A Framework for Leveraging Properties of User Reviews in Recommendation

XI WANG

SUBMITTED IN FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF
Doctor of Philosophy

School of Computing Science

College of Science and Engineering
University of Glasgow

January 2022

Abstract

With the growing volume of information online, it is increasingly harder for users to identify useful information to support their choices when interacting with different items. Review-based recommendation systems, which leverage reviews posted by users on items to estimate the users' preferences, have been shown to be a credible solution for addressing the problem of identifying their preferred items. However, the actual usefulness of these reviews impact the effectiveness of the resulting recommender systems, especially with the enormous volume of available reviews online. In particular, as argued by the widely cited *users' adoption of information* framework, users exhibit distinct preferences for reviews depending on the properties of these reviews (e.g. length, sentiment) when making decisions. Therefore, we argue that not all reviews are equally useful for different users. We aim to effectively modelling the personalised usefulness of reviews through the use of reviews' properties when developing review-based recommendation techniques. Note that, few studies in the literature investigated the effectiveness of leveraging the properties of reviews to develop effective review-based recommendation approaches.

This thesis aims to address this research gap by proposing a review-based recommendation framework. Such a framework models the personalised usefulness of reviews according to various reviews' properties, including the reviews' age, length, sentiment, ratings, helpfulness as judged by the users and helpfulness as predicted by a review helpfulness classifier. In particular, the thesis addresses two main challenges: (i) the availability of the attributes of reviews and (ii) the users' preferences estimation.

The first challenge refers to the difficulty of extracting particular review properties from their corresponding attributes. For example, extraction of the age property relies on the availability of the timestamps of the corresponding reviews. We address the availability of the reviews' attributes to extract their sentiment and helpfulness properties with classification techniques. The sentiment property of reviews is estimated through effective state-of-the-art sentiment classifiers. We first evaluate the estimated reviews' sentiment in comparison to the users' ratings in typical recommendation approaches. Then, we introduce a sentiment attention mechanism

to encode the estimated reviews' sentiment. Our experiments show that the sentiment property can effectively replace the users' ratings when estimating the user preferences. Moreover, by leveraging the estimated sentiment property of reviews, our proposed review-based rating prediction model shows improved performance compared to state-of-the-art rating prediction models. Next, the extraction of the reviews' helpfulness property leverages the reviews' helpful votes (i.e. a type of feedback given by other reviewers providing information on whether the corresponding review is helpful to them). However, the number of helpful votes for each review are not commonly available. In particular, we propose a novel weakly supervised review helpfulness classification correction approach (i.e. the Negative Confidence-aware Weakly Supervised (NCWS) approach), which leverages the confidence in a given review being unhelpful with respect to its age. We experimentally show that NCWS-based classifiers significantly outperform existing review helpfulness classifiers on two public review datasets. Moreover, the estimated helpfulness of reviews by NCWS-based classifiers can significantly improve the performance of a review-based rating prediction model.

Next, to address our second challenge pertaining to the users' preferences estimation, we aim to estimate their preferences when using reviews exhibiting different properties to accurately predict their preferred items. In particular, we propose two novel ranking-based recommendation approaches (named RPRM and BanditProp), which models the users' preferences using different review properties with different techniques. The RPRM model applies the attention mechanism to model the usefulness of reviews according to different review properties. Unlike RPRM, the BanditProp model leverages existing bandit algorithms and introduces a novel contextual bandit algorithm to tackle the users' preference estimation of using specific reviews' properties to identify useful reviews. Our experiments show that RPRM can outperform state-of-the-art review-based recommendation models, and BanditProp can significantly outperform RPRM on two publicly available review datasets. These results validate the effectiveness of leveraging the review properties when examining the usefulness of reviews to improve the performances of review-based recommendation techniques.

Overall, we contribute an effective review-based recommendation framework that make accurate recommendations by leveraging the reviews' associated properties. This framework includes models for extracting properties from reviews, and various techniques that are required to integrate the learned properties, which, in turn and according to our conducted experiments, provide good approximations of a given users' item preferences. These contributions make progress in the development of review-based recommendation techniques and could inspire future directions of research in recommendation systems.

Contents

List of Tables

List of Figures

Acknowledgements

This PhD thesis continued my research interest in developing recommendation techniques since my undergraduate. I still remember my excitement when my first supervisor, Iadh Ounis, accepted my application to be a PhD candidate student. During these four years, I have been fortunate and received immense support, making this thesis possible.

Firstly, I would like to express my sincere thanks to my supervisors, Iadh Ounis and Craig Macdonald. They guided me through the path of doing research and offered me many opportunities of being involved in the information retrieval community. Moreover, through these four years, their encouragement and great patience built up my confidence in doing research and presenting my work.

I must thank my wife, Jun Zhang, for her love, care and many years of support. I would also like to thank my parents and parents in law, who encouraged me whenever I encountered any difficulties. Without them, this work would not be possible.

I am also grateful to my friends and colleagues at the Terrier team and the school of computing science, including Anjie Fang, Zaiqiao Meng, David Maxwell, Ting Su, Graham McDonald, Xiao Yang, Richard McCreadie, Jarana Manotumruksa, Jeff Dalton, Ke Yuan, Siwei Liu, Xiao Wang, Yashon Wu, Jingmin Huang, Zixuan Yi, Zijun Long, Fatma Elsafoury and many others. They have offered me much help and shared much of their knowledge. It has been a great pleasure to work with them.

Chapter 1

Introduction

1.1 Introduction

With the advances of technology and the growth of smart devices, the access to and use of e-commerce services are growing rapidly. As a consequence, the intensive interactions between people and e-commerce platforms are leading to an enormous volume of information online (Jiang et al., 2019a). The information overload makes it difficult for people to find relevant information. To alleviate and decrease the information overload, recommendation systems have been shown to be a promising solution (Zhang et al., 2019; Paul et al., 2017). In particular, recommendation systems are techniques that suggest relevant items to users from an overwhelming number of items (Ricci et al., 2015). They are critical for many online services and e-commerce websites, such as Amazon,[1] Yelp[2] and TripAdvisor,[3] to effectively provide personalised product suggestions to customers.

During the past few decades, recommendation systems have been developed by leveraging different types of user feedback. For example, the users' ratings on items have been widely used in developing recommendation systems to estimate the users' preferences on items (Cremonesi et al., 2010; Liu et al., 2011; Wei et al., 2017), such as in Matrix Factorisation (MF) (Koren et al., 2009), a typical collaborative filtering-based recommendation approach. MF infers the users' and items' latent vectors according to their interactions expressed as ratings. Then, it makes recommendations based on the high correspondence between the latent vectors of users and items, e.g. ranking items based on the dot product of the user and item latent factors (Koren et al., 2009). Another example of user feedback is the users' posted check-in data. Benefitting from

[1] https://www.amazon.com
[2] https://www.yelp.com
[3] https://www.tripadvisor.com

the ubiquity of Global Position System (GPS) chips embedded in smartphones, users can share their real-time interactions with locations or venues and mark a check-in at the corresponding location. As a result, many sequential recommendation studies (Manotumruksa et al., 2018; Tang and Wang, 2018) aimed to learn the users' sequential interaction patterns from the check-in data to estimate the users' next interests. However, these various types of feedback are not equally useful when used in recommendation systems to capture the users' preferences and improve the recommendation performances (Fu et al., 2019; Yu et al., 2013; Wu et al., 2021). Among the available types of users' feedback, the users' posted reviews are frequently used in recommendation systems (Dong et al., 2020; Chen et al., 2018b), since review data contains rich sentiment information about the items. Moreover, reviews are effective for enhancing the performances of recommendation systems, especially when addressing the recommendation task for users / items with few interactions (Zheng et al., 2017). Therefore, with the advantages of using review data, in this thesis, we focus on using review data to develop effective recommendation systems.

Recommendation systems can process the users' posted reviews in different manners. Among the existing approaches of leveraging review data in recommendation systems, a typical strategy is to model the users' preferences based on the textual content of reviews. For example, DeepCoNN (Zheng et al., 2017) merged the reviews posted by a given user (and orthogonally, for a given item) into a single document. Then, it converted the generated documents into latent vectors to represent the users' preferences or the items' attributes. After that, DeepCoNN used these latent vectors as input to a collaborative filtering-based recommender, to predict the users' interests on unseen items. However, the usefulness of the selected reviews can impact the effectiveness of extracting the users' preferences or the items' attributes from reviews (Chen et al., 2018b). In particular, as we mentioned above, the information overload results in an enormous volume of available data online, which includes a huge volume of the users' posted reviews. Indeed, users are unlikely to interact with every available review online to make decisions. In addition, the quality of reviews can vary (Liu et al., 2020b) and users can express different opinions about the usefulness of reviews (Filieri and McLeay, 2014). These factors make it important and challenging to identify the usefulness of reviews from a given user's perspective. Therefore, a challenge in a review-based recommendation approach is to identify personalised useful reviews or to automatically estimate the usefulness of reviews for a given user.

On the other hand, reviews have various associated properties, such as their age, length and sentiment polarity. Each review property implies a characteristic or attribute of a given review. In this thesis, we argue that the *review properties* can be leveraged to effectively capture the usefulness of the reviews, so as to enhance the recommendation performances. Indeed, by using different review properties, we can model the usefulness of reviews from their corresponding

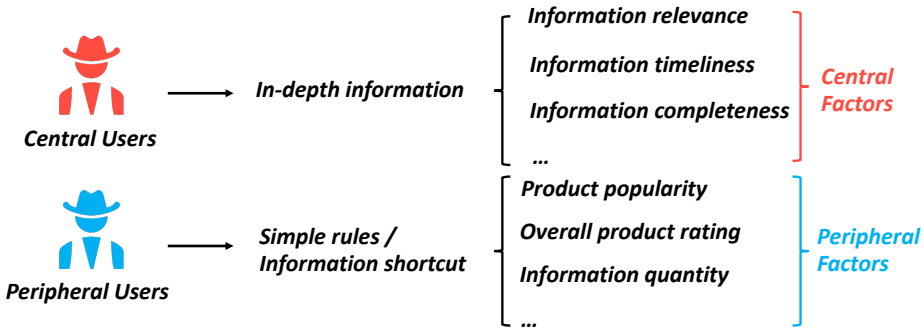


Figure 1.1: The Users' Adoption of Information (UAoI) framework (Sussman and Siegal, 2003).

characteristics. Review properties themselves have also been widely used in the development of recommendation systems (e.g. the timestamp of reviews were used in addressing the sequential recommendation task (Li et al., 2020)). However, there are few studies (e.g. (Liu et al., 2020d; Gao et al., 2020)) that investigated the effectiveness of leveraging review properties in developing a review-based recommendation system. Moreover, from the users' perspective, a review is not equally useful to every user (Filieri and McLeay, 2014). For example, if we consider two users that are about to read reviews to support their decision making, one user might prefer the recent reviews, while another user might be keener to read reviews with rich sentiment information. In this thesis, we aim to address the research gap of estimating the users' preferences on review properties to personalise the usefulness of reviews, so as to develop effective recommendation systems.

In the remainder of this chapter, we start with a discussion of the motivation of this thesis in Section 1.2. After that, we introduce the thesis statement in Section 1.3. In Section 1.4, we describe the contributions of this thesis, followed by acknowledging the origins of materials in Section 1.5. Finally, we briefly present the thesis outline in Section 1.6.

1.2 Motivations

As mentioned in Section 1.1, we aim to develop review-based recommendation systems by leveraging the review properties and estimating the users' personal view on the usefulness of reviews accordingly. This is motivated and supported by a well-cited framework – the users' adoption of information framework (Sussman and Siegal, 2003) – from the social science domain. We provide an overview of the the framework in Figure 1.1, which shows that users follow different behaviour patterns (e.g. the central and peripheral routes), when using various types of infor-

mation, while interacting with items. The central route-based users tend to consider the issues presented by the given information carefully. Meanwhile, the peripheral factors (e.g. rating and overall ranking) have a higher impact on the peripheral route-based users' decision making. Therefore, this users' adoption of information framework states that, when users receive the same information, each user follows a particular scheme or strategy in using different properties or aspects of information and thereby, each user makes different decisions to perform an activity.

As per such framework, in this thesis, we argue that users tend to consider reviews having different properties for examining the items' or products' attributes that support their decision making. Accordingly, we expect the usefulness of reviews in review-based recommendation systems to correlate with the users' preferences in considering various review properties. Therefore, we argue that by leveraging the review properties to examine the usefulness of reviews, we can develop effective recommendation models.

Indeed, according to the Elaboration Likelihood Model (ELM) (Petty and Cacioppo, 2012), – a representative model that instantiates the users' adoption of information framework – the properties of reviews can impact the extent to which the users adopt the given information. Similarly, Filieri and McLeay (2014) also indicated that users give different attitudes toward accepting the suggestions in the available review according to various review properties, such as the timeliness, relevance, and completeness of reviews. Therefore, a review might not be equally useful to different users. In the social science literature, the reviews' usefulness has been widely investigated using the users' adoption of information framework (Aghakhani et al., 2020; Filieri and McLeay, 2014). The usefulness of reviews has also played a vital role in the recommendation literature (Chen et al., 2018b; Lin et al., 2018) when considering the reviews as side information to model the users' features. However, to the best of our knowledge, no previous work considered the users' adoption of information framework to model the usefulness of reviews and then estimate the users' preferences on items.

Overall, motivated by the users' adoption of information framework, we propose to leverage the properties of reviews – posted by the users about the corresponding items – to develop effective recommendation approaches. Hence, in this thesis, we are motivated to answer the following questions: (1) how to propose a model that adequately reflects the properties of reviews; (2) how to effectively combine review properties into review-based recommendation systems to enhance their performances; (3) which review properties better capture the usefulness of reviews according to the users' preferences.

1.3 Thesis Statement

The statement of this thesis is that an enhanced recommendation performance can be attained by leveraging the available properties of the reviews posted by users. As motivated by the users' adoption of information framework, as illustrated in Figure 1.1, we postulate that by distinguishing among the users' personal views on the usefulness of reviews, the recommendation model can effectively apply personalised recommendations and obtain more accurate recommendations.

We focus on six commonly available review properties, namely their age, length, sentiment, rating, helpfulness as judged by the users and helpfulness as predicted by a review helpfulness classifier to enhance the recommendation performance. In particular, we propose an end-to-end recommendation framework, which uses all the six review properties. In particular, by deploying such a recommendation framework, we can develop models that separately consider the users' posted reviews and the items' associated reviews. We hypothesise that we can attain more effective review-based recommendation models by leveraging available review properties (e.g. review age and review length). However, since the sentiment and the helpfulness properties require human effort to label, these two properties are not always available. We postulate that by inferring the sentiment and helpfulness properties from the users' posted reviews, the review-based recommendation model can collect accurate user preferences information, thereby increasing the recommendation effectiveness. Furthermore, we postulate that by estimating the users' views on the use of different review properties to examine the usefulness of reviews, we can model the review data differently for each user, to develop personalised recommendation approaches and make accurate recommendations.

1.4 Contributions

Recall that in the thesis statement (Section 1.3), we aim to develop an effective review-based recommendation framework that uses reviews as well as their associated properties. To achieve this target, we contribute a series of approaches to use the reviews' associated properties in developing review-based recommendation systems. In particular, to implement the review-based recommendation framework, it is essential to address two main challenges: (i) the availability of review attributes and (ii) the estimation of users' preferences. Since some reviews' attributes are not commonly available to extract the corresponding reviews' properties, we aim to tackle the reviews' attribute availability with different techniques to make the reviews properties readily available. Then, we can leverage the learned properties of reviews to estimate the users' pref-

erences on selectively using reviews' properties to examine the usefulness of reviews, which, in turn improves the performances of the review-based recommendation models. In particular, we summarise the main contributions of this thesis on addressing these two challenges above.

- **Availability of Review Attributes:** To correctly estimate the reviews' properties, we need to leverage their corresponding attributes. For example, we rely on the reviews' associated timestamp to calculate their age. However, some attributes are frequently missing, making it harder to correctly estimate the reviews' properties. In particular, we argue that the sentiment and helpfulness properties are not always fully labelled (Guan et al., 2016; Kim et al., 2006). Consequently, we contribute to estimating these two properties with the weakly labelled reviews' attributes.

  To extract the sentiment property from reviews, we conduct two studies. In the first study, we represent the sentiment property as probability scores (namely *property scores*), representing the likelihood of the reviews being polarised (i.e. positive or negative). These sentiment property scores reflect the user preferences since they convey a clear sentiment. Then, we compare the effectiveness when using the sentiment property scores of reviews in comparison to when using the reviews' associated ratings in recommendation systems. In the second study, we encode the scores of the sentiment property into a review-based recommendation system and evaluate its performance.

  Next, we turn to the estimation of the helpfulness property of reviews, which also suffers from the limitations of noisy and insufficient labels (Kim et al., 2006). In particular, we propose a weakly supervised binary classification correction approach (NCWS) to effectively address the review helpfulness classification task, so as to generate reliable helpfulness scores to use to enhance the recommendation outcomes. The NCWS approach tackles the weakly supervised binary classification task by leveraging positive unlabelled learning and using a negative confidence-based loss function to model the unlabelled examples. We examine the effectiveness of integrating NCWS within different popular binary classifiers, including SVM, CNN and BERT-based classifiers on three real-world datasets. We also compare with state-of-the-art classification correction approaches and show the superior performance of NCWS. In particular, we validate the utility of NCWS by using its predicted helpful reviews as input to a state-of-the-art review-based recommendation model (i.e. DeepCONN (Zheng et al., 2017)).

- **Estimation of Users' Preferences:** After addressing the reviews' attribute availability, we have many reviews' properties readily available. Therefore, we turn to address the second

challenge in our proposed review-based recommendation framework, namely the estimation of users' preferences. In particular, we instantiate the proposed recommendation framework with two recommendation approaches that use six commonly available review properties. They are the reviews' age, length, sentiment, rating, helpfulness as judged by users and helpfulness as predicted by a review helpfulness classifier. Then, we estimate the personalised usefulness of reviews accordingly and enhance the recommendation performances. These two recommendation approaches are RPRM and BanditProp, respectively.

The RPRM model leverages the usefulness of reviews to address the recommendation task. To the best of our knowledge, this is the first work that integrates the review properties in estimating the reviews' usefulness in a recommendation model through leveraging how users make use of such reviews when interacting with items (e.g. purchase a product or visit a venue). Moreover, inspired by the users' adoption of information framework, we propose two loss functions and one negative sampling strategy that model the agreement on the importance of review properties between the users and items. We show that RPRM outperforms state-of-the-art approaches on two real-world datasets, and its performance can be further enhanced with the proposed loss functions and negative sampling strategy.

On the other hand, the BanditProp model considers extracting the users' preferences and items' attributes in reviews with different review properties as multiple parallel tasks. Next, we propose to convert the selection of features from such distinct review modelling with varying properties of reviews into a bandit problem. In particular, we integrate various bandit algorithms into BanditProp to predict the users' preferred review properties. We further propose to leverage the users' posted reviews as contextual information to enhance the effectiveness of estimating the users' preferred review properties when making recommendations. We thoroughly evaluate and show the effectiveness of our proposed BanditProp model compared to state-of-the-art recommendation approaches and our attention mechanism-based model (i.e. RPRM) on two public datasets.

Therefore, with two instantiated review-based recommendation models, we have developed effective recommendation approaches by leveraging reviews as well as their associated review properties. This contribution and findings also align with the argument that we made in the thesis statement (Section 1.3). Indeed, by estimating users' preferences on using different review properties to examine the usefulness of reviews, we can develop effective recommendation approaches that model the review data differently for each user.

