GRU layer with 100 hidden units in our model.

[15] https://github.com/bbc/theano-bpr

[16] https://github.com/mquad/hgru4rec

[17] https://github.com/lijingsdu/sessionRec_NARM

The model is defined and trained in Theano.

5.4 Experiments

In this section, we evaluate the performances of our proposed models with four experiments. In the first experiment, we compare our DAINN model with state-of-the-art methods. The second experiment evaluates the influence of session length on the recommendation performance. In the third experiment, we evaluate the significance of different components of our model on recommendation performance. The last experiment explores the effectiveness of different recommendation algorithms in addressing cold-start issues.

5.4.1 Comparison against baselines

In this section, we present the experimental results of all baselines and our DAINN model with well-tuned parameters. To test the effectiveness of our proposed attention mechanism with various side information, we concatenate the representations of day of week, hour of day and location, with high-dimensional representation learned from GRU in NARM model, to learn the final attention score α. The NARM with spatio-temporal enrichment is denoted as NARM+E. As shown in Table 5.3, it can be observed that our method can achieve superior performance than all the other baselines on all datasets. We also can obtain other observations: (1) the recommendation performance increase with the increasing number of k on all baselines. (2) all models perform better on Weibo and Adressa. It is mainly because the sparsity and unbalance characteristics appearing in Adressa dataset, and many meaningless words and noise can be found in Weibo dataset. (3) Among these competitors, Popular-based method get extremely bad results. The reason is that the method only provide user with random (if there are ties with items) or the same popular items, which fails to satisfy users' personalized demands. (4) The deep learning models including HRNN,NARM, NARM+E and DAINN, consistently outperform the other models on both datasets in terms of all evaluation metrics, despite the fact that Item KNN is a very competitive baseline. It is because the latter one cannot generalize the learned representations to new data. (5) Although HRNN considers the dynamics of user behaviors and achieves favorable results on datasets, it still does not adopt the contextual and semantic features and users' long-term interest. (6) NARM and NARM + E performs better than HRNN which can be attributed to the attentionmechanism. However, compared with

Table 5.3: Performance comparison of DAINN with baseline over three datasets.

Method	POP	Item KNN	BPR-MF	HRNN	NARM	NARM + E	DAINN
Adressa							
Recall@5	0.011	0.118	0.112	0.141	0.197	0.213	0.253
Precision@5	0.009	0.108	0.101	0.120	0.195	0.211	0.232
F1-score	0.010	0.113	0.106	0.130	0.196	0.212	0.242
MRR@5	0.002	0.039	0.035	0.051	0.066	0.073	0.112
Recall@10	0.014	0.141	0.133	0.164	0.228	0.241	0.269
Precision@10	0.012	0.121	0.112	0.138	0.212	0.226	0.251
F1-score	0.013	0.131	0.122	0.149	0.220	0.233	0.260
MRR@10	0.003	0.051	0.042	0.064	0.078	0.086	0.125
Recall@20	0.021	0.162	0.156	0.187	0.253	0.265	0.287
Precision@20	0.016	0.134	0.127	0.156	0.232	0.249	0.268
F1-score	0.018	0.147	0.140	0.170	0.242	0.257	0.277
MRR@20	0.005	0.063	0.054	0.078	0.093	0.101	0.136
Last.fm							
Recall@5	0.106	0.201	0.189	0.231	0.303	0.316	0.347
Precision@5	0.089	0.194	0.187	0.221	0.277	0.285	0.334
F1-score	0.097	0.197	0.188	0.226	0.289	0.299	0.340
MRR@5	0.081	0.159	0.137	0.174	0.182	0.180*	0.228
Recall@10	0.118	0.219	0.203	0.256	0.320	0.317*	0.371
Precision@10	0.103	0.216	0.211	0.246	0.295	0.291*	0.359
F1-score	0.110	0.217	0.207	0.251	0.307	0.303*	0.365
MRR@10	0.095	0.173	0.151	0.183	0.194	0.192*	0.242
Recall@20	0.126	0.241	0.225	0.273	0.342	0.348	0.389
Precision@20	0.119	0.231	0.228	0.263	0.317	0.325	0.376
F1-score	0.122	0.236	0.226	0.268	0.329	0.336	0.382
MRR@20	0.107	0.186	0.162	0.194	0.201	0.208	0.253
Weibo-Net-Tweet							
Recall@5	0.087	0.211	0.195	0.216	0.289	0.283*	0.334
Precision@5	0.037	0.151	0.128	0.183	0.249	0.246*	0.304
F1-score	0.052	0.176	0.155	0.198	0.268	0.263*	0.318
MRR@5	0.062	0.141	0.113	0.152	0.163	0.160*	0.181
Recall@10	0.096	0.218	0.207	0.223	0.301	0.308	0.349
Precision@10	0.053	0.168	0.146	0.197	0.265	0.273	0.320
F1-score	0.068	0.190	0.171	0.209	0.282	0.289	0.334
MRR@10	0.075	0.149	0.127	0.158	0.176	0.183	0.194
Recall@20	0.103	0.224	0.215	0.236	0.314	0.326	0.357
Precision@20	0.071	0.184	0.163	0.215	0.282	0.295	0.336
F1-score	0.084	0.202	0.185	0.225	0.297	0.310	0.346
MRR@20	0.083	0.156	0.141	0.169	0.184	0.191	0.202

The numbers with asterisk represent the results of NARM + E are worse than NARM in our experiments

NARM model, NARM+E only improves little in Adressa dataset and appears unstable performance in Last.fm andWeibo dataset.We argue that this is because our attention mechanism performs better on experimental datasets and side information need to be integrated and modelled properly. Otherwise, it may cause counterproductive effect.Besides, similar asHRNN,NARMandNARM+E do not consider users' historical records and relationship between sessions with other users. As a result, our proposed method outperforms NARM + E by (2.0%, 4.6%, 3.6%) with F1-score (k = 20) and (3.5%, 4.5%, 1.1%) with MRR@20 on Adressa, Last.fm andWeibo-Net-Tweet datasets, which also validates the effectiveness of the joint user long-term interest embedding and neighbourhood session embedding.

5.4.2 Evaluation on different session lengths

In this section, we study the impact of different session lengths on the recommendation performance. The attention scheme in our framework is based on the assumption that when a user browsing online, his/her click/play/post behavior frequently revolves his/her main purpose in the current session. However, if the user only clicks a few items, we can hardly capture the user's main purpose. Besides, we also need to make sure that the longer length of session, the better recommendation performance we can achieve for our DAINN model.

The experimental results on Adressa dataset are shown in Figure 5.4. We can learn that: (1) In general, the recommendation performance of our model increases with the increasing number of session length, which indicates that DAINN model can capture user's main purpose more accurately on relatively long sessions. In other words, it needs a process to learn from the existing sequential behaviour features to make a better prediction. (2) However, when the session is too long, namely more than 18 in our experiments, the recommendation accuracy will decline. The reason we consider is that long session will bring more noise so that it increase the uncertainty and randomness of the user's behavior in the current session, which is to say that the user is very likely to click some items aimlessly, and thus it is hard for DAINN to capture the user's main purpose in the current session in this case.

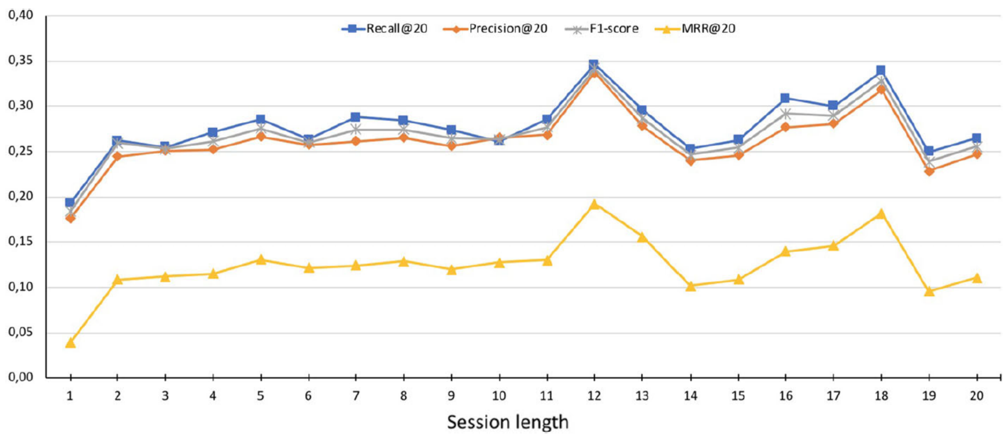


Figure 5.4: The performance among different session lengths on Adressa dataset.

5.4.3 Model component analysis

From Figure 5.2, we can see that our method has four essential components including sessionbased public behavior mining in part (a), item semantic embedding in part (b), user long-term interest modelling in part (c) and attention network. To verify the contribution of each component, we implement four variants of our approach: DAINN-S, DAINN-A, DAINN-U represent DAINN model without session-based public behavior mining, attention network and user long-term interest modelling in our framework. We cannot abandon item semantic embedding part since it is the base for other part.

The comparison results are shown in Figure 5.5. The results showall the components contribute more or less to the final recommendation performance. Several observation can be found: (1) DAINN-A method results in inferior performance with, for instance F1-Score 19.6%, 28.3% and 24.7% of Adressa, Last.fm and Weibo datasets respectively. The results show that the attention scheme can capture the users' main purpose within and across session, and it can adaptively and smartly takes previous knowledge into consideration to capture the users' preference. Besides, the successful utilization of three key factors within attention network brings advantages when recommending items. (2) User long-term interest modelling is also essential in our framework. Compared with DAINN-U model, DAINN model get, for instance 3.6% promotion in terms of F1-score on Adressa dataset. The results demonstrate that users long-term interest modelling can capture users' broad interests and improves the recommendation performance significantly. (3) Our model also can benefit from sessionbased public behavior mining for DAINN gets a promotion from DAINN-S.

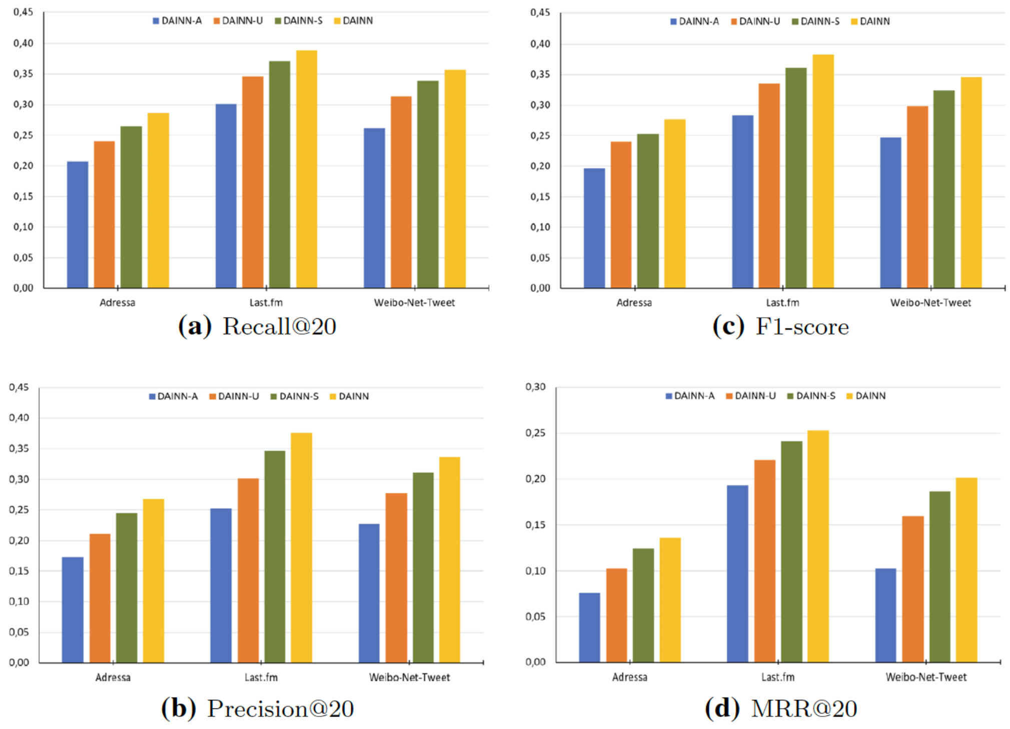
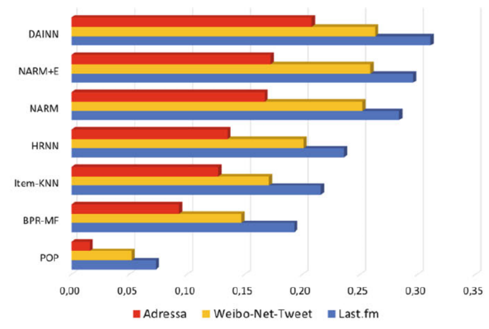


Figure 5.5: Recommendation performance with different components of DAINN model.

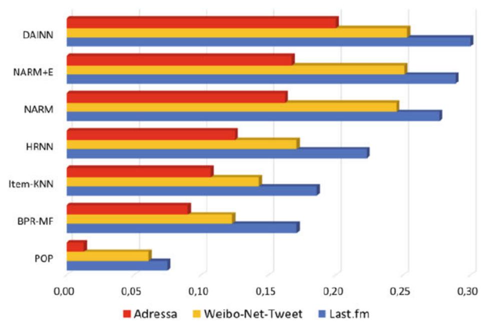
5.4.4 Cold-start problem

Additionally, we also conducted experiments to study the effectiveness of different recommendation algorithms in addressing cold-start issues on three kinds of datasets. As preprocessing, we removed users who have less than 5 events during sessions and more than 2 sessions in training sets. Beside, we also filtered users in testing set who were not contained in training sets. Then, we randomly select 10,000 users among them to conduct the experiments.

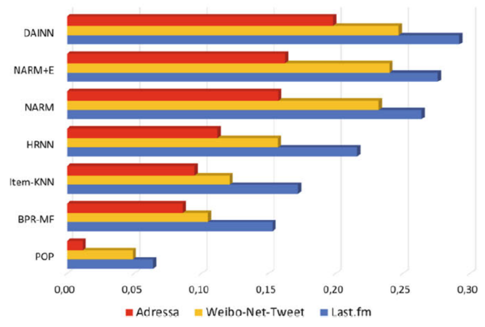
The experimental results are shown in Figure 5.6, from which we have the following observations: (1) our proposed DAINN model and NARM, NARM + E model still performs better consistently than other methods in recommending cold-start cases, which verifies the effectiveness of attention scheme used in session-based recommendation scenario; (2) by comparing the recommendation results in Table 3, the evaluation metrics of nearly all recommendation algorithms decreases, to different degrees, except Popular-based method. It is because the latter two methods consider less historical events than other methods. The recommendation performance of our DAINN model deteriorates more quickly than NARM and NARM + E, which is because the lack of user long-term interests makes DAINN consider only short-term interests of users when recommending items. (3) The recommendation performance of Item-KNN and BPR-MF drop drastically, which from another aspect proves the ineffectiveness of collaborative filtering methods in handling cold-start cases. (4) OurDAINN model still performs better than NARM and NARM + E model especially on Adressa and Last.fm datasets, since DAINN also considers different factors in attention scheme properly: hour of day, day of week and location. Besides, neighbourhood session information also brings positive influence for recommendation tasks.



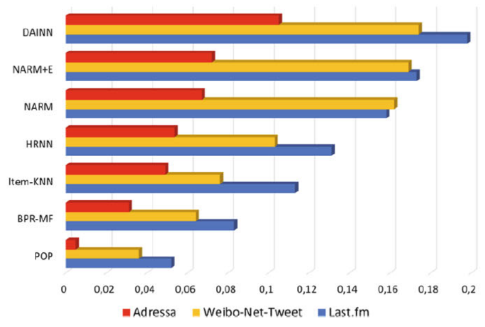
(a) Recall@20



(c) F1-score



(b) Precision@20



(d) MRR@20

Figure 5.6: Recommendation for cold-start users.

5.5 Conclusion

In this chapter, a novel dynamic attention-integrated neural network (DAINN) is proposed to address the problem of personalized session-based recommendation. In order to capture users' interests, we consider item semantic embedding, user long-term interest modelling and session-based public behavior mining in a unified framework, which can be trained end-to-end. By incorporating an attention mechanism into DAINN, our proposed approach can deal with the diverse variance of users' clicking behavior and capture the users' main purpose in the current session. DAINN can effectively learn users' real-time preferences and conduct personalized recommendations.

Though a growing number of publications on session-based recommendation focus on deep learning-based methods, unlike existing studies, our approach can simultaneously model the user's historical interest, the main purpose of the current session and public attention, Which to the best of our knowledge, is not considered by existing researches.

Recently, the attention model has been used in recommender systems and achieves better performance in many recommendation scenarios. Compared with existing related works, the novelty lies in the idea of incorporating users' spatio-temporal reading characteristics into the dynamic attention model for the session-based news recommendation. As far as we know this is the first attempt to capture the diverse variance of users' clicking behavior with a dynamic hybrid attention scheme.

Evaluation on three different real-world datasets demonstrated the effectiveness of the proposed approach. For baseline methods, we select from both deep learning-based methods and attention integrated approaches to verify the effectiveness of our model. Especially, DAINN outperforms NARM with similar attributes integrated setting which verifies the significance of rational and strategic integration of various attributes.

Part III

Exploring Graph Structured Data for User Modelling

Chapter 6

Taxonomic Analyses on Graph Learning-based Recommender Systems

Recent advances in graph-based learning approaches have demonstrated their effectiveness in modelling users' preferences and items' characteristics for Recommender Systems (RSs). Most of the data in RSs can be organized into graphs where various objects (e.g., users, items, and attributes) are explicitly or implicitly connected and influence each other via various relations. Such a graph-based organization brings benefits to exploiting potential properties in graph learning (e.g., random walk and network embedding) techniques to enrich the representations of the user and item nodes, which is an essential factor for successful recommendations. In this chapter, we provide a comprehensive analysis of the Graph Learning-based Recommender Systems (GLRSs). Specifically, we start from a data-driven perspective to systematically categorize various graphs in GLRSs and analyze their characteristics. Then, we discuss the state-of-the-art frameworks with a focus on the graph learning module and how they address practical recommendation challenges such as scalability, fairness, diversity, explainability and so on. Finally, we share some potential research directions in this rapidly growing area.

6.1 Introduction

In the last few decades, the rapid development of Web 2.0 and smart mobile devices has resulted in the dramatic proliferation of online unstructured data, such as news articles. They are explicitly or implicitly connected with each other and can naturally be formed into graphs representing objects and their relationships in varied domains, including e-commerce, social networks, and so on. On the one hand, the interconnection of objects shows a direct or indirect interactive relationship, which provides a more intuitive and effective way for the recommendation systems to explore the hidden relationships between the target user and the recommended items. On the other hand, the data structure of graphs breaks the independent interaction assumption by linking users or items with their associated attributes such that the recommender systems are able to capture not only the user-item interactions but also the rich underlying connections by mining item-item/user-user relations to make more accurate recommendations. Moreover, most recommendation models work as black-boxes that only provide predictive results rather than exhibiting the reasons behind a recommendation, such as collaborative signals in collaborative filtering or knowledge-aware reasoning in knowledge graph-based recommendation. Such black-box nature makes the decision-making process opaque to understand and hampers their further applications. Graph-based recommender systems provide new opportunities in improving the explainability of the recommender systems by using explicit connections between objects in graphs to reveal the recommendation results. Therefore, it is of crucial significance to fully explore the semantic connections and potential relations of the graphs to improve the performance on both the explainability and accuracy of recommendations. Graphs in this chapter can be referred to as anything with an underlying graph structure relevant to recommendation purposes.

However, there exist some problems and challenges in how recommender systems (RSs) can make full use of these data:

- Heterogeneous Objects: The unstructured data can be organized into a graph including different-typed objects and links. Modelling and abstracting such a space of information have been a challenging task encountered in RSs.
- Large-scale Volume: Real graphs, such as social networks, can easily have millions even billions of nodes and edges, which renders most traditional recommendation algorithms computationally infeasible.
- Dynamic Contents: Most real-world graphs are intrinsically dynamic with the addition/deletion of edges and nodes. Meanwhile, similar to graph structure, node attributes also change naturally such that new content patterns may emerge and outdated content patterns will fade.

Recently, graph learning (GL) has exhibited the potential to obtain knowledge embedded in different kinds of graphs. Many GL techniques, such as random walk, graph embedding and graph neural networks, have been developed to learn the complex relations modelled on graphs and achieve great improvement on recommendation performance. An emerging RS paradigm built on GL, namely Graph Learning-based Recommender Systems (GLRSs), has attracted growing attention in both research and industry communities. For example, researchers leverage random walk to propagate users' preference scores from historical item nodes and output a preference distribution over unobserved items, such as ItemRank [113] over item-item correlation graph, RecWalk [114] over the user-item bipartite graph, and TriRank [115] over the user-item-aspect tripartite graph. Moreover, various graph embedding techniques and graph neural networks have been proposed and incorporated into the representation learning of RSs, using direct or multi-hop connections within graphs to enrich the representations of the user and item nodes. These methods further improve the recommendation performance.

To date, there is only a handful of related work devoted to Graph Learning-based Recommender Systems (GLRSs) from multiple perspectives, including data-driven and technology-driven. Wang et al. [116] have analyzed several graph challenges and summarized the recommender systems utilizing graph learning techniques. Although it is the first review on GLRSs, their survey covers a limited number of references and does not go into the technical details on graph learning modules in GLRS and how they address those challenges. Furthermore, they did not give a systematic summarization on existing datasets adopted by graph-learning based recommendation researches. In this chapter, we contribute the most comprehensive overview of state-of-the-art GLRSs. We systematically analyze the benchmark datasets in GLRSs, provide detailed descriptions on representative models, make the necessary comparison, and discuss their solution to practical recommendation issues such as scalability, fairness, diversity, explainability, etc.

Contributions. This chapter aims to provide a thorough analysis of the approaches of graph-learning based recommender systems. It is intended to help both academic researchers and industrial practitioners who are interested in GLRSs and are willing to gain an in-depth understanding of how graphs can help to improve recommendation performance with the help of graph-learning technologies. To this end, the main contributions of this work are summarized as follows:

1) We propose a novel taxonomy to categorize various graphs in GLRSs and analyze their characteristics from a data-driven perspective. We further summarize the corresponding benchmark datasets based on the proposed taxonomy.

2) We conduct a comprehensive taxonomic analysis on existing graph-learning based recommendation approaches and provide a thorough literature review of state-of-the-art

researches through comparison and analysis on different types of graphs associated with model technologies for recommendation purposes. We then analyze the limitations of existing works and suggest future research directions of GLRSs such as dynamicity, interpretability, and fairness.

Organization. The rest of the chapter is organized as follows. In Section 6.2, we review our research methodology on how we collected the related papers and provide an initial analysis on datasets adopted by reference papers. In Section 6.3, we introduce the definitions of the basic concepts required to understand the graph-learning based recommendation problem, followed by a formal problem definition of graph-learning based recommendation. Section 6.4 provides a new taxonomy on graphs that are related to specific datasets. Section 6.5 provides an overview of state-of-the-art GLRSs methods. Section 6.6 discusses the current challenges and suggests future directions, followed by the conclusions in Section 6.7.

6.2 Research Methodology

6.2.1 Paper collection

To achieve a systematical structure of existing researches on graph-based recommendation, this study was performed based on a bibliographic review method in the graph-based recommendation domain. Specifically, we first conducted a comprehensive review of previously published papers concerning GLRSs by searching with the following keywords: “graph learning recommendation”, “graph based recommendation”, “graph neural network recommender system”. We also searched for references with additional keywords such as “social recommendations”, “representation learning based recommender system”, “knowledge graph recommendation”, considering that some articles do not explicitly have “graph” in their titles. Google Scholar was the primary digital library to find relevant papers while other academic search engines such as ACM Digital Library[18], IEEE Xplore[19], Springer[20], Rearch-Gate[21] and Web of Science[22], were also consulted. To ensure the quality

[18] https://dl.acm.org
[19] https://ieeexplore.ieee.org
[20] https://www.springer.com
[21] https://www.researchgate.net/
[22] https://www.webofknowledge.com

of the references, we selected papers published in prestigious and top-tier international conferences and journals including NIPS, ICML, RecSys, CIKM, ICLR, AAAI, IJCAI, WWW, WSDM, KDD, and TOIS, TKDE, UMUAI, etc. Given our interest, we manually checked the keywords, the titles, the abstracts, the conclusions, the tables and figures of the collected papers. In the end, we obtained a collection of 99 papers that meet the mentioned criteria and then are summarized in Section 6.3 and 6.4. Among these selected papers, 9 papers were related to recommender systems on the tree-based graph, 8 papers on recommendations on homogeneous graphs, 15 papers on recommendations on bipartite graphs, 67 papers on recommendations on heterogeneous graphs; 9 papers were related to traditional RSs, 23 papers concentrate on path-based methods, 14 papers on graph embedding based approaches, 51 articles were related to deep learning-based methods for GL-based recommendations. Besides, 5 papers were analyzed to assess open issues for GLRSs and potential directions for future work. Figure 6.1 gives some the statistics of the collected papers with the publication time and venue.

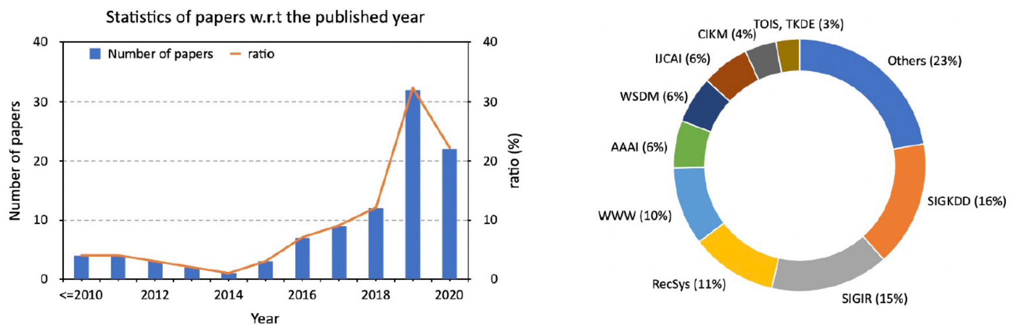


Figure 6.1: Statistics of publications related to GLRSs grouped by the publication year and venue.

6.2.2 Data analysis

Analyzing the collected papers, we made two observations on the utilized datasets: (1) They were across different domains, such as e-commercial and entertainment domains; (2) Some datasets could be used to construct multiple types of graphs for different recommendation purposes, while some were only used to construct one type of graph. For instance, we found nearly all classified graph types were utilized for the Amazon dataset, while only multi-source graph could be found for the Epinions dataset. To clearly make comparisons and show the difference of these datasets in terms of both domains and graph types, we made a detailed comparison of all datasets in Table 6.2.

6.3 Problem Formalization

In this section, we first introduce the definition of the basic concepts in graph-based recommendations and then provide a formal definition of the graph-based recommendation problem.

6.3.1 Basic definitions

The definitions related to GLRSs are as follows.

Definition 1 Graph: A graph is $G = (V, E)$, where $v \in V$ is a node and $e \in E$ is an edge. Each edge e_{ij} is a pair between vertex v_i and v_j. G is associated with a node type mapping function $f_v: V \rightarrow A$ and an edge type mapping function $f_e: E \rightarrow R$. A and R denote the set of node types and edge types respectively. Each node $v_i \in V$ belongs to one particular type, i.e., $f_v(v_i) \in A$. Similarly, for $e_{ij} \in E$, $f_e(e_{ij}) \in R$. When a graph has $e_{ij} \not\equiv e_{ji}$ and $f_e(e_{ij}) \not\equiv f_e(e_{ji})$, it is a *directed graph*. Otherwise, the graph is *undirected*.

Definition 2 Network Schema: The network schema, denoted as $T_G = (A, R)$, is a meta template for a heterogeneous network $G = (V, E)$ with the node type mapping f_v and the edge mapping f_e, which is a directed graph defined over node type set A with edges as relations from R.