1.5 Origins of Material

Most of the material presented in this thesis is based on published studies in various international conferences:

- Chapter 5: We propose to substitute the reviews' associated ratings with the sentiment property scores calculated by various sentiment classifiers. This work was published in ECIR 2019 (Wang et al., 2019d). In addition, this chapter also proposes a property-encoding strategy to embed property scores into the review embeddings, so as to capture the usefulness of reviews according to different review properties. In particular, we use the sentiment property score as an example to verify the performance of such a strategy. This work was published in ECIR 2022 (Wang et al., 2022).

- Chapter 6: We propose to construct a weakly supervised binary classification correction approach (NCWS) to effectively address the review helpfulness classification task. In particular, we apply the NCWS approach within various existing binary classifiers and evaluate the predicted helpful reviews by using them as input to a state-of-the-art review-based recommendation model. This NCWS approach was published in CIKM 2020 (Wang et al., 2020b).

- Chapter 7: We propose the Review Property-based Recommendation Model (RPRM) that leverages the dot-product attention mechanism to estimate the importance of review properties to users and items to enhance the performance of a review-based recommendation approach. We also propose two customised loss functions and one negative sampling strategies to further improve the performance of RPRM in addressing the recommendation task. This work was published in WWW 2021 (Wang et al., 2021a).

- Chapter 8: We propose a multi-task learning-based model, BanditProp as another model, using review properties to personalise the usefulness of reviews, so as to enhance the recommendation performances. In particular, we consider the selection of the use of various review properties as a bandit problem and apply a number of bandit algorithms to address the corresponding task. This work was published in the TWEB journal.

1.6 Thesis Outline

The remainder of this thesis is organised as follows:

- Chapter 2 describes the background of developing recommendation systems. We explore the types of recommendation tasks and discuss the classical collaborative filtering approaches, which are commonly applied techniques in the literature for implementing recommendation models. In particular, we discuss typical memory and model-based collaborative filtering approaches in the literature.

- Chapter 3 discusses the related work in the literature addressing the review-based recommendation task and the use of review properties. First, we describe typical review-based recommendation approaches and then discuss the recommender systems that used the typical review properties (i.e. sentiment, temporal and helpfulness). Moreover, we discuss two main types of user behaviour modelling strategies (i.e. users' conscious and unconscious behaviour). This discussion supports investigating the relationships between the users' preferences and their different behaviours (e.g. users' adoption of information).

- In Chapter 4, we first formally describe the recommendation problem that we aim to address and its used notations. Next, we present and illustrate our proposed framework for review-based recommendation systems that leverages review property information. Such a framework consists of four main components: (1) data preparation, (2) review property extraction, (3) review property encoding, and (4) user preference estimation. In this chapter, we emphasise the challenges within each component. The effectiveness of the four components are experimentally investigated in Chapters 5, 6, 7 and 8.

- Chapter 5 first introduces our proposed technique that represents the sentiment property of reviews into property scores. We evaluate the effectiveness of sentiment property when applied to the recommendation approaches by comparing with the reviews' associated ratings. Next, we introduce our proposed sentiment property-based recommendation model (i.e. SentiAttn) that takes the sentiment property scores into account. We evaluate this model on real-world datasets (i.e. the public available Yelp and Amazon review corpus) and discuss their performances in comparison to state-of-the-art recommendation approaches (e.g. DeepCoNN (Zheng et al., 2017) and NARRE (Chen et al., 2018b)).

- Chapter 6 describes our proposed weakly supervised binary classification correction approach (NCWS), which adjusts the existing state-of-the-art review helpfulness classifiers to generate reliable helpfulness property scores from reviews. In particular, we evaluate the effectiveness of using the helpfulness property of reviews in review-based recommendation approaches by using the predicted helpful reviews as input to a state-of-the-art review-based recommendation model (namely DeepCoNN).

- In Chapter 7, we instantiate our proposed recommendation framework with a review property-based recommendation model, RPRM, that uses six commonly available review properties to estimate the importance of various review properties for users or items. We evaluate the effectiveness of RPRM in estimating users' preferences on items by comparing to existing state-of-the-art recommendation systems on two public datasets (i.e. the public available Yelp and Amazon review corpus).

- Chapter 8 introduces another novel review-property-based recommendation model, BanditProp, that instantiates our proposed recommendation framework. In particular, BanditProp considers the use of various review properties as a multi-task learning problem and then converts the selection of the use of different review properties as a bandit problem. We also introduce novel bandit algorithms to address the bandit problem to enhance the performance of BanditProp in estimating users' preferences on items. We discuss the performances of BanditProp in comparison to state-of-the-art approaches as well as our proposed RPRM model on real-world datasets (namely the Yelp and Amazon datasets).

- Chapter 9 closes this thesis by highlighting the contributions and the conclusions of each chapter. We also discuss some possible future directions for our research.

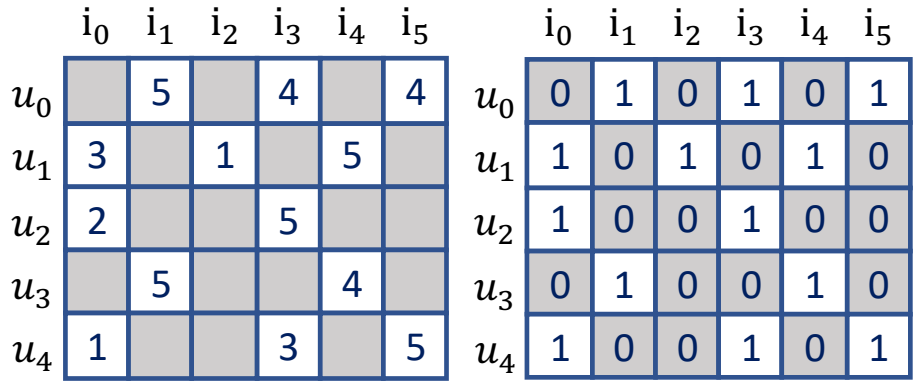
Chapter 2

Background

2.1 Introduction

As introduced in Section 1.1, this thesis aims to leverage the rich available information – the review data in particular – to propose effective recommendation approaches for accurately predicting the users' preferences and outperforming existing recommendation techniques. In particular, we aim to correctly estimate the users' preferences on their unseen items by proposing and developing effective review-based recommendation approaches that build upon the modelling of reviews as well as their associated properties.

In this chapter, we give an overview of the recommendation systems, which summarises the existing work that we consider as a basis of our proposed recommendation framework. In particular, we first discuss the two commonly investigated recommendation tasks (i.e. the rating prediction and ranking-based recommendation tasks) in the literature and their corresponding evaluation metrics. After that, we explore the recommendation algorithms that have been widely applied to develop recommendation models in the literature. Moreover, we consider a common taxonomy of recommendation systems that groups the existing recommendation algorithms into three categories: collaborative filtering (CF), content-based and hybrid recommendation models (Adomavicius and Tuzhilin, 2005). The first two types of recommendation models differ in using the similarities among users and items to make recommendations. The CF models rely on the similarity scores between the users' historical interactions. However, the content-based approaches estimate users' preferences according to the similarity between items (Thorat et al., 2015). Note that the hybrid recommendation approaches combine the first two approaches. In this section, we discuss the recommendation systems with an extensive discussion of the CF techniques, which are the most commonly investigated and developed recommendation models



(a) Explicit feedback (b) Implicit feedback

Figure 2.1: Different forms of user feedback.

in the literature (Burke, 2002; Yang et al., 2014; Mustafa et al., 2017). In particular, we explore classical CF models, including the memory-based methods, classical model-based CF and recent neural model-based CF.

2.2 Taxonomy of Recommendation Tasks

A recommendation system is a general designation of approaches that learn the users' historical behaviours to estimate and predict the users' preferences on their unseen items. Therefore, it is essential to estimate if a recommendation system can effectively make accurate recommendations to the users. Regarding the evaluation methods, in the literature, there are three main types of evaluation strategies for examining the performances of recommendation models (Manotumruksa, 2021), namely (1) a user study, (2) online evaluation, and (3) offline evaluation. However, for the first two of these three types of evaluation methods, it is more challenging than the case of the offline evaluation to efficiently examine the performances of the proposed recommendation approaches (Knijnenburg et al., 2012; Kharitonov, 2016) and obtain trustworthy performance conclusions. For example, it is expensive and difficult to hire qualified users for a user study. Moreover, the users' subjective consciousness can result in a bias in their answers during a user study (Sinha et al., 2001). On the other hand, the online evaluation is another evaluation strategy for recommendation techniques. However, it is costly to deploy recommendation systems for an online evaluation. As a consequence, a common practice is to simulate the online environment via a simulator and then run an algorithm against such simulated online environment. Nevertheless, it is still often difficult to create a proper simulator without introducing a modelling bias (Li

et al., 2011).

Therefore, in this thesis, similar to many existing recommendation approaches and given the effectiveness of applying the offline evaluation on large datasets (Cheng et al., 2018b; Baral et al., 2018; Jiang et al., 2019a), we apply the offline evaluation metrics to evaluate the performance of our proposed recommendation approaches. In the literature, many studies considered various recommendation metrics to evaluate different perspectives of the recommendation outcomes, including prediction accuracy, coverage, novelty, privacy, diversity, scalability, and so on (Gunawardana and Shani, 2015). In this thesis, we aim to improve the recommendation accuracy of state-of-the-art recommendation techniques. The evaluation of other perspectives, other than the prediction accuracy, is beyond the scope of this thesis. There are two types of commonly investigated recommendation tasks: (1) rating prediction and (2) ranking-based recommendation task (Steck, 2013). In the following, we detail these two recommendation tasks and their corresponding evaluation metrics.

2.2.1 Rating Prediction Task

Rating prediction is a classic recommendation task, which aims to address the prediction of users' ratings on their unseen items. Formally speaking, the interactions between a set of users U with size m and a set of items I with size n is represented as a matrix $R^{m\times n}$. In Figure 2.1a, we illustrate the users' explicit feedback on items with an example scale of 1 - 5. Rating prediction models examine the users' preferences via their existing explicit ratings $r \in R$ on a scale of 1 - k and then predict the missing values in the matrix $R^{m\times n}$ (i.e. the grey cells in Figure 2.1a).

On the other hand, to evaluate the accuracy of the made recommendations by a rating prediction model, there are various existing applicable evaluation metrics in the literature. Essentially, an optimal rating prediction model has to correctly estimate the users' ratings on their unseen items. Therefore, the performance of a rating prediction model can be examined by considering the margin between the ground truth $r_{u,i}$ and the predicted value $\hat{r}_{u,i}$ of a given user u on item $i \in I$ that are unseen by the model (i.e. the test data).

$$MAE = \frac{\sum_{(u,i)\in R} |r_{u,i} - \hat{r}_{u,i}|}{|R|} \tag{2.1}$$

$$RMSE = \sqrt{\frac{\sum_{(u,i)\in R} (r_{u,i} - \hat{r}_{u,i})^2}{|R|}} \tag{2.2}$$

In Equation (2.1) and (2.2), we present two commonly used rating prediction evaluation

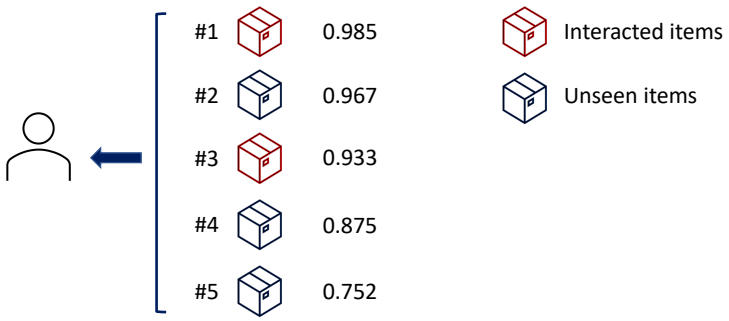


Figure 2.2: An example list of ranked items for recommendation.

metrics: Mean Average Error (MAE) and Root Mean Square Error (RMSE). MAE calculates the averaged absolute difference between the predicted ratings and the ground truth ones. RMSE quadratically scores the margins between the predicted and the ground truth ratings. Therefore, RMSE gives a higher weight to large margins (i.e. big differences between the true and predicted ratings) than the MAE metric. In this thesis, we use these two evaluation metrics to evaluate the proposed rating prediction models.

However, a rating prediction-based recommendation model suffers from limitations, such as the users' inconsistent rating behaviours (Said and Bellogín, 2018). Indeed, users tend to give different levels of ratings on items than other users. Moreover, the users' rating behaviours in giving high scores to items are often unstable and can fluctuate over time (Said et al., 2012). Therefore, a growing number of studies turn to the investigation of the ranking-based recommendation task, which we introduce next.

2.2.2 Ranking Task

For the ranking-based recommendation task, the goal of the recommendation approaches is to correctly rank the users' preferred items on top (Cremonesi et al., 2010). Figure 2.2 gives an example list of top-ranked items as recommendation outcomes to a given user. A ranking-based recommendation model targets to score the users' preferred items higher than others and putting such items on top of a list of recommended items.

Formally speaking, in this recommendation scenario, we create a matrix $R_{U,I}$ that comprises the interactions between a set of users U with size m and a set of items I with size n. A ranking-based recommendation model aims to rank users' preferred or interacted items higher than items that are not preferred by users (Wang et al., 2019a). A commonly used assumption, when developing ranking-based recommendation approaches, is that we consider the users' interacted items as their preferred items (Rendle et al., 2009; Volkovs and Yu, 2015). Therefore, to ag-

gregate the users' interacted items, a recommendation model can leverage both explicit and/or implicit feedback as binary signals. We show the difference between explicit and implicit feedback in Figure 2.1. By comparing Figure 2.1b with Figure 2.1a, we can see that if an item i has been rated or interacted by a user u, for implicit feedback, the corresponding interaction will be indicated by a positive signal (i.e. 1). We can aggregate the positive signals according to different user-item interactions. Examples of those interactions include the users' check-ins, browses, clicks and purchases behaviours. Therefore, a ranking-based recommendation model can leverage richer information than a rating prediction model and give more effective recommendation results (Cremonesi et al., 2010).

Similar to the discussion in Section 2.2.1, to examine the performance of a ranking-based recommendation approach, it is vital to apply appropriate evaluation metrics. In the literature, there are five commonly used evaluation metrics for showing the performances of a ranking-based recommendation approach. These five evaluation metrics are Precision, Recall, Mean Average Precision (MAP), Mean Reciprocal Rank (MRR) and Normalised Discounted Cumulative Gain (NDCG).

As for the Precision and Recall metrics, the Precision metric aims to answer how many recommended items align with the users' preferences. The Recall metric examines the ratio of items of interest for users that have been correctly identified from the item set (Shani and Gunawardana, 2011). In addition, a cutoff N is commonly applied to calculate Precision and Recall by focusing on the top-N items. Hence, we can calculate the Precision@N or Recall@N that examine the top-N recommendations accordingly. However, it is important to evaluate a ranking-based recommendation model according to the suggested items in different ranking positions. Different from the Precision and Recall metrics, the MAP, MRR, and NDCG metrics focus on the ranking of items.

1. **MAP**: The definition of MAP relies on the calculation of Average Precision (AP) over each user. Moreover, AP is equal to the average of the precisions at the ranking positions of the users' every preferred items (Yilmaz et al., 2008). After having the AP score for every user, to calculate the MAP score, we can apply an average operation to the AP scores of the target users to evaluate the performance of a recommendation approach.

2. **MRR**: The MRR metric evaluates a resulted ranking list of items according to when the first relevant item is recommended, instead of focusing on many items. The MRR score is calculated as follows:

$$MRR = \frac{1}{|U|} \sum_{u \in U} \frac{1}{rank_u} \tag{2.3}$$

where $rank_u$ is the rank of user u's first preferred item that is recommended. In this thesis, instead of solely identifying the users' most preferred item, we aim to conduct a comprehensive evaluations of our proposed recommendation approaches across all possible interacted items. Therefore, the MRR metric does not fit our requirements and will not be considered.

3. **NDCG**: To calculate the NDCG scores with a threshold N, NDCG first summing the relevance scores of items from the beginning to position N (i.e. Cumulative Gain). Then, the NDCG metric applies a discount function (e.g. a log function) over the cumulative gains to put different weights on the recommendations at different ranking positions (i.e. Discounted Cumulative Gain (DCG)). In the end, to address the DCG scores in different scales with different number of ranked items, the DCG scores are normalised by comparing to the ideal DCG scores. We refer the reader to Järvelin and Kekäläinen (2002) for a detailed description of the NDCG metric.

Thus far, we have discussed five popular evaluation metrics (i.e. Precision, Recall, MAP, MRR and NDCG) in examining the performances of ranking-based approaches. In this thesis, we adopt the Precision, Recall, MAP and NDCG metrics to evaluate our proposed ranking-based recommenders. Note that the MRR metric is not considered since MRR specifically evaluates if the first preferred item is recommended. However, we aim to evaluate the recommendation techniques in this thesis by focusing on many items (e.g. the top 10 ranked items).

In particular, as we mentioned in Section 2.2.1, compared with the rating prediction task, the ranking-based recommendation better aligns with the user-item interaction behaviours and focuses on users' top-preferred items. Therefore, in this thesis, we aim to propose recommendation approaches that effectively address the ranking-based recommendation task. In the next Section, we turn to the exploration of various classic Collaborative Filtering (CF)-based recommendation approaches in the literature.

2.3 Classical Collaborative Filtering Models

In Section 2.2, we discussed the two main recommendation tasks in the literature. In the following, we explore the classic and typical recommendation approaches that leverage the users' historical interactions to estimate the users' preferences on unseen items. In particular, since the collaborative filtering technique is the most commonly applied in developing recommendation models, we explore the classic CF approaches in the section.

Measures	Equations	Example Study
Cosine Similarity	$cos(u,v) = \frac{\sum_{i \in \mathcal{J}_{uv}} r_{ui} \cdot r_{vi}}{\sqrt{\sum_{i \in \mathcal{J}_u} r_{ui}^2 \sum_{j \in \mathcal{J}_v} r_{vj}^2}}$	Billsus and Pazzani (2000)
Pearson Correlation	$PC(u,v) = \frac{\sum_{u \in \mathcal{U}_{ij}} (r_{ui} - \bar{r}_i)(r_{uj} - \bar{r}_j)}{\sqrt{\sum_{u \in \mathcal{U}_{ij}} (r_{ui} - \bar{r}_i)^2 \sum_{u \in \mathcal{U}_{ij}} (r_{uj} - \bar{r}_j)^2}}$	Deshpande and Karypis (2004)
Adjusted Cosine Similarity	$AC(u,v) = \frac{\sum_{u \in \mathcal{U}_{ij}} (r_{ui} - \bar{r}_u)(r_{uj} - \bar{r}_u)}{\sqrt{\sum_{u \in \mathcal{U}_{ij}} (r_{ui} - \bar{r}_u)^2 \sum_{u \in \mathcal{U}_{ij}} (r_{uj} - \bar{r}_u)^2}}$	Sarwar et al. (2001)
Mean Squared Difference	$MSD(u,v) = \frac{\lvert \mathcal{J}_{u,v} \rvert}{\sum_{i \in \mathcal{J}_{uv}} (r_{ui} - r_{vi})^2}$	Shardanand and Maes (1995)
Spearman Rank Correlation	$SRC(u,v) = \frac{\sum_{i \in \mathcal{J}_{uv}} (k_{ui} - \bar{k}_u)(k_{vi} - \bar{k}_v)}{\sqrt{\sum_{i \in \mathcal{J}_{uv}} (k_{ui} - \bar{k}_u)^2 \sum_{i \in \mathcal{J}_{uv}} (k_{vi} - \bar{k}_v)^2}}$	O'Mahony et al. (2003)
Frequency-Weighted Pearson Correlation	$FWPC(u,v) = \frac{\sum_{i \in \mathcal{J}_{uv}} \lambda_i (r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{\sum_{i \in \mathcal{J}_{uv}} \lambda_i (r_{ui} - \bar{r}_u)^2 \sum_{i \in \mathcal{J}_{uv}} \lambda_i (r_{vj} - \bar{r}_v)^2}}$	Breese et al. (1998)
Item-based Weighted Pearson Correlation	$iWPC(u,v) = \frac{\sum_{i \in \mathcal{J}_{uv}} w_i j (r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{\sum_{i \in \mathcal{J}_{uv}} w_{ij} (r_{ui} - \bar{r}_u)^2 \sum_{i \in \mathcal{J}_{uv}} w_{ij} (r_{vj} - \bar{r}_v)^2}}$	Baltrunas and Ricci (2009)

Table 2.1: Similarity measures for neighbourhood user identification. u and v are two target users, and $\mathcal{J}$ indicates the set of items both users u and v have rated. k refers to the rank of the corresponding item according to its rating score.

Collaborative filtering (CF)-based recommenders rely on the interaction pattern proximity between users or between items to make predictions of the items of interest (Ekstrand et al., 2011). In particular, by examining the similarity between users or between items, the collaborative filtering recommendation approaches can be classified into user-based CF and item-based CF models, respectively. User-based CF models base the prediction of the users' preferences on modelling users' similarity in giving close rating scores or presenting similar interaction frequencies to a shared item set. On the contrary, item-based CF approaches examine if two items have been rated similarly by a shared group of users to estimate if these users will exhibit similar opinions to a group of similar items. In the rest of this section, we discuss the approaches applied in user-based CF models. The item-based CF approaches can be similarly developed. In particular, the CF models have been commonly divided into memory and model-based CF approaches (Bobadilla et al., 2013), which differ at the use of the input data and the techniques to estimate the users' preferences. In the following, we start with the discussion of the memory-based CF techniques and then explore the model-based CF models in the literature.

2.3.1 Memory-based Methods

Memory-based collaborative filtering is also known as the neighbourhood-based CF model, which intuitively predicts the users' preferences according to the tastes of their like-minded users. In particular, it identifies the users' neighbourhoods according to the similarity between their historical feedback on items, which examines if they display similar interests on items. Therefore, a memory-based collaborative filtering model needs to address two main challenges: (1) selectivelyusing effective similarity measures and (2) identifying like-minded users. To address the first challenge, the cosine similarity and the Pearson Correlation are the two most classic approaches used in the CF-based models to calculate users'/items' similarity. In addition, some other cosine similarity and Pearson correlation-variants have also been proposed and applied to the CF-based techniques. In Table 2.1, we summarise representative similarity measures from the literature. Next, for the second challenge, it is also vital to effectively and efficiently leverage the calculated similarity between users to estimate the preferences of target users. To address this challenge, for a given user, some filtering approaches (e.g. Top-N filtering) have been applied to their neighbours (i.e. the similar users) and identify the most similar users to them for the preference estimation. For example, the Top-N filtering (Du et al., 2021), the threshold filtering (BL, 2019) and the negative filtering (Ning et al., 2015) approaches. After applying specific similarity metrics and pre-filtering similar users accordingly, we obtain the users' neighbourhoods so as to predict the missing rating values of the target users on unseen items. A commonly used technique (Do et al., 2010; Su and Khoshgoftaar, 2009) to compute the missing rating values is to use the following equation:

$$r_{ui} = \hat{r}_u + \frac{\sum_{v=1}^{k}(r_{vi} - \hat{r}_v)sim(u,v)}{\sum_{v=1}^{k}|sim(u,v)|} \tag{2.4}$$

where u is the target user and this approach compares their rating behaviours to a set of like-minded users v.

However, the memory-based techniques, which directly use the user-item interaction data, suffer from the data sparsity as well as the scalability problems (Gong et al., 2009) in making good recommendations. Therefore, many studies turned to developing model-based recommendation approaches, which address the problems generated by the memory-based recommendation approaches and further investigate the relationships between user/item-related information and the items of interest for the given users.

2.3.2 Model-based Collaborative Filtering

A model-based recommendation model aims to build an effective recommender upon the observation of the historical user-item interactions as well as the users' preferences and items' attributes (Do et al., 2010). In the following, we discuss the earlier machine learning techniques – before the emergence of deep learning discussed in Section 2.4 – which are commonly applied in recommendation studies to develop collaborative filtering approaches. We refer to these approaches as classical CF models. In the recommendation literature, there are many machine learning techniques that have been applied (e.g. clustering, classification and matrix factorisation approaches (Do et al., 2010)). The clustering model-based approaches rely on the similarity between users or items to make recommendations, similar to the memory-based recommenders. However, instead of identifying like-minded users (namely neighbourhoods), the clustering model-based techniques aggregate users/items into different groups and assume that users within the same group share a similar taste on items or that items in one group present similar features (Ungar and Foster, 1998). Two main used types of clustering techniques are the hierarchical (Zhang et al., 1996; Guha et al., 2000; Karypis et al., 1999) and partitional (MacQueen et al., 1967; Kaufman and Rousseeuw, 2009; Ng and Han, 2002) clustering approaches that build a hierarchical structure of user cluster or divide users into k-clusters without any hierarchical connection. On the other hand, classification-based CF models leverage tailor-made classifiers to predict the users' preferences on predefined classes. Next, the classification-based CF models use these classes to infer users' preferences on items. Two commonly used classification techniques in developing classification-based CF models are support vector machines (Xia et al., 2006; Wang et al., 2015b) and naive Bayes classifiers (Zhang, 2004; Miyahara and Pazzani, 2002).

Among the classical CF models, the matrix factorisation technique is the most commonly used technique in implementing the CF models. The classic matrix factorisation (Koren et al., 2009) first projects users and items to a joint latent factor space and then leverages the inner products as a similarity measure to model user-item interactions. Formally speaking, with a matrix $R^{m\times n}$ as in Figure 2.1, which encapsulates user feedback on items, matrix factorisation approximates $R^{m\times n}$ with the product of two lower dimensional matrices, $P \in R^{m\times d}$ for users and $Q \in R^{n\times d}$ for items, where d is the dimensionality of the joint latent factor space. Hence, the users' preference estimation can be calculated as follows:

$$r_{u,i} = p_u^T q_i \tag{2.5}$$

where p_u and q_i are the latent factors of user u and item i. However, so far, it is not introduced how to calculate the latent factors P and Q of the users and items. In the literature, a classical solution is to compute the Singular Value Decomposition (SVD) of the matrix $R^{m\times n}$ to obtain factorised latent factors (i.e. $R^{m\times n} \approx P^{m\times k} \times D^{k\times k} \times Q^{k\times n}$, where D is a non-negative diagonal matrix and k indicates the size of latent factors). However, SVD is undefined when applied to incomplete matrices, which is common for the user-item interaction matrix R in the recommendation scenario.

Therefore, it is common to apply both the linear system-based matrix factorisation (i.e. $R^{m\times n} \approx P^{m\times k} \times Q^{k\times n}$) – such as the LU decomposition (Bartels and Golub, 1969) – and regularisation techniques on the set of known user-item interactions to generate the latent factors p_u and q_i (Koren et al., 2009; Koren, 2009). The resulting latent factors can be optimised via minimising a regularised loss, which is defined as follows:

$$\min_{q*,p*} \sum_{(u,i)\in R} (r_{u,i} - p_u^T q_i)^2 + \lambda(||q_i||_F^2 + ||p_u||_F^2) \tag{2.6}$$

where $r_{u,i}$ are the available ratings in the matrix $R^{m\times n}$ and $||\cdot||_F^2$ denotes the Frobenius norm of a given matrix. In particular, λ is a parameter that controls the regularisation.

Then, to minimise the loss, which is calculated by Equation (2.6), the Stochastic Gradient Descent (SGD) (Saad, 1998) is frequently applied to identify a local minimum for such a loss function. Before applying SGD, we need to calculate the current prediction error:

$$e_{u,i} = r_{u,i} - p_u^T q^i \tag{2.7}$$

Then, we can leverage the SGD technique to update the users' and items' latent factors as follows:

$$\begin{aligned} q_i &\leftarrow q_i + \gamma(e_{u,i} \cdot p_u - \lambda \cdot q_i) \\ p_u &\leftarrow p_u + \gamma(e_{u,i} \cdot q_i - \lambda \cdot p_u) \end{aligned} \tag{2.8}$$

where γ denotes the step size in updating the parameters. However, the classical MF technique still focuses on predicting the missing rating values. To extend the classical MF model to address the ranking-based recommendation task, many studies (Pessiot et al., 2007; Rendle et al., 2009) have turned to investigating and developing effective ranking-based learning schemes to enhance the matrix factorisation approach when making recommendations. In particular, among them, the Bayesian Personalised Ranking (BPR) (Rendle et al., 2009) is one of the most widely used model learning scheme, which follows a pairwise ranking setup for model training. In particular,

BPR aims to give higher ranks to the users' interacted items than their unseen items according to the calculated preference scores. Hence, BPR creates a corresponding training data: $D_s : U \times I \times I$:

$$D_s := \{(u,i,j)|i \in I_u^+ \wedge j \in I \setminus I_u^+\} \tag{2.9}$$

where (u,i,j) indicates that user u prefers item i over item j and I_u^+ is the set of user u's interacted items. In particular, for each user u, BPR aggregates their interacted items $i \in I^+$ and uniformly samples their unseen items $j \in I \setminus I_u^+$ to constitute a set of training triples $(u,i,j) \in D_s$.

Hence, the objective function of BPR can be formulated as follows:

$$\mathcal{J}(\Theta) = \arg\max_{\Theta} \sum_{(u,i,j)\in D_s} ln(\sigma(\hat{r}_{u,i} - \hat{r}_{u,j})) - \lambda_\Theta ||\Theta||_F^2 \tag{2.10}$$

where $\sigma(\cdot)$ is the logistic function: $\sigma(x) := \frac{1}{1+e^{-x}}$ and the last part of the objective function is the regularisation term that is used to avoid overfitting. Afterwards, similar to the classic matrix factorisation technique, BPR also adopts stochastic gradient descent (SGD) for model training. A general optimisation process of applying SGD to update the model parameter Θ is as follows:

$$\Theta \leftarrow \Theta + \alpha(\sigma(\hat{r}_{u,i} - \hat{r}_{u,j}) \cdot \frac{\partial}{\partial \Theta}(\hat{r}_{u,i} - \hat{r}_{u,j}) + \lambda_\Theta \Theta) \tag{2.11}$$

where α is the learning rate, which controls the step size of updating the parameter Θ. In this thesis, we also consider the MF technique with the BPR learning scheme as one of the important classical baseline approaches to show the performances of our proposed models.

Thus far, we have shown the implementation of memory-based collaborative filtering approaches with representative shallow learning-based machine learning techniques: clustering, classification, and matrix factorisation models. These approaches have been widely applied to develop recommendation techniques. However, it is difficult for these approaches to generating effective representations of users' preferences and items' attributes, especially with sparse auxiliary data (Wang et al., 2015a). Meanwhile, in recent years, deep learning has been shown to be effective in modelling complex relationships among data inputs and generating effective representations when applied to many research fields. A considerable amount of literature has been dedicated to deep learning-based recommendation systems. Therefore, in the next section, we describe various typical deep learning techniques that were used to develop effective collaborative filtering-based models.

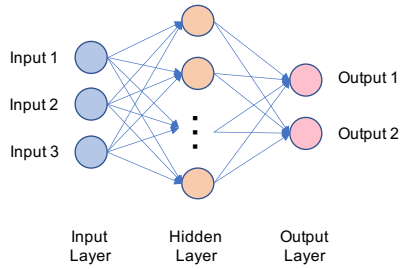


Figure 2.3: The basic architecture of a MLP model.

2.4 Neural Networks and Their Application to Collaborative Filtering

Different from the classic machine learning techniques, deep learning approaches aim to capture the higher level features within the data input (Deng and Yu, 2014). In particular, as explained by Bengio et al. (2013), deep learning methods are:

> "Those approaches that are formed by the composition of multiple non-linear transformations, with the goal of yielding more abstract – and ultimately more useful – representations."

In the recommendation literature, deep learning techniques have assisted many recommendation approaches in effectively capturing the users' preferences on items. In the following, we describe various representative neural architectures in the literature. In particular, we discuss the Multi-Layer Perceptrons (MLPs), Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs) and the attention mechanism as typical neural models.

2.4.1 Multi-Layer Perceptron (MLP)

MLP is a feed-forward neural network model, which is also referred to as a basic neural network (NN). Figure 2.3 presents a basic structure of a MLP model. From the figure, we can see that an MLP model comprises at least three layers of nodes, which are the input, hidden and output layers. In particular, by incorporating multiple hidden layers and activation functions, the MLP model applies a non-linear transformation of the input data to extract features and generate abstractive as well as hierarchical representations for the recommendation task. Moreover, intuitively, the MLP model builds upon the single-layer perceptron network (Taud and Mas, 2018), which computes the output according to the summation of the input data and can be formulated

as follows:

$$\hat{y} = \sigma(Wx + b) \tag{2.12}$$

where the input data x is weighted by a randomly initialised vector W and adjusted by a bias term b. In particular, one of several activation functions (e.g. Rectified Linear Unit (ReLU)), $\sigma(\cdot)$, can be applied to introduce non-linearity into the neural network (Sharma and Sharma, 2017). Then, after stacking multiple single-layer perceptron networks, a MLP model can be described as follows:

$$\hat{y} = a_L(W_L(...a_1(W_1x + b)) + b_L)) \tag{2.13}$$

where L indicates the depth of the corresponding MLP model. However, a MLP is not simply a duplicated set of single-layer perceptrons. A MLP can indicate a convex region to distinguish inputs in a higher dimensional space (Lippmann, 1987). Therefore, the MLP model is capable of dealing with complex data modelling and prediction tasks.

To train an MLP model, in the literature, the back-propagation algorithm (Rumelhart et al., 1986) is the most commonly used technique (Cilimkovic, 2015). We further divide the back-propagation algorithms into two stages (Rojas, 1996): 1) feed-forward and 2) back-propagation. At the first stage, by comparing the calculated and the expected output, the model produces an error signal for every node within the hidden layers. Next, at the second stage, the model calculates the weights within each neural layer via first passing the error signals backwards according to the local gradient and then updating the weights like in the discussed Stochastic Gradient descent in Section 2.3.2,. The model then repeats these two stages until it converges.

In the literature, many recommendation models adopted the MLP technique to model the user-item interactions and estimate the users' preferences on items. For example, Neural Collaborative Filtering (NCF) (He et al., 2017) proposed a dual MLP network to model the user-item interactions. Its score function, which predicts the users' preferences on items is defined as follows:

$$\hat{r}_{u,i} = \sigma_{out}(\sigma_x(...\sigma_1(P^T v_u^U, Q^T v_i^I)) \tag{2.14}$$

where σ_{out} and σ_x indicate the activation functions for generating the output from the x-th neural layer. In particular, $P \in \mathbb{R}^{m \times d}$ and $Q \in \mathbb{R}^{n \times d}$ are the factorised latent vectors of users and items and v_u and v_i are the one-hot identifier of user u and item i, respectively. Then, NCF applies a weighted square loss or binary cross-entropy loss to address the prediction of the users' explicit or implicit feedback (He et al., 2017). In the literature, the MLP model has also been combined with various machine learning techniques to develop recommendation models (e.g. the matrix factorisation technique (Dziugaite and Roy, 2015; Xue et al., 2017) and the factori-

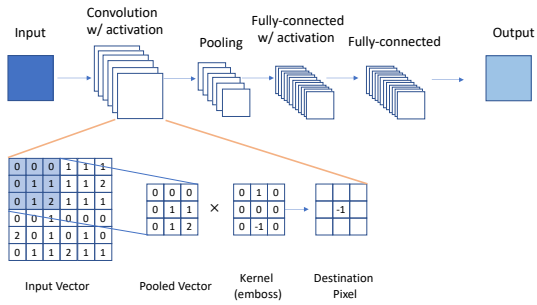


Figure 2.4: A basic CNN architecture (top) and visual illustration of the convolutional layer (bottom).

sation machine (Guo et al., 2017a; Lian et al., 2018b) that introduced by Rendle (2010)). Many studies have also proposed recommendation model by stacking the MLP layers differently and creating different architectures (Chen et al., 2017a; Covington et al., 2016). For example, the Wide&Deep learning model (Cheng et al., 2016), which separately uses both the single-layer and multilayer perceptrons to memorise the direct feature from input data and generate abstract representations (Cheng et al., 2016). However, to capture a complex data structure of the user-item interactions, a MLP model needs to be sufficiently complex with a large number of parameters, which makes it difficult in learning the desired target function (Rendle et al., 2020). To address such difficulty, different neural models have been proposed and applied to model complex data, such as the Convolutional Neural Networks that we discuss in the next Section.

2.4.2 Convolutional Neural Networks (CNNs)

Similar to MLP, the CNNs model is another type of feed-forward networks and was initially developed with local sensitive neurons to model the grid-structured inputs (Liu et al., 2015). We can regard CNNs as a regularised version of MLP. CNNs introduce the filter scheme with smaller patterns to address pattern recognition. The use of the filter scheme reduces the complexity of the MLP model. The architecture of basic CNNs comprises five main components (O'Shea and Nash, 2015): 1) the input layer; 2) the convolutional layer; 3) the pooling layer; 4) the fully connected layer; and 5) the output layer. The *input layer* feeds the data input to the model. The *convolutional layer* applies multiplications of a set of weights (i.e. kernels or filters) to a restricted area (i.e. receptive field) of the previous layer. In Figure 2.4, we draw the convolutional layer with an arbitrary input and an kernel. In particular, by calculating the receptive field, the CNN model can identify the dependent relationships among local information. Next, the *pooling layer* down-samples a given input to further reduce the number of parameters within a neural

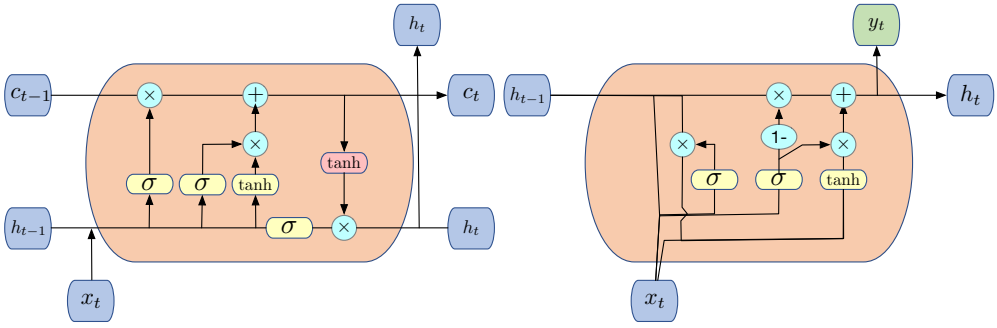
network model. Then, the *fully-connected* layer functions similarly to the MLP model, which learns the given input to generate general and abstract representations. In the end, the *output layer* makes the prediction based upon the calculated feature vectors from the previous layers. As for the learning and model training technique, the back-propagation and stochastic gradient descent strategies are also the most commonly applied approaches to update the parameters of a CNN model (Wang et al., 2017b; Zheng et al., 2019; Dogo et al., 2018).

Therefore, by considering the space efficiency and the ability to capture the surrounding features, many studies (Wang et al., 2019b; Yan et al., 2019; You et al., 2019) integrated CNNs to the vanilla collaborative filtering (CF) model. CASER (Tang and Wang, 2018) converted the users' sequential interactions with items into a 2-dimensional latent space. In particular, CASER leveraged the convolutional filter to capture the sequential patterns and any skipping behaviour in user-item interactions. Another example, ConvMF (He et al., 2018), introduced the convolutional filter to the NCF model to further identify higher-order correlations between the features within the user-item interaction latent vectors. However, even though CNNs-based approaches can capture the correlations between features in a higher dimensional view, CNNs suffer from the difficulty in modelling data exhibiting some specific patterns. For example, it is challenging for CNNs to capture long-term recurrent patterns in user-item interactions (Xu et al., 2019), like when users repeatedly visit a venue within a time period. The traditional CNNs-based techniques also do not consider the temporal information among user activities (Mao et al., 2018), such as session-based user-item interactions. However, much of the recommendation literature, e.g. (Kang and McAuley, 2018; Chen et al., 2018d; Liu et al., 2016), emphasized the importance of modelling the users' sequential behaviours to predict accurate recommendations. Therefore, to address the above concern, we discuss the recurrent neural network, which learns the interactions between users and items in a sequential manner.