Definition 3 Homogeneous Graph: A homogeneous graph $G_{homo} = (V, E)$, is a graph in which $|A| = |R| = 1$. This is to say that all nodes in G_{homo} belong to a single type and all edge to one single type.

Definition 4 Tree Graph: A tree graph $G_{tree} = (V, E)$, is a graph in which all nodes are connected with each other and there is no cycles in G_{tree}. The *leaf* node in a tree graph has degree 1, where *degree* of a vertex v, denoted as $d(v)$, is defined as the number of vertices that are adjacent to v.

Definition 5 Heterogeneous Graph: A heterogeneous graph $G_{heter} = (V, E)$ can be defined as a graph in which $A > 1$ and/or $R > 1$.

Definition 6 Bipartite Graph: A bipartite graph $G_{bipar} = (V, E)$ is a graph in which nodes are partitioned in two sets A_1 and A_2 where $A = A_1 \cup A_2$ and $A_1 \cap A_2 = \emptyset$. Each edge $e_{ij} \in E$ of G_{bipar} connects a node v_i with a node v_j where $f_v(v_i) \in A_1, f_v(v_j) \in A_2$

and $v_i, v_j \in V$. A bipartite graph is a special type of heterogeneous graph.

Definition 7 Knowledge Graph: A knowledge graph $G_{know} = (V, E)$ is a directed graph whose nodes are entities and edges are *subject-property-object* triple facts. Each edge of the form *(head entity, relation, tail entity),* denoted as $< h, r, t >$, indicates a relationship of rr from entity h to entity t. $h, t \in V$ are entities and $r \in E$ is the relation. Entities and relations in a knowledge graph are usually of different types, such that $A \cap R = \emptyset$. Knowledge graphs can be viewed as another type of heterogeneous graphs.

The network schema of a heterogeneous graph specifies type constrains on graph objects/nodes and relationships of links/edges between the objects/nodes. The constraints make the heterogeneous graph semi-structured data, guiding the exploration of the semantics of the graph.

6.3.2 Problem definition

Given a user set U and an item set I, for each user $u \in U$ who has interacted with an item set $I_u^+ \in I$, the recommendation problem can generally be seen as a mapping function $Y = argmax_u\ f(U, I)$generating the corresponding recommendation results from I_u^- that are of interests to the user.

However, there is no formal definition of GLRSs to date due to different implementations of various models on different datasets with specific characteristics. Graphs in GLRSs can be built upon user-item interactions as well as other auxiliary information. For instance, considering the graph $G = (V, E)$, where nodes in V can represent e.g. users, items and other named entities, while edges in E can represent e.g. purchase, clicks, social relations as well as other relationships among entities. In this chapter, we formulate the GLRS problem in a general perspective using U, I and G as inputs to generate the corresponding recommendation results Y by modelling graph properties, as well as other user/item characteristics out of graphs:

$$Y = argmax_u f(U, I, G) \tag{1}$$

where G can be of different types, e.g. homogeneous, heterogeneous, bipartite, tree-based etc, based on specific recommendation scenarios, while Y can be of different forms, e.g. rating scores, possible links, classifications, or ranked lists.

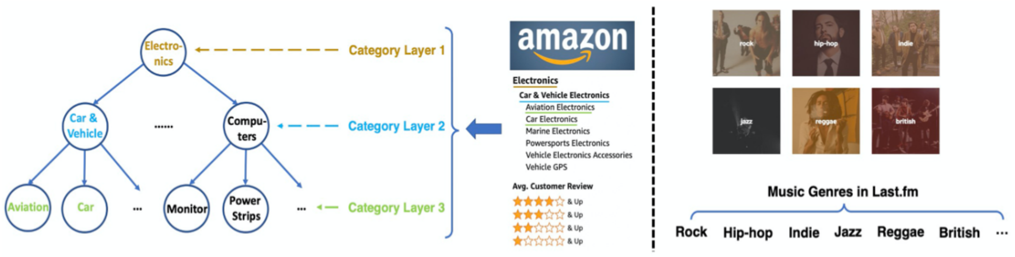


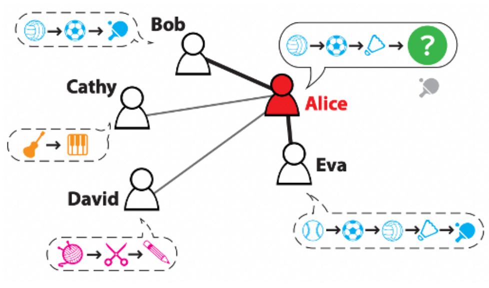
Figure 6.2: Tree-based graph illustration.

6.4 Data-driven graph taxonomy

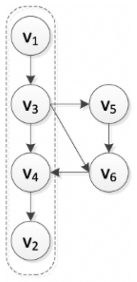
Generally, most of the objects in GLRSs, e.g., users, items, attributes, and contexts, are explicitly or implicitly connected with and influence each other through various relations, e.g., social friendship, category hierarchies and user-to-item relationships. These relations essentially result in natural graphs and contribute significantly to the performance of GLRSs. To achieve a systematic understanding, we propose a new taxonomy to categorize the graphs by the presence of their intrinsic data characteristics, including tree-based graphs, homogeneous graphs, bipartite graphs and heterogeneous graphs. The taxonomy is shown in Table 6.3 (Appendix B).

6.4.1 Tree-based graphs

A tree-based graph where the items are organized with a hierarchy by a certain attribute of them (e.g., the category), is a natural yet powerful structure for human knowledge. It provides a machine- and human-readable description of a set of items and their parallel or hierarchical relationships like *affiliatedTo*, *subClass* and *isAPartOf* relations. Such hierarchy relations between items have been widely studied and proven to be effective in generating high-quality recommendations [117-123]. Typical domains of tree-based graphs in GLRSs consist of online products (e.g., the Amazon web store [124]), foods (e.g., Gowalla [125]), movies (e.g., IMDB) and music (e.g., Last.fm). Figure 6.2 illustrates an example of tree-based graphs in Amazon and Last.fm to organize electronics or music by categories/genres. If a user buys a Monitor, she may possibly prefer Power Strips to match her Monitor instead of Aviation Electronics. This is due to both Monitor and Power Strips belonging to a higher layer category – Computers according to their inherit electrical characteristics. If a user prefers one song under a certain genre, she is more likely to favor



(a) Attributed User Friend Network [127]



(b) Non-attributed Item Co-occurrence Graph [47]

Figure 6.3: Non-/attributed homogeneous graph illustration.

other songs under this genre. By considering the *affiliatedTo* relations among items in tree-based graphs, recommendations can be generated in a more accurate and diverse manner.

6.4.2 Homogeneous graphs

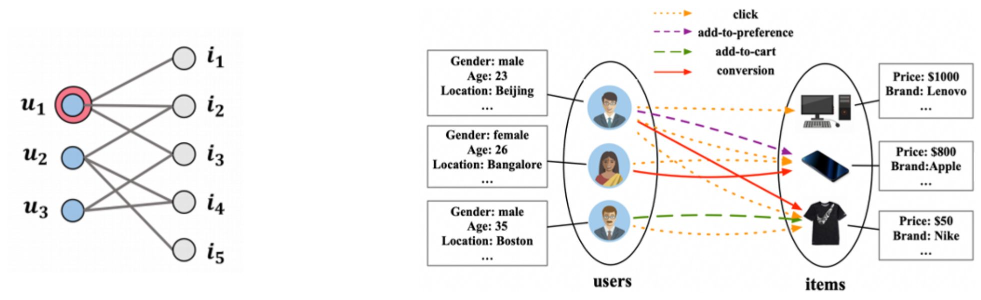
A homogeneous graph is a graph with a single type of objects and links in GLRSs. Typical examples are user graphs in social networks, which include online or offline social relations between users. Two connected users in a user graph usually share similar preferences and influence each other by recommending items. Therefore, it is necessary to take user relations into account when making recommendations, especially for cold-start users. Besides, the co-occurrence of items in a user behaviour sequence connects all the items together and thus results in a homogeneous item graph. The co-occurrence relations in item graphs not only reflect certain latent relations between items, but also reveal some behaviour patterns of users. It has been proven that the fusion of co-occurrence relations between items can yield significant performance enhancements [126]. The nodes of these homogeneous graphs are without attribute information, which are referred to as *non-attributed homogeneous graphs*. In practice, many real-world networks usually have attributes with their nodes that are also important for making sense of modelling network topological as well as contextual information for recommendation purpose. Such networks with node attributes and a single type of nodes as well as edges are named *attributed homogeneous graphs*[23]. An example can be found in a friend network where edges

[23] Some articles also categorize graphs into directed and undirected graphs. In our point of view, the

represent friendship (e.g. follow, like) between two users, and nodes represent users with attributes e.g. demographic information, or a sequence of items the user interacted with [127]. In such a case, both social influence and user attributes can help to learn user preferences and thus affect the recommendation performance. Figure 6.3 illustrates examples of non-/attributed homogeneous graphs to help resolve the cold start issue of recommender systems.

6.4.3 Bipartite graphs

As an extension of network theory, the bipartite graph has attracted significant attention in areas like social network analysis [128]. It divides network nodes into two types and edges exist only between different types of nodes. User-item interactions are basic requirements for recommender systems and can be naturally considered a bipartite graph, where the nodes represent users and items, and user nodes are linked with those interacted item nodes. The edges of the bipartite graph can either be a single type or multiple types of interactions, e.g. click, like, purchase or view. Considering this multi-typed property of nodes and edges, the bipartite graph can be viewed as a special case of the heterogeneous graph. Figure 6.4 shows examples of two kinds of bipartite graphs in the GLRS scenario.



u_1
u_2
u_3
i_1
i_2
i_3
i_4
i_5
click
add-to-preference
add-to-cart
conversion
Gender: male Age: 23 Location: Beijing ...
Gender: female Age: 26 Location: Bangalore ...
Gender: male Age: 35 Location: Boston ...
Price: $1000 Brand: Lenovo ...
Price: $800 Brand:Apple ...
Price: $50 Brand: Nike ...
users
items

(a) User-Item Interaction Network with a Single Type of Edges [213]

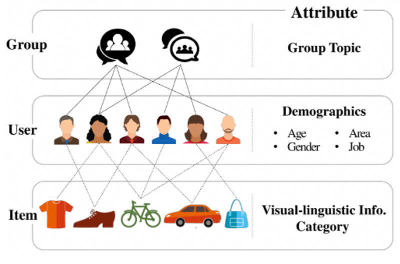
(b) User-Item Interaction Network with Different Types of Edges [193]

Figure 6.4: Bipartite graph illustration.

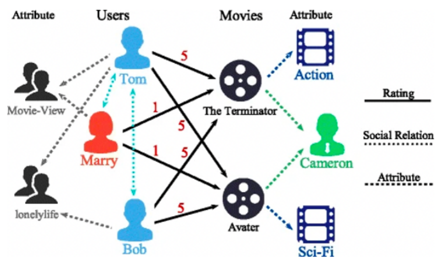
In addition to user-item interactions, auxiliary information of user/item can also be constructed as a bipartite graph. For instance, users and their correlated activities performed

undirected graph can be readily converted into a directed graph by replacing each edge with two oppositely directed edges. Thus in this chapter, without loss of generality, we assume that all graphs are directed graphs.

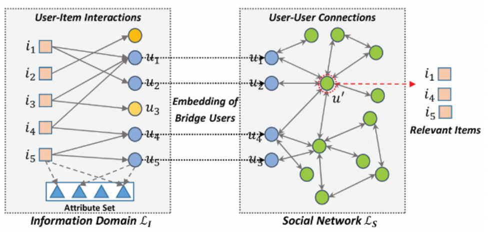
in a given location are considered as two types of nodes which can be formed as a location-



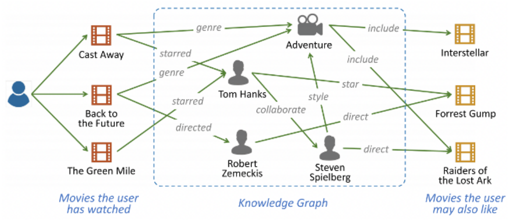
(a) Attributed HIN [286]



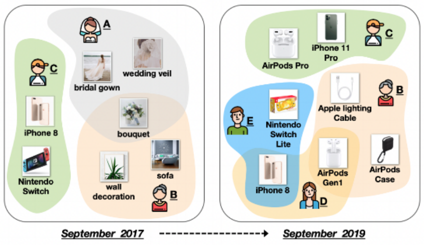
(b) Non-attributed HIN [166]


User-Item Interactions
User-User Connections
Embedding of Bridge Users
Relevant Items
Attribute Set
Information Domain $\mathcal{L}_I$
Social Network $\mathcal{L}_S$

(c) Multi-source HIN [280]



(d) Knowledge Graph [261]



(e) Sequential Hypergraphs [219]

Figure 6.5: Heterogeneous graph illustration.

activity graph. Items and their attributes (e.g., pin-boards for Pinterest dataset [129]) can also be seen as two types of nodes in forming an item-entity bipartite graph.

6.4.4 Heterogeneous graphs

Different from widely used homogeneous graphs which include only same-typed objects or links, heterogeneous graphs (also referred to as heterogeneous information networks (HINs)) consider multi-typed interacting components that fuse more information and contain richer semantics in nodes and links [130]. The heterogeneous graph can be converted into a homogeneous graph through network projection or ignoring graph heterogeneity, while it will make significant information loss [131]. Although the heterogeneous graph is ubiquitous in the real world, it is non-trivial to model the heterogeneity of the links and objects with the large-scale properties from both topological and contextual perspectives, let alone the dynamic properties for some evolving networks. The recent development of artificial intelligence sheds light on the possibility of processing more complicated networks than before, and thus the study of heterogeneous networks ushered in a breakthrough. Heterogeneous graphs can further be categorized as follows:

Attributed/Non-attributed HINs. To enrich its information content, objects in a HIN are typically associated with diverse attributes. For instance, a "user" object on Facebook is related with demographic attributes like age, gender, workplace, school, while a "photo" object has contextual attributes like date/time, tag, album. Such HIN can be categorized as Attributed HIN or AHIN [132]. Otherwise heterogeneous graphs that are not affiliated with additional attributes are categorized as Non-attributed HIN or NAHIN for short. Figure 6.5(a) is an example of AHIN which contains multiple types of objects including group, user, item, that are associated with attributes of group topic, demographics (e.g. age, area, gender and job), visual information. Many recent studies regard node attributes as a new kind of nodes that convert an AHIN to a NAHIN, as shown in Figure 6.5(b).

Multi-source graphs. To address the data sparsity and cold start problems suffered by recommender systems, a great number of recommendation algorithms have proposed to leverage side information of users or items from multiple sources. For instance, user profiles across different networks connected through anchor links (e.g. the link which connects the same entity from different platforms is call an anchor link), item profiles from different communities, user social relationships through information sharing platforms, are leveraged to improve both recommendation accuracy and diversify recommendation output. Accordingly, multiple heterogeneous subgraphs are built upon various sources which are then jointly learned for recommendation tasks. Apart from user-item interactions which provide the basic but crucial information for recommendation strategies, user social networks provide auxiliary information for modelling user preferences through discovering social relationships and other users with similar interest patterns; item auxiliary information and co-occurrence in user interaction records provide more comprehensive and detailed knowledge revealing fine-grained user behaviour. Figure 6.5(c) illustrate a multi-source graph with social relations and user-item interaction information. Though it is generally considered that the more information we can obtain for user interest analysis and modelling learning process, the more diversified and accurate recommendation results as well as better recommendation experience the user will get, we still face challenges from multiple perspectives. For instance, how to apply anchor links to provide recommendations on one or more platforms with available data sources while reducing noisy and inconsistent information from these multiple data sources. Besides, different side information of users/items from multiple data sources can come in various forms, e.g. multi-modalities. Then how to integrate such multi-source multi-model data into one recommendation framework in a mutually-consistent fashion is still an open research problem.

Table 6.1: A collection of commonly used knowledge graphs.

	Name	**Domain Type**	**Main Knowledge Source**
General KG	YAGO [133]	Cross-Domain	Wikipedia
	Freebase[137]	Cross-Domain	Wikipedia, NNDB, FMD, MusicBrainz
	DBpedia [134]	Cross-Domain	Wikipedia
	Satori [294]	Cross-Domain	Web Data
	CN-DBPedia [138]	Cross-Domain	Baidu Baike, Hudong Baike, Wikipedia (Chinese)
	NELL [138]	Cross-Domain	Web Data
	Wikidata [295]	Cross-Domain	Wikipedia, Freebase
	Google Knowledge Graph [296]	Cross-Domain	Web data
	Facebooks Entities Graph [297]	Cross-Domain	Wikipedia, Facebook data
	ConceptNet [140]	Cross-Domain	Web data
	MultiWordNet [141]	Cross-Domain	Princeton WordNet
	Babelfy [142]	Cross-Domain	BabelNet
	Open Multilingual Wordnet [143]	Cross-Domain	Wiktionary, Unicode Common Locale Data Repository
Domain Specific KG	Bio2RDF [144]	Biological Domain	Public bioinformatics databases, NCBIs databases
	KnowLife [145]	Biomedical Domain	Scientific literature, Web portals
	IMDB [298]	Movie Domain	Web data
	KnowIME [146]	Intelligent Manufacturing Domain	Internet, Baidu Encyclopedia, and related intelligent manufacturing websites

Knowledge graphs. Knowledge graphs (KGs) is a multi-relational graph composed of entities as nodes and relations as different types of edges as illustrated in Figure 6.5(d). Each edge of KG represents a triple of the form *(head entity, relation, tail entity)*, also called a *fact*, indicating that two entities are connected by a specific relation. Recent years have witnessed a rapid growth in KG application of recommendation, resulting in promising improvements in both recommendation accuracy and explainability due to the rich structured information that KG provides about the items. Existing KGs, e.g. Yago [133], DBPedia [134], provide auxiliary message apart from user-item interactions. The relational properties in KGs break down the independent interaction assumption by linking items with their attributes. Meanwhile, the introduction of KGs alleviates the data sparsity and cold-start issues raised in recommender systems [221,247,203,204]. However, the various types of entities and relations in KGs also pose the challenge of capturing semantically interconnected information for effective recommendations. In addition, how to reasonably and vividly provide recommendation results through KG internal reasoning and the linkage among user-item interactions, deserves more concerns. For instance, how to impartially and convincingly explain the reasoning process of the recommendation list to the target user. The resources of all KGs used for GLRSs have been collected and displayed in Table 6.1.

Hypergraphs. Hypergraphs are defined as a generalization of graphs in which the edges are arbitrary non-empty subsets of the vertex set [135]. Instead of having edges between pairs of vertices, hypergraphs have edges that connect sets of two or more vertices. Correspondingly, such edges of hypergraphs are called *hyperedge*. The motivation for introducing hypergraphs is two-fold [136]: first, the data correlations can be more complex than the pairwise relationship, which is difficult to modell with traditional graph structures; second, the data representations can be multi-model, which means that data can be connected through e.g. text information, visual information, or social connections, which is difficult to capture with the traditional graphs. Thus, hypergraph is a way to model a more general data structure. An example of hypergraph is illustrated in Figure 6.5(e), in which the hypergraph consists of three nodes with one hyperedge in September 2017, and four nodes with two hyperedges in September 2019.

6.4.5 Graph comparison

Graphs, ranging from flat tree-based structure to complex network structure, from homogeneous network to heterogeneous ones, evolve from both structural and contextual sides. As summarized in Table 6.3, though many different datasets are overlapped across various graphs and recommendation tasks, it is undeniable that there is no perfect graph type that can embrace all types of data or solve all problems of recommender system.

However, we can still have several observations. First, tree-based graphs are mainly derived from such as e-commerce datasets (e.g. Amazon and JD) in which types of commodities can be broken down by the category's level of granularity. Such kind of tree structure provides recommender systems a paradigm which can be further refined from a ``top-down" manner. Second, homogeneous graph can be formed by leveraging user social networks (user graph) or interlink of items (item graph). Session graph is kind of directed item graph with nodes of items that are clicked by the user and links of relations that are according to clicked order. Homogeneous graphs can be used to mine the relationships between a single type of nodes (e.g., users/items/sessions) to make targeted modelling for recommendations. Third, most bipartite graphs are constructed based on user-item interactions, which is the preliminary requirements for RS and can be naturally formed into a graph with user and item two groups of nodes. Heterogeneous graphs are suitable for most cases while can be subdivided into various categories according to specific scenarios. For instance, multi-source graphs, as the name suggests, are established upon data collected from multiple websites or cross-domain sources, and they are all associated with users' social connections like friend, chat, follow etc. KGs are generally leveraged in the domains of movie, music, books or news, where named entities are highly recognizable and have been witnessed, and most of which can be found in the corresponding entries in e.g. Wikipedia, DBpedia, Yago, etc. Hypergraphs have recently been introduced to represent the connections between sample groups such as user groups according to social relationships, or item groups according to user's co-purchase history, which breaks the traditional node-to-node pattern, pursuing a higher-level representation of data structure.

With its diversity, heterogeneous graphs with various aspects of information can be used to model more complex situations than other types of graphs so as to better solve the cold-start and data sparsity issues of recommender systems. However, such complexity of both graph structure and content leads to a more complicated modelling process and brings challenges that cannot be ignored. With the development of technology, the types of graphs that computers can process tend to be more and more complex and fine-grained, which suggests that RS can handle more sophisticated and multi-dimensional problems and scenarios.

6.5 Graph-based models for GLRS

In this section, we will go through various models adopted in GLRS and analyze them from the technical perspective, as shown in Figure 6.6.

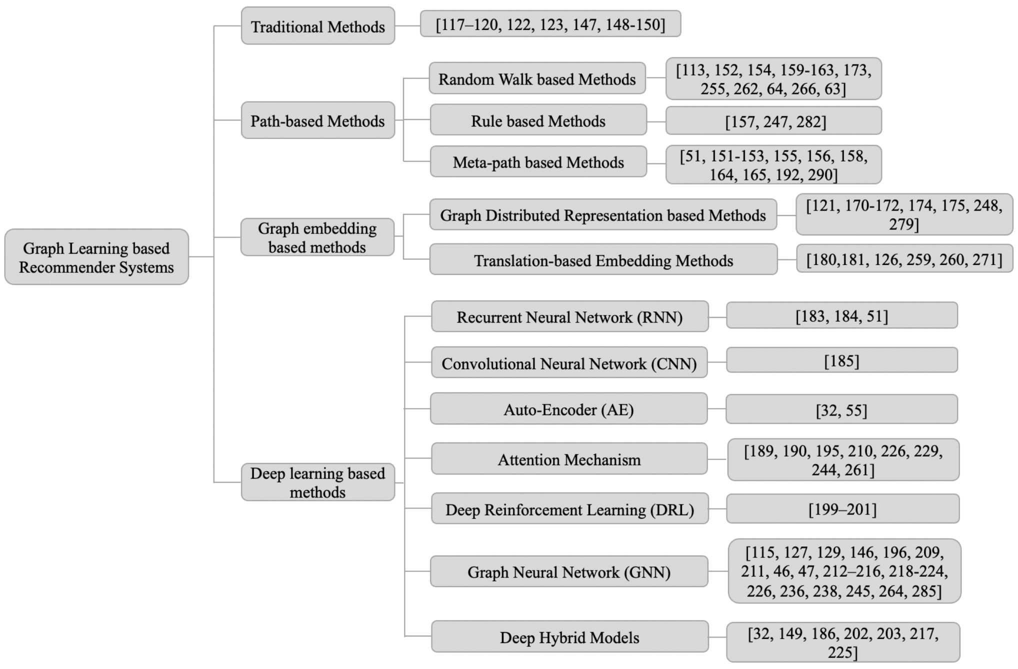


Figure 6.6: A categorization of GLRS methods from the technical perspective.

6.5.1 Traditional methodologies

Traditional recommender systems mainly aim to learn an effective prediction function based on user-item interactions and are generally classified into Collaborative filtering (CF)-based, content-based and hybrid approaches. Content-based recommendations use user/item side information in addition to user-item interactions for recommendation. However, it suffers from the cold-start issue and inefficiency because of the high time complexity. CF-based methods are one of the most popular algorithms in RSs, where Memory-based approaches, also referred to as neighborhood-based CF, are among the earliest techniques which aggregate the interests of neighbors for recommendation. Specifically, they calculate user-user or item-item similarity derived from the user-item interaction matrix for recommendation. In fact, the user-item interaction matrix can be regarded as the adjacency matrix of the bipartite graph with user and item two types of nodes, the recommendation algorithms based on which can thus be regarded as GLRS. For simplicity and clarity, this chapter only refers to those that explicitly indicate to exploit the graph as input or part of the input of the RSs. For instance, [148] calculates user-based similarity followed by k-nearest neighbors for recommendation. In addition to being convenient to implement and maintain, such memory-based CF is not applicable for large-scale data for their inefficiency in searching among large-scale database pool. Other traditional graph-based RSs build predictive models on user-item matrices to uncover the implicit user behaviour patterns. Such model-based GLRSs are originally introduced for processing large-scale datasets and meanwhile improving the recommendation performance compared with memory-based GLRSs, representatives among which can be latent factor models (LFMs). LFM has shown its effectiveness and efficiency in RSs by factorizing user-item interaction matrix into two low-rank user-specific and item-specific matrices which are then utilized to make further predictions on user's future behaviour [149]. To overcome the cold-start problem which LFMs usually suffer from, many works attempt to leverage additional information to improve the recommendation performance. For instance, [150] also model social network information namely user social relationships, as regularization term into LFM. Traditional GL-based recommenders strive to model user preferences on the basis of explicit or implicit interactions with the assumption that all user-item interactions in the historical sequences are equally important, which is not always true in real-world scenarios. The user's next behaviour depends not only on the static long-term preferences but also on the current intent to a large extent. Despite the limited improvements on recommendation performance, the investigation into how to efficiently and fully exploiting nodes as well as relations on graphs has not reached its full potential.

6.5.2 Path-based methods

For heterogeneous graphs with multiple types of nodes and relations, the basic idea for earlier recommendation strategies is to leverage path-based semantic relatedness between users and items over the constructed heterogeneous graphs [151-157]. Different from similarity-based methods based upon the item/user attributes, path-based methods especially emphasize on the essential role of links in graphs, and links between start node and end node can form a path serving recommendation purpose. In this case, the underlying relationships via network propagation show particularly importance for indirectly connected objects. Earlier studies leverage a series of predefined rules to generate path on the constructed graphs followed by different similarity measurements for ranking the

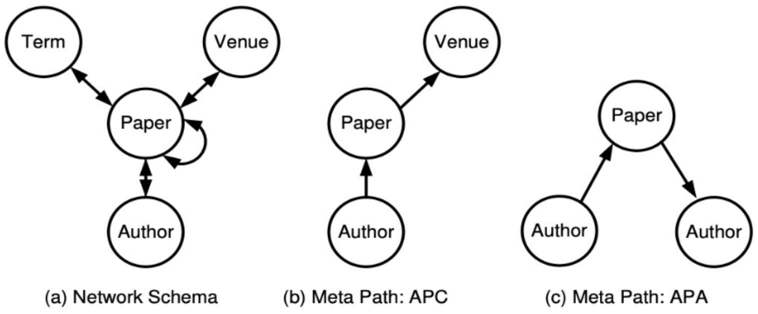


Figure 6.7: Bibliographic network schema and meta paths [164].

candidate items for recommendation [51,158]. Another graph tracing algorithm initially designed for homogeneous networks are random walk-based algorithms [159]. It starts at a node and follows outgoing edges, uniformly at random or according to predefined transition probability, until the stop condition is reached. The output paths indicate the context of connected vertices. The randomness of walks gives the ability to explore the graph with considering both the global and local structural information by walking through neighboring vertices. Random walk mechanism enables to capture complex, high-order and indirect relations between nodes for recommendations. Due to this advantages, random walk and its various variants are favored for a long period in GLRS domain generating paths in homogeneous as well as heterogeneous graphs [160-163].