2.4.3 Recurrent Neural Networks (RNNs)

Other than the MLP and CNN techniques, in the recommendation literature, the Recurrent Neural Networks (RNNs) are effective in capturing the users' dynamic preferences and modelling a sequential interaction context. In the following, we first discuss the architecture of RNNs and their variants as well as the use of RNNs in recommendation systems with example approaches.

RNNs comprise three main components: the input, hidden and output layers. In particular, a feed-forward network calculates different weights for each node of the network. However, differently from the feed-forward network, RNNs share the parameters across the neural layers of the network by introducing the hidden state mechanism. The hidden state is a representation



(a) LSTM (b) GRU

Figure 2.5: The recurrent Unit of LSTM and GRU networks.

of previous inputs up to the current time step. At first, RNNs initialise the hidden state h_0. Then, each hidden layer (i.e. recurrent unit R_t) takes the data X_t and the hidden state h_{t-1} of the previous time steps as input. Next, RNNs generate a corresponding output y_t as well as the hidden state h_t. Formally speaking, as explained by Elman (1990), the hidden state h_t and the output y_t are calculated as follows:

$$h_t = \sigma_h(X_t W_h + U_h h_{t-1} + b_h) \tag{2.15}$$

$$y_t = \sigma_y(W_y h_t + b_y) \tag{2.16}$$

where W_h, U_h and W_y are coefficients and b_h and b_y are the used bias terms. The predictions also leverage σ_h and σ_y, which are two activation functions. In particular, to train RNNs, the basic backpropagation algorithm is applied at each step (i.e. the backpropagation through time algorithm (Werbos, 1990)) to update the gradients, which is formulated as follows:

$$\frac{\partial \mathcal{L}^T}{\partial W} = \sum_{t=1}^{T} \frac{\partial \mathcal{L}^T}{\partial W}\bigg|_t \tag{2.17}$$

Therefore, by leveraging the hidden states, RNNs can model the input data with varying lengths and learn to memorize the historical information. Given these advantages, many studies have adopted RNNs for the development of recommendation systems (Wu et al., 2016; Zhang et al., 2014a). However, along with the increasing size of the user-item interactions, there are longer sequences that a RNN model has to learn, which leads to two common problems (Bengio et al., 1994): vanishing gradient and exploding gradient. Due to these two problems, the gradient will become zero or infinite exponentially fast (Bengio et al., 1994). To address the exploding

gradient problem, the gradient clipping approach (i.e. $clip(g,\theta) = g \cdot max(1, \frac{\theta}{||g||})$) is commonly applied in the literature. Note that, g is the gradient and θ is the hyperparameter. To alleviate the gradient vanishing problem when training the RNN model, two RNN-variants, Long Short Term Memory (LSTM) and Gated Recurrent Unit (GRU), have been proposed accordingly. The first of which, LSTM (Hochreiter and Schmidhuber, 1997), was devised to deal with the long-term dependency problem. In particular, LSTM introduced a cell state and a gating mechanism to the recurrent unit to enable a recurrent neural network with a constant error flow via an unchanged cell state. In Figure 2.5a, we present a recurrent unit of LSTM. Figure 2.5a shows that the calculation of a hidden state h_t relies on not only both input x_t and the previous hidden state h_{t-1}, but also on the previous cell state c_{t-1}. We detail the calculation of the hidden state h_t as follows:

$$\begin{aligned} f_t &= \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \\ i_t &= \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \\ o_t &= \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \\ \tilde{c}_t &= \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \\ c_t &= f_t \circ c_{t-1} + i_t \circ \tilde{c}_t \\ h_t &= o_t \circ \sigma_h(c_t) \end{aligned} \tag{2.18}$$

where W, U, and b are associated parameters and $\circ$ indicates the element-wise product. The cell state c_{t-1} is the 'memory' of the network that saves important information during the sequence modelling. Meanwhile, f_t, i_t and o_t refer to the forget gate, input gate and output gate, respectively. These gates decide how much information we keep from the previous states (forget gate), which values to update (input gate) and what to output (output gate) (Gers et al., 2000).

On the other hand, the other RNNs-variant, GRU (Cho et al., 2014) can be regarded as a simplified version of LSTM. In Figure 2.5b, we depict the recurrent unit of a GRU model. According to the figure, we can see that a GRU does not include the cell states and also has fewer gates than the recurrent unit of LSTM. In particular, for the recurrent unit of a GRU model, the calculation of the hidden states can be formulated as follows:

$$\begin{aligned} z_t &= \sigma_g(W_z x_t + U_z h_{t-1} + b_z) \\ r_t &= \sigma_g(W_r x_t + U_r h_{t-1} + b_r) \\ \hat{h}_t &= \phi_h(W_h x_t + U_h(r_t \circ h_{t-1}) + b_h) \\ h_t &= (1 - z_t) \circ h_{t-1} + z_t \circ \hat{h}_t \end{aligned} \tag{2.19}$$

where W, U and b are the model's parameters like the ones in vanilla-RNN and LSTM. As for the applied gate mechanism, GRU differs from LSTM by leveraging fewer gates and merely uses the update gate (z_t) and reset gate (r_t). The reset gate r_t decides how much information to keep from the previous state while the update gate z_t determines the extend to which the previous hidden state h_{t-1} and the newly calculated hidden state $\hat{h}_t$ can be leveraged (Cho et al., 2014). In particular, we compare the recurrent unit of GRU with the recurrent unit of LSTM and observe that GRU is a lighter model than LSTM and can also alleviate the burden of memory cost. Therefore, in the recommendation literature, the GRU technique has been widely applied, especially when dealing with session-based or sequential recommendations (Bharadhwaj et al., 2018; Liu et al., 2018a; Hidasi et al., 2015). For example, to address session-based recommendation, GRU4Rec (Hidasi et al., 2015) applied the GRU technique to model the users' sequential item-click behaviour and predict users' preferred items to interact next. Similarly, Chu et al. (2017) also considered GRU to address users sequential behaviour. However, they emphasised the impact of the users' recent interactions on the users' future behaviours and limit the sequence to a small constant value. Moreover, GRU has also been frequently integrated with other machine learning techniques (e.g. variational inference (Broderick et al., 2013; Song et al., 2019) and Generative Adversarial Networks (GAN) (Bharadhwaj et al., 2018)) to propose effective recommendation systems. An example is RecGAN (Bharadhwaj et al., 2018), which is implemented by combining GRU with GAN. In particular, it constructed the generator and discriminator by utilising two individual GRU models. The generator predicts the users' ratings on items, while the discriminator judges if the generated ratings align with the true rating distribution. Thus far, we have discussed three types of neural models (namely MLP, CNNs and RNNs). However, recently, many studies argued that it is difficult for these techniques to effectively identify the important parts of the input that contribute to the recommendations. Next section, we discuss the attention mechanism, which was proposed to address the above challenge.

2.4.4 Attention Mechanism

Different from the introduced neural models above, the attention mechanism-based recommendation models aims at selectively considering the most relevant information (Vaswani et al., 2017) instead of the whole input, to make practical recommendations. The attention mechanism is a function that calculates the dynamic weights of different parts of the input in deciding the corresponding output (e.g. the representation vector of users' preferences). Many studies commonly used one of three approaches to implement the attention mechanism (i.e. additive, dot-product and multi-head attention). In particular, to explain these three mechanisms, we

assume that we aim to model a sequence of items that have been interacted by a given user. We denote such a sequence as $S = \{i_0, i_1, ..., i_m\}$ and each unique item is represented by a corresponding one-hot embedding vector. Then, we apply the attention mechanism over the sequence S and learn a representation vector to indicate the users' preferences. For the *additive mechanism* (Bahdanau et al., 2015), the weight a of each item that contributes to the generation of the output O with a weighted sum (i.e. $O = \sum_{j=1}^{m} a_j S_j$). For each of the weight a_j, it is computed by:

$$a_j = \frac{exp[f(S_j)]}{\sum_{k=1}^{m} exp[f(S_k)]} \tag{2.20}$$

where $f(\cdot)$ is a feed-forward neural network that is applied over each of the item latent vector. Next, the dot-product and the multi-head attention mechanisms involve another group of notions (i.e. query Q, key K, value V). The query Q is the target, the key K is the source, and the attention weights are applied to the value V. In our assumed user-item interaction scenario, the query Q is the output O. Both key K and value V are the input, namely the item sequence S. Then, we formulate the calculation of weights via a dot-product attention as follows:

$$W(Q,K,V) = softmax(\frac{QK}{\sqrt{d_k}})V \tag{2.21}$$

where d_k is the dimension of K. On the other hand, the multi-head attention mechanism argues that learning information from different representations (i.e. multi-heads) can improve the models' performances (Vaswani et al., 2017). We also formulate the multi-head attention mechanism as follows:

$$\begin{aligned} MultiHead(Q,K,V) &= [h_1 \oplus ... \oplus h_n]W^O \\ \text{where } h_i &= W(QW_i^Q, KW_i^K, VW_i^V) \end{aligned} \tag{2.22}$$

and W_i^Q, W_i^K, W_i^V and W_O indicate the learned parameter metrics that project the representations into different variants.

By leveraging the additive, dot-product and multi-head attention mechanisms above and introducing customised or optimised variants, many studies have proposed attention mechanism-based CF approaches, to examine the relationships between the users' historical interacted items and the unseen items, and have shown state-of-the-art performances (Fu et al., 2018; Zhou et al., 2018; Lv et al., 2020). Zhou et al. (2018) also argued that the attention mechanism is capable of dynamically capturing the users' behaviours.

In particular, a typical application of the dot-product attention mechanism (or self-attention)

is to the latent semantic spaces, where it models the users' behaviours to build an individual behaviour representation. It is also common for attention mechanisms to be adopted to learn the importance of user/item features. For example, the ACF model (Chen et al., 2017b) applied the attention mechanism on the users' historical items to identify which items and which parts of the items features make effective user representations. Therefore, given the effectiveness of the attention mechanism in identifying helpful information and generating individual representations, in Chapters 5, 7, and 8, we adopt and leverage the attention mechanism to enhance the models' recommendation performances.

2.5 Conclusions

In this chapter, we have provided a comprehensive overview of the recommendation systems in the literature. We illustrated two commonly investigated recommendation tasks: rating prediction and ranking-based recommendation tasks. In particular, we have discussed that it is more practical to recommend a list of users' top-preferred items than predicting the ratings of users' unseen items. Therefore, in this thesis, we aim to develop effective ranking-based recommendation approaches.

Next, we also explored various collaborative filtering techniques, which are the most commonly used approaches to build recommendation models. In particular, we discussed the memory-based CF (Section 2.3.1), classical model-based CF (Section 2.3.2) and recent neural CF models(Section 2.4). We shown that the recent neural models, such as the introduced MLP, CNNs, RNNs and the attention mechanisms, are more effective techniques than memory-based and classical CF models to develop practical recommendation approaches. Therefore, in this thesis, our proposed recommendation framework will take the recent neural models into account to model the features from user-item interactions, reviews as well as their associated properties, so as to develop effective recommendation techniques. However, thus far, we have not discussed how the existing recommendation techniques leveraged the reviews and their associated properties to make recommendations. In the next chapter, we explore the existing work that related to the main contributions of this thesis, which is the use of reviews as well as reviews' associated properties to build effective recommendation techniques.

Chapter 3

Related Work

As we previously discussed in Chapter 2, in this thesis, we aim to develop top-n ranking-based recommendation models and base our models' architectures on deep learning techniques. However, recall that in Section 1.4, different from the recommendation approaches in the literature, the main contributions of this thesis are based on the use of reviews as well as their various associated properties, such as sentiment and helpfulness, to address recommendation tasks.

Therefore, to distinguish our main contributions from the existing work, in this chapter, we explore the recommendation techniques in the literature that consider the review text as input to identify the users' interests and estimate their preferences on target items (Section 3.1). Next, since we also aim to use the reviews' associated features / characteristics in the development of the recommendation approaches, we also discuss, in Sections 3.2, 3.3 and 3.4, the related works that leveraged various review properties in recommendation tasks. In particular, we focus on discussing the reviews' sentiment, timing and helpfulness, which are the typical and commonly used features in various review-based recommendation scenarios (such as those exemplified by the Amazon and Yelp datasets). On the other hand, in this thesis, we argue that reviews are not equally useful to different users and users are likely to selectively use reviews exhibiting different properties. Therefore, next, to learn to model the users' behaviour, we explore the studies that investigated the users' behaviour when interacting with online data. In particular, we describe two typical user behaviour modelling strategies in the literature (i.e. user conscious and unconscious behaviour modelling). We detail the outline of this chapter as follows:

- Section 3.1 explores many review-based recommendation approaches. Moreover, we first use a representative review-based recommendation model (i.e. DeepCoNN (Zheng et al., 2017)) to discuss the use of review text in a deep learning-based recommendation model. We show that a review-based recommender commonly comprises three components: (1)

textual content modelling; (2) user/item feature learning; and (3) users' preferences prediction. Then, we compare the differences among the existing review-based recommendation approaches based on these three components. We also identify the limitations of these approaches.

- Section 3.2 discusses the recommendation approaches that take the review sentiment information into account. In particular, we group these review sentiment-based recommenders into two categories: the studies that leveraged the (1) sentiment orientation and (2) aspect-level sentiment analysis. We compare the differences between the approaches from these two classes and identify their limitations.

- In Section 3.3, we explore the recommendation studies that used the review timing information. To compare the differences among such approaches, we also categorise them into (1) sequential recommenders and (2) Context-aware recommendation approaches. Then, we identify the limitations of these existing approaches.

- In Section 3.4, we discuss how the review helpfulness has been used in addressing the recommendation task. In particular, we show that existing review helpfulness-based recommenders relied on the use of a learned review helpfulness classifier. Therefore, we also summarise the available review features that have been leveraged to train review helpfulness classifiers. Then, we illustrate the limitations of the existing review helpfulness-based approaches.

- Section 3.5 explores two types of users' behaviour modelling techniques, namely the modelling of the users' conscious and unconscious behaviour. We show how the modelling of the users' behaviour has been leveraged in the recommendation literature. In particular, we show that the users' adoption of information, the modelling of a type of users' unconscious behaviour, can be used to address the estimation of users' preferences on different types of reviews.

- Section 3.6 gives the concluding remarks for this chapter.

3.1 Review-based Recommendation Models

In this thesis, we aim to develop an effective review-based recommendation framework. The main objective of applying a recommendation model is to recommend items that the users might be interested in, based on the observed users' behaviour and a designed technique to estimate

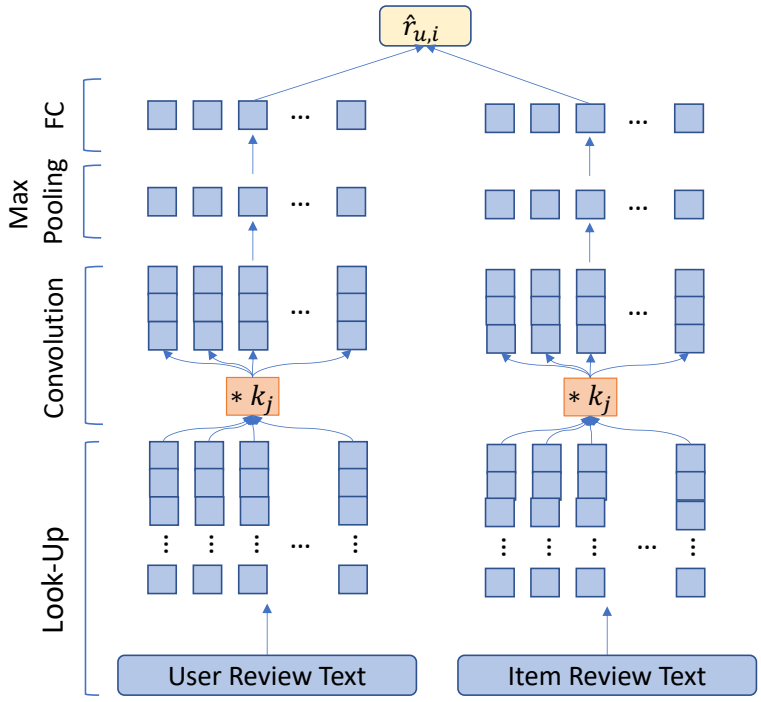


Figure 3.1: The architecture of the DeepCoNN model. Figure adopted from (Zheng et al., 2017).

the users' preferences. The users' posted reviews encapsulate rich semantic information, such as a possible explanation of the users' preferences or a description of specific item attributes (Chen et al., 2015). Therefore, many recommendation models, especially the content-based recommendation approaches, leveraged the reviews data to create user/item representations and to address the recommendation task (Chen et al., 2020; He et al., 2015; Almahairi et al., 2015; Li et al., 2019a; Ni et al., 2019). In particular, many review-based approaches based their recommendations on the semantic similarity between the review content of users and items (Zheng et al., 2017; Chen et al., 2018b), which encodes additional relationships among the users and items. Indeed, the posted reviews by users are valuable in modelling the interactions among users and items from a textual semantic perspective. However, we argue that the existing review-based recommendation techniques ignored the reviews' associated properties (e.g. the reviews' sentiment, length and age) in measuring the usefulness of reviews. In particular, recall the discussion of the users' adoption of information framework in Section 1.2. Users present distinct preferences in using different types of information as per the corresponding information properties while selecting their preferred items. Consequently, differently from the existing recommendation techniques, we introduce these review properties to the review-based recommendation model for improved recommendation performances. In the following, we discuss many typical

and representative review-based recommendation techniques to show how to encode the review data, so as to learn the users' preferences and the items' attributes for recommendations.

To illustrate the use of review text in a deep learning-based recommendation model, we use the DeepCoNN model (Zheng et al., 2017) as an example, which is a representative review-based recommendation model in the literature. We also consider DeepCoNN as a possible baseline to compare to our proposed recommendation framework, so as to show the performances of our proposed review-based recommendation approaches. DeepCoNN was built upon the semantic similarity between the review content of the corresponding user-item pairs. We present the architecture of the DeepCoNN model in Figure 3.1. The DeepCoNN model was devised to tackle the prediction of ratings that users might give to their unseen items (namely the rating prediction task discussed in Section 2.2.1). As for a given user-item pair, DeepCoNN aggregates reviews and concatenates them into individual 'documents' d^u and d^i for the user u and item i, respectively. Then, it models these documents to learn the users' preferences and items' attributes. Next, DeepCoNN applies a popular word embedding technique (i.e. word2vec (Mikolov et al., 2013)) to project the text of reviews into the latent space. We formulate this process as follows:

$$V_{1:l} = \theta(d_1) \oplus \theta(d_2) \oplus \theta(d_3) \oplus ... \oplus \theta(d_l) \tag{3.1}$$

where $V_{1:l}$ is the resulting latent vector of reviews and d_k refers to the k_{th} word within the document d and l is the cutoff for defining the max number of l words. Then, DeepCoNN applies convolution, which we have introduced in Section 2.4, to the latent vectors of review documents. Next, the resulting feature vectors are fed into a fully connected layer to make user/item representation vectors. After modelling the review document via a series of neural networks, for a user-item pair (u,i), DeepCoNN calculates a corresponding feature vector pair (x_u, x_i). To address the prediction of ratings that users would give to the corresponding items, DeepCoNN uses a factorisation machine (Rendle, 2012) to implement the prediction layer that estimates the users' ratings on the target items. We formulate the factorisation machine as follows:

$$\hat{r} = \hat{w}_0 + \sum_{j=1}^{|\hat{z}|} \hat{w}_j \hat{z}_j + \sum_{j=1}^{|\hat{z}|} \sum_{k=j+1}^{|\hat{z}|} \langle \hat{v}_j, \hat{v}_k \rangle \hat{z}_j \hat{z}_k \tag{3.2}$$

where $\hat{z}$ is given by concatenating the calculated representation vectors (i.e. $(x_u \oplus x_i)$). The first summand $\hat{w}_0$ is the global bias. Within the second summand, $\hat{w}_j$ linearly learns the weight of variable $\hat{z}$ at index j. After that, the third summand captures the second order interactions within the feature vector $\hat{z}$ and $\langle \hat{v}_j, \hat{v}_k \rangle = \sum_{f=1}^{|\hat{z}|} \hat{v}_{j,f} \hat{v}_{k,f}$.