To integrate different types of objects and links in heterogeneous networks, the work of [164] proposed the concept of meta-path, which are learned by many later researches [152,154]. Specifically, a meta-path [164] is a path defined on the graph of network schema $T\_G = (A, R)$, and normally denoted in the form of $A_1 \overset{R_1}{\rightarrow} A_2 \overset{R_2}{\rightarrow} A_3 \ldots \overset{R_l}{\rightarrow} A_{l+1}$, which defines a composite relation $R = R_1 \circ R_2 \circ \ldots \circ R_l$ between types A_1 and A_{l+1}, where $\rightarrow$ explicitly shows the direction of a relation from graph G, $\circ$ denotes the composition operator on relations. Figure 6.7 illustrates two examples of meta-paths 6.7(b) and 6.7(c) derived from network schema 6.7(a). When a user-specific meta-path e.g. $P = (A_1 A_2 \ldots A_l)$ has been given, several similarity measures can be defined for pair-wise nodes comparison, namely to compare $v_i \in A_1$ and $v_j \in A_l$ according to a series of paths derived based on P, referred to path instances. Random walk is one representative to generate paths instances $p \in P$ following the predefined meta-path schema [152]. To further learn the

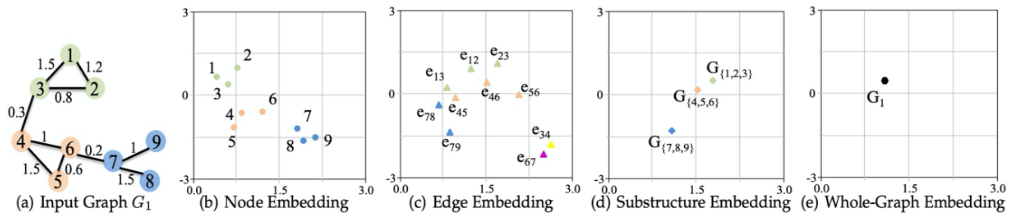


Figure 6.8: A toy example of embedding a graph into 2D space with different granularities [165]. $G_{1,2,3}$ denotes the substructure containing node v_1, v_2, v_3.

attributed HIN for better recommendations, later studies attempt to combine meta-paths with traditional latent model, e.g. FM [37], MF [166,154]. Though random-walk based similarity measures require less domain knowledge compared with meta-path based measures, the latter one turns out to be more meaningful and interpretable in most GLRSs [164].

Despite that path-based similarity strategies have achieved initial success in improving RS accuracy to some extent, challenges still exist. First, meta-path based similarities rely on explicit path reachability and would be affected by the sparse and noisy input data on recommendation performance, especially for links that are accidentally formed but do not convey meaningful information for recommendations. Second, the explicit path relatedness derived from the path-based similarity methods does not necessarily have a positive impact on recommendation performance. For instance, the work of [153] learns a linear weighting mechanism to integrate the extracted meta-paths for the subsequent recommendations, ignoring the complicated mapping mechanism of HIN. Third, path-based similarity strategies need to generate similarity scores for all candidate items at each step for every user which reduce the effectiveness of the system and thus difficult to be applied to the large-scale scenario.

6.5.3 Graph embedding-based methods

The motivation of applying graph embedding (GE) strategies lie in that they can provide an effective yet efficient way to solve the graph analytics problem. Specifically, graph embedding converts a graph into a low dimensional space in which the graph information can be retained as much as possible. By representing graph as a (or a set of) low-dimensional vector(s), graph algorithms can be applied efficiently. Figure 6.8 illustrates how graph embedding projects a graph into the vector space in different granularities, e.g. w.r.t. node/edge/substructure/whole graph [165]. Some researches differentiate graph representation learning and graph embedding by comparing the dimension of the output embedding vectors with the dimension of the inputs [165]. Graph embedding focuses on learning the low-dimensional representations, while graph representation learning doesn't require the learned representations to be low dimensional. Though they have slight differences, we do not make a special distinction in this analysis. Essentially, the two methods aim to project graph into the vector space while preserving the graph structure and capturing the connectivity information within graph to serve the recommendation task. The mapping can be defined as:

$$f: v_i \rightarrow \boldsymbol{x}_i \in R^d \tag{2}$$

where $d \gg |V|$, and $\boldsymbol{x}_i = \{x_1, x_2, \ldots, x_d\}$ is the embedded or learned vector that captures the structural properties of node v_i.

Different from path-based methods, GE-based methodologies can capture more complex and higher-order relationships among nodes and provide more efficient computation for various recommendation tasks by automatically discovering and mapping a network's structural properties into a latent space. After that, the output node representations can be leveraged to various recommendation tasks, e.g. link prediction, rating prediction. The recent advances on GE-based GLRSs have been largely influenced by the skip-gram model [167], which learns word representation in latent vector space according to its context in a sentence. Later works, e.g. DeepWalk [168], LINE [169], and Node2vec [170] are inspired by skip-gram and have achieved encouraging success in GLRSs [171-176]. For instance, the authors of [175] apply DeepWalk which aims to maximize the average logarithmic probability of all vertex context pairs in random walk sequence, to learn user and item representations on a multi-source graph to consider item structure, textual content and tag information simultaneously which are then used for collaborative filtering. In [172] the authors generate user and item representations with Node2vec, an extension of DeepWalk by leveraging a biased random walk to navigate the neighborhood nodes, on heterogeneous knowledge graph, which are then used to compute property-specific relatedness scores between users and items as the input for the learning to rank approach, resulting in optimizing top-N item recommendations.

Another research line of GE-based method adopts translation-based embedding models inspired by [177], e.g. TransE [178]. Different from DeepWalk related methods, TransE explicitly models entities and relationships among entities into the same space or different spaces while preserving certain information of the graph, which is later generalized into hyperplane translation(TransH [179]) and translation in separate entity space and relation spaces (TransR [180]). The basic idea behind TransE is that, the relationship between two entities corresponds to a translation between the embeddings of entities, that is, $\boldsymbol{h} + \boldsymbol{r} = \boldsymbol{t}$ where $\boldsymbol{h}$, $\boldsymbol{t}$, and $\boldsymbol{r}$ represents head entity, tail entity and relation between $\boldsymbol{h}$ and $\boldsymbol{t}$ in triplet (h, r, t) in a graph. Researchers attempted to adopt such translation-based models for knowledge graph embedding for recommendation [181,126,183]. For example, Wang et al. [164] assign a basic representation and various relational ones for each item via TransE, which are then combined dynamically by temporal kernel functions, providing both recommendations and explanations. Chen et al. [182] adopt TransH to embed the objects' social relationships into a shared lower-dimensional space and learn user's dynamic preference via probabilistic model. Finally, the recommendation list is generated with item-based collaborative filtering.

6.5.4 Deep learning-based methods

Deep learning (DL) has driven a remarkable revolution in recommender applications as can be seen that the number of research publications on deep learning-based recommendation methods has increased exponentially recently. To draw an overall concept of this field, we further classify the existing DL-based methods into RNN, CNN, auto-encoder, attention mechanism, reinforcement learning, graph neural network, transformer-based methods and deep hybrid models as shown in Figure 6.6.

Recurrent Neural Network (RNN). Recurrent neural networks (RNNs) and their variations e.g. gated recurrent units (GRU) [107], long and short-term memory (LSTM) [183], are proposed to model the temporal dynamics and sequential evolution of content information. The original superiority of RNNs can well capture the dependencies among items in time-sensitive user-item interaction sequences or in session-based recommendation settings. However, the limitations lie in that it is difficult to model dependencies in a longer sequence, and training is burdened with high cost, especially with the increase of sequence length. Thus, some works combine RNN with other mechanisms to balance this disadvantage of RNN. For instance, Huang et al. [184] design a memory-module to extract user's fine-grained preference on taxonomy together with a GRU layer to learn the sequential pattern. Wang et al. [57] make it different by adopting an LSTM layer to model the sequential dependencies of entities and relations on KG, generating path representations followed by a pooling operation to obtain prediction signal for user-item pairs.

Besides, using RNN for long sequence modelling also suffer from the vanishing and exploding gradient problem which is a common problem in many types of neural networks, e.g. feed-forward neural network, CNN. Despite its limitations, the RNN-based approach still dominates in sequential recommendations due to its recurrent nature that matches the natural way of our brain to read one after another in sequence mode.

Convolutional Neural Network (CNN). Convolutional neural networks (CNNs) [185] are capable of extracting local and global representations from heterogeneous data sources such as textual and visual information. It performs well in processing data with grid-like topology and capturing spatial relationships of the inputs, such as the relations between different parts of the image, or dependence between words. Besides, the parameter sharing mechanism of CNN filters applied across different parts of inputs to produce a feature map can help reduce the complexity of the model and avoid model overfitting. The typical structure of CNN consists of convolution layers, pooling layers and feed-forward full-connected layers. CNN and its variants are mostly utilized for the feature extraction process in GLRSs as well as other recommendation tasks. For instance, Wang et al. [32] leverage a multi-channel CNN to learn news embeddings w.r.t. words and entities extracted from news titles and knowledge graph.

However, CNN also suffers from gradient vanishing and exploding problems with deep architecture. Furthermore, CNN does not encode the position and orientation of the input object, and it lacks the ability to deal with the spatial variants of input data. Therefore, a large amount of training set containing various cases is required to achieve a certain performance.

Auto-Encoder. Basic auto-encoder (AE) contains an encoder which encodes (projects) high-dimensional inputs $\boldsymbol{X}$ to low dimensional hidden representations $\boldsymbol{Z}$, and a decoder which decodes (re-projects) hidden representations $\boldsymbol{Z}$ to the output $\widehat{\boldsymbol{X}}$ that looks like the original input $\boldsymbol{X}$. The objective is designed to minimize the reconstruction error, and find the most efficient and informative compact representations for the inputs. To apply AE to graph-structured data for recommendation purpose, Zhang et al. [55] first use TransE to

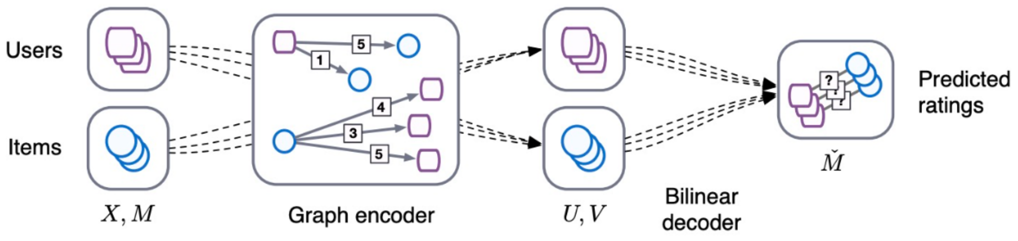
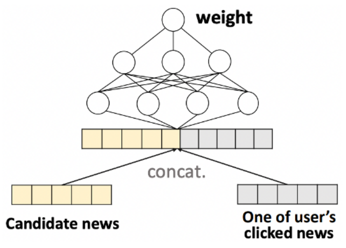


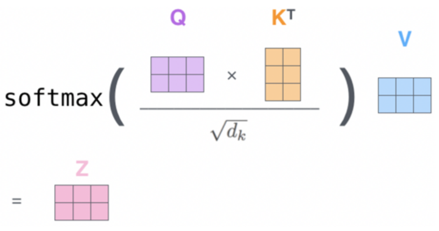
Figure 6.9: Illustration of the graph auto-encoder framework in GLRS [186].

learn graph topological information from the knowledge base. Then a stacked denoising auto-encoders and stacked convolutional auto-encoders are adopted to learn textual and visual representations of items, which are as input for collaborative filtering framework. Later, Berg et al. [186] consider recommender system as a matrix completion task, and propose to apply a graph auto-encoder to produce latent features of user and item nodes through a form of message passing on the bipartite user-item interaction graph. The learned latent user and item representations are used to reconstruct the rating links through a bilinear decoder. Generally speaking, graph auto-encoder takes node feature embedding $\boldsymbol{X}$ and adjacency matrix A as inputs, generating latent variable $\boldsymbol{Z}$ as output through encoder (inference model). To reconstruct the graph structure data, the decoder (generative model) takes $\boldsymbol{Z}$ as input and output a reconstructed adjacency matrix $\widehat{\boldsymbol{X}}$. Based on [186], Zhang et al. [187] take a step further by proposing a new stacked and reconstructed graph convolutional networks, which takes low-dimensional user and item embeddings as the input to the model and solve the cold start problem by reconstructing the masked node embeddings with a block of graph encoder-decoder in the training phase. Figure 6.9 illustrates how auto-encoder operates on graph-structured data for recommendation purpose. The problem of auto-encoder framework might be that it usually leads to a local optimum due to the back-propagation algorithm it employs [188], which may also be the common problem of most deep learning-based methods that adopt back-propagation for training procedure. Besides, the encoder-decoder architecture requires that the complete sequence of information must be captured by a single vector, which poses problems in holding on to information at the beginning of the sequence and encoding long-range dependencies.

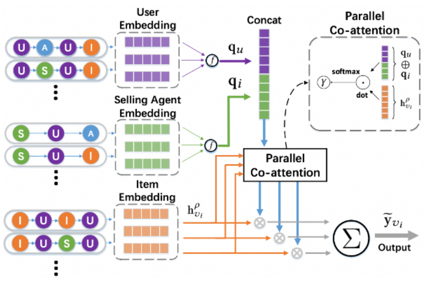
Attention mechanism. Attention mechanism [188] is motivated by human visual attention. For example, people only need to focus on specific parts of visual inputs to understand or



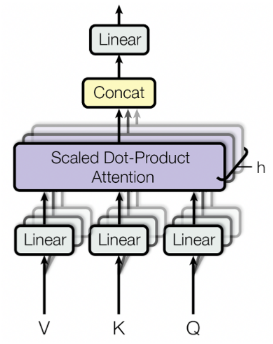
(a) An example of vanilla attention mechanism [32].



(b) The self-attention calculation in matrix form*.



(c) The architecture of the co-attentive embeddings fusing mechanism, taking item embedding as an example [196].



(d) Illustration of the multi-head attention mechanism [192].

* http://jalammar.github.io/illustrated-transformer/

Figure 6.10: Different attention mechanisms (vanilla attention, self-attention, co-attention and multi-head attention) in GLRS.

recognize them. Attention mechanism is proposed to filter out the uninformative features from raw inputs and reduce the side effects of noisy data. The effectiveness of the attention-based methods in RSs has been verified and aroused considerable attention over recent years. In attention-based GLRSs, inputs are weighted with attention scores and outputs are normally vectors that combine different importance of the inputs. Attention mechanism can be used to allow the learning process to focus on parts of a graph that are more relevant to a specific task. Generally, it can be used in conjunction with MLP, CNN, RNN and other deep learning-based architectures. Thus the heart of the attention-based methods is how to obtain and calculate the attention weights of each input parts.

There are three attention mechanisms commonly used in recent studies: (1) *vanilla attention* mechanism learns the attention scores for the input data by transforming the representations of input data via fully-connected layers, and then adopting a softmax layer to normalize the scores [190,191,32]. Han et al. [190] propose to use multi-layer MLP to learn user/item aspect-level representation based on extracted meta-paths. Then an attention mechanism is adopted to weigh the contribution of different aspect-level latent factors to user/item final representations. Wang et al. [32] combined the vanilla attention layer with CNN layer to match candidate news to each piece of clicked news and aggregate the user's history with different weights. (2) *Self-attention* mechanism [192] is proposed to gain exposure recently as it can replace RNN and CNN in the sequence learning, achieving better accuracy with lower computation complexity. It focuses on the self-matching of a sequence whereby the attention weights are calculated by the multiplication between key and query vectors transformed from the input sequence. For instance, Cen et al. [193] adopt self-attention to capture the influential factors between different edge types of the neighbors of a specific node on the multiplex heterogeneous graph. (3) *Co-attention* mechanism focuses on co-learning and co-matching of two sequences whereby the attention weights of one sequence are conditioned on the other sequence, and vice verse. Some studies prefer to classify co-attention and self-attention as one category [194,195], but for clarity, in this analysis we describe them separately. In [196], the authors design a parallel co-attention mechanism to dynamically infer the primary reasons of the user purchase decision, assigning higher attention weights to more relevant meta-paths extracted on the graph. Other studies adopt attention variations based on these three categories. For instance, in [171] the authors adapt skip-gram to a merged heterogeneous user-item interaction and use social networks followed by a multi-layer and multi-head attention [192] mechanism to learn different importance of entities. Multi-head attention makes an improvement of self-attention mechanisms to draw global dependencies between inputs and outputs by eschewing the use of recurrence in neural network and running through an attention mechanism several times in parallel. In [197] the authors adopt Sentence-BERT [198], a language model that based on multi-head attention and bidirectional training procedure, to explore the potential links between item based on reviews. The learned item representations from BERT are then used to generate item subgraph according to the cosine similarities between all items.

Figure 6.10 illustrates the abovementioned attention mechanism architectures for GLRSs. The core of the attention mechanism of focusing on the most relevant parts of the input by providing a direct path to the input helps to alleviate the bottleneck problem of the vanishing gradient and resolve the disadvantage of encoder-decoder architecture that has the problem of remembering long sequence dependencies. However, one believes that the attention mechanism adds more weight parameters to the model, which increase training time, especially for long input sequences.

Deep Reinforcement Learning (DRL). Reinforcement learning (RL) uses trial-and-error experience with an agent that learns a good behavior by modifying or acquiring new behaviors and skills incrementally. During such a learning process, the agent interacts with the environment and must make value judgements so as to select good actions over bad. Actions that get them to the target outcome are rewarded (reinforced). Deep reinforcement learning [199] goes a step further by incorporating deep neural networks to represent knowledge acquisition progress.

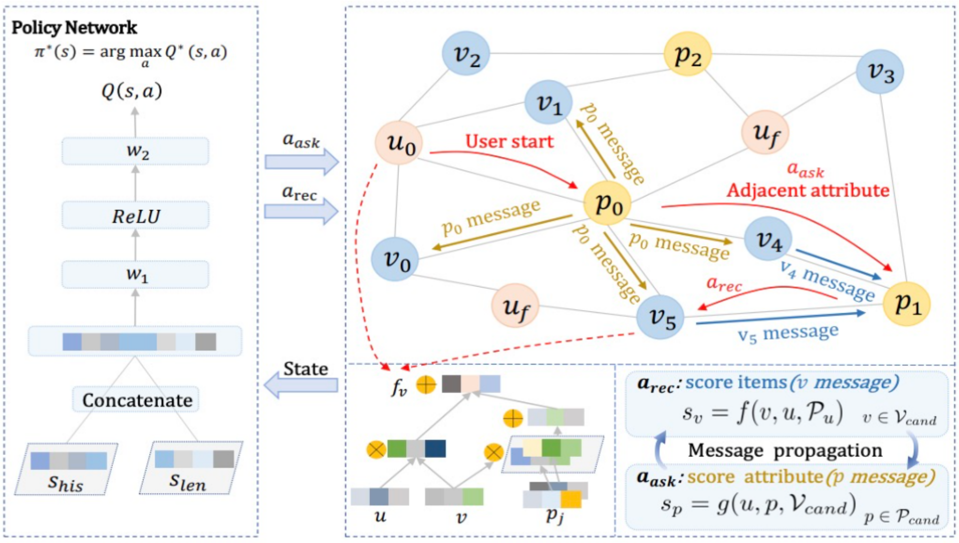
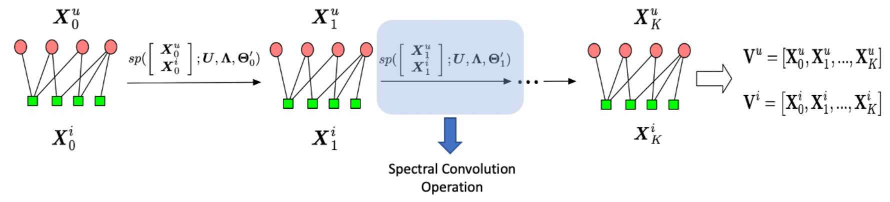

Policy Network
$\pi^*(s) = \arg\max_a Q^*(s,a)$
$Q(s,a)$
W_2
ReLU
W_1
Concatenate
s_{his}
s_{len}
a_{ask}
a_{rec}
State
u_0
User start
p_0 message
a_{ask}
Adjacent attribute
v_4 message
a_{rec}
v_5 message
f_v
u
v
p_j
a_{rec}: score items (v message)
$s_v = f(v, u, \mathcal{P}_u)$ $v \in \mathcal{V}_{cand}$
Message propagation
a_{ask}: score attribute (p message)
$s_p = g(u, p, \mathcal{V}_{cand})$ $p \in \mathcal{P}_{cand}$

Figure 6.11: Deep reinforcement learning based GLRS with knowledge graph [202].

In GLRSs, one can take path generation procedure as a decision-making process for training with RL, so that the optimal recommendation results as well as the interpretation of the results can be generated at the same time [200-203]. For instance, Xian et al. [201] propose to use RL approach where an agent starts from a given user, and learn to navigate to the potential items of interest on the knowledge graph. After that, the reasoning path history can serve as a genuine explanation for the recommendation results. Similarly, Song et al. [200] formulate the generation of user-to-item paths as a sequential decision process. Specifically, it defines the target user as the initial state and the walks on the constructed heterogeneous user-item-entity graph as actions. In the work of [204], the authors adopt KG to improve the sample efficiency as well as interactive recommendation performance by applying a deep Q-network to fit on samples from the local graph of KG rather than the whole graph. Interestingly, we can find that most DRL-based recommendation approaches utilize KG as an important medium to learn user-to-item inference. Figure 6.11 illustrates a typical example of adopting KG with DRL for recommendation purpose. It is probably due to the explicit association between the target user and items that can reveal the user's potential interests, and meanwhile, the semantic space of KG can also helpful in extracting user's preference while learning.

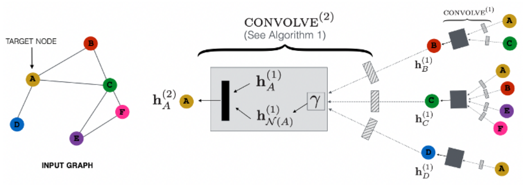
DRL-based approaches have great potential in decision-making and long-term planning in dynamic environment [205]. However, the ideal way to train a DRL model to learn the optimal recommendation policy is to train the agent online, which cannot be always satisfied. One commonly used training strategy is to make use of offline logged data directly, but it will suffer from the estimation bias problem under the real-time interaction setting [293]. Besides, similar to other deep learning-based methods, DRL-based approaches also lack of interpretability. More importantly, few appropriate platforms or resources for developing and testing DRL-based methods in academia [207].

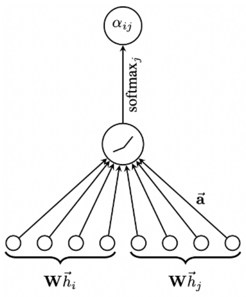
Graph Neural Networks (GNN). Graph neural network (GNN) enjoys a massive hype as recent works have witnessed a boost of performance in RSs. It motivated from CNN and graph embedding and designed specifically on graph-structured data in the non-Euclidean domain. GNN can be applied from simple homogeneous graphs to complex heterogeneous ones with attributes, and achieves improvements in recommendation results by capturing the higher-order interaction in user-item relationships through iterative propagation resulting in better user/item representations. Specifically, GNN aims to iteratively aggregate feature information from neighbors and integrate the aggregated information with the current node representation [208]. Further, it can simultaneously model the diffusion process on the graph with the RNN kernel. Following the existing work of [209], we categorize GNN as spectral [210,211,146] and non-spectral methods [46,47,212,187 ,129,196,197,122-127], as illustrated in Figure 6.12(a)(b). The spectral GNN based on spectral graph theory [208] which studies connections between combinatorial properties of a graph and the eigenvalues of matrices associated with the graph, e.g. laplacian matrix. It



X_0^u
X_0^i
$sp(\left[\begin{smallmatrix}X_0^u\\X_0^i\end{smallmatrix}\right];U,\Lambda,\Theta_0')$
X_1^u
X_1^i
$sp(\left[\begin{smallmatrix}X_1^u\\X_1^i\end{smallmatrix}\right];U,\Lambda,\Theta_1')$
X_K^u
X_K^i
$V^u = [X_0^u, X_1^u, ..., X_K^u]$
$V^i = [X_0^i, X_1^i, ..., X_K^i]$
Spectral Convolution Operation

(a) The feed-forward procedure of spectral collaborative filtering [210].





(b) Overview of GNN architecture using depth-2 convolutions [129]. (c) Illustration of the graph attention network [231].

Figure 6.12: Graph neural network based GLRS.

focuses on the connectivity of the graph rather than geometrical proximity. For instance, [147] performs spectral clustering to form user community w.r.t. user side information e.g. geographical regions, user's active timestamp, which are then considered to sort all candidate items to generate ranked list for recommendation. Zhang et al. [210] propose to use spectral convolution operation in the spectral domain of the bipartite user-item graph to alleviate the cold-start problem of RS.

The non-spectral methods mainly include aggregator and updater to learn multi-layer graph. The aggregator is responsible for collecting information from neighborhood nodes and related edges, while the updater aims to merge the propagation information around the central node and collected through aggregator. Normally, GNN is utilized to learn the representations of nodes and links of the graphs, which are then used for the following recommendation strategies, e.g. rating prediction and link prediction etc. For instance, Monti et al. [265] propose a GCN-based method for recommender systems for the first time, in which GCN is a variant of GNN and used to aggregate information from two auxiliary user-user and item-item graphs with the convolutional operation. The latent factors of users and items were updated after each aggregation step, and a combined objective function of GCN and MF is used to train the model. Ying et al. [129] propose to use GCN to generate item embeddings from both graph structure as well as item feature information with random walks for recommendations. It can be applied to very large-scale web recommenders and has been deployed in Pinterest to address a variety of real-world recommendation tasks. In [230] the authors adopt a GNN layer for modelling both local and global influence of user social relations on constructed user social graph. Graph attention networks (GATs) [231] is an enhanced version of GNN which utilize masked self-attention layers to limit the shortcomings of prior graph convolutional based methods. An attention weight $\alpha_i \in [0,1]$ is assigned to the neighborhood nodes of a target node n_t, where $\sum_{i \in N(t)} \alpha_i = 1$ and $N(t)$ denotes the set of neighboring nodes of n_t. One advantage of applying attention to graphs is to avoid the noisy part of a graph so as to increase the signal-to-noise ratio in information processing. Specifically, GAT aims to compute the attention coefficients

$$\alpha_{ij} = \frac{\exp\left(LeakyReLU(\vec{a}^T[W\vec{h}_i || W\vec{h}_j])\right)}{\sum_{k \in N_i} \exp\left(LeakyReLU(\vec{a}^T[W\vec{h}_i || W\vec{h}_k])\right)} \tag{2}$$

where a and W is the weight matrix. h_k is neighbor node embedding of node n_i whose node embedding is h_i. Figure 6.12(c) illustrates the schematic diagram of the attention operation of GAT.

Despite their verified effectiveness in the community of graph-based recommendations, they suffer from the expensive computation overhead with the exponential growth of the neighborhood size as the layers stacked up [129]. Besides, researchers empirically show that the performance of GNN quickly degenerates when the number of layers is deep owing to that the informative neighbours will diminish in large amount irrelevant neighbors [232]. To solve this, Xu et al. [196] design a relation-aware GNN with attention mechanism to prioritize neighbors based on their importance. Then a meta-path defined receptive field sampler is integrated to derive the node embeddings as well as address the rapid growth of the multiple-hop neighborhood of each node, which is followed by a co-attention mechanism for differentiating purchase motivations. In [187], the authors also point out that training GCN-based models for rating prediction faces the label leakage issue, which results in the overfitting problem and significantly degrades the final performance, which can be improved by removing the sampled edges.

Although some problems have been proposed by researchers for improvement or solutions, other challenges still exist and deserve more attention from both academia and industry.

First, compared with traditional deep neural networks which can stack tens or hundreds of layers to achieve better performance, most GNN-based architectures are usually no more than three layers. Despite the possible problem proposed by [232,233] shows empirically that stacking multiple GNN layers will result in over-smoothing, we believe that a deeper GNN can capture more significant information benefiting recommendation purpose for some larger-scale complex networks. Second, current GNN is mainly applied for static graph, but how to apply GNN for dynamic graphs with changing structures is still an open challenge.

Deep Hybrid Models. Graph In order to deal with more complicated and diverse problems, as well as process more complex graphs, many graph-based recommendation models utilize more than one deep learning techniques. The flexibility of neural blocks in deep neural networks makes it possible to combine several neural components to complement one another and form a more powerful hybrid model. The use of a variety of different DL-based components can also maximize the strengths and improve the defects of a single technology to a certain extent. In [58] the authors employ a batch of bi-directional recurrent networks [234] to learn the semantic representations of each path. Then an attention gated hidden layer is applied to learn the different importance of the derived paths between two entities followed by a pooling operation and a fully-connected layer for rating prediction. Zhang et al. [187] propose to leverage multi-link GCN as an encoder and two-layer feedforward neural network as a decoder to learn the user and item (users and items are denoted as nodes) representations, considering both node's content information as well as structural information. Some studies leverage different techniques to learn various graph features, such as node attributes and graph structure. Zhang et al. [55] construct a knowledge base to learn user potential preferences, where the item nodes associate with textual and visual features as their attributes. To model such multi-modal information, the authors first apply a network embedding (TransR) approach to extract item's structural representations by considering the heterogeneity of both nodes and relationships, followed by a stacked denoising auto-encoder and stacked convolutional auto-encoder to extract items' textual and visual representations respectively. Finally, the pair-wise ranking between items is considered to learn the CF architecture.

Deep hybrid approaches have become a trend in solving complex recommendation problems, facing the complicated and changeable network structure for modelling dynamic user preferences.

6.5.5 Discussion of Graph-based Recommendation Models

In this section, we present the main idea and the basic technical details of each class of graph-learning based recommendation approaches. From the descriptions above, we make several observations: (1) conventional graph-learning based approaches may suffer from information loss, e.g. rule-based or nearest neighbor based approaches. Some of them ignore long-term or high-order dependencies such as the latent factor model. (2) For path-based methods, they either rely on domain knowledge which may not always be applicable e.g. meta-path based similarities, and/or require explicit path reachability which may incorporate noisy, meaningless paths and thus do not always have a positive impact on recommendation results. Besides, recommendations that rely on similarity measures cannot be easily applied to large-scale networks. (3) Graph embedding based methods pave the way for more complex and high-order features among nodes and links modelling. More and more state-of-the-art GLRSs leverage GE combined with deep learning approaches such as attention network for more efficient recommendation tasks. (4) Compared with other approaches, deep learning-based methods have been a well-deserved dominance in the current GLRS research field. They can be applied to more complicated graphs with multi-type nodes and links, as well as additional attributes associated with graphs. Meanwhile, they generally obtain better performance in terms of accuracy than traditional GLRSs. Besides, deep learning-based methods are more robust to sparse data and can adapt to the varied magnitude of the input (with the help of e.g. attention mechanism). However, interpretability and efficiency are still the main concerns for most GE-based and deep learning-based GLRSs which need to be further studied in the future. (5) From Table 6.3, by relating graph types to their associated modelling technologies, we observe that tree graphs are modelled mainly by traditional methods, while other types of graphs are modelled and learned mainly through deep-learning and graph embedding based methods. Besides, attention mechanisms become especially prevalent which are adopted to nearly all types of graphs for selecting branches, filtering noisy nodes, and learning better nodes and edges representations for recommendation purposes.

6.6 Challenges and open issues in GLRS

Graph-based recommendation algorithm is an exciting and rapidly growing research area that attracts attention from both industrial and academic domains. While existing works have established a solid foundation for GLRSs research, this section reveals several challenges and promising prospective research directions. Specifically, there are two types of challenges: (1) Challenges still unsolved by graphs, which includes diversity, adaptability, explainability, and fairness issues; (2) Challenges caused by graphs and their limitations, including scalability, dynamic graph, and complex heterogeneity learning issues. We will explain separately on these issues.

6.6.1 Diversity

Diversity has been acknowledged to help increase user's satisfaction more than accuracy for state-of-the-art recommender system, while probably is detrimental to average accuracy. McNee et al. [235] observed that ignoring the diversity of the recommended list leads to sub-optimal performance. Such a phenomenon seems more obvious on active users since they exhibit some ``tendency" from their historical interaction records. However, there is never a shortage of researches on improving the accuracy of recommendation, but only a handful of works focus on optimizing the diversity of recommendations. Ziegler et al. [236] present a topic diversification method to balance and diversify recommendation lists to reflect the user's complete spectrum of interests. Later, Sun et al. [237] design a new training framework based on bayesian GNN by incorporating a random graph generative model based on node-copying [238]. The proposed framework is based on the observation that GNN-based models for recommendation usually aggregating neighborhood nodes to learn the node embeddings may lead to recommending similar items to the previous items interacted by the user. Isufi et al. [239] point out that the nearest neighbor graph connects entities based on their similarities and is responsible for improving accuracy, while the furthest neighbor graph connects entities based on their dissimilarities and is responsible for diversifying recommendations. Based on this observation, they develop a model that learns joint convolutional representations from a nearest neighbor and a furthest neighbor graph to establish an accuracy-diversity trade-off for recommendation purpose. Since state-of-the-art research has reached the point where going beyond pure accuracy and the consideration of real user experience becomes indispensable for further advances, we believe that studies on fairness will become an inevitable and essential research direction in the community of GLRSs.

6.6.2 Adaptability

Real networks are constantly evolving, and thus new applications should not require repeating the learning process all over again. This is a challenging task since even on the same information space, different types of entities and relations among them may play different roles for different purposes. For instance, one can recommend a hotel to a traveler group or a flight-hotel-package to a traveler, a user to each other in social network or a user to a game corporation for targeted advertising. Because of such distinct recommendation scenarios, the parameters need to be optimized accordingly even for the same model architecture, which is called fine-tuning. The optimization is generally extremely challenging, and it usually requires more training data for complex networks for a specific purpose. GPT-3 [240] is a general language model that can be applied without any gradient updates or fine-tuning for language generation tasks. However, it is not specific for graph processing or for recommendation tasks, which destined involve redundant parameters and structures considering the 175 billion parameters. In addition, due to the huge amount of training set and cost, people cannot perform low-cost retraining or fine-tuning for different purposes. Therefore, considering the increasingly complex heterogeneous networks, we believe that a highly adaptable model structure and training algorithm is urgent in the community of recommendations and has extremely high research value.

6.6.3 Explanability and persuasiveness

A good explanation for recommendation results can help to improve the transparency, persuasiveness, effectiveness, trustworthiness and satisfaction of recommender systems. It can also facilitate system designer for better system debugging. Earlier studies leverage top aspect/topic words as an explanation for the recommended items [115]. With the surge of deep learning-based approaches in GLRSs, it is even harder to provide convincing explanations and calibrate why the recommendation models are effective and thus to yield a robust model for varied scenarios. Ma et al. [247] provide recommendation explanations according to the learned reasoning rules on heterogeneous graph with ground-truth item associations in the knowledge graph. The emerging of attention mechanisms has more or less eased the non-interpretable concerns of deep learning-based recommendation on graphs. The learned attention weights can tell which parts of the input graph contribute more than others with higher attention scores to the recommendation results. However, it only shows part of the transparency of the model rather than the entire architecture. In addition, the current explanation on graphs is presented by highlighting certain branches of the graph with target user and recommended item to form a path, which we argue lacks reliable evaluation metrics the explainability and persuasiveness for target users, such as GLUE [242] for language understanding. Besides, we need to make sure the target population who we provide explanations for. They can be the end-users or researchers. For both groups, whether they are satisfied with this explanation and whether this explanation is sufficient still requires further empirical verification. Such verification cannot just stay on a small-scale user study or case study. It requires more target audience participation and different situations should also be taken into account.

6.6.4 Fairness

Fairness has attracted growing attention recently, especially in the context of intelligent decision making systems, e.g. recommender systems. Current RSs discriminate unfairly among the users in terms of recommendation performance, and further, the systems may discriminate between users in terms of explanation diversity [243]. One reason is probably related to the issue of data imbalance, such as inactive users who are disinclined to make sufficient interactions for recommendation basis. The imbalanced data can easily lead to bias observation on graphs where paths inference from user-to-item usually participate in the process of graph learning and meanwhile provide recommendation explanations. Research works delving into this data disparity issue include that Fu et al. [243] propose to solve the fairness on the user side from both individual- and group-level for KG enhanced explainable RSs. Specifically, they reveal that the unfairness issue is due to data imbalance through an empirical study on the e-commercial dataset, namely Amazon. Then they propose fairness metrics in term of path diversity as well as recommendation performance disparity based on KG. Another possible reason lies in the incomplete evaluation metrics, which renders the inclination of the learning objectives on accuracy driven for recommendation purpose.

Despite some studies considering the influence of fairness in GLRSs, related research is still quite limited with many issues remaining to be focused on. For instance, whether the construction and learning process of the graphs affects the fairness and discrimination of the ranking of recommendation results, whether there exists algorithmic bias among various graph-learning technologies, and whether fairness, bias and discrimination conflict with the accuracy of graph-based recommendations.

6.6.5 Scalability

Scalability is an essential factor that affects the applicable of recommendation models in real-world scenarios. To deal with large-scale graphs, most existing models choose to adopt a sampling scheme to construct subgraphs following the sampling strategy proposed in GraphSage [244]. Some use random walk strategy to get the neighborhood nodes and links, while others consider using the shortest path algorithm for subgraph construction. Another algorithm to increase model scalability is to use clustering scheme. Whether using sampling or clustering, a model will lose part of the graph information. The scalability is gained at the price of corrupting graph completeness. A node may miss its influential neighbors with a bad sampling strategy, and a graph may be deprived of a distinct structural pattern by clustering. Though subgraph strategy makes GNN-based algorithms applicable no matter how large-scale the whole graph is, the shortcoming is that the node representation should be recalculated for each propagation layer. Thus, how to tradeoff the algorithm scalability and graph integrity could be one of the further research directions. More researches can be studied on the sampling strategy in integrating more informative information from neighborhood nodes and link while minimizing the harm to the graph integrity. Recently, Kyriakidi et al. [58] propose to adopt graph databases as base to improve the data scalability and meanwhile build recommendation models on top. Different from other graph-based recommender systems which focus on model complexity when considering the efficiency problem, the work of [248] resort recommendation problem to the result of path traversal computations from start node to the end node, which shed light on a new perspective of improving the scalability of GL-based recommendations.

6.6.6 Recommendation on Dynamic Graph

Most current GLRSs are able to model stable static graphs, while fail to model dynamic graphs with changing structures. When nodes and edges appear or disappear from time to time, existing GL-based recommendation models cannot change adaptively. Recommender systems based upon static graph processing technologies are unable to capture the changing attributes of the graph resulting in the detention of modelling user's dynamic interest, which will affect the display of recommendation results and finally influence the user's experience. Solutions for this research question have been actively researched on and we believe it is an important research direction for future work. Besides, changes in user preferences or item attributes (e.g. co-interacted by users) revealed by dynamic properties of the graph may span multiple platforms and involve different fields. How to transfer such dynamic characteristic to the cross-platform or cross-fields, and how to jointly learning its impact on recommendation performance is also one of the future trends.

6.6.7 Complex Heterogeneity Learning

Apart from the user-item bipartite graph, most heterogeneity of graphs is reflected in the integration of different side information. Side information has been demonstrated a high degree of effectiveness in improving recommendation performance, especially for the data sparsity and cold-start issues. It can appear in different forms: textual, visual, or audio information; structure or non-structure. In current studies of graph-based recommendations, side information are extensively involved either as extra attributes of nodes or edges, or as sources being learned to constructed heterogeneous graphs, or as external resources outside the graph, which are learned in parallel with the graph and then integrated at a high level. Despite the variety of utilizing side information, no study can indicate which fusion strategy is better in general cases, or suitable for which recommendation scenarios. These we believe are extremely significant references and guidance for future researcher and technology users. Besides, some side information are from multi-sources such as user social relationships pointing out user-user interlinks directly, while item-item relations may from co-interaction pattern from specific user or user group. To integrate multi-sourced side information for recommendation purpose, some works learn representations from different graphs separately and combine the vectors from different sources. Some works combine different graphs into a large-scale heterogeneous graph which is then learned in a unified way. These two kinds of integration strategy can both contribute to the improvement of recommendations, but there is no evidence showing that which one is better. This is thus another research question for further study.

6.7 Conclusion

The study systematically investigated the graph learning-based recommendation. Compared with the conventional RSs, the recommendation algorithms based on graph-structured data have an advantage in solving the sparsity and cold-start problems with improved accuracy by mining and leveraging the explicit as well as implicit relations revealed on graphs. In GLRSs, the core is how to process graph-structured data, how to learn and obtain adequate information from the graph to fulfil the final recommendation purpose, and how to adapt the graph operation process to more complex and diverse graph structures as well as large-scale node and edges in real-world. Looking at the changes of GLRSs in recent years, the graph structure is from homogeneous to heterogeneous, the graph attribute from zero to multiplex, the technology used from the traditional recommendation algorithm to deep learning-based models that have been popular recently, the evaluation of recommendation performance shifted from focusing only on accuracy and click-through rate to increasingly multidimensional development. Such development sheds light on a new perspective for the community of recommendations and practitioners. We argue that the current graph-based recommendation algorithms are far from being fully developed, and further research investment and empirical studies are still needed.

Chapter 7

Demystifying Knowledge-aware User Intents for Session-based News Recommendation

Knowledge Graph (KG) has shown significant success in recommendation tasks due to its ability to expose the implicit connections between entities. Although it has been widely recognized that relationships between entities have indispensable value while modelling KG, existing studies overlook the co-effects caused by entity-relation interactions and the semantics carried by relation connectivities, which is non-trivial in news recommendation with diverse user intents and perplexed relations among news entities. In this paper, we propose a novel Relational Knowledge-aware Heterogeneous Graph Attention Network (ReKaH_GAT) to model entity-relation interactions and user intents explicitly for session-based news recommendation. Specifically, we design an original transformation schema from traditional KG to Entity-Relation Interaction (ERI) graph where the complex graph structure, entity and relation semantics are embedded in a unified way. A novel heterogeneous graph attention network with a self-attentive layer is applied subsequently to learn the context and intent embeddings from a session-specific ERI. Meanwhile, the semantic session embedding learned from pre-trained multi-lingual BERT is combined with contextual and intentional session embeddings to achieve a robust news recommendation. Extensive experiments are conducted on two real-world news datasets, demonstrating the effectiveness and explainability of our approach on recommendation tasks.

7.1 Introduction

Knowledge graph (KG), which can provide fruitful and structured side information for user-item interactions, has attracted increasing attention in improving the accuracy and explainability of recommender systems. KG appears as a type of directed heterogeneous graph in which nodes correspond to entities with different properties, and edges correspond to relations. They can provide semantic connectivity information among user/item via the different types of edges and nodes in between. Besides, the extra connectivity information derived from KGs endows the recommender systems with reasoning and interpretable capabilities [57].

Due to the highly practical value, many kinds of approaches for knowledge-aware recommendations have been proposed and can roughly fall into two categories, path-based [149,175] and embedding-based methods [55,222,221]. The advantages of path-based methods lie in the ability to provide explanations by searching in the connectivity information of users and items in KGs with pre-defined meta-paths. However, many relations excluded from meta-paths can hardly be captured accurately and specified the holistic semantics of paths. Besides, meta-paths are always selected requiring domain knowledge, and thus cannot be automatically uncovered the unseen connectivity patterns. Though embedding-based approaches can automatically fuse semantical information via entities from KGs to user/item embeddings and closely project similar users/items in the embedding space [55], they lack the reasoning ability for recommendation results, especially when multi-hop relations occur in KG. Recent works combine these two methods for recommendations and meanwhile provide explanations [57-59], while none of them is designed for the news domain.

Different from other domains, e.g., music and e-commerce, recommending news articles to online users has been recognized as a challenging problem because of several remarkable characteristics [299]: short shelf lives, continuous, anonymous users with few user profiles available, large-scale and complex relations among news entities. Although some recent studies on session-based news recommendations [217, 218] preliminary model internal relationships among a sequence of news articles with KG, they overlook the co-effects caused by entity-relation interactions and the semantics carried by relation connectivities. Moreover, user intent, which is also an important influential factor beyond user preference, is the lack of consideration. In fact, a user usually has multiple intents, driving him/her to consume different items. For example, the user may click a football news article highlighting the football player or specific football team while focusing on another music news article emphasizing a specific singer, reflecting different user intents on either football player/team or music singer. Thus, how to accurately mine user intents based on limited interaction records within each session while providing effective recommendations is the key to modern news recommender systems.

To this end, we propose a novel Relational Knowledge-aware Heterogeneous Graph Attention Network (ReKaH_GAT) to model entity-relation interactions and diverse user intents explicitly for session-based news recommendation. Specifically, we firstly construct an original entity-relation interaction (ERI) graph for each session where the complex graph structure, entity and relation semantics are embedded in a unified way. Then, a novel heterogeneous graph attention network (HGAT) is proposed to attentive aggregate information from different kinds of neighbouring nodes and capture the co-effects of entity-relation interactions. Afterwards, we adopt self-attention mechanisms to distil useful contextual information and user intents for session representation learning and explainability. Meanwhile, based on the purely attention-based sequence-to-sequence model, Transformer [192], sequential semantics of news articles within sessions are considered during the training procedure to enhance representation learning as well. To the best of our knowledge, we are the first to explicitly model user intents behind the complex interactions among knowledge entities and relations for better session representation learning and explanations. Experimental results on real-world benchmark datasets demonstrate the effectiveness and explainability of our approach.

7.2 Our Approach

In this section, we elaborate our proposed method in detail. Firstly, we formulate the problem, then explain how to learn session semantic embedding. After that, we introduce the knowledge-aware heterogeneous graph embedding module for session contextual embeddings and session intentional embedding. Finally, we explain model training process.

7.2.1 Problem Formulation

This Chapter focuses on session-based news recommendation, which aims to predict possible item a user will click next without user profiles or clicked history out of the current session. In this scenario, let $\mathcal{I} = \{i_1, i_2, \dots, i_{|\mathcal{I}|}\}$ denotes a set of items (news articles) involved in all sessions. For any anonymous session s, it can be denoted as a list $s = [i_1, i_2, \dots, i_s]$ of articles clicked by the user ordered by timestamp. $\mathcal{E}_s = \{e_1, e_2, \dots, e_n\}$ and $\mathcal{R}_s = \{r_1, r_2, \dots, r_m\}$ denote a set of entities and relations extracted from s and knowledge graph.

The goal of our model is to take s, $\mathcal{E}_s$ and $\mathcal{R}_s$ as input, and output probability distribution

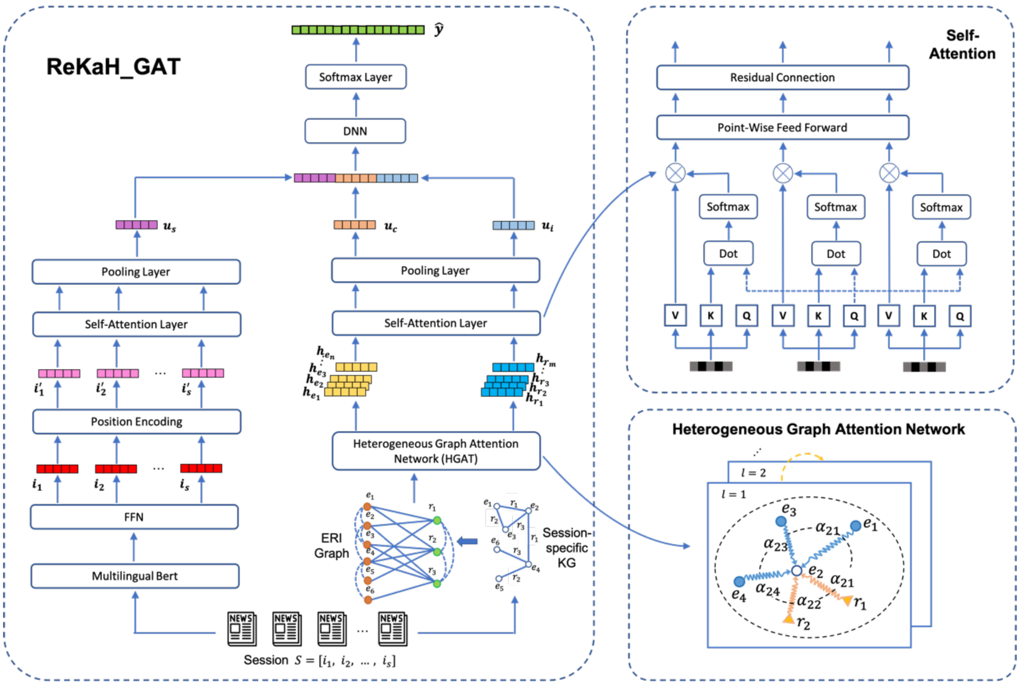


Figure 7.1: The proposed ReKaH_GAT framework. ERI is constructed for each session. Entity and relation nodes are explicitly learned with the proposed HGAT and their influences are captured with self-attention layer. Session representations are learned from semantic, contextual and intentional embeddings for the next clicked article recommendation.

$\hat{y}$ for all possible items in $\mathcal{I}$. The items with top-K values in $\hat{y}$ will be candidate items for recommendation.

7.2.2 The ReKaH_GAT Architecture

The model framework is shown in Figure 7.1. ReKaH_GAT consists of semantic encoding module, knowledge-aware heterogeneous graph embedding (KHGE) module and recommendation, which will be introduced in detail in the following parts.

Semantic Encoding. As BERT has been proved to be very beneficial for various downstream Natural Language Processing (NLP), we leveraged multi-lingual BERT[24] model [291] which is trained based on multi-lingual corpora and fine-tuned with news article classification objectives to initialize the item embeddings in separate mode. Specifically, given the current session s, we first learn item representations with multi-lingual BERT followed by a fully connected layer, represented as $\boldsymbol{S} = [\boldsymbol{i}_1, \boldsymbol{i}_2, \dots, \boldsymbol{i}_s], \boldsymbol{S} \in \mathbb{R}^{s\times d}$. Then a self-attention layer is leveraged to merge all article information into one single output vector with respect to the different importance of different clicked articles in the current session. Following the work of [192], in order to take into account different positions of a clicked article in the whole session sequence, we introduce positional encoding before the self-attention layer.

$$\boldsymbol{S'} = \boldsymbol{S} + \boldsymbol{P} \tag{1}$$

$$\boldsymbol{E} = SA(\boldsymbol{S'}) = softmax(\frac{(\boldsymbol{S'W^Q})(\boldsymbol{S'W^K})^T}{\sqrt{d}})(\boldsymbol{S'W^v}) \tag{2}$$

where $\boldsymbol{P} \in \mathbb{R}^{s\times d}$ is the positional encoding matrix. $\boldsymbol{W}^* \in \mathbb{R}^{d\times d}$ is trainable weight matrix converting article embeddings to key, value, query matrices where $* \in \{Q, V, K\}$. Similar to previous work [300], to endow the non-linearity to each layer while considering the interactions between different latent dimensions, a two feed-forward network (FFN) is applied:

$$FFN(\boldsymbol{E}) = ReLU(\boldsymbol{W}_1\boldsymbol{E} + \boldsymbol{b}_1)\boldsymbol{W}_2 + b_2 \tag{3}$$

However, a deeper network may cause overfitting or vanishing gradients in model training. Meanwhile, to reduce the transmission loss caused by matrix operations and propagate the last visited article's embedding to the final layer for enhancing the key role of the last visited article [301, 302], following the work of [192, 300], we leverage dropout, layer normalization and a residual connection after the FFN layer:

$$\boldsymbol{F}_j = FFN(\boldsymbol{E}_j) + Dropout(LayerNorm(FFN(\boldsymbol{E}_j))) \tag{4}$$

Here we get the article embedding $\boldsymbol{F}_j \in \mathbb{R}^d$. $\boldsymbol{E}_j$ is the j-th article embedding after self-attention layer. To achieve a better article embeddings, one may need multiple self-attention block for more complex article transitions based on $\boldsymbol{F}$. For simplicity, we define

[24] We choose multi-lingual BERT instead of ordinary BERT model because our experimental datasets include Norwegian and German news, which are both covered by multi-lingual BERT.

the above whole self-attention machanism as:

$$\boldsymbol{F} = SAN(\boldsymbol{S}') \tag{5}$$

Thus, for multiple stacks of self-attention blocks, the k-th block is defined as: $\boldsymbol{F}^{(k)} = SAN(\boldsymbol{F}^{(k-1)})$, where $\boldsymbol{F}^{(0)} = \boldsymbol{S}'$ and $\boldsymbol{F}^{(k)} = \boldsymbol{X}$. To get the final semantic session embedding $\boldsymbol{u}_s$, we leverage the average pooling layer: $\boldsymbol{u}_s = 1/s \sum_{j=1}^{s} \boldsymbol{X}_j$. Other pooling operation such as max pooling, summation or attention pooling can be used, but average pooling lead to best results in our cases.

KHGE Module. Despite their well recognized importance in affecting user-item interactions, relations in KGs are normally ignored [52] or modelled with entities in triplets [55, 56] while neglecting the rich information contained in relation correlations [180]. Hence, we define an ERI graph in each session, and model user intentional properties by learning the distribution of the relations in ERI. In this way, we can avoid the noise brought by too many irrelevant relations, and simultaneously concentrate the user's intents to a reasonable domain. Specifically, for a set of entities $\mathcal{E}_s$ extracted from news articles and their direct neighbours in the KG, we define two types of nodes and three types of edges in the ERI graph. Two types of nodes represent entity and relation, respectively. Three types of edges are: 1) entity node e_i and relation node r_i are connected if r_i is incident upon e_i in the original KG. 2) Two entity nodes e_i and e_j are linked if there is an edge between them in the original KG. 3) Two relation nodes r_i and r_j are connected if they are incident upon the same entity node in the original KG. In this way, ERI graph can simultaneously maintain graph topological structure, node (entity) and edge (relation) semantics and contextual information, transformed from the original KG.