In summary, a commonly seen review-based recommendation approach involves three main components: 1) textual content modelling (e.g. the use of word embeddings in DeepCoNN); 2) the feature learning of the users/items; 3) the prediction of the users' preferences on items. Many studies have proposed to develop practical review-based recommendation approaches by varying the implementation of the above three components.

1. Textual Content Modelling

The textual content modelling process focuses on extracting valuable information from the review text to help estimating the users' preferences on items. We group the commonly applied techniques into two families, i.e. the aspect-level and review-level modelling approaches. The first of which captures the users' fine-grained preferences based on the extracted or learned review aspects (Guan et al., 2019; Baral et al., 2018; Cheng et al., 2018b). Indeed, a review expresses users' preferences on different factors. For instance, "*The pizzas are so fresh, the staff is lovely, service is fast and they're so well priced.*" implies a positive feedback on the food, service and price of a corresponding pizza restaurant. We divide the review aspect modelling into two sequential stages: aspect extraction and aspect-level estimation of users' preferences. As for the aspect extraction, several language rules, sequence labelling techniques, and external corpora/toolkits have been used in the literature. For example, Baral et al. (2018) used the term frequency of nouns as well as the dependency parsing (Manning et al., 2014); Zhang et al. (2010) leveraged the conditional random field algorithm; Zhang et al. (2014c) used the public sentiment corpora MPQA[1] and HowNet;[2] the use of toolkit opinion parser (Qiu et al., 2011) in (Bauman et al., 2017); Xu et al. (2021) and Guan et al. (2019) made use of the sentires toolkit (Zhang et al., 2014b). For the recommendation models that used these techniques, their performances highly relied on the used linguistic rules and the usefulness of the used toolkits. However, with the growing volume of reviews, it is difficult for the rule-based approaches and classic toolkits to model the infrequent features (Jin et al., 2009).

Meanwhile, to address the above limitation, related works developed abstractive review aspects to model the users' fine-grained preferences. For example, ALFM (Cheng et al., 2018b) defined review aspects via topic modelling and regarded the probability distribution of topics as 'aspects'. Similarly, Cheng et al. (2018a) proposed a review-based recommendation model, A^3NCF that fused the topic vectors and embedding vectors to define 'aspects'. However, these abstract 'aspects' are not the items' aspects (e.g. service, food flavour of a restaurant). They are the different distributions of the lexical terms in reviews, which still lack of effective way to identify users' preferences from reviews.

[1] http://mpqa.cs.pitt.edu/corpora/

[2] http://www.keenage.com/

On the other hand, another approach to model the textual content consists in review-level techniques, which are commonly implemented via embedding techniques. One-hot embeddings, word2vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014) techniques are three widely used approaches that encode review text into latent space (Dong et al., 2020; Tay et al., 2018; Liu et al., 2019a). Then, the recommendation models use the produced embedding vectors as input to model the information within the review text. However, the above solutions are limited in providing static embedding of review text with weak semantic interpretability (Qiu et al., 2020). Along with the development of natural language processing (NLP) and deep learning techniques, some studies proposed contextual pre-trained embedding techniques, such as ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019), to address the aforementioned concern. For example, inspired by the Cloze task (Taylor, 1953), BERT introduced a 'masked language model' (MLM) as the pre-training objective to learn the contextual pre-trained embeddings of text. Hence, the contextual embedding techniques allow models to capture the semantic information within the review text effectively and are growingly popular in the recommendation field (Zhang et al., 2021; Bai et al., 2021; Shamir et al., 2021).

Compared with the previous textual content modelling technique, namely the review aspect-level modelling, both aspect-level and review-level techniques have pros and cons. As for the aspect-level modelling of the review content, the model can learn fine-grained user preferences on different aspects of a given item. However, the effectiveness of the resulting review-based recommendation models highly rely on the use of the review aspect extraction strategies and the quality of the extracted review aspects. In particular, using a specific review aspect set may also lower the generalisability of a recommendation system. For example, the aspects extracted from reviews on restaurants are different from the aspects that are included in book reviews. However, differently, the embedding techniques can leverage the pre-trained language models and the semantic proximity between lexical terms to effective extract valuable information from the review text (Şenel et al., 2018). Therefore, this thesis will consistently use the embedding techniques to convert review text into latent vectors and pass them as input to our proposed recommendation models.

2. User/item Feature Learning

After modelling the textual content of reviews, a review-based recommendation model needs to learn users' or items' features from the learned textual content of reviews. For example, recall from the description of the DeepCoNN model, it implements the feature learning of users and items from review embeddings via a convolution operation, then uses a max-pooling operation followed by a fully connected layer. However, there are alternative approaches in addressing the feature learning task in the recommendation literature. Indeed, many studies have been

devoted to optimising feature learning to develop effective recommendation models. For example, Hyun et al. (2018) applied a two-staged convolution operation to the review data to make rating predictions. The first stage extracts feature vectors from each review, and then the second stage extracts the features from the learned vectors through convolution operators to provide user/item representation vectors.

Moreover, since the review text can be understood as a sequence of characters or words, Sun et al. (2020b) applied a pre-trained LSTM model to the word embedding vectors of reviews and used the last hidden state as the representation of the input review. Then, Sun et al. (2020b) fused the review representation vector with the user/item embeddings and passed it to MLP layers to make rating predictions. However, these studies ignored the fact that the quality and usefulness of the reviews markedly vary with the increasing number of users and the available reviews they posted online (Chen et al., 2018b).

Therefore, many studies started investigating the different usefulness of reviews and how it contributes to the improvement of models' recommendation performances. For example, Chen et al. (2018b) applied an attention mechanism to estimate the usefulness of different reviews. Then, Chen et al. (2018b) concatenated the weighted review embeddings with the one-hot embedding vectors of users/items as the feature vectors that encode the users' preferences and items' attributes. Recently, the attention mechanism has been adopted in other recommendation studies to model the review input (Liu et al., 2020a, 2019c; Cheng et al., 2018a). For instance, Li et al. (2019a) introduced a recommendation model (called RNS), which includes a sequence of convolution operators applied over the users' posted reviews on their recently interacted items. Then, RNS uses two consecutive attention layers to model the learned feature vectors in order to build the representation vector of the users' recent preferences. Some studies also considered the attention mechanism to examine the usefulness of the review content over distinct granularity levels: Tay et al. (2018) applied attention to identify the valuable review input via modelling at word and review levels; Xing et al. (2019) investigated the distinct impacts of parts of the review content at word and sentence levels on the model's rating prediction performance.

However, we argue that a limitation of the classic attention mechanism when applied to model the review input is that it captures the usefulness of reviews by relying on historical data, which often does not generalise to reviews that are unseen by the trained model with infrequent features (Veličković et al., 2018). To address the above concern and produce better user/item features, as we discussed in Chapter 1.3, we propose a recommendation framework that leverages the reviews' associated properties to model the usefulness of reviews, so as to make accurate recommendations. In particular, to show the performances of our review property-based recommendation framework, we also compare our proposed approaches to the attention mechanism-

based recommendation techniques, e.g. (Chen et al., 2018b). We detail the description of it in the next chapter.

3. User Preferences Prediction

Thus far, we have discussed the various widely used techniques in learning the review data and generating the features of users/items. In the following, we explore the related works in the literature that considered different techniques of estimating the users' preferences on items by leveraging the learned user/item features. Essentially, we regard most preference estimation approaches as a similarity measurement between the feature vectors of users and items. In particular, we review the approaches that have been applied in the literature to examine the users' preferences and items.

First, recall the description of the example recommender (namely DeepCoNN). It used the factorisation machine (Equation (3.2)) to model the user and item feature vectors to address the rating prediction. The *factorisation machine* has been widely applied (Guo et al., 2017a; Tay et al., 2018; Baral et al., 2018) to address the score prediction problem – either the ranking or rating scores – because of its ability in modelling the complex relationships in the data input.

On the other hand, together with the development of the deep learning techniques, many studies have devised various neural networks to implement a non-linear projection of the learned user/item features to address the users' preference estimation. For example, Tang and Wang (2018) and Li et al. (2017) built multiple fully-connected hidden layers to non-linearly transform the learned feature vectors and then used a regression function to generate the score for ranking or rating the items. However, as argued by Rendle et al. (2020), learning an effective multi-layer perceptron function to model the complex relationships in the data requires a huge number of parameters to learn and using enormous training data to obtain the target function. Instead, the *dot product* function is more efficient in dealing with the similarity measurement between a given user and item to predict the users' preferences on items (Rendle et al., 2020).

As a consequence, the dot product, or in general the inner-product, has become the top choice when considering an approach to make the users' preference estimation upon the available user/item feature vectors (Shi et al., 2018; Hyun et al., 2018; Cheng et al., 2018b; Chin et al., 2018). We are aware that there are some other distinct approaches (e.g. element-wise product (Chen et al., 2018b), cosine similarity (Zhu et al., 2019) and customised sigmoid function (Li et al., 2019b)) in the literature to calculate the distance between the user and item features. However, in this thesis, we use the dot-product function to calculate the scores of users' preferences on items, so as to address the ranking-based recommendation task, since it is the most commonly used function.

On the other hand, according to the above discussions, we found that many review-based

recommendation models were designated to address the classic rating prediction task when dealing with the users' preference prediction. These recommendation models include many widely cited approaches, such as DeepCoNN (Zheng et al., 2017), NARRE (Chen et al., 2018b) and D-Attn (Seo et al., 2017), all of which provide baselines with which any new review-based recommendation approaches ought to be compared against.

Recall from the thesis statement in Section 1.3, our proposed recommendation models need to accurately infer the reviews' associated properties from reviews. Then, to show the effectiveness of the inferred reviews' properties when applied to the recommendation approaches, we first compare our models to the existing review-based recommendation baselines in addressing the rating prediction task. Next, after aggregating the reviews' associated properties, we aim to develop effective ranking-based recommendation approaches.

Thus far, we have explored many review-based recommendation approaches in the literature. In particular, we showed the advantages of modelling review data via embedding techniques, using attention mechanisms and the dot product function on top of it. Attention mechanisms have been widely applied in the field to deal with review modelling and user/item feature generation. However, arguably, the classic attention mechanisms rely on the historical trained data and have a difficulty in capturing the usefulness of unseen reviews with infrequent features. To address such a limitation, in this thesis, we propose to leverage the reviews' associated properties to better estimate the usefulness of reviews to predict the users' preferences on items. In particular, to show that we correctly infer the reviews' associated properties, we compare our proposed approaches that encode the inferred reviews' properties to the existing well-cited baseline approaches in addressing the rating prediction task.

Moreover, in the following, we explore the recommendation approaches that considered the reviews' associated properties or characteristics as side information to model the users' preferences and items' attributes.

Recall that in Chapter 1, we call a 'review property' a characteristic attribute of the reviews. In Figure 3.2, we show some possible review properties of an example review. The example shows a recent review (30/03/2020) that briefly discusses (in 90 words) a user's positive opinion (rating 5 and positive sentiment) regarding their experience of interacting with a given item. Many other customers also consider this review helpful (14 helpful votes). Several studies, such as (Wang et al., 2018c; Sahoo et al., 2012; Liu et al., 2021), have recognised the value of reviews' associated properties in modelling data to capture the users' preferences on items. In the following sections (i.e. Sections 3.2, 3.3 and 3.4), we explore the related works that used review sentiment, time and helpfulness, which are commonly used, to show the existing techniques of applying review properties into the development of recommendation approaches.

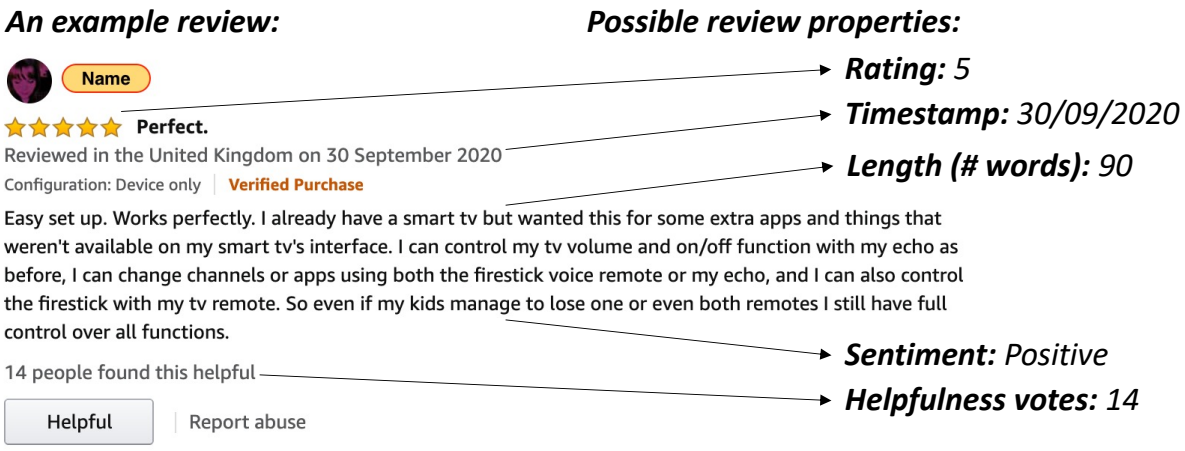


Figure 3.2: Review property examples.

3.2 Using Review Sentiment in Recommendation

The review sentiment encodes a user's view or opinion towards a specific experience of their interaction with an item. In particular, the review sentiment tells a positive sentiment if the users' real experiences align with the users' ideal expectations. Otherwise, it shows a negative opinion. For example, for a review of having dinner in a restaurant, users might evaluate various aspects of their experience, including the environment, the service, the offered food in the menu, the price, the served food and beverages, etc. Then, within the corresponding review, the user may express their personal experience by telling the level of their satisfaction (i.e. sentiment) on some focused aspects of the restaurant. Therefore, it is vital to identify the sentiment of reviews to capture the users' preferences. In the recommendation literature, many studies (Ziani et al., 2017; Wang et al., 2018c; Yun et al., 2018) have devoted themselves to leveraging the sentiment of reviews to develop approaches in making customised recommendations for users. Note that, since users' explicit ratings also present users' sentiment regarding their interaction experience with items, in this thesis, we include the users' explicit ratings as a type of sentiment information within the users' posted reviews.

There are many sentiment analysis strategies in the literature that have been applied to identify the users' opinions within reviews. We categorise these strategies into (1) overall sentiment orientation and (2) aspect-level sentiment. The overall sentiment orientation indicates an averaged or summarised users' opinions towards a given item. On the other hand, the aspect-level sentiment gives a fine-grained description of the users' view about various extracted aspects of a given item (e.g. the service quality of a restaurant and the sound quality of a CD product).

(1) Sentiment orientation-based approaches

We start with a brief overview of the existing sentiment analysis tools. Following (Medhat et al., 2014), we group the sentiment analysis approaches into two categories: (1) lexicon-based approaches and (2) machine learning techniques. The first of which are based on a set of pre-aggregated sentiment lexicons or sentiment terms and then they use statistical or semantic methods to calculate the sentiment orientation of a given text. The second type of sentiment analysis approaches, namely machine learning techniques, involves various learning algorithms to model the features within a text to identify its associated sentiment.

Lexicon-based approaches considered various available resources to support the analysis of the sentiment orientation of a given text (e.g. WordNet (Miller, 1995) and thesaurus (Mohammad et al., 2009)). For example, Yang et al. (2013) took the NTLK toolkit (Bird, 2006) and SentiWordNet3.0 (Baccianella et al., 2010) into account to model the users' posted tips (i.e. a brief users' feedback in few sentences) to learn and extract the users' historical preferences on items. In particular, the learned sentiment scores are further adopted to build the user-item preference matrix as input to a matrix factorisation model to make rating predictions. Similarly, Kumar et al. (2020) applied a lexicon and a rule-based sentiment analysis tool (i.e. VADER (Hutto and Gilbert, 2014)) to classify the sentiment ratings of tweets – a type of microblogging data without associated ratings – on a scale of 0-10. In particular, they leveraged the learned sentiment scores to calculate the similarity between items. Wang et al. (2018a) developed a location-based recommendation system, which considered the NRC Word-Emotion Association Lexicon dictionary (Mohammad et al., 2013) to analyse geotagged tweets to extract the users' emotions on green spaces. By analysing the sentiment of tweets, the system captures a general reputation of a given geographical region. The resulting system allows users to obtain personalised recommendations of green spaces in real-time and interactively. However, similar to the review text modelling approaches that we introduced in Section 3.1, the main limitation of the lexicon-based techniques is their weak generalisability. Indeed, they based their sentiment analysis on available toolkits or previously learned knowledge, making it difficult to learn from infrequent features to estimate users' sentiments accurately.

On the other hand, different from the lexicon-based approaches, machine learning-based techniques automatically extract valuable information from the review text to predict users' sentiments. In particular, machine learning-based approaches have grown rapidly in conducting effective sentiment analysis. The common applied machine learning techniques are Bayesian methods (Kang et al., 2012; Ortigosa-Hernández et al., 2012), linear classifiers (e.g. support vector machine (Joachims, 1996; Li and Li, 2013)) and deep learning techniques (Gao et al., 2019; Dang et al., 2021). For example, Ziani et al. (2017) used the Support Vector Machine with a set of extracted features (e.g. number of positive words, number of negative words, and

number of adverbs) to learn the sentiment information within reviews. Then, Ziani et al. (2017) addressed the users' preference estimation by considering the opinions of the users' neighbours. Such opinions are presented with the sentiment polarity scores of neighbours' associated reviews and calculated by the learned SVM. In particular, with the advances in machine learning, it has been shown that the machine learning-based approaches can provide leading performances in sentiment analysis (Gao et al., 2019). For instance, Dang et al. (2021) introduced a state-of-the-art rating prediction model by leveraging a pre-trained BERT model to initialise the text embeddings and built a sequence of neural networks (i.e. CNN and LSTM networks) on top to estimate the sentiment orientation of a given text. In particular, the rating prediction is made by enriching a CF model with the learned sentiment scores from reviews enrich the rating data. According to the discussions above, the machine learning-based techniques (e.g. the CNNs model and the pre-trained BERT (Devlin et al., 2019) model) can effectively predict the sentiment orientation of a given review text. However, in the recommendation literature, no study leverage the learned sentiment information to estimate the usefulness of reviews. In this thesis, we aim to fill this gap by weighting the usefulness of reviews according to the learned sentiment orientation of reviews.

(2) Aspect-level sentiment analysis

Apart from the sentiment orientation-based approaches, many recommendation models (Da'u and Salim, 2019; Yun et al., 2018; Zhang et al., 2017) emphasised the usefulness of aspect-level sentiment analysis to obtain fine-grained user opinions on items as well as effective and explainable recommendation results. For example, Da'u and Salim (2019) proposed a sentiment-aware recommendation model (called SDRA), which assumes that each review comprises two components: 1) m aspect topics and 2) k sentiment topics with positive and negative polarities. Therefore, by applying the topic modelling technique and the fully connected neural network, SDRA can obtain the aspect and sentiment aware representation latent vectors for both users and items. Then, SDRA feeds the learned representation vectors into a factorisation machine to compute users' preferences on unseen items. In particular, many studies also leveraged more recent deep learning techniques to model the fine-grained sentiment expressed by users in review texts (Baral et al., 2018). For example, Baral et al. (2018) proposed a recommendation model (i.e. ReEL) which devised a CNN-based sentence-aspect classifier to extract the users' opinions on relevant aspects (e.g. food flavour, environment, and service) within review sentences. By aggregating the learned features from the aspect-aware modelling of reviews and other contextual features, ReEL made comprehensive estimations of the users' preferences over items. Aspect-aware sentiment analysis approaches are effective in providing explainable recommendation results (He et al., 2015; Baral et al., 2018). We agree that aspect-aware sentiment analysis could effectively

estimate the fine-grained users' preferences on different aspects of items. However, it isn't easy to obtain the ground truth of users' sentiment on item aspects (Huang et al., 2020). Therefore, it is challenging to estimate the usefulness of reviews according to the aspect-aware sentiment analysis of reviews.

In summary, we categorised the sentiment analysis into sentiment orientation-based and aspect-level sentiment analysis approaches. The sentiment orientation-based techniques can effectively identify the overall sentiment polarity compared to the aspect-level sentiment analysis approaches. In particular, it is challenging to estimate the reviews' fine-grained aspect-level sentiment accurately. Therefore, in this thesis, we consider the learned sentiment polarities as a type of valuable side information. However, in the recommendation literature, such learned sentiment polarities of reviews have not been considered in measuring the usefulness of reviews to enhance the recommendation performance. Moreover, given the high performances of machine learning techniques in sentiment analysis, in this thesis, we use the machine learning-based approaches to model the overall sentiment expressed by users in the review text, so as to estimate the usefulness of reviews.

3.3 Using Review Timing in Recommendation

Aside from the review sentiment discussed in the previous section, many studies have widely considered the review timing property. As shown in Figure 3.2, a given review has an associated timestamp that indicates when the review was posted. In particular, this review timing also points out to the approximate time of the user-item interactions. The review timing has been used as an essential information, in the literature, to model the users' behaviour while interacting with items. Indeed, many studies have devised various techniques to use the review timing in different manners. In particular, for the review timing-aware recommendation approaches in the literature, we propose to classify those existing approaches into sequential recommenders and context-aware recommenders. The sequential recommenders emphasise the modelling of the users' recurrent and sequential behaviour while interacting with items. On the other hand, the context-aware approaches add the review timing as side information to the other available data input, which allows a model to be aware of when the interaction happened and users' latest preferences.

(1) Sequential Recommenders

Recall that in Section 2.1, an effective recommendation model aims to correctly predict the users' preferences on their unseen items. This indicates a potential temporal linkage between

the users' historical interactions and the users' future behaviour. In particular, a recommendation model can identify the sequential pattern within the user-item interaction data by leveraging the review timing. Therefore, various sequential recommendation techniques have been proposed, which use time, based on the learned sequential pattern from the user-item interactions. The early contributions (Sahoo et al., 2012; Rendle et al., 2010; Kanagal et al., 2012) to sequential recommendation mainly started with the use of Markov Chains (Gagniuc, 2017), which is a stochastic model that predicts the probability of the event that will happen next solely based on the previous event. An example model is Factorizing Personalised Markov Chains (FPMC) (Rendle et al., 2010). FPMC calculates the probability of the next item that the user is going to visit by considering the Markov chains technique defined as follows:

$$p(i \in B_t | B_{t-1}) := \frac{1}{|B_{t-1}|} \sum_{l \in B_{t-1}} p(i \in B_t | l \in B_{t-1}) \tag{3.3}$$

In the above definition, FPMC introduces the notion of *basket* when modelling of the sequential interactions between users and items. A *basket B* refers to a set of items at a time t. Therefore, FPMC predicts if an item will be interacted in the next *basket* given the previous set of interacted items (i.e. B_{t-1}). In particular, to avoid independent modelling of transitions, FPMC also adopts the factorization model to capture the transition probability between the previously interacted with and the next items for the given users. This improvement extends the basic Markov Chains from focusing on the most recent interaction to considering the impact of the users' long-term interactions. In particular, to better learn from the users' historical interactions, a higher-order Markov chain has been proposed and implemented to learn the users' preferences (Kanagal et al., 2012; Zhang and Nasraoui, 2007).

Markov Chains-based techniques underline the importance of capturing the transition between the users' sequential interactions. They have also been shown to be effective when applied to model users' sequential behaviour. However, a Markov chain still highly relies on its assumption that the next state is determined only by the current state. Many studies have also extended the basic Markov Chains to a higher-order version or leveraged the factorisation model like FPMC (Rendle et al., 2010) to address the above limitation. However, the Markov Chain-based approaches can still only model the local sequential behaviours between users' adjacent actions, without learning a global information from user-item interaction sequences (Liu et al., 2018b). Therefore, to better address the above concern, the next generation of sequential recommenders started to incorporate the recurrent neural networks (RNNs), a neural network model capable of processing data input in variable length. Compared with the Markov Chain-based techniques, a main advantage of a RNNs model is its ability to 'memorise' (Sherstinsky, 2020).

Indeed, we provided a detailed description of RNNs and their commonly used variants in Section 2.4. In the literature, the RNNs and their variants have been widely used to address the pattern recognition of the users' sequential behaviour and predict their next item of interest (Yu et al., 2016; Villatel et al., 2018; Devooght and Bersini, 2017). For example, Yu et al. (2016) devised a RNNs-based sequential recommendation model (i.e. DREAM) to capture the users' dynamic preferences over time. RNNs calculate a representation (i.e. the hidden state) of the users' preferences after modelling each data input in a sequence, as follows:

$$h_t^u = f(X \cdot b_t^u + R \cdot h_{t-1}^u) \tag{3.4}$$

where $X \cdot b_t^u$ calculates the representation of the users' current interest based on the learned feature from session b_t^u. X is a transition matrix to convert the representation of sessions into the users' interest. In particular, h_{t-1}^u is the learned user's preferences from the previous interactions. Therefore, by leveraging the learned dynamic representations, DREAM estimates the users' favoured items at each time and predicts their future preferences. However, it is difficult for an RNNs-based recommendation model to model the long term dependency among the user-item interactions due to the introduced gradient vanishing and exploding problem in Section 2.4. In the literature, the RNNs-variants (i.e. LSTM and GRU) have been commonly used in sequential recommenders (Hidasi et al., 2015; Wu et al., 2017) to address this limitation. However, in the literature, it has been discussed that it is also difficult for these techniques to identify the useful user-item interactions in a sequence. Therefore, recently, the attention mechanisms (Vaswani et al., 2017) have been commonly applied to combine with the RNNs-based models to capture the critical hidden states in a sequence. AttRNN (Chen and Li, 2019) is an example, which applies the dot-product attention mechanism over the learned hidden states to calculate the updated latent vectors of the users' dynamic preferences. The corresponding function is:

$$o_t^u = f(W[h_t^u, c_t^u]) \tag{3.5}$$

where c_t^u is obtained by applying the dot-product mechanism (Equation (2.21)) over the corresponding hidden state (i.e. h_t^u). Similarly, Zeng et al. (2019) also applied the dot-product mechanism as well as a position encoder over the GRU-processed sequential data input to enhance the representation of the users' preferences.

Among these RNNs-based and attention mechanism-based techniques, most leveraged the review timing as ground truth to order the user-item interactions in a sequence. However, these approaches did not consider the review timing as a factor in measuring the usefulness of the data

input from user-item interactions (e.g. the users' posted reviews). On the other hand, there is another group of common review timing-based approaches, which leverage the review timing is also an essential contextual factor to estimate the users' preferences on items.

(2) Context-aware Recommenders

The context of user-item interactions plays an important role for a recommendation system in modelling the users' preferences. In particular, the review timing is one of the most commonly used contextual factors to estimate the users' preferences and to make effective personalised recommendations. However, different from the sequential recommenders mainly based on a set of techniques, various techniques have been proposed to encode the review timing into modelling the users' preferences. For example, Manotumruksa et al. (2018) proposed to encode the review timing via the one-hot embedding technique and used the learned latent vector as input to learn the users' preferences at a given time step. In particular, in the literature, the recommendation studies not only used the review timing to index the user-item interactions in different time steps, but also used it to model the user-item interactions in different manners. For example, Ye et al. (2020) investigated two temporal patterns: (1) absolution time pattern and (2) relative time pattern. The first captures temporal impact at a certain time point, while the relative time pattern investigates how the time interval between interactions influences two interactions. For the absolute time pattern, Ye et al. (2020) adopted an embedding technique, like that of of (Manotumruksa et al., 2018) to transform the temporal value t into latent vectors ϕ_t. However, Ye et al. (2020) mapped the temporal embedding into a pre-defined set of latent groups to avoid distinct granularity of bucketization. Note that bucketization is the operation that discretises continuous variables into discrete values. On the other hand, the relative time pattern modelling relies on the computed embeddings of time steps from the first stage. Then, Ye et al. (2020) leveraged the subtracted latent vectors of two steps to represent the corresponding time interval (i.e. $\Delta\phi_{t,t+1} = \phi_{t+1} - \phi_t$). Another example of leveraging the review timing as context is (Wang et al., 2020a), which proposed a temporal kernel function to encode the temporal dynamics of different relations (e.g. complementary and substitutional items) between the interacted items as well as to manage the impact of the previous relational items.

Thus far, we have discussed two main families of review timing-aware recommendation studies: sequential recommenders and context-aware recommendation systems. The context-aware recommendation approaches – such as (Ye et al., 2020; Wang et al., 2020a) – have considered the review timing to differ the impact of the users' recent and old interactions with items to the estimation of the users' preferences on items. However, to the best of our knowledge, there is no review timing-aware recommendation approaches in the literature that connect the review timing to the usefulness of reviews. Recall that, in this thesis, we aim to model the

personalised usefulness of reviews via the use of different review properties – including the review timing – to better capture the users' preferences and make effective recommendations. In particular, to show the benefits of our proposed techniques, we also compare them to the recent review timing-aware approaches, such as DREAM (Yu et al., 2016) and CASER (Tang and Wang, 2018). Aside from both the sentiment and temporal properties, existing recommendation studies also considered the review helpfulness, which has been frequently used to examine the usefulness of reviews (Kim et al., 2006; Lu et al., 2010). In the next section, we explore the use of review helpfulness in the development of recommendation approaches.

3.4 Using Review Helpfulness in Recommendation

As shown in Figure 3.2, reviews can have associated helpful votes to indicate their helpfulness to other users in making decisions. However, the helpfulness of reviews has not seen much coverage in the literature for developing recommendation models. Indeed, some recommendation approaches leveraged the review helpfulness to model the quality of reviews to develop review-based recommendation approaches (Li et al., 2021a; Liu et al., 2021). For example, Li et al. (2021a) argued for the importance of excluding unhelpful/noisy reviews to learn the users' preferences as well as the items' attributes from reviews. Therefore, the contributions of (Li et al., 2021a) are twofold. First, Li et al. (2021a) constructed a review helpfulness classifier to divide reviews into two classes (i.e. helpful and unhelpful reviews). At the second stage, Li et al. (2021a) kept only the helpful reviews and used them to model the user-item interactions within a recommendation system. On the other hand, Liu et al. (2021) proposed a multi-task learning framework, DARMH, to deal with the rating prediction as well as the review helpfulness classification tasks. As for the rating prediction task, DARMH used the helpfulness scores of reviews to measure the importance of reviews to address the rating prediction task. The experimental results have shown that by varying the ratio of helpful reviews, DARMH exhibited different rating prediction performances with a wide margin. In particular, DARMH can obtain a lower rating prediction error rate with more helpful reviews.

According to the example studies (Liu et al., 2021; Li et al., 2021a), we see that the use of the reviews' helpfulness highly relies on the ground truth of reviews' helpful votes and an accurate review helpfulness classifier. Indeed, both Liu et al. (2021) and Li et al. (2021a) analysed the helpfulness of reviews via a trained review helpfulness classifier on the available reviews' helpful votes. However, the use of the reviews' helpfulness property commonly suffers from a widely discussed challenge: *sparse review helpfulness ratings* (Tang et al., 2013; Moghaddam

et al., 2012; Hong et al., 2017). Therefore, in the literature, various review helpfulness classification models have been proposed by leveraging different review features (such as review length, percentage of nouns). Specifically, we classify such prior studies into five categories with varying features, including structural features, lexical features, syntactic features, metadata features, and contextual features.

1. Structural Features: These features capture the structure and formatting of the users' comments. Many studies consider structural features as strong features in detecting helpful reviews. Liu et al. (2007) and Lu et al. (2010) leveraged the average sentence length and the number of sentences; Kim et al. (2006) explored the effectiveness of the length of comments, the percentage of question sentences and the number of exclamation marks. In general, these studies agree that the length of the reviews is one of the most effective features in review helpfulness classification.

2. Lexical Features: Lexical features analyse the words used in the comments. Both Kim et al. (2006) and Tsur and Rappoport (2009) used the TF-IDF score of each word and each bigram as features. Moreover, many deep learning approaches use word embedding for word representations. Word embedding-based approaches have been adopted by Chen et al. (2018a, 2016) and have been shown to have a better expressiveness than other hand-crafted features.

3. Syntactic Features: These features capture the linguistic properties of the users' comments. Kim et al. (2006) investigated the effectiveness of different syntactic tokens including the percentage of nouns, the percentage of verbs, the percentage of adjectives and the percentage of adverbs. In addition, sentiment words were considered by Yang et al. (2015), who obtained significant improvements compared to using simple lexical features.

4. Metadata Features: These features focus on the relationship between review helpfulness and user ratings. Both Kim et al. (2006) and Huang et al. (2015) found a positive correlation between review helpfulness rating and review star ratings.

5. Contextual Features These features mainly focus on the review behaviour of review writers as well as the connection between the review writer and readers. Huang et al. (2015) investigated the historical review helpfulness ratings of the review writers and Lu et al. (2010) studied the influence of the connection between the review writers and readers on the review helpfulness ratings. They concluded that the contextual features contribute to enhance the accuracy of the review helpfulness classification. However, these contextual features are harder to obtain than the other previous types of features.

In Chapter 6, we further investigate the development of effective review helpfulness classifiers to alleviate the sparsity of review helpfulness labels. In particular, we use some representative features from the aforementioned listed feature sets (e.g. structural and lexical features)

and apply them to classic and state-of-the-art classification models (e.g. support vector machine, CNN/BERT-based classifier). Then, we use the proposed classifier to predict the helpfulness labels of reviews as input to a recommendation approach.

In summary, we showed that the helpfulness of reviews had been merely investigated in the literature while developing recommendation approaches. In particular, the existing work that took the reviews' helpfulness into account mainly focused on the helpful reviews and ignored the unhelpful reviews, which could result in an information loss and a biased estimation of the users' preferences (Li et al., 2021b). On the other hand, the existing review helpfulness classification approaches also suffer from the limitation of sparse review helpfulness ratings. Therefore, in comparison to the existing recommendation approaches, this thesis aims to address the challenge of using the sparse ratings to correctly estimate the reviews' helpfulness, which is then applied to weight the usefulness of reviews for developing recommendation models.

Thus far, we have discussed three commonly available properties of reviews. These properties can be found in several public recommendation datasets. According to the discussed related work in Sections 3.2, 3.3 and 3.4, it has been shown that by incorporating the reviews' associated properties, such as review sentiment, timing and helpfulness, recommendation approaches can improve their performances in making accurate recommendations. These review properties allow models to capture the usefulness of reviews, the information quality, and the users' dynamic preferences etc.

However, as we can see from the above discussion, most of the existing contributions leveraged one specific property of reviews at a time and ignored the impact of taking various other properties of reviews into the development of recommendation approaches. Therefore, it is still unclear: (1) how to use multiple review properties in an end-to-end recommendation framework and (2) how to learn the users' preferences and model the users' interactions with reviews exhibiting different properties for a recommendation model.

Recall that in Chapter 1, the contributions of this thesis are motivated by an existing user behaviour modelling framework (i.e. the users' adoption of information framework). Therefore, in the following, we explore many user behaviour modelling strategies in the literature that modelled the users' behaviour while interacting with the review data and items.

3.5 User Behaviour Modelling

It is fundamentally vital for a recommendation system to discover and model the users' behaviour while interacting with the items, so as to identify the items of interest (Beheshti et al.,

2020). Therefore, in the literature, many studies have been devoted to investigating the users' behaviour patterns to guide the data mining and contribute to learning the users' decision-making process. In particular, in the following, we classify the commonly used users' behaviour patterns in the recommendation literature into two classes: 1) conscious behaviour patterns and 2) unconscious behaviour patterns.

Users' conscious behaviours are responses given by the users when these users are aware of their made judgements (Boag, 2017). In other words, users are aware of the answers they are offering, and the attitude to value of items can be measured directly by a self-report (Hong et al., 2021). To model the conscious behaviour patterns, we need to capture the users' conscious responses and examine how they can contribute to the development of recommendation systems. Various types of users' conscious responses have been used in the recommendation literature. For example, the users' ratings and reviews are two commonly seen responses, where users are aware that they are making judgement about the value of an item. These two types of conscious responses have been leveraged as input to develop recommendation systems to predict the users' preferences (Wang et al., 2014; Liu et al., 2021; He et al., 2015; Zhang et al., 2014b), and have been discussed in Sections 2.3 and 3.1. However, as discussed in (Hong et al., 2021), the conscious responses of users are likely to be incorrect, inconsistent or even false reactions.

Different from the conscious responses, the users' unconscious behaviour patterns are extracted from the users' behaviour without conscious awareness, which is less likely to be falsely reported by a given user. As for the unconscious behaviour patterns, they capture the users' unconscious biases while they are interacting with items (Hahn et al., 2014). As a consequence, many studies have developed their recommendation models based on the users' unconscious behaviour patterns. For example, the users' social connections (Fan et al., 2019; Sun et al., 2020a; Zhou et al., 2020b; Liu et al., 2020c) have shown to play a vital role in the users' decision making process (Sinha et al., 2001). Indeed, users tend to present similar preferences on items with their friends (Ma et al., 2008). Another example is the users' browsing behaviours, such as the users' browsing history, clicked items and the time spent on the information of different items, which have been frequently used in the development of many recommendation systems (Ou et al., 2021; Velàsquez and Palade, 2007; Garcin et al., 2013). However, to the best of our knowledge, there is no existing recommendation approach investigating the users' unconscious behaviour of interacting with online reviews, which is essential to develop our proposed recommendation framework to make accurate recommendations. Note that many types of users' unconscious behaviours are also noisy to be used in developing the corresponding techniques. For example, the effectiveness of click models, which estimate the probability of clicking documents by users in the information retrieval domain, has been shown to be influenced by various types of pre-

sentation bias (e.g. position-bias) (Craswell et al., 2008; Yue et al., 2010). Nevertheless, many studies still consider the users' unconscious behaviours as an invaluable source of information, such as using the click logs as relevant signals linking queries and documents (Craswell et al., 2008). Therefore, it is vital to introduce appropriate techniques to leverage such users' unconscious behaviours and avoid the possible included biases. In this thesis, instead of focusing on predicting the exact types of online reviews users would like to interact with, we aim to leverage and model the users' distinct preferences in using reviews exhibiting different properties (i.e. the users' unconscious behaviour) for the down-stream personalised recommendations.