Corresponding to different types of nodes and edges with different semantics, we define two adjacency matrices $\boldsymbol{A}_s^{(in)} \in \mathbb{R}^{(n+m)\times(n+m)}$ and $\boldsymbol{A}_s^{(out)} \in \mathbb{R}^{(n+m)\times(n+m)}$ representing incoming and outgoing connections between nodes respectively. For simplicity, we use $\boldsymbol{A}_s$represent either $\boldsymbol{A}_s^{(in)}$ or $\boldsymbol{A}_s^{(out)}$for they have same propagation rule but do not share parameters. Then we adopt Graph Convolutional Network (GCN) [303] to learn node embeddings due to its simplicity and effectiveness in learning and aggregating semantic influence and connections from neighbour nodes. For each ERI graph $\mathcal{G}_s$, the layer-wise propagation rule is as follows:

$$\boldsymbol{H}^{(l+1)} = \sigma(\boldsymbol{M}^{-\frac{1}{2}}(\boldsymbol{A}_s + \boldsymbol{I})\boldsymbol{M}^{-\frac{1}{2}}(H)^{(l)}\boldsymbol{W}^{(l)}) \tag{6}$$

where $\boldsymbol{I}$ is an identity matrix added self-connections and $\boldsymbol{M}$is the degree matrix. $\boldsymbol{W}^{(l)}$ is a layer-specific trainable transformation matrix and $\boldsymbol{H}^{(l)} \in \mathbb{R}^{(n+m)\times d}$ denotes the hidden representations of nodes in $\mathcal{G}_s$ in the l-th layer. $\boldsymbol{H}^{(0)}$ is randomly initialized with uniform distribution. Unfortunately, GCN cannot be directly applied to the ERI graph due to the node and edge heterogeneity issue. To this end, we propose a heterogeneous graph attention network with respect to different types of neighbours and their different impact on the central node. Specifically, given a central node $o \in \mathcal{E} \cup \mathcal{R}$ with type $\tau \in \{entity, relation\}$, we define a type vector as $\boldsymbol{h}_\tau = \sum_{o' \in \tau}(A_{s,oo'} + I)h_{o'}$, which is the sum of neighbour node embeddings with type τ. Then the attention score of different types of τ can be calculated as:

$$\alpha_\tau = softmax(\sigma(\boldsymbol{w}_\tau(\boldsymbol{h}_o \oplus \boldsymbol{h}_\tau))) \tag{7}$$

where $\boldsymbol{w}_\tau$ is the attention vector for the node type τ. After that, we also need to consider different types of edges related to the central node o. Particularly, there are four types of edges $\mu \in \{e_i \rightarrow e_j, e_i \rightarrow r_i, r_i \rightarrow r_j, r_i \rightarrow e_j\}$ in which $\rightarrow$ is edge direction from head node to tail node, $e_i, e_j \in \mathcal{E}_s$ and $r_i, r_j \in \mathcal{R}_s$. Similarly, we define edge vector as $\boldsymbol{h}_\mu = \sum_{oo' \in \mu}(A_{s,oo'} + I)h_{o'}$. The attention weights with respect to different edge types can be calculated as:

$$\alpha_{\tau,\mu} = softmax(\sigma(\boldsymbol{w}_\mu \alpha_\tau(\boldsymbol{h}_o \oplus \boldsymbol{h}_\mu))) \tag{8}$$

where $\boldsymbol{w}_\mu$ is the attention vector for the edge type μ. Then, we incorporate the hierarchical attention mechanism into GCN as:

$$\boldsymbol{H}^{(l+1)} = \sigma(\textstyle\sum_{\tau,\mu} A'_{\tau,\mu} H^{(l)}_{\tau,\mu} W^{(l)}_{\tau,\mu}) \tag{9}$$

where $A'_{\tau,\mu}$ is the attention matrix, whose element represents the hierarchical attention score $\alpha_{\tau,\mu}$ with respect to τ type of node and μ type of edge. The output of HGAT are entity embeddings $H_e = \{h_e, e \in \mathcal{E}_s\}$ and relation embeddings $H_r = \{h_r, r \in \mathcal{R}_s\}$. To learn different importances of entities/relations affecting the user's next article decision, we adopt a self-attention layer with FFN, followed by an average pooling layer to learn session contextual embeddings with respect to entities and session intentional embeddings with respect to relations:

$$\boldsymbol{\mu}_c = \frac{1}{n}\textstyle\sum_{k=1}^{n} SAN(\boldsymbol{H}_{e,k}) \quad , \quad \boldsymbol{\mu}_i = \frac{1}{m}\sum_{k'=1}^{n} SAN(\boldsymbol{H}_{r,k'}) \tag{10}$$

Different from the self-attention layer in learning session semantic embedding, we do not adopt residual connection here since there is no sequential order w.r.t. entities and relations.

Finally, we compute the hybrid session embedding $\boldsymbol{v}$ by taking feed-forward layer over the concatenation of the semantic, contextual and intentional session embedding vectors as $v = FFN(\boldsymbol{\mu}_s \oplus \boldsymbol{\mu}_c \oplus \boldsymbol{\mu}_i)$.

Model Learning. After obtaining the session embedding, we compute the prediction score of each candidate news article $i_j \in \mathcal{I}$ as $\hat{z}_j = \boldsymbol{v}^T \boldsymbol{i}_j$. Then we apply a softmax layer to get the probability distribution of the items appearing to be the next in s, $\hat{y} = softmax(\hat{z})$. To train the model, we define the loss function as the cross-entropy of the prediction $\hat{\boldsymbol{y}}$ and the ground truth y with L2-norm.

$$\mathcal{L}(\hat{\boldsymbol{y}}) = -\sum_{k=1}^{|\mathcal{I}|} y_k \log(\hat{y}_k) + (1 - y_k)\log(1 - \hat{y}_k) + \eta||\Theta||_2 \tag{11}$$

where η is regularization factor and Θ is model parameters. Finally, the Back-Propagation Through Time (BPTT) algorithm is adopted to train the whole model.

Time Complexity. The time cost of ReKaH_GAT mainly comes from the self-attention layer, the feed-forward network and the heterogeneous graph attention network. The computational complexity of self-attention layer is $\mathcal{O}((s^2 + n^2 + m^2)d)$, where s, n, m represent the input length, number of entities and number of relations of ERI graph, and d is the embedding size. Correspondingly, the time complexity of the feed-forward network is $\mathcal{O}\big((s + n + m)d^2\big)$. For HGAT, the time cost is mainly from the attention computing and aggregation scheme. The computational complexity of HGAT with L layers is $\mathcal{O}(L((n + m)d^2 + |\mathcal{E}'|d)$ where $|\mathcal{E}'|$ denotes the number of edges in ERI graph. Thus, in total, the time complexity of the whole training epoch is $\mathcal{O}(|S|\big((s^2 + n^2 + m^2)d\big) + (s + n + m)d^2 + L((n + m)d^2 + |\mathcal{E}'|d)))$, where $|S|$ is the total number of sessions.

7.3 Experiments and Results

7.3.1 Experimental Settings

Datasets. We perform experiments on two real-word datasets which are publicly accessible and vary in terms of language and size. The details of them are shown in Table 7.1.

Table 7.1: Statistics of the two datasets.

Dataset	#sessions	#items	#events	#Avg. session len	#Avg. words / title	#Avg. entities / news	#KG entities	#KG relations	#relation types
Adressa	9,211,140	48,486	113,579,695	12.00	6.64	26.5	10,708,665	15,553,942	85
CLEF	3,008,150	16,043	10,936,023	11.70	5.10	4.28	12,724,741	17,004,376	85

1) Adressa Dataset[25]. This is a Norwegian news dataset published in [227] which contains about 113 million events over a 90-day period from 1 January to 31 January 2017. Each event represents that a user read a particular news article. The entities from each news article have been included in the dataset.

2) CLEF Newsreel Dataset (CLEF for short) [293]. This dataset is originally published for offline evaluation of an online news recommendation campaign. The dataset contains interactions between users and news articles from 8 different publishing sites in February 2016. In our experiments, we select session sequences in which the articles were clicked by the users for our experimental dataset.

In Adressa dataset, the entities from each news article have been included in the dataset extracted by Adressa company. For CLEF dataset, we extract news entities with spaCy[26], and select session sequences in which the articles were clicked by the users for our experimental dataset. For both datasets, we discard the sessions with events less than 3. It should be noted that each benchmark dataset constitutes a respective knowledge graph with entities and relations crawled from YAGO 4 [63], a large-scale multi-lingual knowledge graph built upon Wikipedia. We choose YAGO since it contains entities and relations in under-resourced language e.g., Norwegian. Following the work of [201], there is no constraint on possible path patterns, and thus any path between a user and the recommended item should be considered[27]. The testing set is built with the last event of each session. The remaining events form the training set.

Baselines. In this subsection, we compare ReKaH_GAT with four groups of recommendation baselines. 1) Session-based or sequencial recommendation framework GRU4Rec [41], NARM [43] and SR-GNN [46]. 2) News recommendation framework NPA [31] and DKN [32]. 3) Recommendation with considering both KG entities and relations: KGAT [222] and KGIN [60].

1) GRU4Rec. It is a session-based recommendation model that utilizes session-parallel mini-batch training process and also employ RNNs to model sequences.

[25] https://www.ntnu.no/wiki/display/smartmedia/SmartMedia+Program.

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

[27] The experimental KGs can be available at https://www.dropbox.com/sh/i0akwet3cw5o9jx/AAAHtU8nS76FMxvx1YUQXBk5a?dl=0.

2) NARM. It employs RNNs with attention mechanism to capture the user's main purpose and sequential patterns.

3) SR-GNN. It is a session-based recommendation framework which our work is mainly based on. It models each session sequence as a directed graph and a GNN followed by an LSTM layer is employed to learn item embeddings.

4) NPA. It is a deep neural news recommendation approach with personalized attention. CNN network is used to learn the hidden representations of the news based on news titles.

5) DKN. It employs CNN to learn entity as well as relation embeddings. A news-level attention network is adopted to capture user intentions when providing recommendations.

6) KGAT. It employs TransR to learn entity and relation embeddings of KG and adopt an attention mechanism to capture the connectivity importance within paths. KGAT can thus provide explanations upon the learned attention weights.

7) KGIN. It considers user-item relationships at the finer granularity of intents and long-range semantics of relational paths under the GNN paradigm.

Evaluation Metrics. We evaluate all the methods by the following two metrics:

1) Recall@K. It is defined as the fraction of cases where the article actually clicked in the next timestamp in current session is among the top K articles recommended [61].

2) MRR@K. Mean Reciprocal Rank (MRR) is defined as the average of the reciprocal ranks of the desired items [62]. The rank is set to zero if it is above K.

We show the performance when $K = \{5,10,20\}$, as a larger value of K is usually ignored for a typical top-K recommendation [39].

Parameter Settings. We implement our ReKaH_GAT in Tensorflow. The embedding size d is set to 64 and 128 for Adressa and CLEF datasets, respectively. The number of self-attention blocks is set to 2. We initialize all parameters using a Gaussian distribution with a mean of 0 and a standard deviation of 0.1. The mini-batch Adam optimizer is exerted to optimize the parameters with an initial learning rate set to 0.001 and will decay by 0.1 after every 3 epochs. Note that the size of neighbours of an entity may vary significantly over the KG. To keep the computation more efficient, we uniformly sample a fixed number h of neighbours when the number of neighbours is more than h. We set h to 8 and 16 for Adressa and CLEF datasets, respectively. The hop number L is set to 2 and 3 for Adressa

Table 7.2: Overall recommendation performance w.r.t. Recall@K and MRR@K for Adressa and CLEF datasets. The best results are highlighted in boldface. “*" indicates the improvements are statistically significant for p-value < 0.01 with paired t-test.

Datasets	Metrics	GRU4Rec	NARM	SR-GNN	NPA	KGAT	DKN	KGIN	**ReKaH_GAT**	-w/o RE	-w/o HGAT
Adressa	Recall@5	0.1379	0.1970	0.2903	0.3318	0.3692	0.2247	0.3856	**0.4134***	0.3726	0.3402
	Recall@10	0.1623	0.2284	0.3742	0.4163	0.4551	0.3297	0.4731	**0.4969***	0.4628	0.4336
	Recall@20	0.1852	0.2531	0.4615	0.4962	0.5264	0.4176	0.5583	**0.5811***	0.5387	0.5069
	MRR@5	0.0503	0.0664	0.1886	0.2032	0.2285	0.1653	0.2449	**0.2531***	0.2372	0.2146
	MRR@10	0.0636	0.0782	0.2028	0.2192	0.2475	0.1903	0.2653	**0.2701***	0.2569	0.2307
	MRR@20	0.0773	0.0931	0.2406	0.2522	0.2851	0.2063	0.3006	**0.3112***	0.2886	0.2649
CLEF	Recall@5	0.2353	0.3429	0.5074	0.5436	0.5759	0.3942	0.5909	**0.6253***	0.5871	0.5636
	Recall@10	0.2873	0.4057	0.5742	0.5921	0.6108	0.4674	0.6344	**0.6613***	0.6219	0.5991
	Recall@20	0.3594	0.4921	0.6537	0.6703	0.6842	0.5447	0.6926	**0.7082***	0.6876	0.6772
	MRR@5	0.1768	0.2033	0.3526	0.3637	0.3842	0.3177	0.4106	**0.4237***	0.3974	0.3699
	MRR@10	0.2264	0.2836	0.4015	0.4232	0.4467	0.3584	0.4602	**0.4796***	0.4533	0.4291
	MRR@20	0.2882	0.3613	0.4414	0.4571	0.4852	0.4136	0.4970	**0.5128***	0.4896	0.4618

and CLEF datasets, respectively by grid search. For all the baselines, we used the implementation provided by their respective authors to avoid unfair comparison due to faulty implementation. The parameters of the baselines are tuned to achieve the best performance in our problem. Experiments are conducted on GPU machines of Nvidia GeForce GTX TITAN X. The final performances are reported after 5 runs with the average test results.

7.3.2 Experimental Results

Recommendation Performance. Table 7.2 presents the recommendation performance of our ReKaH_GAT as well as other baselines on both datasets, and we can make the following observations. (1) Most methods with contextual information for user/item modelling (e.g., KGAT, KGIN and NPA) outperform methods only leveraging user-item interactions (e.g., SR-GNN, NARM and GRU4Rec). This is because contextual information can bring more informative knowledge for learning latent factors of user preferences. However, DKN slightly underperforms SR-GNN in both datasets, which is probably because the latter one considers the sequential patterns and capture the transitions among nodes with constructed graph of each session which is crucial in modelling news recommendation tasks. (2) Among ID-based methods (SR-GNN, NARM and GRU4Rec), the methods using attention mechanism during user/item modelling (SR-GNN and NARM) outperform the method without attention mechanism such as GRU4Rec, which probably because different parts of news and contexts contribute differently for recommendations. It is beneficial to select important features when modelling user preferences for better recommendation performance. SR-GNN performs better than NARM, which shows the importance of modelling item transitions in session settings and sequential patterns when only few user-item interactions are available. (3) NPA achieves better performance than DKN while underperforms KGAT in two datasets, indicating the effectiveness of the attention mechanism which can be used to learn the explicit information among words and interactions. Meanwhile, high-order connectivity in graph structured data has the positive effect of modelling user/item profiles for recommendations. Though DKN incorporates KG when modelling user preferences, it probably cannot effectively utilize the word orders and semantic meanings in between. ReKaH_GAT and KGIN outperform KGAT in both datasets, which is probably benefit from relational modelling in uncovering user intents, resulting in more powerful representations of sessions. (4) ReKaH_GAT consistently outperforms all baselines across two datasets in terms of all evaluation metrics. More specifically, it achieves improvements over the strongest baseline with respect to recall@20 by 2.28%, 1.56% and mrr@20 by 1.06%, 2.21% in Adressa and CLEF datasets, respectively. Although KGIN also explores the hidden user intents by relational modelling on KG, the differences lie in the way we model entities and relations w.r.t. entity-relation interactions and entity/relation semantic information in each session, while KGIN model entity/relation separately without considering the mutual influence between them. Besides, ReKaH_GAT model sequential patterns in each session by leveraging self-attention with residual connection to learn the transitional information among articles in each session. Finally, we find out that all methods achieve better performance in CLEF than Adressa. One possible reason is that more entities can be aligned with YAGO in CLEF than Adressa, and there are more noises from KG in the latter dataset, which shows the importance of the quality of KG.

Ablation Study. To verify the impact of KG relations and HGAT in affecting recommendation performance, we make two variants by 1) discarding all KG relations and learn entity embeddings with GAT, termed -w/o RE, and 2) removing HGAT scheme and using random initialization for entity and relation embeddings, termed -w/o HGAT. The results are summarized in Table 7.2. We can observe that compared with ReKaH_GAT, removing relations results in a consistent performance drop, which shows the importance of relation embedding. Although entities are incorporated, it lacks modelling the internal relationships among entities in a finer-grained way. Without HGAT, as expected, the recommendation performance drops significantly, since the co-effects of entity-relation interactions and the complex node transitions cannot be well-captured.

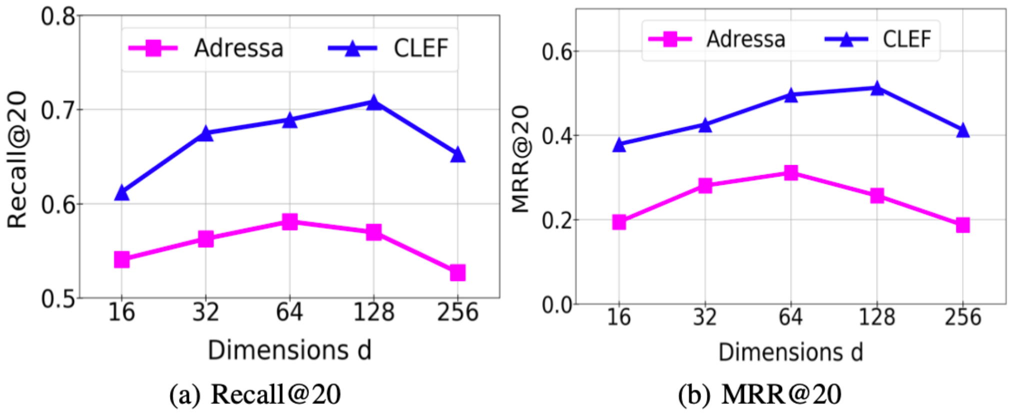


Figure 7.2: Effect of embedding size d on ReKaH_GAT.

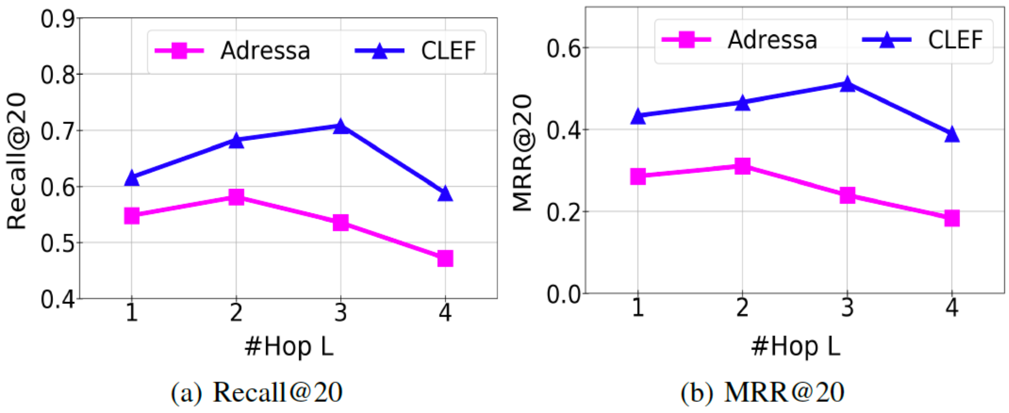
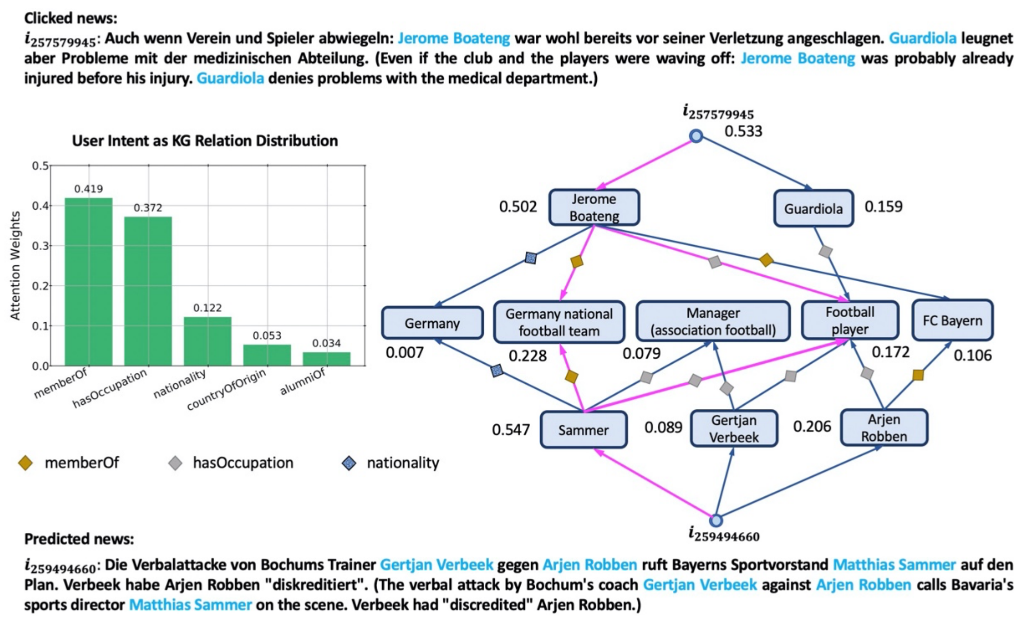


Figure 7.3: Effect of different hops L on performance of ReKaH_GAT.

Parameter Sensitivity. ReKaH_GAT involves a number of hyper-parameters which may affect the final recommendations. Here we examine how different choices of embedding size d and the number of hops L affect the performance of ReKaH_GAT.

We first investigate how the embedding size d influences the recommendation results by varying d in set $\{16, 32, 64, 128, 256\}$. The results are shown in Figure 7.2, from which we can observe that performance initially improves with the increasing number of d. This is probably attributed to more useful semantic information encoded in node embeddings. However, the performance drops when d further increases, as a too large d, e.g., larger than

Figure 7.4: Explanation of user intents with respect to KG entities and relations in real case. Best viewed in color.



Clicked news:

$i_{257579945}$: Auch wenn Verein und Spieler abwiegeln: Jerome Boateng war wohl bereits vor seiner Verletzung angeschlagen. Guardiola leugnet aber Probleme mit der medizinischen Abteilung. (Even if the club and the players were waving off: Jerome Boateng was probably already injured before his injury. Guardiola denies problems with the medical department.)

Predicted news:

$i_{259494660}$: Die Verbalattacke von Bochums Trainer Gertjan Verbeek gegen Arjen Robben ruft Bayerns Sportvorstand Matthias Sammer auf den Plan. Verbeek habe Arjen Robben "diskreditiert". (The verbal attack by Bochum's coach Gertjan Verbeek against Arjen Robben calls Bavaria's sports director Matthias Sammer on the scene. Verbeek had "discredited" Arjen Robben.)

64 or 128 for different datasets, may introduce noises which mislead the subsequent prediction.

Then we further investigate the influence of hop number L to recommendations by varying the maximal hop number of L in $\{1, 2, 3, 4\}$. The results are shown in Figure 7.3, from which we can observe that the best performance is achieved when L is set to 2 or 3. We result the consequence to the trade-off between the positive messages from long-distance dependency and negative signals from noises.

Case Study (Q4). To give an intuitive impression of the explainability of ReKaH_GAT, we present an example to show how user intents can be expressed through KG entities and relations. For the predicted piece of news, we backtrack through the adjacency matrix to get the connection between the predicted article and one of the user clicked articles with high attentive scores on relations and entities. As demonstrated in Figure 7.4, we can have the following observations: 1) The learned attention weight distribution on KG relations reflect the importance to influence the user next clicked article. In our case, $memberOf$

and $hasOccupation$ have the highest score among other relations appearing in the current session, which shows the user may pay more attention to some organizations (e.g. football teams) and a specific group of occupations (e.g. football player in this case). 2) However, merely providing the distribution of the relation can only provide part of the user's intentions, and the combination of relation and entity can more effectively reflect what users want. For instance, from path $“Jerome\ Boateng” \xrightarrow{memberOf} "Germany\ national\ football\ team"$, we can see that by connecting entity with relation, the model will focus on the specific domain, e.g. football in this case, leading to a more reasonable prediction result by one of the paths $"Sammer" \xrightarrow{memberOf} "Germany\ national\ football\ team"$. In most cases, we also find that relations with high attention weights tend to be in correlation with entities with high attention weights as well, which we believe is due to the joint modelling of entities and relations as well as their internal correlations.

7.4 Conclusion

In this paper, we have proposed a novel method named Relational Knowledge-aware Heterogeneous Graph Attention Network (ReKaH_GAT) to model complex entity-relation interactions and diverse user intents explicitly for session-based news recommendation. An original Entity-Relation Interaction (ERI) graph is designed to integrate the complex knowledge graph structure, entity and relation semantics in a unified way. Furthermore, we propose a novel heterogeneous graph attention network with a self-attention mechanism to extract user intents for enhancing the session representation learning. Extensive experiments on two real-world news datasets demonstrate the effectiveness and explainability of ReKaH_GAT.

Compared with existing work, we are the first that explicitly model KG relations for better expressing user intents with learned relation distributions from the ERI graph in each session. Though KGIN has also explored the user intents w.r.t. aggregation of KG relations, it does not consider entity-relation correlations. In contrast, we design a hierarchical attention mechanism considering different importance of neighbour nodes and edges to the central node, to learn better entity/relation embeddings.

Part IV Conclusions

Chapter 8

Conclusions

In this chapter, we briefly summarizes the thesis by presenting our contributions, reflections of our work and potential future research directions.

8.1 Contributions

Online platforms create a unique environment for content creation, sharing and consumption, which open new possibilities for new applications. One critical factor that affects the success of online applications is mining user interests from user behavior data. By analyzing user data, service providers can retain original users, develop potential customers, and generate revenue through customized advertising delivery. The challenge of accurately and efficiently identifying user interests has been studied for many years. Due to the complexity, variability and randomness, many questions regarding them remain open.

To this end, the main contributions of this thesis specifically related to model user interests from streaming data are twofold. First, we investigated the dynamic nature of user interests and studied in-depth attribute features affecting and shaping user behaviors. Second, we identified the significance of auxiliary features from knowledge graphs which are then used for recommendation purposes. Specifically, our contributions of this thesis can be summarized as follows.

8.1.1 Mining Attribute Information for User Modelling

We propose to use Recurrent Neural Network for personalized time aware user interests prediction based on Twitter datasets. To overcome the data sparsity problem, we enrich our user behavior matrix with the help of Mediawiki api, and we also leverage multiple aspects of users' activities, including tweets, retweets and comments. For the first time, our Neural Time Series Forecasting Model (NTSF) extracts the common interests patterns and predicts their interested topics for a specific user by introducing various features. Experimental results on real datasets show that NTSF outperforms the classic and state-of-art methods on prediction problems.

Then we presented a Deep Joint Network (DeepJoNN) for session-based recommendations. The proposed model allows combining various item features such as ID, category, keywords and entities, which then are transformed into character-level input matrix to the model. DeepJoNN consists of two parts of deep neural networks coupled together in a hierarchical way and thus could extract contextual patterns and process long and short-term dependencies simultaneously. In the comparison of the state-of-the-art baselines, DeepJoNN achieved nearly 11% and 12% improvements on datasets of Adressa and Last.fm respectively w.r.t. Recall value. Additionally, we also explored the influence of different parameter settings and conducted experiments on different loss functions. Our model also performed competitively on cold start users without user profiles.

Further, different from existing works assuming that all user-item interactions are equally important, a novel dynamic attention-integrated neural network (DAINN) is proposed to address the problem of personalized session-based recommendation in Chapter 5. In order to capture users' interests, we consider item semantic embedding, user long-term interest modelling and session-based public behavior mining in a unified framework, which can be trained end-to-end. By incorporating an attention mechanism into DAINN, our proposed approach can deal with the diverse variance of users' clicking behavior and capture the users' main purpose in the current session. DAINN can effectively learn users' real-time preferences and conduct personalized recommendations. Evaluation on three different real-world datasets demonstrated the effectiveness of the proposed approach.

8.1.2 Exploring Graph Structured Data for User Modelling

We proposed a novel method named Relational Knowledge-aware Heterogeneous Graph Attention Network (ReKaH_GAT) to model complex entity-relation interactions and diverse user intents explicitly for session-based news recommendation. An original Entity-Relation Interaction (ERI) graph is designed to integrate the complex knowledge graph structure, entity and relation semantics in a unified way. Furthermore, we propose a novel heterogeneous graph attention network with a self-attention mechanism to extract user intents for enhancing the session representation learning. Extensive experiments on two real-world news datasets demonstrate the effectiveness and explainability of ReKaH_GAT.

In the future, we will further explore the evolution of user intents and the integration of long-term user preference.

8.2 Answers to research questions

Here, we elaborates on the methods and our findings by answering the research questions raised in Section 1.3.

RQ1: How can attribute features of users and items can be learned and integrated for effective user modelling and recommendation?

To answer this research question, we discussed different attribute features in influencing user modelling and established different ways of modelling these features in different tasks. Specifically, this research question can be answered through the following three sub-questions.

RQ1.1: What kinds of temporal patterns can be leveraged to predict dynamic user interests in textual data streams?

In Chapter 3, we propose a neural time series forecasting model (NTSF) to fit and predict user preference trends according to time. In this model, we integrate emerging/hot topic detection to deal with the short-term aspects and use Fast Fourier Transformation (FFT) to differentiate the cyclic behavior of users. Considering the nonstationary and nonlinear characteristics appearing through user interest patterns, Long Short-Term Memory Recurrent Neural Network (LSTM-RNN) is employed to balance the influence of short and long term aspects, as well as adjust the model parameters according to historical results. Our experimental results with extensive Twitter datasets verify the effectiveness of our approach by outperforming the best baseline by 11.98% and 29.58% on average on the Pearson Correlation score and MSE respectively on predicting user interested topics.

RQ1.2: How item-level semantic features can be applied to alleviate the cold start issue in session-based recommendation?

In Chapter 4, we proposed a method that combines user click events within-session and news contextual features to predict the next click behavior of a user. The model consists of two different kinds of hierarchical neural networks to learn article contextual properties and temporal sequential patterns in streams of clicks. Character-level embedding over input features is adopted to allow integrating different types of data and reduce engineering computation. Besides, we also introduced a time-decay method to compute the freshness of news articles within a time slide. Experimental results show that our model outperforms the best baseline by 65.58%, 44.61% and 86.49%, 27.04% on average on recall and MRR score on Adressa and Last.fm datasets, respectively on session-based recommendation tasks.

RQ1.3: How the spatial-temporal, semantic, inter- and intra-session features can be integrated into a unified framework for modelling complex dynamic user interests and effective recommendations?

In Chapter 5, we propose a novel neural network framework, a dynamic attention-integrated neural network, to tackle the problems. Specifically, we propose a dynamic neural network to model users' dynamic interests over time in a unified framework for personalized news recommendations. News article semantic embedding, user interests modelling, session-based public behavior mining and an attention scheme used to learn the attention score of user and item interaction within sessions are four key factors for online sequences mining and recommendation strategy. Experimental results on three real-world datasets show significant improvements over several baselines and state-of-the-art methods on session-based recommendations. Specifically, our model outperforms the best baseline by 34.65%, 21.63%, 5.76%, and 7.78%, 13.69%, 11.61% on average on MRR and F1-score on Adressa, last.fm and Weibo datasets.

RQ2: How auxiliary feature from knowledge graph can be employed to understand user intents and improve the explainability of recommender systems?

After a systematic analysis of the existing graph-learning-based recommender systems, we integrated the knowledge graph into the process of user modelling for better discovering the potential associations among users and items. Specifically, the answer to this research question is on the basis of the decomposition and answers of the following two sub-research questions.

RQ2.1: How to understand data-driven mechanisms behind graph-learning based recommendation approaches through taxonomic assessment on recent advances?

Through a systematically study on the graph learning-based recommendation, we proposed a novel taxonomy from the data-oriented perspective and conducted a comprehensive taxonomic analysis from the technology-oriented perspective that gives a more systematic and in-depth understanding of existing GLRSs. Moreover, we summarized our observations in Chapter 6. Compared with the conventional RSs, the recommendation algorithms based on graph-structured data have an advantage in solving the sparsity and cold-start problems with improved accuracy by mining and leveraging the explicit as well as implicit relations revealed on graphs. In graph learning-based recommender systems, the core is how to process graph-structured data, how to learn and obtain adequate information from the graph to fulfil the final recommendation purpose, and how to adapt the graph operation process to more complex and diverse graph structures as well as large-scale node and edges in real-world. Looking at the changes of GLRSs in recent years, the graph structure is from homogeneous to heterogeneous, the graph attribute from zero to multiplex, the technology used from the traditional recommendation algorithm to deep learning-based models that have been popular recently, the evaluation of recommendation performance shifted from focusing only on accuracy and click-through rate to increasingly multidimensional development. Such development sheds light on a new perspective for the community of recommendations and practitioners. We argue that the current graph-based recommendation algorithms are far from being fully developed, and further research investment and empirical studies are still needed.

RQ2.2: Does explicitly modelling of relations in KG help capture user intents for improving session-based recommendation performance and explainability?

In Chapter 7, we propose a novel Relational Knowledge-aware Heterogeneous Graph Attention Network (ReKaH_GAT)) for session-based news recommendation. To achieve this goal, we innovatively design a transformation schema from traditional KG to Entity-Relation Interaction (ERI) graph to simultaneously consider the complex graph structure, entity and relation semantics. A novel heterogeneous hierarchical graph attention network is proposed to learn the different importance of different types of nodes and edges in an ERI graph. Meanwhile, user intents are captured through a probability distribution on relations in KG for better mining the user behaviors. Parallelly, the semantic session embedding learned from pre-trained multi-lingual BERT is combined with contextual and intentional session embeddings to achieve a robust news recommendation. Extensive experiments on two real-world news datasets demonstrate the effectiveness and explainability of ReKaH_GAT. More specifically, it achieves improvements over the strongest baseline with respect to recall@20 by 2.28%, 1.56% and mrr@20 by 1.06%, 2.21% in Adressa and CLEF datasets, respectively.

8.3 Reflections and Future Directions

This section presents limitations and possible future work related to the methods propsed in this thesis.

8.3.1 Reflections

Despite the effectiveness of our proposed methods in modelling user preferences in different scenarios, some limitations need to be discussed below:

First, in all of our proposed frameworks, the outputs are from the softmax layer that indicates the probability distributions of the next items that the target user will interact with. Meanwhile, the training set should include all candidate items. Such model architect and training strategy degrade the efficiency of the model. When a new item appears, the model needs to be retained to learn new item profiles and adjust model parameters. In the scenarios with high timeliness requirements, such as the news recommendation field, retraining will reduce the efficiency of recommendation, resulting in bad user experience due to the inability to provide users with timely recommendations. Apart from this, our DAINN model outputs word distributions instead of item distributions which alleviates the retaining problem to some extent. However, the output distributions may be too scattered, resulting in a small probability for each word when facing a large-scale vocabulary. Therefore, the words in the vocabulary need to be selected in the preprocessing stage.

Besides, all attributes and auxiliary features we exploited in this thesis are information with text as a carrier, whereas the factors that affect users' behavior in the online environment also exist in other different forms such as images and videos. Today, when the fast-moving era has arrived, people are very likely to be attracted by images and short videos rather than pure text messages. As such, exploiting multi-modal fusion techniques for user behavior modelling should also be taken into consideration.

Furthermore, due to different datasets and task objectives, the evaluation criteria are different, especially comparing chapter 3 with the later chapters. In chapter 3, we targeted the collection of active users' clicking records for 1 year in Twitter whereas the Adressa dataset spans 3 months and basically an unbiased collection of all online users' clicking events during the targeted period. Compared with Twitter dataset, most Adressa users are anonymous and thus only contain the clicking records of the current session, which makes it difficult to mine the users' periodic interest in such a short period of time. In contrast, later user modelling approaches for recommendation purposes can be evaluated systematically and all performed on Adressa dataset.

In addition in chapter 5, although we initially explored the influence of some attribute combinations in affecting recommendation performance, we did not consider all attributes in a unified framework. Theoretically, we can integrate all factors into the same model, but that will increase training costs. Besides, which attributes are more important in determining user interests needs to be taken into account, especially for different users, for the same user in different time periods, the determining factor may be different. Unreasonable integration may reduce the recommended performance and increase the computational cost. In the future, it will be a good research direction to explore how to effectively integrate the complex and diverse factors in a unified framework.

8.3.2 Future Directions

We believe the Reflections on mining users' interest from online streams will continue to remain an important research area. In the following, we present several potential directions related to user modelling for future work.

First, in chapter 7, we provide a case study on the explainability of our proposed model on the testing dataset. However, the actual impact and the acceptance of these explanations on real users haven't been tested and explored. Therefore, one possible future direction is to see what the real-world users think of our proposed recommendation method and the provided explanations. We also would like to integrate KG into our DAINN model to see how the KG can affect the recommendation performance and how different parts can be influenced on the overall recommendation results.

Besides, most current graph-based recommenders are able to model stable static graphs, while failing to model dynamic graphs with changing structures. When nodes and edges appear or disappear from time to time, existing graph learning-based recommendation models cannot change adaptively. Recommender systems based upon static graph processing technologies are unable to capture the changing attributes of the graph resulting in the detention of modelling user's dynamic interest, which will affect the display of recommendation results and finally influence the user's experience. Solutions for this research question have been actively researched on and we believe it is an important research direction for future work. Besides, changes in user preferences or item attributes (e.g. co-interacted by users) revealed by dynamic properties of the graph may span multiple platforms and involve different fields. How to transfer such dynamic characteristics to the cross-platform or cross-fields, and how to jointly learn its impact on recommendation performance is also one of the future trends.

Another potential research direction in the future is how to adapt the existing user modelling framework to new research fields or new applications. Real networks are constantly evolving, and thus new applications should not require repeating the learning process all over again. This is a challenging task since even in the same information space, different types of entities and relations among them may play different roles for different purposes. For instance, one can recommend a hotel to a traveller group or a flight-hotel-package to a traveller, a user to each other in a social network or a user to a game corporation for targeted advertising. Because of such distinct recommendation scenarios, the parameters need to be optimized accordingly even for the same model architecture, which is called finetuning. The optimization is generally extremely challenging, and it usually requires more training data for complex networks for a specific purpose. In addition, due to the huge amount of training set and cost, people cannot perform low-cost retraining or fine-tuning for different purposes. Therefore, considering the increasingly complex heterogeneous networks, we believe that a highly adaptable model structure and training algorithm is urgent in the community of user modelling and has extremely high research value.

Bibliography

[1]. Fischer, G., 2001. User modeling in human–computer interaction. User modeling and user-adapted interaction, 11(1), pp.65-86.

[2]. Chen, W.H., Hsu, C.C., Lai, Y.A., Liu, V., Yeh, M.Y. and Lin, S.D., 2020. Attribute-Aware Recommender System Based on Collaborative Filtering: Survey and Classification. Frontiers in Big Data, 2, p.49.

[3]. Zang, C., Cui, P. and Faloutsos, C., 2016, August. Beyond sigmoids: The nettide model for social network growth, and its applications. In Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining (pp. 2015-2024).

[4] Joachims, T., 1996. A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization. Carnegie-mellon univ pittsburgh pa dept of computer science.

[5] Blei, D.M., Ng, A.Y. and Jordan, M.I., 2003. Latent dirichlet allocation. the Journal of machine Learning research, 3, pp.993-1022.

[6] Blei, D.M. and Lafferty, J.D., 2006, June. Dynamic topic models. In Proceedings of the 23rd international conference on Machine learning (pp. 113-120).

[7] Bergland, G.D., 1969. A guided tour of the fast Fourier transform. IEEE spectrum, 6(7), pp.41-52.

[8] Cooley, J.W. and Tukey, J.W., 1965. An algorithm for the machine calculation of complex Fourier series. Mathematics of computation, 19(90), pp.297-301.

[9] Khater, S., Elmongui, H.G. and Gracanin, D., 2014, December. Tweets you like: Personalized tweets recommendation based on dynamic users interests. In Proc. Int. Conf. Social Comput.(SocialCom) (pp. 14-15).

[10] Yin, H., Cui, B., Chen, L., Hu, Z. and Zhou, X., 2015. Dynamic user modeling in social media systems. ACM Transactions on Information Systems (TOIS), 33(3), pp.1-44.

[11] Abel, F., Gao, Q., Houben, G.J. and Tao, K., 2011, July. Analyzing user modeling on twitter for personalized news recommendations. In international conference on user modeling, adaptation, and personalization (pp. 1-12). Springer, Berlin, Heidelberg.

[12] Abel, F., Gao, Q., Houben, G.J. and Tao, K., 2011, June. Analyzing temporal dynamics

in twitter profiles for personalized recommendations in the social web. In Proceedings of the 3rd international web science conference (pp. 1-8).

[13] Ahmed, A., Low, Y., Aly, M., Josifovski, V. and Smola, A.J., 2011, August. Scalable distributed inference of dynamic user interests for behavioral targeting. In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 114-122).

[14] Ke Tao, Fabian Abel, Qi Gao, and Geert-Jan Houben. 2011. Tums: twitter-based user modeling service. In Extended Semantic Web Conference. Springer, 269–283.

[15] Longfei Wu and Nianlong Luo. 2014. Social streams recommendation in sina microblog with relation of user and interest. In Information Science and Technology (ICIST), 2014 4th IEEE International Conference on. IEEE, 480–483.

[16] Kira Radinsky, Krysta Svore, Susan Dumais, Jaime Teevan, Alex Bocharov, and Eric Horvitz. 2012. Modeling and predicting behavioral dynamics on the web. In Proceedings of the 21st international conference on World Wide Web. ACM, 599–608.

[17] Sarah Masud Preum, John A Stankovic, and Yanjun Qi. 2015. Maper: A multiscale adaptive personalized model for temporal human behavior prediction. In Proceedings of the 24th ACM International on Conference on Information and Knowledge Management. ACM, 433–442.

[18] Yongfeng Zhang, Min Zhang, Yi Zhang, Guokun Lai, Yiqun Liu, Honghui Zhang, and Shaoping Ma. 2015. Daily-aware personalized recommendation based on feature-level time series analysis. In Proceedings of the 24th international conference on world wide web. ACM, 1373–1383.

[19] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In Proceedings of the 19th international conference on World wide web. ACM, 661–670.

[20] Abhinandan S Das, Mayur Datar, Ashutosh Garg, and Shyam Rajaram. 2007. Google news personalization: scalable online collaborative filtering. In Proceedings of the 16th international conference on World Wide Web. ACM, 271–280.

[21] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th international

conference on World Wide Web. ACM, 285–295.

[22] Thomas Hofmann. 2004. Latent semantic models for collaborative filtering. ACM Transactions on Information Systems (TOIS) 22, 1 (2004), 89–115.

[23] James H Martin and Daniel Jurafsky. 2000. Speech and language processing. International Edition 710 (2000), 25.

[24] Junyang Rao, Aixia Jia, Yansong Feng, and Dongyan Zhao. 2013. Personalized news recommendation using ontologies harvested from the web. In International Conference on Web-Age Information Management. Springer, 781–787.

[25] Liu, Jiahui, Peter Dolan, and Elin Rønby Pedersen. 2010. Personalized news recommendation based on click behavior. In Proceedings of the 15th international conference on Intelligent user interfaces. ACM, 31-40.

[26] Lian, Jianxun, Fuzheng Zhang, Xing Xie, and Guangzhong Sun. 2018. Towards Better Representation Learning for Personalized News Recommendation: a Multi-Channel Deep Fusion Approach. In IJCAI, 3805-3811.

[27] Okura, Shumpei, Yukihiro Tagami, Shingo Ono, and Akira Tajima. 2017. Embedding-based news recommendation for millions of users. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 1933-1942.

[28] Zheng, Guanjie, Fuzheng Zhang, Zihan Zheng, Yang Xiang, Nicholas Jing Yuan, Xing Xie, and Zhenhui Li. 2018. DRN: A deep reinforcement learning framework for news recommendation. In Proceedings of the 2018 World Wide Web Conference. ACM, 167-176.

[29] Wu, Chuhan, Fangzhao Wu, Suyu Ge, Tao Qi, Yongfeng Huang, and Xing Xie. 2019. Neural news recommendation with multi-head self-attention. 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), 6390-6395.

[30] Wu, Chuhan, Fangzhao Wu, Mingxiao An, Tao Qi, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019. Neural news recommendation with heterogeneous user behavior. 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), 4876-4885.

[31] Wu, Chuhan, Fangzhao Wu, Mingxiao An, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019. NPA: neural news recommendation with personalized attention. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM, 2576-2584.

[32] Wang, Hongwei, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. DKN: Deep knowledge-aware network for news recommendation. In Proceedings of the 2018 world wide web conference. ACM, 1835-1844.

[33] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. Computer 42, 8 (2009).

[34] Andriy Mnih and Ruslan R Salakhutdinov. 2008. Probabilistic matrix factorization. In Advances in neural information processing systems. 1257–1264.

[35] Amr Ahmed, Choon Hui Teo, SVN Vishwanathan, and Alex Smola. 2012. Fair and balanced: Learning to present news stories. In Proceedings of the fifth ACM international conference on Web search and data mining. ACM, 333–342.

[36] Figueiredo, Flavio, Bruno Ribeiro, Jussara M. Almeida, and Christos Faloutsos. "TribeFlow: Mining & predicting user trajectories." In Proceedings of the 25th international conference on world wide web, pp. 695-706. International World Wide Web Conferences Steering Committee, 2016.

[37] Shani, Guy, David Heckerman, Ronen I. Brafman, and Craig Boutilier. 2005. An MDP-based recommender system. Journal of Machine Learning Research, 6(9).

[38] Vasile, Flavian, Elena Smirnova, and Alexis Conneau. "Meta-prod2vec: Product embeddings using side-information for recommendation." In Proceedings of the 10th ACM Conference on Recommender Systems, pp. 225-232. ACM, 2016.

[39] Hidasi, Balázs, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. arXiv preprint arXiv:1511.06939 (2015).

[40] Tan, Yong Kiam, Xinxing Xu, and Yong Liu. 2016. Improved recurrent neural networks for session-based recommendations. In Proceedings of the 1st workshop on deep

learning for recommender systems, 17-22.

[41] Hidasi, Balázs, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. Parallel recurrent neural network architectures for feature-rich session-based recommendations. In Proceedings of the 10th ACM conference on recommender systems, pp. 241-248. ACM, 2016.

[42] Jannach, Dietmar, and Malte Ludewig. 2017. When recurrent neural networks meet the neighborhood for session-based recommendation. In Proceedings of the Eleventh ACM Conference on Recommender Systems. ACM, 306-310.

[43] Li, Jing, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural attentive session-based recommendation. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. ACM, 1419-1428.

[44] Liu, Qiao, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. STAMP: short-term attention/memory priority model for session-based recommendation. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM, 1831-1839.

[45] Quadrana, Massimo, Alexandros Karatzoglou, Balázs Hidasi, and Paolo Cremonesi, 2017. Personalizing session-based recommendations with hierarchical recurrent neural networks. In proceedings of the Eleventh ACM Conference on Recommender Systems. ACM, 130-137.

[46] Wu, Shu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-based recommendation with graph neural networks. In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, no. 01, pp. 346-353.

[47] Xu, Chengfeng, Pengpeng Zhao, Yanchi Liu, Victor S. Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. 2019. Graph Contextualized Self-Attention Network for Session-based Recommendation. In IJCAI, vol. 19, pp. 3940-3946.

[48] Yu, Feng, Yanqiao Zhu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2020. TAGNN: Target attentive graph neural networks for session-based recommendation. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 1921-1924.

[49] GRAD-GYENGE, László; FILZMOSER, Peter; WERTHNER, Hannes. 2015.

Recommendations on a knowledge graph. In: 1st International Workshop on Machine Learning Methods for Recommender Systems, MLRec, 13-20.

[50] M Ross Quillan. Semantic memory. In Semantic Information Processing, pages 227-270. MIT Press, Cambridge, MA, 1968.

[51] Catherine, Rose; Cohen, William. Personalized recommendations using knowledge graphs: A probabilistic logic programming approach. In: Proceedings of the 10th ACM Conference on Recommender Systems. ACM, 2016. p. 325-332.

[52] Chaudhari, Sneha; Azaria, Amos; Mitchell, Tom. An entity graph based recommender system. AI Communications, 2017, 30.2: 141-149.

[53] Xiao Yu, Xiang Ren, Yizhou Sun, Bradley Sturt, Urvashi Khandelwal, Quanquan Gu, Brandon Norick, and Jiawei Han. Recommendation in heterogeneous information networks with implicit user feedback. In RecSys, pages 347-350. ACM, 2013.

[54] Chen Luo, Wei Pang, Zhe Wang, and Chenghua Lin. Hete-cf: Social-based collaborative filtering recommendation using heterogeneous relations. In: 2014 IEEE International Conference on Data Mining. IEEE, 2014. p. 917-922.

[55] Zhang, F., Yuan, N. J., Lian, D., Xie, X., & Ma, W. Y. Collaborative knowledge base embedding for recommender systems. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. ACM, 2016. p. 353-362.

[56] Huang, Jin, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y. Chang. Improving sequential recommendation with knowledge-enhanced memory networks. In The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, pp. 505-514. ACM, 2018.

[57] Wang, Xiang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. Explainable reasoning over knowledge graphs for recommendation. In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, no. 01, pp. 5329-5336. 2019.

[58] Sun, Zhu, Jie Yang, Jie Zhang, Alessandro Bozzon, Long-Kai Huang, and Chi Xu. Recurrent knowledge graph embedding for effective recommendation. In Proceedings of the 12th ACM Conference on Recommender Systems, pp. 297-305. 2018.

[59] Zhu, Qiannan, Xiaofei Zhou, Jia Wu, Jianlong Tan, and Li Guo. A knowledge-aware

attentional reasoning network for recommendation. In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 04, pp. 6999-7006. 2020.

[60] Wang, X., Huang, T., Wang, D., Yuan, Y., Liu, Z., He, X. and Chua, T.S. Learning Intents behind Interactions with Knowledge Graph for Recommendation. In Proceedings of the Web Conference 2021, pp. 878-887. ACM, 2021.

[61]. Powers, D.M., 2020. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. arXiv preprint arXiv:2010.16061.

[62]. Voorhees, E.M., 1999, November. The TREC-8 question answering track report. In Trec (Vol. 99, pp. 77-82).

[63] Gulla, J.A., Zhang, L., Liu, P., Özgöbek, Ö. and Su, X., 2017, August. The adressa dataset for news recommendation. In Proceedings of the international conference on web intelligence (pp. 1042-1048).

[64] Schedl, M., 2016, June. The lfm-1b dataset for music retrieval and recommendation. In Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval (pp. 103-110).

[65] Zhang, J., Liu, B., Tang, J., Chen, T. and Li, J., 2013, August. Social influence locality for modeling retweeting behaviors. In IJCAI (Vol. 13, pp. 2761-2767).

[66] Kille, B., Hopfgartner, F., Brodt, T. and Heintz, T., 2013, October. The plista dataset. In Proceedings of the 2013 international news recommender systems workshop and challenge (pp. 16-23).

[67] Xiang, Zheng, and Ulrike Gretzel. Role of social media in online travel information search. Tourism management 31, no. 2 (2010): 179-188.

[68] Baltrunas, Linas, and Xavier Amatriain. Towards time-dependant recommendation based on implicit feedback. In Workshop on context-aware recommender systems (CARS'09), pp. 25-30. 2009.

[69] Panniello, Umberto, Alexander Tuzhilin, Michele Gorgoglione, Cosimo Palmisano, and Anto Pedone. Experimental comparison of pre-vs. post-filtering approaches in context-aware recommender systems. In Proceedings of the third ACM conference on Recommender systems, pp. 265-268. 2009.

[70] Michelson, Matthew, and Sofus A. Macskassy. Discovering users' topics of interest on twitter: a first look. In Proceedings of the fourth workshop on Analytics for noisy unstructured text data, pp. 73-80. 2010.

[71] Stoyanovich, Julia, Sihem Amer-Yahia, Cameron Marlow, and Cong Yu. Leveraging Tagging to Model User Interests in del. icio. us. In AAAI Spring Symposium: Social Information Processing, pp. 104-109. 2008.

[72] Wen, Zhen, and Ching-Yung Lin. On the quality of inferring interests from social neighbors. In Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 373-382. 2010.

[73] Chua, Freddy Chong Tat, Richard J. Oentaryo, and Ee-Peng Lim. Modeling temporal adoptions using dynamic matrix factorization. In 2013 IEEE 13th International Conference on Data Mining, pp. 91-100. IEEE, 2013.

[74] Gao, Rui, Bibo Hao, Shuotian Bai, Lin Li, Ang Li, and Tingshao Zhu. Improving user profile with personality traits predicted from social media content. In Proceedings of the 7th ACM conference on recommender systems, pp. 355-358. 2013.

[75] Li, Lei, Li Zheng, Fan Yang, and Tao Li. Modeling and broadening temporal user interest in personalized news recommendation. Expert Systems with Applications 41, no. 7 (2014): 3168-3177.

[76] Nazerfard, Ehsan, and Diane J. Cook. Using Bayesian Networks for Daily Activity Prediction. In AAAI workshop: plan, activity, and intent recognition. 2013.

[77] Yin, Hongzhi, Bin Cui, Ling Chen, Zhiting Hu, and Zi Huang. A temporal context-aware model for user behavior modeling in social media systems. In Proceedings of the 2014 ACM SIGMOD international conference on Management of data, pp. 1543-1554. 2014.

[78] Paulescu, Marius, Eugenia Paulescu, Paul Gravila, and Viorel Badescu. Weather modeling and forecasting of PV systems operation. Springer Science & Business Media, 2012.

[79] Xu, Zhiheng, Yang Zhang, Yao Wu, and Qing Yang. Modeling user posting behavior on social media. In Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval, pp. 545-554. 2012.

[80] Li, Chenliang, Aixin Sun, and Anwitaman Datta. Twevent: segment-based event detection from tweets. In Proceedings of the 21st ACM international conference on Information and knowledge management, pp. 155-164. 2012.

[81] Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. nature 323, no. 6088 (1986): 533-536.

[82] Epure, Elena Viorica, Benjamin Kille, Jon Espen Ingvaldsen, Rebecca Deneckere, Camille Salinesi, and Sahin Albayrak. Recommending personalized news in short user sessions. In Proceedings of the Eleventh ACM Conference on Recommender Systems, pp. 121-129. 2017.

[83] Chung, Junyoung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. In NIPS 2014 Workshop on Deep Learning, December 2014. 2014.

[84] Graves, Alex. Generating sequences with recurrent neural networks. arXiv preprint arXiv:1308.0850 (2013).

[85] Zhang, Xiang, Junbo Zhao, and Yann Lecun. Character-level convolutional networks for text classification. Advances in Neural Information Processing Systems 2015 (2015): 649-657.

[86] Kim, Yoon, Yacine Jernite, David Sontag, and Alexander Rush. Character-aware neural language models. In Proceedings of the AAAI conference on artificial intelligence, vol. 30, no. 1. 2016.

[87] Tuan, Trinh Xuan, and Tu Minh Phuong. 3D convolutional networks for session-based recommendation with content features. In Proceedings of the eleventh ACM conference on recommender systems, pp. 138-146. 2017.

[88] Lei, Tao, Regina Barzilay, and Tommi Jaakkola. Molding CNNs for text: non-linear, non-consecutive convolutions. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 1565-1575. 2015.

[89] Rendle, Steffen, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, pp. 452-461. 2009.

[90] Zheleva, Elena, John Guiver, Eduarda Mendes Rodrigues, and Nataša Milić-Frayling.

Statistical models of music-listening sessions in social media. In Proceedings of the 19th international conference on World wide web, pp. 1019-1028. 2010.

[91] Baur, Dominikus, Jennifer Büttgen, and Andreas Butz. Listening factors: A large-scale principal components analysis of long-term music listening histories. In Proceedings of the SIGCHI conference on human factors in computing systems, pp. 1273-1276. 2012.

[92] Adomavicius, Gediminas, and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE transactions on knowledge and data engineering 17, no. 6 (2005): 734-749.

[93] Su, Xiaoyuan, and Taghi M. Khoshgoftaar. A survey of collaborative filtering techniques. Advances in artificial intelligence 2009. 2009.

[94] Weimer, Markus, Alexandros Karatzoglou, Quoc Le, and Alex Smola. Cofirank-maximum margin matrix factorization for collaborative ranking. In Advances in Neural Information Processing Systems, 21st Annual Conference on Neural Information Processing Systems 2007, pp. 222-230. 2007.

[95] Schafer, J. Ben, Joseph Konstan, and John Riedl. Recommender systems in e-commerce. In Proceedings of the 1st ACM conference on Electronic commerce, pp. 158-166. 1999.

[96] Kim, Y. Convolutional neural networks for sentence classification. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1746–1751. 2014

[97] Agarwal, Deepak, and Bee-Chung Chen. Regression-based latent factor models. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 19-28. 2009.

[98] Melville, Prem, Raymond J. Mooney, and Ramadass Nagarajan. Content-boosted collaborative filtering for improved recommendations. Aaai/iaai 23 (2002): 187-192.