On the other hand, in the literature, the users' adoption of information framework is another technique that models users unconscious behaviour, when interacting with online information, such as the online reviews. Indeed, the information adoption concerns how consumers modify their behaviour by making use of the suggestions made in online reviews (Sussman and Siegal, 2003; Filieri and McLeay, 2014). The communication routes and the customers' involvement in a consumer opinion sharing website might persuade a customer to visit a particular destination or purchase a specific product (Tang et al., 2012). A number of user studies have examined various properties of reviews that influence the users' adoption of information (Chan and Ngai, 2011; Filieri and McLeay, 2014; Wu and Shaffer, 1987; Qahri-Saremi and Montazemi, 2019; Cheung et al., 2012). These studies observed that the properties of reviews are correlated to the level of the users' adoption of the information in those reviews – indeed, such correlations are important motivations for our present work. For example, Filieri and McLeay (2014) used the Elaboration Likelihood Model (ELM) (Petty and Cacioppo, 2012) to connect the factors and properties of reviews (e.g. review length, review timing, review credibility) to the users' adoption of information. In particular, some users would prefer to process information about a product that simply has a good overall rating. On the other hand, some other users would consider in-depth information to make decisions (e.g. review accuracy, review timeliness). Furthermore, Sussman and Siegal (2003) considered the information usefulness as a mediator between the exploration of information and the information adoption by users and showed a strong linkage between the usefulness of the information and the users' decision making.

Recall from Section 1.3, in this thesis, we argue that, by leveraging the users' preferences in relation to the reviews' properties, a review-based recommendation model can obtain improved recommendation effectiveness. The use of review properties provides additional insights about the users' information processing behaviour and their personalised preferences on the items' attributes as conveyed by the reviews' properties. In particular, as we discussed above, the users' adoption of information framework also argues that users tend to consider different types of reviews to support their choices. However, no existing study has explored the application

of the users' adoption of information in the recommendation domain. Therefore, we aim at developing an effective ranking-based recommendation framework that builds on top of the users' adoption of information framework.

3.6 Conclusions

In this chapter, we first discussed the various review-based recommendation studies according to their differences in textual content modelling, user/item feature learning, and the users' preferences prediction (Section 3.1) . We compared the existing recommendation models that modelled the users' preferences in different ways. In particular, we concluded that, according to the comparison between different review-based recommendation techniques, we leverage the embedding techniques to model the review text in this thesis. Then, we propose to leverage the reviews' associated properties to model the usefulness of reviews. Next, we consider applying the dot-product function to the learned user/item features to predict the users' preferences on items in addressing the ranking-based recommendation task. Moreover, we also explored the application of various commonly used review properties in the recommendation literature (Sections 3.2, 3.3 and 3.4). In particular, we concluded that it is still unclear how to leverage reviews' associated properties to identify the usefulness of reviews to different users. Furthermore, to build connections between the reviews' associated properties and the usefulness of reviews, we explored the users' behaviour modelling techniques in the recommendation literature (Section 3.5). We showed the benefits of capturing the users' unconscious patterns compared to the conscious ones. In particular, among the users' unconscious patterns, the users' adoption of information framework agrees with our argument that users tend to consider different types of reviews to support their choices. Therefore, this thesis aims to develop an effective recommendation framework to make effective recommendations on top of the users' adoption of information framework. In the next chapter, we describe our proposed recommendation framework and discuss the challenges to address within each framework component.

Chapter 4

A Review-based Recommendation Framework

4.1 Introduction

Recall from Section 1.1 that this thesis aims to leverage reviews, as well as their associated properties to develop effective review-based recommendation systems. Indeed, as we mentioned in Chapter 3.1, reviews posted by users have played an important role when estimating the users' preferences on items. In particular, in Sections 3.2, 3.3, and 3.4, we have shown that many studies considered the reviews' associated properties to improve the performances of existing recommendation models (e.g. the timing property of the reviews is used in sequential recommendation systems (Ye et al., 2020; Wang et al., 2020a)). However, to the best of our knowledge, no study in the literature took the reviews' associated properties into account to tailor the estimation of the usefulness of reviews according to the users' preferences. To fill the above research gap, in this chapter, we propose a review-based recommendation framework, which consists of four main components: (1) data preparation, (2) review property extraction, (3) review property encoding and (4) user preference estimation. For each of these four components, we describe its functionality and how it contributes to estimating the users' preferences on items effectively.

In the rest of this chapter, we first state the problem we are addressing in this thesis in Section 4.2, which is to propose a ranking-based recommendation framework that leverages both reviews and their associated properties. The main goals of developing a review-based recommendation framework that leverages review properties are: (1) to capture the users' preferences towards reviews exhibiting different properties; and (2) to accurately learn the users' preferences and items' attributes within the selected reviews, so as to improve the performances of

recommendation models in ranking the users' preferred items in the top. Next, in Section 4.3, we discuss how the behaviour of users interacting with different types of reviews and their properties exhibit users' preferences. After that, to summarise the available review properties in the the public review datasets (e.g. the Amazon and Yelp datasets), in Section 4.4, we propose a taxonomy of review properties that we consider in this thesis. Next, to leverage the reviews as well as the introduced taxonomy of review properties in recommendation approaches, we propose a review-based recommendation framework. In particular, in Section 4.5, we provide an overview of the proposed recommendation framework and illustrate the functionality of each of its components in a stepwise manner. Finally, in Section 4.6, we discuss the main challenges of the proposed recommendation framework and the tackled research questions.

4.2 Problem Statement

As denoted in Chapter 1, in this thesis, we aim to develop top-n ranking-based recommendation models and base our models' architectures on deep learning algorithms. In particular, we aim to leverage the users' posted reviews and their associated reviews as additional information to the user-item interactions, so as to better learn the users' preferences on items.

First, we describe the problem statement of the ranking-based recommendation task that we aim to address in this thesis. As we discussed in Section 2.2.2, a ranking-based recommendation model aims to rank items for users according to their preferences effectively. A recommendation task involves connecting two key entity types, namely: the set of users $U = \{u_1, u_2, ..., u_N\}$ with size N and the set of items $I = \{i_1, i_2, ..., i_M\}$ with size M.

To address the ranking-based recommendation task, we aim to estimate the users' preferences on items accurately so that we rank the items that a given user might find the most interesting in higher ranks by estimating the users' preferences. In particular, we denote the users' interacted items as $i^+ \in I^+$ and their unseen items as $i_- \in I^-$. We expect that an effective ranking-based recommendation model can learn the users' preferences, which, in turn accurately rank the interacted items higher than the unseen ones.

In this thesis, we investigate the use of the reviews as well as their associated properties to develop an end-to-end review-based recommendation framework. Each user u or item i has an associated set of reviews, e.g. posted by that user, C_u, or posted on that item, C_i. Furthermore, the reviews of a user or associated to an item can be described using k review properties $\mathbf{P} = \{P_1, P_2, ..., P_k\}$. For example, to capture the preferences of user u for a given property P_1, we estimate the corresponding review property scores for each review in the review set of user u,

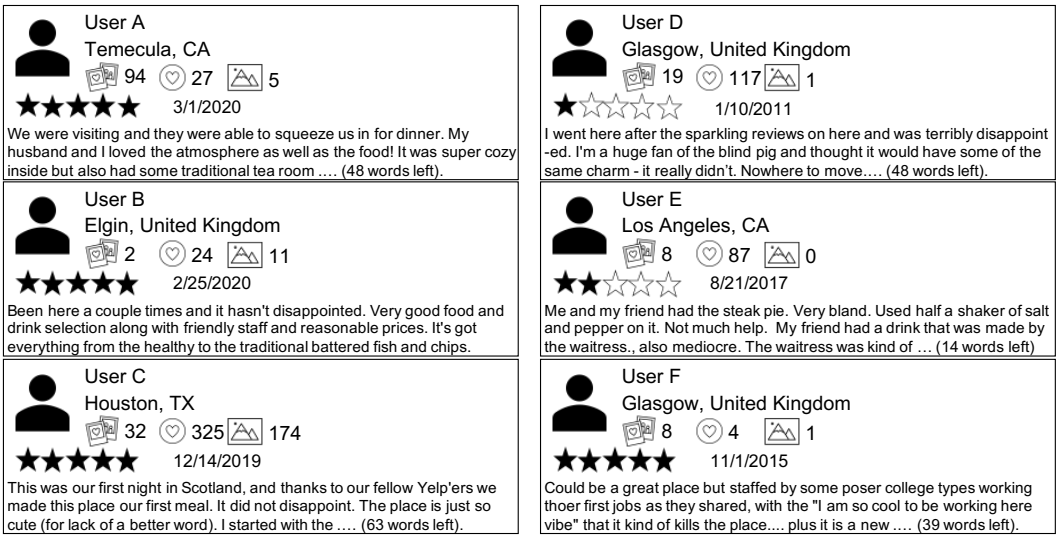
i.e. $P_{1,u} = \{p_{1,1}, p_{1,2}, ..., p_{1,|C_u|}\}$, where $p_{1,t}$ is the property score of the t^{th} review of user u. For example, for the length (or time) property, the score will correspond to the length of the review (or its time resp.). These scores could be computed for any property, provided that the property values are mapped into scalars in the range of [0, 1] using an adequate function.

Indeed, for the length property, a longer review will have its length property score closer to 1 than other shorter reviews. Hence, the property scores depict the usefulness of reviews. In other words, the computed review property scores enable the modelling of the reviews from different perspectives and examine the relationship between the review usefulness and the review properties. In particular, according to the discussed users' adoption of information framework in Section 3.5, different users prefer to use different types reviews (i.e. different properties) to examine the items' attributes, so as to decide the items they will interact with. Therefore, in the next section, we discuss the interactions between users and reviews to explain the use of review properties in a review-based recommendation framework.

4.3 Users' Interactions with Reviews

As we stated in Section 4.2, we aim to address a ranking-based recommendation task. A recommendation model has to comprehensively learn the users' preferences on items to make accurate recommendations. In the literature, many recommendation studies (Manotumruksa et al., 2018; Massimo and Ricci, 2018; Nepal et al., 2012) have shown that it is important to learn the users' behaviour when interacting with items to learn and estimate the users' preferences. In this thesis, we argue that users tend to selectively use reviews exhibiting different properties to identify their preferred items' attributes and decide which items to interact with. In particular, the information overload results in an enormous amount of reviews online (Sun et al., 2019). As a consequence, there is a need for users to browse and choose their preferred reviews to use according to specific criteria. For example, recall that in Section 3.5, the users' adoption of information framework (Sussman and Siegal, 2003; Filieri and McLeay, 2014) has shown that many users selectively use reviews according to their preferred review properties (e.g. overall star ratings and recency) when deciding which items to interact with. Moreover, many platforms have also been aware of such users' behaviour when users selectively use reviews according to various review properties. For example, in Figure 4.1, we present two groups of reviews on the Yelp website[1] as examples. In particular, these two groups of reviews are about one restaurant. Each group contains the top-ranked reviews sorted by some specific review properties (i.e.

[1] Yelp website: https://www.yelp.com/



(a) Reviews sorted by recency. (b) Reviews sorted by lowest rating first.

Figure 4.1: Reviews sorted by different properties on the Yelp website.

recency and lowest rating first, respectively). The Yelp website allows users to choose from various review properties, including the two properties above, to rank reviews. Note that, for the reviews in Figure 4.1, to avoid privacy concerns, after aggregating the information within the collected reviews, we have anonymized the corresponding reviewers by removing their names as well as the profile pictures.

For the example reviews in Figure 4.1, we present two groups of the top-ranked reviews that are sorted by two different associated properties (namely the recency and lowest rating first), respectively. The first (Figure 4.1a) shows the three most recent reviews. These three reviews describe the recent features and attributes of the corresponding item (i.e. a restaurant in this context). On the other hand, the reviews within the second group in Figure 4.1b have the lowest ratings among all posted reviews. These low rating-based reviews discuss the negative points of the corresponding item. For example, user D mentioned that the restaurant is too crowded to stay in, and user E complained about the flavour of the food. In particular, by summarising the information in these two groups of reviews (i.e. the most recent and lowest rated reviews), we can identify the recent features of the given item and its negative points. These examples indicate that users can easily capture comprehensive features about items without exploring all the reviews by focusing on reviews exhibiting different properties. Suppose a recommendation system can estimate the users' preferences using different types of reviews. In that case, the recommendation model can focus on leveraging the reviews that it deemed to be useful to model the features of items to make accurate recommendations. In addition, according to the users'

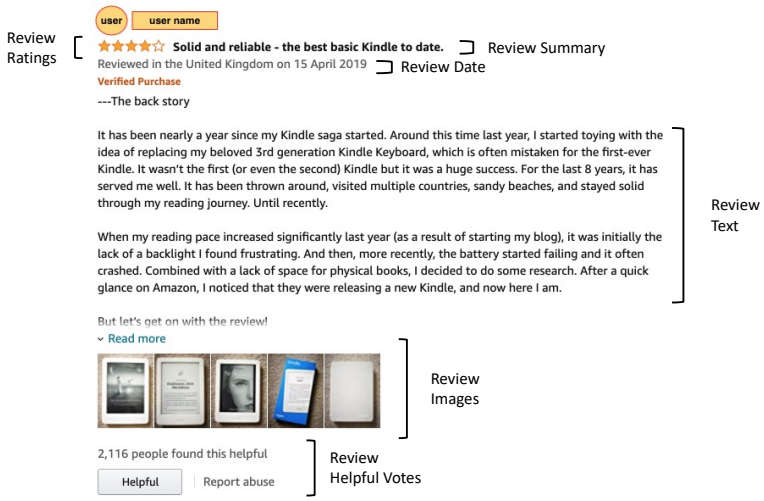


Figure 4.2: An example review on the Amazon website. For privacy reason, we anonymize the user's personal information.

adoption of information framework (Petty and Cacioppo, 2012), different types of users tend to consider reviews with different associated properties to support their choice about which items to interact with. Therefore, we argue that it is essential for a recommendation model to identify and estimate the users' preferences towards using various review properties. However, thus far, we have not discussed the commonly available properties of reviews that we can use to capture the users' preferences on different types of reviews. Therefore, in the next section, we first summarise review properties in public review datasets. After that, we summarise those review properties by proposing a taxonomy of review properties that we will use in this thesis.

4.4 A Taxonomy of Review Properties

According to the discussion in Section 4.3, the reviews' associated properties play an important role in the process of users deciding which reviews to consider to examine the items' attributes as well as which items to interact with. However, thus far, for simplicity, we only briefly discussed examples of the use of the reviews' associated properties in the users' interactions with online reviews. In this section, we propose a taxonomy of review properties that are commonly seen on various platforms and highlight their differences.

First, we present a review collected from the Amazon website,[2] an e-commerce site, in Figure 4.2 as an example to illustrate the associated properties of a given review. This review

[2]https://www.amazon.com

Public Datasets	Available Review Attributes
Yelp	Review ID, User ID, Business ID, Ratings, Date, Review Text, Helpful, Funny, Cool
Amazon	User ID, Product ID, User Name, Helpful, Review Text, Ratings, Summary, Review Time
TripAdvisor	Review Text, Ratings
IMDB	Review Text, Binary Sentiment Class
Airbnb	Review ID, User ID, User Name, Review Text, Review Date

Table 4.1: Available attributes of reviews within the existing pubic datasets.

describes a user's experience of purchasing a product from the Amazon website. The presented review contains many attributes, including the reviewer's personal information (i.e. profile image and name), a rating (4 out of 5), a review summary, the review date (15 April 2019), the review text, the images, and the number of helpful votes from other users (2116 for this review). We can derive various reviews' associated properties that describe the attributes or characteristics of the given reviews from the available information above. One example is the review sentiment extracted from the users' ratings or classified according to the review text. Another example is the length of reviews calculated by counting the number of tokens within the review text. However, the representations of reviews vary across different websites. For example, the review summary in the Amazon reviews is not shown in the Yelp reviews. Moreover, many public review datasets have been aggregated from different websites (e.g. Yelp and Amazon) with different available review attributes. To compare the review attributes available for use in order to derive the reviews' associated properties, we gather the attributes of reviews within five existing public review datasets in Table 4.1, including Yelp,[3] Amazon,[4] TripAdvisor,[5] IMDB[6] and Airbnb.[7]

According to the aggregated review attributes in Table 4.1, the review text is commonly available in most review datasets. However, aside from the review text, each review dataset comprises different available review attributes. For example, the TripAdvisor dataset has both the review text and the users' ratings, while the IMDB dataset has the review text and the sentiment classes (e.g. 1 or -1 that indicate a positive or a negative opinion). Nevertheless, we still can extract properties of reviews from the available attributes of reviews to estimate the users' preferences on different types of reviews. For instance, by using the IMDB dataset, we can extract

[3]https://www.yelp.com/dataset
[4]https://jmcauley.ucsd.edu/data/amazon/
[5]https://www.kaggle.com/andrewmvd/trip-advisor-hotel-reviews
[6]https://www.kaggle.com/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews
[7]https://www.kaggle.com/airbnb/seattle

Review Property Categories	Examples of Review Properties
Lexical	Review Length, Percentage of Nouns, Percentage of Verbs, Percentage of Adjectives
Sentiment	Users' Ratings, Binary Sentiment Class, Sentiment Scores Calculated by a Learned Classifier
Helpfulness	Helpfulness Votes, Helpfulness Scores Calculated by a Learned Classifier
Event-based	Review Timing, Review Location

Table 4.2: Example properties in different review property categories.

the review length and the percentage of nouns, verbs and adjectives from the review text. We can also extract the review sentiment property from the sentiment class attribute. Next, based on the aggregated attributes of reviews in Table 4.1, we propose a new taxonomy of review properties that we can extract from the reviews and group review properties into four classes (namely the lexical, sentiment, helpfulness and event classes). In particular, in Table 4.2, we present the possible review properties that we can extract from the reviews' attributes. We describe such a taxonomy of review properties as follows:

1. **Lexical:** The lexical class of the reviews' properties relies on the lexical analysis (Paice, 2018) of reviews, which first converts the reviews into a sequence of textual strings (i.e. lexical terms). Next, we investigate the associated meaning of the generated lexical terms. The lexical class of the reviews' properties reflects the characteristics of given reviews according to the extracted lexical terms from the reviews' text. For example, the number of lexical terms within a given review (i.e. the review length in Table 4.2), which shows if a review is likely to contain sufficient useful information. In this thesis, we use the review length property as a representative instance of the lexical class of review properties, since it has been commonly used in the literature to identify the usefulness of reviews (Kim et al., 2006; Korfiatis et al., 2012).

2. **Sentiment:** The sentiment class of the reviews' properties reflects the users' opinions, attitudes and emotions expressed in reviews. The reviews' sentiment property can be generated by applying a sentiment analysis technique to the review text or calculated according to the users' ratings. Therefore, the review sentiment property can help other users to quickly identify a positive or negative opinions from the corresponding reviews. In this category, we use both the users' ratings and the sentiment scores calculated by a learned classifier in our proposed review-based recommendation framework. As we can see in Table 4.1, the users' ratings are commonly available in many review datasets (i.e. in the

Yelp, Amazon and TripAdvisor datasets), but not always (e.g. there are no users' ratings in IMDB and Airbnb datasets). To address the missing sentiment property, we leverage the sentiment scores calculated by a learned classifier on the reviews' text to estimate the sentiment property of reviews.

3. **Helpfulness:** The helpfulness class of the reviews' properties indicates the helpfulness level of a given review to other users. In particular, this helpfulness level of a review relies on the number of helpful votes given by other reviewers to the corresponding review. The helpfulness class of the reviews' properties can be estimated either according to the reviews' helpful votes or by applying a machine learning model to the reviews' text (Lee et al., 2018b). In this thesis, we consider both of these alternatives to indicate the helpfulness of reviews. The reason is similar to the discussion of the sentiment property. Indeed, the reviews' helpful votes are not commonly available. Therefore, we leverage the helpfulness scores estimated through a classifier (i.e. the confidence of a review text being helpful judged by a classifier) on the reviews' text to estimate the helpfulness of reviews.