[99] Gopalan, Prem, Laurent Charlin, and David M. Blei. Content-based recommendations with Poisson factorization. In NIPS, vol. 14, pp. 3176-3184. 2014.

[100] Wang, Chong, and David M. Blei. Collaborative topic modeling for recommending scientific articles. In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 448-456. 2011.

[101] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems 25 (2012): 1097-1105.

[102] Tai, Kai Sheng, Richard Socher, and Christopher D. Manning. Improved semantic representations from tree-structured long short-term memory networks. arXiv preprint arXiv:1503.00075 (2015).

[103] Socher, Richard, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In Proceedings of the 2013 conference on empirical methods in natural language processing, pp. 1631-1642. 2013.

[104] Lai, Siwei, Liheng Xu, Kang Liu, and Jun Zhao. Recurrent convolutional neural networks for text classification. In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 29, no. 1. 2015.

[105] Hong, James, and Michael Fang. Sentiment analysis with deeply learned distributed representations of variable length texts. Stanford University Report (2015): 1-9.

[106] Wu, Chen, and Ming Yan. Session-aware information embedding for e-commerce product recommendation. In Proceedings of the 2017 ACM on conference on information and knowledge management, pp. 2379-2382. 2017.

[107] Cho, Kyunghyun, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259 (2014).

[108] Pan, Yingwei, Tao Mei, Ting Yao, Houqiang Li, and Yong Rui. Jointly modeling embedding and translation to bridge video and language. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4594-4602. 2016.

[109] Cui, Peng, Zhiyu Wang, and Zhou Su. What videos are similar with you? learning a common attributed representation for video recommendation. In Proceedings of the 22nd ACM international conference on Multimedia, pp. 597-606. 2014.

[110] Ding, Yi, and Xue Li. Time weight collaborative filtering. In Proceedings of the 14th ACM international conference on Information and knowledge management, pp. 485-492. 2005.

[111] Greene, Derek, and James P. Cross. Exploring the political agenda of the european parliament using a dynamic topic modeling approach. Political Analysis 25, no. 1 (2017): 77-94.

[112] Zhang, Jing, Biao Liu, Jie Tang, Ting Chen, and Juanzi Li. Social influence locality for modeling retweeting behaviors. In IJCAI, vol. 13, pp. 2761-2767. 2013.

[113] Gori, Marco, Augusto Pucci, V. Roma, and I. Siena. Itemrank: A random-walk based scoring algorithm for recommender engines. In IJCAI, vol. 7, pp. 2766-2771. 2007.

[114] Nikolakopoulos, Athanasios N., and George Karypis. Recwalk: Nearly uncoupled random walks for top-n recommendation. In Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, pp. 150-158. 2019.

[115] He, Xiangnan, Tao Chen, Min-Yen Kan, and Xiao Chen. Trirank: Review-aware explainable recommendation by modeling aspects. In Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, pp. 1661-1670. 2015.

[116] Wang, Shoujin, Liang Hu, Yan Wang, Xiangnan He, Quan Z. Sheng, Mehmet Orgun, Longbing Cao, Nan Wang, Francesco Ricci, and Philip S. Yu. Graph Learning Approaches to Recommender Systems: A Review. arXiv preprint arXiv:2004.11718 (2020).

[117] Menon, Aditya Krishna, Krishna-Prasad Chitrapura, Sachin Garg, Deepak Agarwal, and Nagaraj Kota. Response prediction using collaborative filtering with hierarchies and side-information. In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 141-149. 2011.

[118] Koenigstein, Noam, Gideon Dror, and Yehuda Koren. Yahoo! music recommendations: modeling music ratings with temporal dynamics and item taxonomy. In Proceedings of the fifth ACM conference on Recommender systems, pp. 165-172. 2011.

[119] Mnih, Andriy. Taxonomy-informed latent factor models for implicit feedback. In Proceedings of KDD Cup 2011, pp. 169-181. PMLR, 2012.

[120] Kanagal, Bhargav, Amr Ahmed, Sandeep Pandey, Vanja Josifovski, Jeff Yuan, and Lluis Garcia-Pueyo. Supercharging recommender systems using taxonomies for learning user purchase behavior. arXiv preprint arXiv:1207.0136 (2012).

[121] He, Ruining, Chunbin Lin, Jianguo Wang, and Julian McAuley. Sherlock: sparse

hierarchical embeddings for visually-aware one-class collaborative filtering. arXiv preprint arXiv:1604.05813 (2016).

[122] Yang, Jie, Zhu Sun, Alessandro Bozzon, and Jie Zhang. Learning hierarchical feature influence for recommendation by recursive regularization. In Proceedings of the 10th ACM Conference on Recommender Systems, pp. 51-58. 2016.

[123] Sun, Zhu, Jie Yang, Jie Zhang, and Alessandro Bozzon. Exploiting both vertical and horizontal dimensions of feature hierarchy for effective recommendation. In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31, no. 1. 2017.

[124] McAuley, Julian, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. Image-based recommendations on styles and substitutes. In Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval, pp. 43-52. 2015.

[125] Liu, Xin, Yong Liu, Karl Aberer, and Chunyan Miao. Personalized point-of-interest recommendation by mining users' preference transition. In Proceedings of the 22nd ACM international conference on Information & Knowledge Management, pp. 733-738. 2013.

[126] Wang, Chenyang, Min Zhang, Weizhi Ma, Yiqun Liu, and Shaoping Ma. Make it a chorus: knowledge-and time-aware item modeling for sequential recommendation. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 109-118. 2020.

[127] Song, Weiping, Zhiping Xiao, Yifan Wang, Laurent Charlin, Ming Zhang, and Jian Tang. Session-based social recommendation via dynamic graph attention networks. In Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, pp. 555-563. 2019.

[128] Tay, David BH, and Zhiping Lin. Design of near orthogonal graph filter banks. IEEE Signal Processing Letters 22, no. 6 (2014): 701-704.

[129] Ying, Rex, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 974-983. 2018.

[130] Sun, Yizhou, and Jiawei Han. Mining heterogeneous information networks: a

structural analysis approach. Acm Sigkdd Explorations Newsletter 14, no. 2 (2013): 20-28.

[131] Shi, Chuan, Yitong Li, Jiawei Zhang, Yizhou Sun, and S. Yu Philip. A survey of heterogeneous information network analysis. IEEE Transactions on Knowledge and Data Engineering 29, no. 1 (2016): 17-37.

[132] Li, Xiang, Yao Wu, Martin Ester, Ben Kao, Xin Wang, and Yudian Zheng. Semi-supervised clustering in attributed heterogeneous information networks. In Proceedings of the 26th International Conference on World Wide Web, pp. 1621-1629. 2017.

[133] Suchanek, Fabian M., Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In Proceedings of the 16th international conference on World Wide Web, pp. 697-706. 2007.

[134] Lehmann, Jens, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann et al. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. Semantic web 6, no. 2 (2015): 167-195.

[135] Agarwal, Sameer, Kristin Branson, and Serge Belongie. Higher order learning with graphs. In Proceedings of the 23rd international conference on Machine learning, pp. 17-24. 2006.

[136] Feng, Yifan, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, no. 01, pp. 3558-3565. 2019.

[137] Bollacker, Kurt, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In Proceedings of the 2008 ACM SIGMOD international conference on Management of data, pp. 1247-1250. 2008.

[138] Xu, Bo, Yong Xu, Jiaqing Liang, Chenhao Xie, Bin Liang, Wanyun Cui, and Yanghua Xiao. CN-DBpedia: A never-ending Chinese knowledge extraction system. In International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, pp. 428-438. Springer, Cham, 2017.

[139] Carlson, Andrew, Justin Betteridge, Richard C. Wang, Estevam R. Hruschka Jr, and Tom M. Mitchell. Coupled semi-supervised learning for information extraction. In Proceedings of the third ACM international conference on Web search and data mining,

pp. 101-110. 2010.

[140] Speer, Robyn, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31, no. 1. 2017.

[141] Pianta, Emanuele, Luisa Bentivogli, and Christian Girardi. MultiWordNet: developing an aligned multilingual database. In First international conference on global WordNet, pp. 293-302. 2002.

[142] Moro, Andrea, Alessandro Raganato, and Roberto Navigli. Entity linking meets word sense disambiguation: a unified approach. Transactions of the Association for Computational Linguistics 2 (2014): 231-244.

[143] Bond, Francis, and Kyonghee Paik. A survey of wordnets and their licenses. Small 8, no. 4 (2012): 5.

[144] Belleau, François, Marc-Alexandre Nolin, Nicole Tourigny, Philippe Rigault, and Jean Morissette. Bio2RDF: towards a mashup to build bioinformatics knowledge systems. Journal of biomedical informatics 41, no. 5 (2008): 706-716.

[145] Ernst, Patrick, Cynthia Meng, Amy Siu, and Gerhard Weikum. Knowlife: a knowledge graph for health and life sciences. In 2014 IEEE 30th International Conference on Data Engineering, pp. 1254-1257. IEEE, 2014.

[146] Yan, Hehua, Jun Yang, and Jiafu Wan. KnowIME: A System to Construct a Knowledge Graph for Intelligent Manufacturing Equipment. IEEE Access 8 (2020): 41805-41813.

[147] Farseev, Aleksandr, Ivan Samborskii, Andrey Filchenkov, and Tat-Seng Chua. Cross-domain recommendation via clustering on multi-layer graphs. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 195-204. 2017.

[148] Phuong, Tu Minh, and Nguyen Duy Phuong. Graph-based context-aware collaborative filtering. Expert Systems with Applications 126 (2019): 9-19.

[149] Zhao, Huan, Quanming Yao, Jianda Li, Yangqiu Song, and Dik Lun Lee. Meta-graph based recommendation fusion over heterogeneous information networks. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data

Mining, pp. 635-644. 2017.

[150] Gao, Jingyue, Xiting Wang, Yasha Wang, and Xing Xie. Explainable recommendation through attentive multi-view learning. In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, no. 01, pp. 3622-3629. 2019.

[151] Ma, Hao, Dengyong Zhou, Chao Liu, Michael R. Lyu, and Irwin King. Recommender systems with social regularization. In Proceedings of the fourth ACM international conference on Web search and data mining, pp. 287-296. 2011.

[152] Shi, Chuan, Zhiqiang Zhang, Ping Luo, Philip S. Yu, Yading Yue, and Bin Wu. Semantic path based personalized recommendation on weighted heterogeneous information networks. In Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, pp. 453-462. 2015.

[153] Yu, Xiao, Xiang Ren, Quanquan Gu, Yizhou Sun, and Jiawei Han. Collaborative filtering with entity similarity regularization in heterogeneous information networks. IJCAI HINA 27 (2013).

[154] Yu, Xiao, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. Personalized entity recommendation: A heterogeneous information network approach. In Proceedings of the 7th ACM international conference on Web search and data mining, pp. 283-292. 2014.

[155] Feng, Wei, and Jianyong Wang. Incorporating heterogeneous information for personalized tag recommendation in social tagging systems. In Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 1276-1284. 2012.

[156] Zheng, Jing, Jian Liu, Chuan Shi, Fuzhen Zhuang, Jingzhi Li, and Bin Wu. Recommendation in heterogeneous information network via dual similarity regularization. International Journal of Data Science and Analytics 3, no. 1 (2017): 35-48.

[157] Shi, Chuan, Jian Liu, Fuzhen Zhuang, S. Yu Philip, and Bin Wu. Integrating heterogeneous information via flexible regularization framework for recommendation. Knowledge and Information Systems 49, no. 3 (2016): 835-859.

[158] Catherine, Rose, Kathryn Mazaitis, Maxine Eskenazi, and William Cohen. Explainable entity-based recommendations with knowledge graphs. arXiv preprint

arXiv:1707.05254 (2017).

[159] Andersen, Reid, Christian Borgs, Jennifer Chayes, Uriel Feige, Abraham Flaxman, Adam Kalai, Vahab Mirrokni, and Moshe Tennenholtz. Trust-based recommendation systems: an axiomatic approach. In Proceedings of the 17th international conference on World Wide Web, pp. 199-208. 2008.

[160] Bagci, Hakan, and Pinar Karagoz. Context-aware friend recommendation for location based social networks using random walk. In Proceedings of the 25th international conference companion on world wide web, pp. 531-536. 2016.

[161] Jamali, Mohsen, and Martin Ester. Trustwalker: a random walk model for combining trust-based and item-based recommendation. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 397-406. 2009.

[162] Jiang, Zhengshen, Hongzhi Liu, Bin Fu, Zhonghai Wu, and Tao Zhang. Recommendation in heterogeneous information networks based on generalized random walk model and bayesian personalized ranking. In Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, pp. 288-296. 2018.

[163] Eksombatchai, Chantat, Pranav Jindal, Jerry Zitao Liu, Yuchen Liu, Rahul Sharma, Charles Sugnet, Mark Ulrich, and Jure Leskovec. Pixie: A system for recommending 3+ billion items to 200+ million users in real-time. In Proceedings of the 2018 world wide web conference, pp. 1775-1784. 2018.

[164] Sun, Yizhou, 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, no. 11 (2011): 992-1003.

[165] Cai, Hongyun, Vincent W. Zheng, and Kevin Chen-Chuan Chang. A comprehensive survey of graph embedding: Problems, techniques, and applications. IEEE Transactions on Knowledge and Data Engineering 30, no. 9 (2018): 1616-1637.

[166] Shi, Chuan, Binbin Hu, Wayne Xin Zhao, and S. Yu Philip. Heterogeneous information network embedding for recommendation. IEEE Transactions on Knowledge and Data Engineering 31, no. 2 (2018): 357-370.

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

[168] Perozzi, Bryan, 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. 2014.

[169] Tang, Jian, 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. 2015.

[170] Grover, Aditya, 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. 2016.

[171] Verma, Janu, Srishti Gupta, Debdoot Mukherjee, and Tanmoy Chakraborty. Heterogeneous edge embedding for friend recommendation. In European Conference on Information Retrieval, pp. 172-179. Springer, Cham, 2019.

[172] Palumbo, Enrico, Giuseppe Rizzo, and Raphaël Troncy. Entity2rec: Learning user-item relatedness from knowledge graphs for top-n item recommendation. In Proceedings of the eleventh ACM conference on recommender systems, pp. 32-36. 2017.

[173] Jiang, Zhuoren, Yue Yin, Liangcai Gao, Yao Lu, and Xiaozhong Liu. Cross-language citation recommendation via hierarchical representation learning on heterogeneous graph. In The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, pp. 635-644. 2018.

[174] Vijaikumar, M., Shirish Shevade, and M. Narasimha Murty. SoRecGAT: Leveraging Graph Attention Mechanism for Top-N Social Recommendation. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 430-446. Springer, Cham, 2019.

[175] Gao, Li, Hong Yang, Jia Wu, Chuan Zhou, Weixue Lu, and Yue Hu. Recommendation with multi-source heterogeneous information. In IJCAI International Joint Conference on Artificial Intelligence. 2018.

[176] Ali, Zafar, Guilin Qi, Khan Muhammad, Bahadar Ali, and Waheed Ahmed Abro. Paper recommendation based on heterogeneous network embedding. Knowledge-Based Systems 210 (2020): 106438.

[177] Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean.

Distributed representations of words and phrases and their compositionality. arXiv preprint arXiv:1310.4546 (2013).

[178] Bordes, Antoine, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In Neural Information Processing Systems (NIPS), pp. 1-9. 2013.

[179] Wang, Zhen, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 28, no. 1. 2014.

[180] Lin, Yankai, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 29, no. 1. 2015.

[181] Ai, Qingyao, Vahid Azizi, Xu Chen, and Yongfeng Zhang. Learning heterogeneous knowledge base embeddings for explainable recommendation. Algorithms 11, no. 9 (2018): 137.

[182] Chen, Yuanyi, Mingxuan Zhou, Zengwei Zheng, and Dan Chen. Time-aware smart object recommendation in social Internet of Things. IEEE Internet of Things Journal 7, no. 3 (2019): 2014-2027.

[183] Hochreiter, Sepp, and Jürgen Schmidhuber. Long short-term memory. Neural computation 9, no. 8 (1997): 1735-1780.

[184] Huang, Jin, Zhaochun Ren, Wayne Xin Zhao, Gaole He, Ji-Rong Wen, and Daxiang Dong. Taxonomy-aware multi-hop reasoning networks for sequential recommendation. In Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, pp. 573-581. 2019.

[185] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems 25 (2012): 1097-1105.

[186] Berg, Rianne van den, Thomas N. Kipf, and Max Welling. Graph convolutional matrix completion. arXiv preprint arXiv:1706.02263 (2017).

[187] Xu, Huance, Chao Huang, Yong Xu, Lianghao Xia, Hao Xing, and Dawei Yin. Global Context Enhanced Social Recommendation with Hierarchical Graph Neural

Networks. In 2020 IEEE International Conference on Data Mining (ICDM), pp. 701-710. IEEE, 2020.

[188] Tian, Fei, Bin Gao, Qing Cui, Enhong Chen, and Tie-Yan Liu. Learning deep representations for graph clustering. In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 28, no. 1. 2014.

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

[190] Han, Xiaotian, Chuan Shi, Senzhang Wang, S. Yu Philip, and Li Song. Aspect-Level Deep Collaborative Filtering via Heterogeneous Information Networks. In IJCAI, pp. 3393-3399. 2018.

[191] Wang, Pengfei, Hanxiong Chen, Yadong Zhu, Huawei Shen, and Yongfeng Zhang. Unified collaborative filtering over graph embeddings. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 155-164. 2019.

[192] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. arXiv preprint arXiv:1706.03762 (2017).

[193] Cen, Yukuo, Xu Zou, Jianwei Zhang, Hongxia Yang, Jingren Zhou, and Jie Tang. Representation learning for attributed multiplex heterogeneous network. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1358-1368. 2019.

[194] Zhang, Shuai, Lina Yao, Aixin Sun, and Yi Tay. Deep learning based recommender system: A survey and new perspectives. ACM Computing Surveys (CSUR) 52, no. 1 (2019): 1-38.

[195] Sun, Zhu, Qing Guo, Jie Yang, Hui Fang, Guibing Guo, Jie Zhang, and Robin Burke. Research commentary on recommendations with side information: A survey and research directions. Electronic Commerce Research and Applications 37 (2019): 100879.

[196] Xu, Fengli, Jianxun Lian, Zhenyu Han, Yong Li, Yujian Xu, and Xing Xie. Relation-aware graph convolutional networks for agent-initiated social e-commerce

recommendation. In Proceedings of the 28th ACM international conference on information and knowledge management, pp. 529-538. 2019.

[197] Liu, Siwei, Iadh Ounis, Craig Macdonald, and Zaiqiao Meng. A Heterogeneous Graph Neural Model for Cold-Start Recommendation. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 2029-2032. 2020.

[198] Reimers, Nils, and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. arXiv preprint arXiv:1908.10084 (2019).

[199] Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves et al. Human-level control through deep reinforcement learning. nature 518, no. 7540 (2015): 529-533.

[200] Song, Weiping, Zhijian Duan, Ziqing Yang, Hao Zhu, Ming Zhang, and Jian Tang. Explainable knowledge graph-based recommendation via deep reinforcement learning. arXiv preprint arXiv:1906.09506 (2019).

[201] Xian, Yikun, Zuohui Fu, S. Muthukrishnan, Gerard De Melo, and Yongfeng Zhang. Reinforcement knowledge graph reasoning for explainable recommendation. In Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval, pp. 285-294. 2019.

[202] Lei, Wenqiang, Gangyi Zhang, Xiangnan He, Yisong Miao, Xiang Wang, Liang Chen, and Tat-Seng Chua. Interactive path reasoning on graph for conversational recommendation. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2073-2083. 2020.

[203] Wang, Xiang, Yaokun Xu, Xiangnan He, Yixin Cao, Meng Wang, and Tat-Seng Chua. Reinforced negative sampling over knowledge graph for recommendation. In Proceedings of The Web Conference 2020, pp. 99-109. 2020.

[204] Zhou, Sijin, Xinyi Dai, Haokun Chen, Weinan Zhang, Kan Ren, Ruiming Tang, Xiuqiang He, and Yong Yu. Interactive recommender system via knowledge graph-enhanced reinforcement learning. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 179-188. 2020.

[205] Silver, David, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George

Van Den Driessche, Julian Schrittwieser et al. Mastering the game of Go with deep neural networks and tree search. nature 529, no. 7587 (2016): 484-489.

[206] Chen, Minmin, Alex Beutel, Paul Covington, Sagar Jain, Francois Belletti, and Ed H. Chi. Top-k off-policy correction for a REINFORCE recommender system. In Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, pp. 456-464. 2019.

[207] Fang, Hui, Danning Zhang, Yiheng Shu, and Guibing Guo. Deep Learning for Sequential Recommendation: Algorithms, Influential Factors, and Evaluations. ACM Transactions on Information Systems (TOIS) 39, no. 1 (2020): 1-42.

[208] Wu, Zonghan, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S. Yu Philip. A comprehensive survey on graph neural networks. IEEE transactions on neural networks and learning systems (2020).

[209] Wu, Shiwen, Wentao Zhang, Fei Sun, and Bin Cui. Graph Neural Networks in Recommender Systems: A Survey. arXiv preprint arXiv:2011.02260 (2020).

[210] Zheng, Lei, Chun-Ta Lu, Fei Jiang, Jiawei Zhang, and Philip S. Yu. Spectral collaborative filtering. In Proceedings of the 12th ACM conference on recommender systems, pp. 311-319. 2018.

[211] Wang, Chenyang, Weizhi Ma, Min Zhang, Chong Chen, Yiqun Liu, and Shaoping Ma. Toward Dynamic User Intention: Temporal Evolutionary Effects of Item Relations in Sequential Recommendation. ACM Transactions on Information Systems (TOIS) 39, no. 2 (2020): 1-33.

[212] Fan, Wenqi, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. Graph neural networks for social recommendation. In The World Wide Web Conference, pp. 417-426. 2019.

[213] Wang, Xiang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. Neural graph collaborative filtering. In Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval, pp. 165-174. 2019.

[214] Wang, Xiang, Hongye Jin, An Zhang, Xiangnan He, Tong Xu, and Tat-Seng Chua. Disentangled Graph Collaborative Filtering. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1001-1010.

2020.

[215] Wu, Le, Peijie Sun, Yanjie Fu, Richang Hong, Xiting Wang, and Meng Wang. A neural influence diffusion model for social recommendation. In Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval, pp. 235-244. 2019.

[216] Wu, Qitian, Hengrui Zhang, Xiaofeng Gao, Peng He, Paul Weng, Han Gao, and Guihai Chen. Dual graph attention networks for deep latent representation of multifaceted social effects in recommender systems. In The World Wide Web Conference, pp. 2091-2102. 2019.

[217] Wang, Wen, Wei Zhang, Jun Rao, Zhijie Qiu, Bo Zhang, Leyu Lin, and Hongyuan Zha. Group-Aware Long-and Short-Term Graph Representation Learning for Sequential Group Recommendation. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1449-1458. 2020.

[218] Sheu, Heng-Shiou, and Sheng Li. Context-aware graph embedding for session-based news recommendation. In Fourteenth ACM conference on recommender systems, pp. 657-662. 2020.

[219] Wang, Jianling, Kaize Ding, Liangjie Hong, Huan Liu, and James Caverlee. Next-item recommendation with sequential hypergraphs. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1101-1110. 2020.

[220] Wang, Hongwei, Miao Zhao, Xing Xie, Wenjie Li, and Minyi Guo. Knowledge graph convolutional networks for recommender systems. In The world wide web conference, pp. 3307-3313. 2019.

[221] Wang, Hongwei, Fuzheng Zhang, Mengdi Zhang, Jure Leskovec, Miao Zhao, Wenjie Li, and Zhongyuan Wang. Knowledge-aware graph neural networks with label smoothness regularization for recommender systems. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 968-977. 2019.

[222] Wang, Xiang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. Kgat: Knowledge graph attention network for recommendation. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 950-958. 2019.

[223] Fan, Shaohua, Junxiong Zhu, Xiaotian Han, Chuan Shi, Linmei Hu, Biyu Ma, and Yongliang Li. Metapath-guided heterogeneous graph neural network for intent recommendation. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2478-2486. 2019.

[224] Zhao, Jun, Zhou Zhou, Ziyu Guan, Wei Zhao, Wei Ning, Guang Qiu, and Xiaofei He. Intentgc: a scalable graph convolution framework fusing heterogeneous information for recommendation. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2347-2357. 2019.

[225] Yang, Zuoxi, and Shoubin Dong. HAGERec: hierarchical attention graph convolutional network incorporating knowledge graph for explainable recommendation. Knowledge-Based Systems 204 (2020): 106194.

[226] Abugabah, Ahed, Xiaochun Cheng, and Jianfeng Wang. Dynamic Graph Attention-Aware Networks for Session-Based Recommendation. In 2020 International Joint Conference on Neural Networks (IJCNN), pp. 1-7. IEEE, 2020.

[227] Zhou, Kun, Wayne Xin Zhao, Shuqing Bian, Yuanhang Zhou, Ji-Rong Wen, and Jingsong Yu. Improving conversational recommender systems via knowledge graph based semantic fusion. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1006-1014. 2020.

[228] Shuman, David I., Sunil K. Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. IEEE signal processing magazine 30, no. 3 (2013): 83-98.

[229] Monti, Federico, Michael M. Bronstein, and Xavier Bresson. Geometric matrix completion with recurrent multi-graph neural networks. arXiv preprint arXiv:1704.06803 (2017).

[230] Chen, Jiawei, Can Wang, Sheng Zhou, Qihao Shi, Yan Feng, and Chun Chen. Samwalker: Social recommendation with informative sampling strategy. In The World Wide Web Conference, pp. 228-239. 2019.

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

[232] Liu, Ziqi, Chaochao Chen, Longfei Li, Jun Zhou, Xiaolong Li, Le Song, and Yuan Qi. Geniepath: Graph neural networks with adaptive receptive paths. In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, no. 01, pp. 4424-4431. 2019.

[233] Li, Qimai, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, no. 1. 2018.

[234] Schuster, Mike, and Kuldip K. Paliwal. Bidirectional recurrent neural networks. IEEE transactions on Signal Processing 45, no. 11 (1997): 2673-2681.

[235] McNee, Sean M., John Riedl, and Joseph A. Konstan. Being accurate is not enough: how accuracy metrics have hurt recommender systems. In CHI'06 extended abstracts on Human factors in computing systems, pp. 1097-1101. 2006.

[236] Fu, Zuohui, Yikun Xian, Ruoyuan Gao, Jieyu Zhao, Qiaoying Huang, Yingqiang Ge, Shuyuan Xu et al. Fairness-aware explainable recommendation over knowledge graphs. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 69-78. 2020.

[237] Sun, Jianing, Wei Guo, Dengcheng Zhang, Yingxue Zhang, Florence Regol, Yaochen Hu, Huifeng Guo et al. A framework for recommending accurate and diverse items using bayesian graph convolutional neural networks. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2030-2039. 2020.

[238] Zhang, Yingxue, Soumyasundar Pal, Mark Coates, and Deniz Ustebay. Bayesian graph convolutional neural networks for semi-supervised classification. In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, no. 01, pp. 5829-5836. 2019.

[239] Isufi, Elvin, Matteo Pocchiari, and Alan Hanjalic. Accuracy-diversity trade-off in recommender systems via graph convolutions. Information Processing & Management 58, no. 2 (2021): 102459.

[240] Brown, Tom B., Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan et al. Language models are few-shot learners. arXiv preprint arXiv:2005.14165 (2020).

[241] Hamilton, William L., Rex Ying, and Jure Leskovec. Inductive representation

learning on large graphs. arXiv preprint arXiv:1706.02216 (2017).

[242] Wang, Alex, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. arXiv preprint arXiv:1804.07461 (2018).

[243] Farnadi, Golnoosh, Pigi Kouki, Spencer K. Thompson, Sriram Srinivasan, and Lise Getoor. A fairness-aware hybrid recommender system. arXiv preprint arXiv:1809.09030 (2018).

[244] He, Ruining, 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, pp. 507-517. 2016.

[245] Wang, Xiao, Ruijia Wang, Chuan Shi, Guojie Song, and Qingyong Li. Multi-component graph convolutional collaborative filtering. In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 04, pp. 6267-6274. 2020.

[246] Chen, Lei, Le Wu, Richang Hong, Kun Zhang, and Meng Wang. Revisiting graph based collaborative filtering: A linear residual graph convolutional network approach. In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 01, pp. 27-34. 2020.

[247] Ma, Weizhi, Min Zhang, Yue Cao, Woojeong Jin, Chenyang Wang, Yiqun Liu, Shaoping Ma, and Xiang Ren. Jointly learning explainable rules for recommendation with knowledge graph. In The World Wide Web Conference, pp. 1210-1221. 2019.

[248] Kyriakidi, Marialena, Georgia Koutrika, and Yannis Ioannidis. Recommendations as Graph Explorations. In Fourteenth ACM Conference on Recommender Systems, pp. 289-298. 2020.

[249] Lu, Yuanfu, Yuan Fang, and Chuan Shi. Meta-learning on heterogeneous information networks for cold-start recommendation. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1563-1573. 2020.

[250] Tang, Jiliang, Huiji Gao, and Huan Liu. mTrust: Discerning multi-faceted trust in a connected world. In Proceedings of the fifth ACM international conference on Web search and data mining, pp. 93-102. 2012.

[251] Massa, Paolo, and Paolo Avesani. Trust-aware recommender systems. In

Proceedings of the 2007 ACM conference on Recommender systems, pp. 17-24. 2007.

[252] Zhao, Tong, Julian McAuley, and Irwin King. Leveraging social connections to improve personalized ranking for collaborative filtering. In Proceedings of the 23rd ACM international conference on conference on information and knowledge management, pp. 261-270. 2014.

[253] Richardson, Matthew, and Pedro Domingos. Mining knowledge-sharing sites for viral marketing. In Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 61-70. 2002.

[254] Wang, Jinpeng, Wayne Zhao, Yulan He, and Xiaoming Li. Leveraging product adopter information from online reviews for product recommendation. In Proceedings of the International AAAI Conference on Web and Social Media, vol. 9, no. 1. 2015.

[255] Huang, Zan, Daniel D. Zeng, and Hsinchun Chen. Analyzing consumer-product graphs: Empirical findings and applications in recommender systems. Management science 53, no. 7 (2007): 1146-1164.

[256] Li, Xin, and Hsinchun Chen. Recommendation as link prediction in bipartite graphs: A graph kernel-based machine learning approach. Decision Support Systems 54, no. 2 (2013): 880-890.

[257] Huang, Zan, Daniel Zeng, and Hsinchun Chen. A comparative study of recommendation algorithms in e-commerce applications. IEEE Intelligent Systems 22, no. 5 (2007): 68-78.

[258] Cantador, Iván, Peter Brusilovsky, and Tsvi Kuflik. Second workshop on information heterogeneity and fusion in recommender systems (HetRec2011). In Proceedings of the fifth ACM conference on Recommender systems, pp. 387-388. 2011.

[259] Harper, F. Maxwell, and Joseph A. Konstan. The movielens datasets: History and context. Acm transactions on interactive intelligent systems (tiis) 5, no. 4 (2015): 1-19.

[260] Wang, Hongwei, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 417-426. 2018.

[261] Wang, Hongwei, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. Exploring high-order user preference on the knowledge graph for

recommender systems. ACM Transactions on Information Systems (TOIS) 37, no. 3 (2019): 1-26.

[262] Xin, Xin, Xiangnan He, Yongfeng Zhang, Yongdong Zhang, and Joemon Jose. Relational collaborative filtering: Modeling multiple item relations for recommendation. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 125-134. 2019.

[263] Cheng, Haibin, Pang-Ning Tan, Jon Sticklen, and William F. Punch. Recommendation via query centered random walk on k-partite graph. In Seventh IEEE International Conference on Data Mining (ICDM 2007), pp. 457-462. IEEE, 2007.

[264] Yao, Weilong, Jing He, Guangyan Huang, Jie Cao, and Yanchun Zhang. A graph-based model for context-aware recommendation using implicit feedback data. World wide web 18, no. 5 (2015): 1351-1371.

[265] Monti, Federico, Michael M. Bronstein, and Xavier Bresson. Geometric matrix completion with recurrent multi-graph neural networks. arXiv preprint arXiv:1704.06803 (2017).

[266] Zheng, Yong, Bamshad Mobasher, and Robin Burke. Carskit: A java-based context-aware recommendation engine. In 2015 IEEE International Conference on Data Mining Workshop (ICDMW), pp. 1668-1671. IEEE, 2015.

[267] Yin, Zhijun, Manish Gupta, Tim Weninger, and Jiawei Han. Linkrec: a unified framework for link recommendation with user attributes and graph structure. In Proceedings of the 19th international conference on World wide web, pp. 1211-1212. 2010.

[268] Tang, Lei, Xufei Wang, and Huan Liu. Uncoverning groups via heterogeneous interaction analysis. In 2009 Ninth IEEE International Conference on Data Mining, pp. 503-512. IEEE, 2009.

[269] Sharma, Aneesh, Jerry Jiang, Praveen Bommannavar, Brian Larson, and Jimmy Lin. GraphJet: Real-time content recommendations at Twitter. Proceedings of the VLDB Endowment 9, no. 13 (2016): 1281-1292.

[270] De Domenico, Manlio, Antonio Lima, Paul Mougel, and Mirco Musolesi. The anatomy of a scientific rumor. Scientific reports 3, no. 1 (2013): 1-9.

[271] Ziegler, Cai-Nicolas, Sean M. McNee, Joseph A. Konstan, and Georg Lausen.

Improving recommendation lists through topic diversification. In Proceedings of the 14th international conference on World Wide Web, pp. 22-32. 2005.

[272] Cao, Yixin, Xiang Wang, Xiangnan He, Zikun Hu, and Tat-Seng Chua. Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences. In The world wide web conference, pp. 151-161. 2019.

[273] Tang, Jie, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. Arnetminer: extraction and mining of academic social networks. In Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 990-998. 2008.

[274] Purushotham, Sanjay, Yan Liu, and C-C. Jay Kuo. Collaborative topic regression with social matrix factorization for recommendation systems. arXiv preprint arXiv:1206.4684 (2012).

[275] Wang, Jianling, and James Caverlee. Recurrent recommendation with local coherence. In Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, pp. 564-572. 2019.

[276] Radev, Dragomir R., Pradeep Muthukrishnan, Vahed Qazvinian, and Amjad Abu-Jbara. The ACL anthology network corpus. Language Resources and Evaluation 47, no. 4 (2013): 919-944.

[277] Hersh, William, Chris Buckley, T. J. Leone, and David Hickam. OHSUMED: An interactive retrieval evaluation and new large test collection for research. In SIGIR'94, pp. 192-201. Springer, London, 1994.

[278] Liang, Dawen, Laurent Charlin, James McInerney, and David M. Blei. Modeling user exposure in recommendation. In Proceedings of the 25th international conference on World Wide Web, pp. 951-961. 2016.

[279] Cho, Eunjoon, Seth A. Myers, and Jure Leskovec. Friendship and mobility: user movement in location-based social networks. In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 1082-1090. 2011.

[280] Wang, Xiang, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. Item silk road: Recommending items from information domains to social users. In Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information

Retrieval, pp. 185-194. 2017.

[281] Gao, Huiji, Jiliang Tang, and Huan Liu. gSCorr: Modeling geo-social correlations for new check-ins on location-based social networks. In Proceedings of the 21st ACM international conference on Information and knowledge management, pp. 1582-1586. 2012.

[282] Farseev, Aleksandr, Liqiang Nie, Mohammad Akbari, and Tat-Seng Chua. Harvesting multiple sources for user profile learning: a big data study. In Proceedings of the 5th ACM on International Conference on Multimedia Retrieval, pp. 235-242. 2015.

[283] Shi, Daqian, Ting Wang, Hao Xing, and Hao Xu. A learning path recommendation model based on a multidimensional knowledge graph framework for e-learning. Knowledge-Based Systems 195 (2020): 105618.

[284] Li, Raymond, Samira Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. Towards deep conversational recommendations. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, pp. 9748-9758. 2018.

[285] Ostuni, Vito Claudio, Tommaso Di Noia, Eugenio Di Sciascio, and Roberto Mirizzi. Top-n recommendations from implicit feedback leveraging linked open data. In Proceedings of the 7th ACM conference on Recommender systems, pp. 85-92. 2013.

[286] Kim, Kyung-Min, Donghyun Kwak, Hanock Kwak, Young-Jin Park, Sangkwon Sim, Jae-Han Cho, Minkyu Kim, Jihun Kwon, Nako Sung, and Jung-Woo Ha. Tripartite heterogeneous graph propagation for large-scale social recommendation. arXiv preprint arXiv:1908.02569 (2019).

[287] Vargas-Govea, Blanca, Gabriel González-Serna, and Rafael Ponce-Medellın. Effects of relevant contextual features in the performance of a restaurant recommender system. ACM RecSys 11, no. 592 (2011): 56.

[288] Nikolakopoulos, Athanasios N., Vassilis Kalantzis, Efstratios Gallopoulos, and John D. Garofalakis. EigenRec: generalizing PureSVD for effective and efficient top-N recommendations. Knowledge and Information Systems 58, no. 1 (2019): 59-81.

[289] Jamali, Mohsen, and Martin Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In Proceedings of the fourth ACM conference on Recommender systems, pp. 135-142. 2010.

[290] Agarwal, Deepak, Rahul Agrawal, Rajiv Khanna, and Nagaraj Kota. Estimating rates of rare events with multiple hierarchies through scalable log-linear models. In Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 213-222. 2010.

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

[292] Rendle, Steffen. Factorization machines. In 2010 IEEE International Conference on Data Mining, pp. 995-1000. IEEE, 2010.

[293] Scarselli, Franco, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. IEEE transactions on neural networks 20, no. 1 (2008): 61-80.

[294] Microsoft Satori. https://blogs.bing.com/search/2013/03/21/understand-your-world-with-bing

[295] Wikidata. http://www.wikidata.org/

[296] Google Knowledge Graph. https://blog.google/products/search/introducing-knowledge-graph-things-not/

[297] Facebooks Entities Graph. https://engineering.fb.com/2013/03/06/core-data/under-the-hood-building-out-the-infrastructure-for-graph-search/

[298] IMDb. https://www.imdb.com/

[299] de Souza Pereira Moreira, Gabriel, Felipe Ferreira, and Adilson Marques da Cunha. News session-based recommendations using deep neural networks. In Proceedings of the 3rd Workshop on Deep Learning for Recommender Systems, pp. 15-23. 2018.

[300] Kang, Wang-Cheng, and Julian McAuley. Self-attentive sequential recommendation. In 2018 IEEE International Conference on Data Mining (ICDM), pp. 197-206. IEEE, 2018.

[301] Rendle, Steffen, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In Proceedings of the 19th international conference on World wide web, pp. 811-820. 2010.

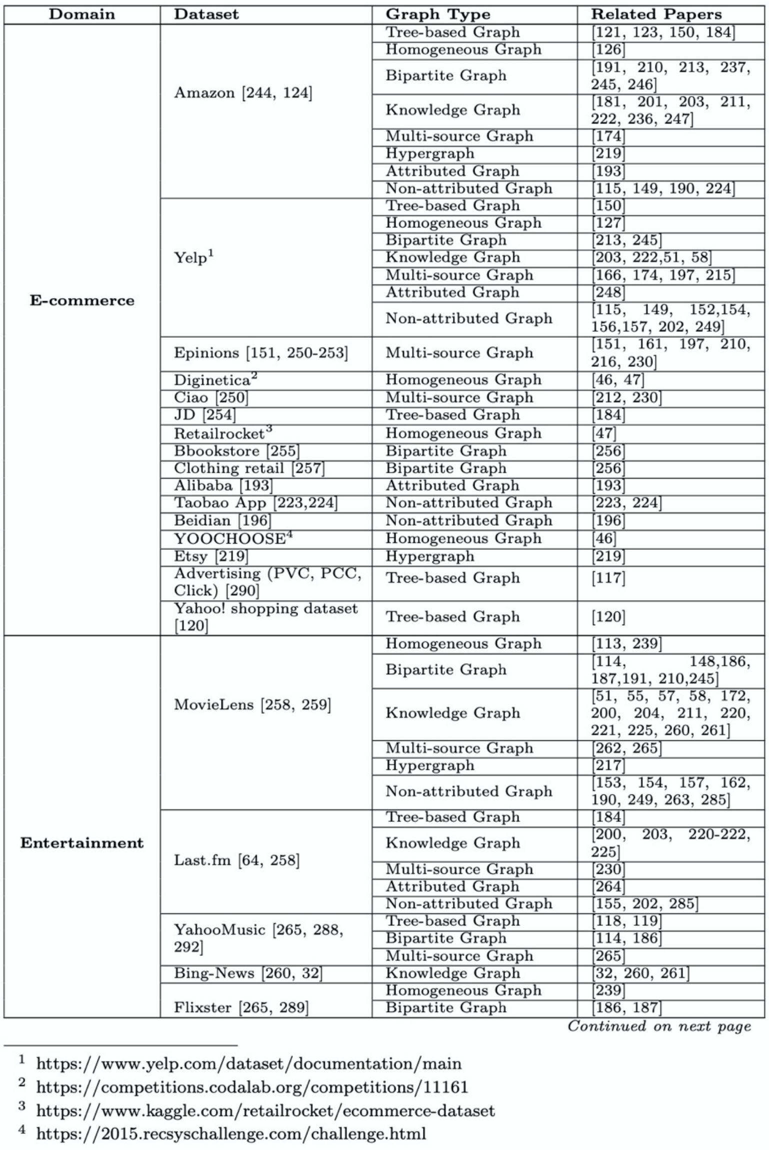
[302] He, Ruining, Wang-Cheng Kang, and Julian McAuley. Translation-based

recommendation. In Proceedings of the eleventh ACM conference on recommender systems, pp. 161-169. 2017.

[303] Kipf, Thomas N., and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

Appendix A: Statistics of datasets commonly used in GLRS

Table 6.2: Statistics of datasets commonly used in GLRS



Domain	Dataset	Graph Type	Related Papers
E-commerce	Amazon [244, 124]	Tree-based Graph	[121, 123, 150, 184]
		Homogeneous Graph	[126]
		Bipartite Graph	[191, 210, 213, 237, 245, 246]
		Knowledge Graph	[181, 201, 203, 211, 222, 236, 247]
		Multi-source Graph	[174]
		Hypergraph	[219]
		Attributed Graph	[193]
		Non-attributed Graph	[115, 149, 190, 224]
	Yelp[1]	Tree-based Graph	[150]
		Homogeneous Graph	[127]
		Bipartite Graph	[213, 245]
		Knowledge Graph	[203, 222,51, 58]
		Multi-source Graph	[166, 174, 197, 215]
		Attributed Graph	[248]
		Non-attributed Graph	[115, 149, 152,154, 156,157, 202, 249]
	Epinions [151, 250-253]	Multi-source Graph	[151, 161, 197, 210, 216, 230]
	Diginetica[2]	Homogeneous Graph	[46, 47]
	Ciao [250]	Multi-source Graph	[212, 230]
	JD [254]	Tree-based Graph	[184]
	Retailrocket[3]	Homogeneous Graph	[47]
	Bbookstore [255]	Bipartite Graph	[256]
	Clothing retail [257]	Bipartite Graph	[256]
	Alibaba [193]	Attributed Graph	[193]
	Taobao App [223,224]	Non-attributed Graph	[223, 224]
	Beidian [196]	Non-attributed Graph	[196]
	YOOCHOOSE[4]	Homogeneous Graph	[46]
	Etsy [219]	Hypergraph	[219]
	Advertising (PVC, PCC, Click) [290]	Tree-based Graph	[117]
	Yahoo! shopping dataset [120]	Tree-based Graph	[120]
Entertainment	MovieLens [258, 259]	Homogeneous Graph	[113, 239]
		Bipartite Graph	[114, 148,186, 187,191, 210,245]
		Knowledge Graph	[51, 55, 57, 58, 172, 200, 204, 211, 220, 221, 225, 260, 261]
		Multi-source Graph	[262, 265]
		Hypergraph	[217]
		Non-attributed Graph	[153, 154, 157, 162, 190, 249, 263, 285]
	Last.fm [64, 258]	Tree-based Graph	[184]
		Knowledge Graph	[200, 203, 220-222, 225]
		Multi-source Graph	[230]
		Attributed Graph	[264]
		Non-attributed Graph	[155, 202, 285]
	YahooMusic [265, 288, 292]	Tree-based Graph	[118, 119]
		Bipartite Graph	[114, 186]
		Multi-source Graph	[265]
	Bing-News [260, 32]	Knowledge Graph	[32, 260, 261]
	Flixster [265, 289]	Homogeneous Graph	[239]
		Bipartite Graph	[186, 187]

Continued on next page

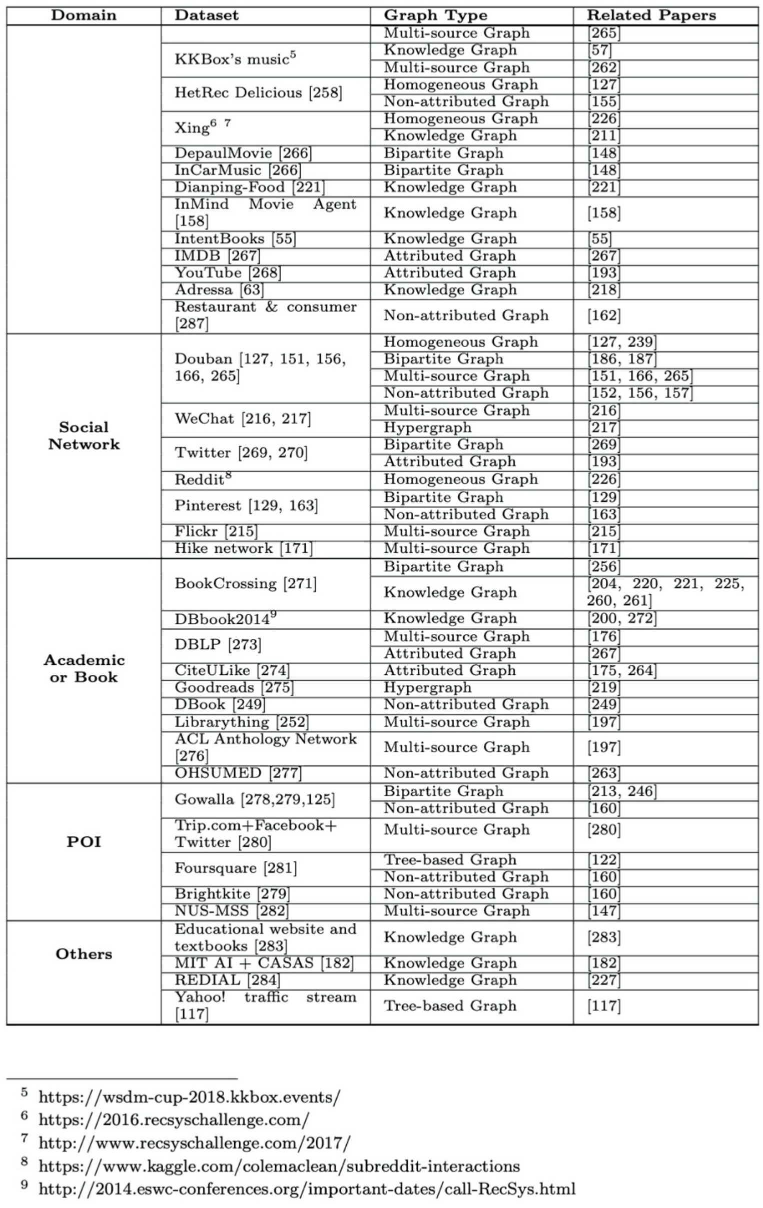
[1] https://www.yelp.com/dataset/documentation/main

[2] https://competitions.codalab.org/competitions/11161

[3] https://www.kaggle.com/retailrocket/ecommerce-dataset

[4] https://2015.recsyschallenge.com/challenge.html

Table 6.2 – ***continue from previous page***



Domain	Dataset	Graph Type	Related Papers
		Multi-source Graph	[265]
	KKBox's music[5]	Knowledge Graph	[57]
		Multi-source Graph	[262]
	HetRec Delicious [258]	Homogeneous Graph	[127]
		Non-attributed Graph	[155]
	Xing[6] [7]	Homogeneous Graph	[226]
		Knowledge Graph	[211]
	DepaulMovie [266]	Bipartite Graph	[148]
	InCarMusic [266]	Bipartite Graph	[148]
	Dianping-Food [221]	Knowledge Graph	[221]
	InMind Movie Agent [158]	Knowledge Graph	[158]
	IntentBooks [55]	Knowledge Graph	[55]
	IMDB [267]	Attributed Graph	[267]
	YouTube [268]	Attributed Graph	[193]
	Adressa [63]	Knowledge Graph	[218]
	Restaurant & consumer [287]	Non-attributed Graph	[162]
Social Network	Douban [127, 151, 156, 166, 265]	Homogeneous Graph	[127, 239]
		Bipartite Graph	[186, 187]
		Multi-source Graph	[151, 166, 265]
		Non-attributed Graph	[152, 156, 157]
	WeChat [216, 217]	Multi-source Graph	[216]
		Hypergraph	[217]
	Twitter [269, 270]	Bipartite Graph	[269]
		Attributed Graph	[193]
	Reddit[8]	Homogeneous Graph	[226]
	Pinterest [129, 163]	Bipartite Graph	[129]
		Non-attributed Graph	[163]
	Flickr [215]	Multi-source Graph	[215]
	Hike network [171]	Multi-source Graph	[171]
Academic or Book	BookCrossing [271]	Bipartite Graph	[256]
		Knowledge Graph	[204, 220, 221, 225, 260, 261]
	DBbook2014[9]	Knowledge Graph	[200, 272]
	DBLP [273]	Multi-source Graph	[176]
		Attributed Graph	[267]
	CiteULike [274]	Attributed Graph	[175, 264]
	Goodreads [275]	Hypergraph	[219]
	DBook [249]	Non-attributed Graph	[249]
	Librarything [252]	Multi-source Graph	[197]
	ACL Anthology Network [276]	Multi-source Graph	[197]
	OHSUMED [277]	Non-attributed Graph	[263]
POI	Gowalla [278,279,125]	Bipartite Graph	[213, 246]
		Non-attributed Graph	[160]
	Trip.com+Facebook+ Twitter [280]	Multi-source Graph	[280]
	Foursquare [281]	Tree-based Graph	[122]
		Non-attributed Graph	[160]
	Brightkite [279]	Non-attributed Graph	[160]
	NUS-MSS [282]	Multi-source Graph	[147]
Others	Educational website and textbooks [283]	Knowledge Graph	[283]
	MIT AI + CASAS [182]	Knowledge Graph	[182]
	REDIAL [284]	Knowledge Graph	[227]
	Yahoo! traffic stream [117]	Tree-based Graph	[117]

[5] https://wsdm-cup-2018.kkbox.events/
[6] https://2016.recsyschallenge.com/
[7] http://www.recsyschallenge.com/2017/
[8] https://www.kaggle.com/colemaclean/subreddit-interactions
[9] http://2014.eswc-conferences.org/important-dates/call-RecSys.html

Appendix B: The graph taxonomies by specific datasets in GLRS

Table 6.3: The graph taxonomies by specific datasets in GLRS

Graph Categorization		Dataset	Recommendation Tasks	Methodology	Related Papers
Tree-based Graph		Amazon, JD, Last.fm, Yelp, Foursquare, Yahoo! datasets (Music, shopping and traffic streams), advertising datasets (PVC, PCC, CLICK)	Sequential Recommendation, Top-N Recommendation, Next-item Recommendation, Rating Prediction	EFM, MF + Logistic Regression, Latent Factor Model + k-order Markov Chains, MF + Hierarchical Embedding, Memory Network	[117-123,150,184]
Homogeneous Graph		Diginetica, Retailrocket, YOOCHOOSE, Douban, Delicious, Yelp, Xing, Reddit, Movielens, Flixster	Sequential Recommendation, Top-N Recommendation, Rating Prediction	GNN, RNN, Graph Attention Network, Graph Embedding, Random Walk + biased PageRank	[113,127,175,126, 46,47,226,239]
Bipartite Graph		Bookstore, Clothing Retail, BookCrossing, Movielens, Flixster, Douban, YahooMusic, Amazon, Hetkee, Gowalla, Yelp, Foursquare, DePaulMovie, InCarMusic, Twitter, Pinterest	Link Prediction, Top-N Recommendation, Next-item recommendation, Rating Prediction	Random Walk + SVM, Random Walk + Markov Chains, Auto-encoder, Collaborative Filtering, GNN, Graph Embedding + Attention, Regularized GNN, GCN + Encoder-Decoder, Bayesian GNN + Graph Generative Model	[114,129,148,163, 186,187,191,210, 213,214,246,237, 245,256,269]
Heterogeneous Graph	Knowledge Graph	Adressa, Movielens, BookCrossing, Bing-News, Yelp, Last.fm, Dianping, KKBox's Music, InMind Movie Agent, DBbook, Amazon, RecSys Challenge 2017, IntentBooks, Educational Website + Textbooks, MIT AI + CASAS, REDIAL	Sequential Recommendation, Top-N Recommendation, Next-item Recommendation	GCN + RNN, Graph Embedding, Bi-RNN, CNN + Attention, Path-based Methods, Regularized GNN, Graph Attention Network, LSTM, Personalized PageRank, Representation Learning + Markov Decision Process, Fairness Constrain (regular), Auto-encoder, Constrained Path Learning	[158,172,181,182, 57,32,55,200,201, 203,204,211,217, 220-222, 225,227,236, 247,51,58,260,261, 272,283]
	Multi-source Graph	Hike, Ciao, Epinion, WeChat, Amazon, Yelp, LibraryThing, Trip.com + Facebook + Twitter, Douban, Flickr, KKBox, MovieLens, DBLP, ACL Anthology Network, Last.fm, NUS-MSS, Flixster, Yahoo! Music	Social Recommendation, Link Prediction, Rating Prediction, Top-N Recommendation	DeepWalk, GNN + Attention Network, GAT + Multi-armed Bandit, Random Walk + Skip-gram, Bert + GNN, Collaborative Filtering + Graph Regularization, MF + Regularization, Layer-wise Influence Propagation + Neural Network	[147,151,161,166, 171,173,174,176, 197,212,215,216, 230,262,265,280]
	Hypergraph	WeChat, MovieLens, Amazon, Etsy, Goodreads	Sequential Recommendation, Top-N Recommendation	GNN + Self-attention, Hypergraph Convolution Network + Self-attention	[217,219]
	Attributed Graph	DBLP, IMDB, Amazon, YouTube, Twitter, Alibaba, Yelp, CiteULike, Last.fm	Link Prediction, Top-N Recommendation, Next-item Recommendation, Conversational Recommendation	Random Walk, Graph Embedding, GNN + personalized PageRank + Attention, Graph Database (Neo4j)	[193,248,264,267, 286]
	Non-attributed Graph	Brightkite, Gowalla, Foursquare, Amazon, MovieLens, Yelp, Taobao, DBook, Beidian, Douban, OHSUMED, Delicious, Last.fm, Restaurant & consumer	Rating Prediction, Top-N Recommendation, Intent Recommendation	Random Walk, Metapath-based Collaborative Filtering + Attention, matrix factorization + factorization machine, GNN, Graph Embedding, GNN + Attention, Metapath-based Similarity + Regularization, Metapath + Gradient Boosting Regression Tree, Spectral GNN + RNN	[115,149,152-157, 160,162,190,192, 202,223,224,249, 263,285]

ISBN 978-82-326-5551-9 (printed ver.)
ISBN 978-82-326-6086-5 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (online ver.)

NTNU

Norwegian University of
Science and Technology