4. **Event-based:** The event-based class of the reviews' properties describes the contextual factors of the users' experience when interacting with items. For example, the event-based class of the reviews' properties can be related to the time and location of when and where the review was posted. A recommendation system can identify the time and location of the reviews by leveraging this type of properties to make more accurate recommendations. For example, to recommend local restaurants to a given user, a recommender needs to examine the reviews that were posted locally. In comparison to the reviews' location attribute, the review's timestamp is more frequently included in the public review datasets. Therefore, in this thesis, we use the review age property, extracted from the reviews' timestamp, which indicating if a review was recently posted, as a representative instance of the event-based class of review properties.

In summary, we have discussed four categories of review properties (namely the lexical, sentiment, helpfulness and event-based classes). In this thesis, we use six representative review properties as instances of these four categories, including the reviews' length, age, ratings, the sentiment scores calculated by a learned classifier, the helpful votes and the helpful scores calculated through a learned classifier. Next, we describe our proposed review-based recommendation framework that leverages these four types of review properties to estimate the users' preferences of using various review properties to examine the items' attributes. Then, based on the learned items' features, the recommendation framework estimates the users' preferences on items.

Figure 4.3: Overview of the Review Property-based Recommendation Framework.

4.5 Framework Overview

This thesis aims to leverage different types of the reviews' associated properties to effectively identify the users' preferred reviews that exhibit specific review properties and estimate the users' preferences on items accordingly. Consequently, we propose a review-based recommendation framework to address the corresponding research task introduced in Section 2.2. In Figure 4.3, we present our proposed review-based recommendation framework, which comprises four main components: (1) data collection; (2) review property extraction; (3) review property encoding; and (4) user preference estimation. We link these four components in a sequential order. In Figure 4.3, each dashed box corresponds to a single component. From the first component to the last one, we start by collecting the interaction data of users and items and their associated reviews. Then, we conduct the review property extraction process and encode the obtained review properties in the learned review features. Next, based on the obtained data, we leverage the users' preferences on various review properties to rank the predicted users' preferred items in the top. In the following, we illustrate the functionality of each component and formalise its input and output, respectively.

- **Data Collection:** As we mentioned above, our proposed framework aims to address the recommendation task by leveraging the reviews as well as their associated properties. Therefore, there is a need to aggregate the data to prepare the required information and evaluate the proposed approaches. The collected data should comprise a set of users $\mathbf{U} = \{u_1, u_2, ..., u_M\}$ and items $\mathbf{I} = \{i_1, i_2, ..., i_N\}$, where M and N refer to the number of the users and items within the collected datasets. Moreover, we should also collect the user-item interactions $R^{M \times N}$ between users U and items I to learn and estimate the users' preferences on items. In particular, since we are addressing a review-based recommendation task, the users' and items' associated reviews C_U and C_I are also important elements to be aggregated in the data collection step. Note that, in this thesis, we address the data

collection by using a number of existing public datasets, i.e. the Amazon and Yelp datasets in Table 4.1, which have sufficient review data to extract review properties, to examine the performances of our proposed approach and to avoid possible concerns in collecting new data (e.g. transgressing the data sharing protocols or the user privacy issue). After this stage, we aim to have the review datasets comprise the identification information of users, items, their associated reviews and the interaction records between users and items.

- **Review Property Extraction:** At this stage, we use the aggregated data, namely $\{U, I, R, C\}$ from the previous stage (i.e. the data collection stage) as input to conduct the review property extraction process. In particular, a given review c, like the one in Figure 4.2, has various associated review properties. Each property describes a characteristic of a given review. For example, the sentiment property refers to the sentiment expressed within the text of review or the associated rating. However, the reviews' associated properties are shown in different formats within the aggregated data. For example, the time and location of when and where the review was posted are encapsulated in timestamp and city name formats. To effectively leverage the review property data, we propose to convert the reviews' associated properties into digital values as follows:

$$p_c = f_p(c, p) \tag{4.1}$$

where p_c is the review property score that describes the property p of review c. In particular, $f_p(.)$ calculates the property scores of reviews and can be implemented by a mapping function or a machine learning model. Note that some specific review properties, such as the location and categorical properties, cannot be easily transferred into a singular value. Therefore, in dealing with such types of review properties, we need to introduce carefully designed and tailored mapping functions. The main rule is that such a mapping function can effectively generate values capable of reflecting the corresponding review property. For example, the resulting value can be in a singular form, ranging from 0 to 1, measuring the extent of the corresponding property encoded in a given review. Meanwhile, the resulting value can also be in a binary or other categorical forms, showing whether a review exhibits a particular property.

Afterwards, for reviews within the review set C, we have a set of the reviews' associated properties $\mathbf{P}$ that describe reviews (e.g. the reviews' sentiments and the reviews' helpfulness).

- **Review Property Encoding:** Recall from Section 4.2 that, in this thesis, we aim to ad-

dress the estimation of the users' preferences towards using reviews exhibiting different properties so as to improve the personalised recommendation outcomes. As a consequence, our proposed recommendation framework needs to model the users' behaviour in using reviews that exhibit different properties. In particular, our proposed framework first learns an initial feature representation X_c of a given review c. After that, for a given review property p, it integrates the calculated review property score p_c to generate a variant of the review's feature representation (i.e. $X_{c,p}$). This process can be described by the following equation:

$$X_{c,p} = f_e(X_c, p_c) \tag{4.2}$$

where f_e is the used review property encoding technique. Afterwards, this review property encoding stage outputs the reviews' various representations $X_{C,P}$ according to the considered reviews' associated properties.

- **User Preference Estimation:** With the aggregated data and the reviews' feature representations that are encoded with various reviews' properties, the primary function of this stage is to estimate the users' preference on items by considering reviews exhibiting different properties. Essentially, for a given user, the proposed recommendation framework first identifies their preferred type of reviews to examine the attributes of items and then it predicts which items need to be ranked higher than others according to the learned users' preferences. We formally describe this process as follows:

$$\hat{R}_{u,i} = f_r(u, i, f_c(u, c, P, X_u, X_i)) \tag{4.3}$$

where $f_c(.)$ identifies the users' preferences on reviews' exhibiting different properties. In particular, given a pair of user and item (i.e. u and i) as well as a set of review properties P, $f_c(.)$ calculates personalised weights of different review properties in P and encodes such weights to the review representations X_u and X_i (i.e. the representation of the users' and items' associated reviews). For example, if a user prefers using recent reviews, $f_c(.)$ is likely to apply a higher weight to the timeliness property of reviews for the resulting review representation. Afterwards, $f_r(.)$ estimates the users' preferences in interacting with different items.

Once these four stages have been completed, our review-based recommendation framework can effectively identify the users' preferences on their preferred types of reviews exhibiting different properties. Then, the recommendation framework can leverage the learned knowledge to support the estimation of the users' preferences on items. However, it is essential for our

proposed recommendation framework to address some corresponding challenges to achieve this target. Therefore, in the following, we describe the main challenges within the proposed recommendation framework and the corresponding main research questions.

4.6 Challenges Within the Framework

After describing the functionality of each component of our review property-based recommendation framework, we now discuss the main challenges within this framework and their associated research questions. In particular, in Section 4.6.1, we discuss the availability of the review attributes within a number of existing public datasets. We also describe the research questions we aim to answer in the process of addressing the reviews' attributes availability challenge. In Section 4.6.2, we describe the challenge of estimating the users' preferred review properties and apply these users' preferences to recommendation approaches. Similarly, we also describe the corresponding research questions.

4.6.1 Availability of Review Attributes

In this thesis, one of the main targets of our proposed recommendation framework is to effectively estimate the users' preferences towards reviews exhibiting various review properties. Recall that in Section 4.4, we proposed a taxonomy of review properties that categorise the properties into lexical, sentiment, helpfulness and event-based classes.

However, in practice, some reviews' attributes are not available in the review datasets to extract the corresponding review properties. For example, the TripAdvisor dataset in Table 4.1 contains only the review text and rating, which makes it difficult to estimate both the helpfulness and event-based classes of the review properties. Consequently, one of the main challenges within the framework is the availability of the reviews' attributes to extract various review properties.

An intuitive way to address the availability of the review attributes is to conduct additional data collection, as required, to ensure that the framework has sufficient review attributes to extract the corresponding review properties. However, most of the current review datasets contain anonymised review data due to privacy concerns. As a consequence, it is difficult to address the problem of the missing reviews' attributes by augmenting the reviews' associated attributes.

On the other hand, according to the available review attributes within Table 4.1, we observe that, for all the datasets, the review text is commonly available to be used. Therefore, instead of using the corresponding review attributes to directly extract the properties of reviews, we

propose to develop machine learning approaches that leverage the review text to estimate the missing properties of reviews. For example, in a given dataset, some of the reviews may have the required review attributes but not the rest. Therefore, we leverage the review text and the available review attributes (e.g. sentiment class or helpful votes) as input to train classifiers and then predict the review properties of reviews. For example, by using the users' ratings and mapping them into binary labels, we can train a binary sentiment classifier on the reviews' text to estimate the sentiment property of reviews.

In particular, the *availability of review attributes* challenge affects the effectiveness of the developed classifiers and the quality of the generated review properties. In particular, the data containing the reviews' attributes is frequently limited, imprecise and unbalanced (Chan et al., 2020; Chen et al., 2018a). For example, the class distribution of the sentiment labels is commonly imbalanced in the review datasets (i.e. a review dataset frequently has more positive reviews than negative ones) (Chan et al., 2020). The helpfulness labels of the reviews are also often limited, and it is common that the review datasets lack sufficient helpfulness labels to build good review helpfulness estimators (Chen et al., 2018a).

In Chapter 5, we perform two consecutive studies to examine the quality of generating the sentiment property of reviews by using various machine learning models. In particular, we leverage the resulting estimated sentiment property by proposing a sentiment property encoding mechanism (i.e. a sentiment attention mechanism) in a novel recommendation model (i.e. denoted by SentiAttn). Moreover, recall from Section 3.1 that many widely cited review-based recommendation approaches are designed to address the rating prediction task. Therefore, to evaluate the effectiveness of using the extracted and estimated sentiment property of reviews in a recommendation model, we compare the performance of our proposed SentiAttn model with various review-based recommendation approaches (e.g. DeepCoNN (Zheng et al., 2017) and NARRE (Chen et al., 2018b)) in addressing the rating prediction task. In particular, our investigation aims to answer the following research questions: (i) Can the estimated sentiment property of reviews sufficiently capture the users' preferences so as to replace the users' ratings for the purposes of effective item recommendation? (ii) Does the SentiAttn model that encodes the sentiment property of reviews outperform existing review-based recommendation approaches?

Next, similar to the sentiment property, the helpfulness property is also not commonly available in several existing public review datasets (e.g. the TripAdvisor and IMDB datasets in Table 4.1). Therefore, in Chapter 6, we propose a novel binary review helpfulness classification correction approach, called the Negative Confidence-aware Weakly Supervised (NCWS) approach, to improve the prediction accuracy of the helpfulness property of reviews and examine the effectiveness of applying the predicted helpfulness properties of reviews to address the rating

prediction task. In particular, our investigation aims to answer the following research questions: (i) Can our proposed NCWS approach improve the performances other classification approaches (i.e. the SVM, CNNs and BERT-based classifiers) on all used datasets (i.e. the Yelp and Amazon datasets)? (ii) Does the learned helpfulness of reviews by using our proposed NCWS approach benefit an existing state-of-the-art review-based rating prediction model (i.e. DeepCoNN (Zheng et al., 2017)) with a higher accuracy?

In Chapters 5 and 6, we will introduce the sentiment and helpfulness classifiers we are going to apply to estimate the often missing sentiment and helpfulness properties of reviews. These chapters address the availability of the reviews' attributes and prepare the review properties to be used in our proposed review-based recommendation framework. However, several challenges remain when we aim to predict the users' preferences when using different review properties and recommending items to users.

4.6.2 Estimation of Users' Preferences

Aside from the availability of the reviews' attributes, we need to address others main challenges in our proposed framework. These challenges are twofold. First, we need to investigate how to identify the users' preferred review properties. Second, we also need to integrate the learned users' preferences on the types of reviews within the recommendation approaches to make effective recommendations. Recall that, in Section 3.5, according to the users' adoption of information framework, different users show different interests in using the various reviews' associated properties to examine the usefulness of reviews. Even though this users' adoption of information framework has been well-cited (Sussman and Siegal, 2003; Petty and Cacioppo, 2012; Filieri and McLeay, 2014), it has not yet been considered within a recommender to support the estimation of the users' preferences on items. Consequently, it is still unclear how to effectively integrate the estimation of the users' preferred review properties into a review-based recommendation model. Therefore, in Chapters 7 and 8, we will investigate the approaches to address this research gap, which effectively estimate the users' preferences on review properties and then predict the users' preferences on items accordingly.

To propose effective techniques, we are aware of that there is a big difference between the available number of items and the reviews' associated properties. Indeed, many datasets contain enormous quantities of items. However, there are much fewer available properties to describe reviews. Therefore, some existing approaches used to rank the users' preferred items are not applicable to address the users' preferred review properties. For example, the classic Bayesian Personalised Ranking (BPR) approach (Rendle et al., 2009) leverages the pairwise comparison

of the users' preferences on two items (i.e. one interacted item and one non-observed item) for training. However, in the interactions between users and the available reviews' associated properties, there is no ground truth indicating if one review property is preferred by a user than another review property. This limitation makes the usual traditional methods (e.g. the BPR learning) inapplicable for estimating the users' preferred review properties. Consequently, we need to propose new customised approaches to estimate the users' preferences on the reviews' properties. In addition, it is essential that the proposed recommendation techniques can effectively leverage the learned users' preferences on the reviews' properties, so as to improve the performances of recommendation approaches.

To address the introduced challenges, in Chapters 7 and 8, we propose two individual end-to-end recommendation models. These two models instantiate our proposed review-based recommendation framework by first estimating the users' preferences on various review properties and then on the items of interest.

In particular, in Chapter 7, we introduce a novel review-based recommendation model, called RPRM, which estimates the users' preferences on the review properties via an attention mechanism. This study focuses on answering the following research questions: (i) Which review properties are the most effective when examining the usefulness of reviews to make ranking-based recommendations? (ii) Can RPRM, which leverages various review properties outperform the existing strong recommendation approaches (e.g. CASER, JRL and DeepCoNN) on the two used datasets (i.e. the Yelp and Amazon datasets)?

On the other hand, in Chapter 8, we leverage the bandit algorithm, a commonly used approach to model the users' choices of using different review properties (Xu et al., 2020; Li et al., 2010; Cañamares et al., 2019), in a novel multi-task learning-based recommendation model, called BanditProp, to estimate the users' preferences on various review properties. In particular, BanditProp explores various existing bandit search algorithms and proposes novel neural contextual bandit algorithms (i.e. ConvBandit) to learn the users' choices in using different types of reviews. In Chapter 8, we aim to answer the following research questions: (i) Can the BanditProp model that leverages both a multi-task framework and the bandit search algorithms outperform the RPRM model of Chapter 7, which uses the attention mechanism? (ii) Can our proposed ConvBandit algorithm outperform other well-known bandit search algorithms (e.g. the greedy search (Kuleshov and Precup, 2000) and UCB search (Kaufmann et al., 2012) algorithms) when applied to the BanditProp model?

Both RPRM and BanditProp instantiate our proposed review-based recommendation framework. In Chapters 7 and 8, we will experimentally validate the effectiveness of these two different approaches when modelling the users' preferences on using different types of reviews to

improve the performances of recommendation models.

4.7 Conclusions

This chapter first stated the problem we are addressing in this thesis. Next, in Section 4.3, we discussed the users' behaviour when interacting with reviews, which illustrates the necessity of identifying the usefulness of reviews according to their various associated properties. Then, in Section 4.4, we presented a novel taxonomy of review properties, which categorizes the reviews' associated properties into four classes. We also discussed the six review properties that we will use in this thesis. To leverage reviews and their associated review properties, in Section 4.5, we presented a novel review-based recommendation framework that considers the users' preferences on various review properties and items. This recommendation framework comprises four components. In particular, we discussed the functionality of each component within the recommendation framework. Moreover, in Section 4.6, we described the challenges of implementing the recommendation framework as well as the research questions we aim to answer so as to address the discussed challenges. Next, starting from Chapter 5, we discuss the studies we will be conducting to address the challenges we have identified and presented.

Chapter 5

Sentiment Property Extraction and Integration

5.1 Introduction

In our thesis statement (see Section 1.3), we postulated that we can obtain an enhanced recommendation performance by leveraging the available properties of reviews posted by users. Therefore, in Chapter 4, we proposed a review-based recommendation framework, which comprises four main components leveraging reviews to estimate the users' preferences on items. Moreover, as argued in Section 4.6 regarding the implementation of the proposed recommendation framework, we need to address various sub-tasks and challenges within the four main components. In particular, recall from the discussion in Section 4.6.1, that some review attributes are not readily available in the review datasets to extract the corresponding review properties, e.g. the reviews' sentiment and helpfulness properties.

Indeed, according to the description in Section 4.5, the functionality of the second component of our proposed review-based recommendation framework is to extract the reviews' associated properties from the output of the data collection stage. Moreover, given the proposed taxonomy of review properties in Section 4.4, we focus on extracting four categories of review properties (namely the lexical, sentiment, helpfulness and event-based classes). Each category of review properties implies a specific characteristic of the given reviews. Moreover, the extraction of each property could be addressed by leveraging the corresponding attributes of reviews (e.g. the users' ratings or the use of review text to extract the review sentiment property). In this chapter, we focus on the extraction of the sentiment property as well as tackling the possible unavailability of the required reviews' attributes. As we described in Section 4.4, the senti-

ment property includes the users' ratings (Pang and Lee, 2005) or extracted from the review text (Fang and Zhan, 2015) as both encapsulate the users' preferences towards the items. However, the users' ratings are not always available in the public datasets (e.g. the Airbnb data in Table 4.1). As a consequence, many studies in the literature proposed various sentiment classifiers to estimate the sentiment property of the reviews. However, to the best of our knowledge, we found no study in the literature that investigated whether the use of the sentiment information of the review text can be a reliable surrogate for the explicit ratings. On the other hand, it is also unclear if the use of the extracted sentiment property can support the identification of the reviews' usefulness, so as to improve the recommenders' performances. Therefore, in this chapter, we conduct two consecutive studies. The first study examines the reliability of using the sentiment information as a surrogate for the users' ratings when addressing a ranking-based recommendation task. The second study aims at improving the rating prediction performances of recommenders by leveraging the extracted sentiment property from the reviews. The remainder of this chapter is structured as follows:

- Section 5.2 presents the first study, which investigates whether the sentiment information from the review text can be an effective surrogate for the users' ratings when addressing a ranking-based recommendation task.
- Section 5.3 provides the details of the second study, which leverages the sentiment property of the reviews within a review-based rating prediction approach to estimate the users' preferences towards items.
- Section 5.4 summarises insights gained from our conducted studies in this chapter.

5.2 Comparison of Sentiment Analysis and Ratings in Ranking-based Recommendation

Recall from Section 1.1 that many recommendation systems (Manotumruksa et al., 2016a; Lian et al., 2018a) have been proposed to address the data overload problem on e-commerce websites and Location-Based Social Networks (LBSN). These systems automatically suggest items for users to interact with based on the users' profiles and interaction histories. In particular, the users' ratings of items are widely used in various recommendation systems to elicit the users' preferences, including in collaborative filtering systems (Schafer et al., 2007; Hu et al., 2014), Matrix Factorization (MF) approaches (Koren et al., 2009) and more recent advanced recommendation approaches (Wang et al., 2021b; Zhao et al., 2020; Wang et al., 2020c).

However, the user ratings, a type of the reviews' sentiment property, are not always available in public review datasets (e.g. the Airbnb dataset in Table 4.1). Moreover, the user ratings are not always effective in representing the users' preferences. For example, it has been reported that users have distinct and inconsistent rating behaviour (Hu and Pu, 2013) and find it difficult to provide accurate feedback on the items when faced with selecting among multi-rating values (Cosley et al., 2003). Several previous studies aimed to assess the impact of the users' location (Hu and Pu, 2013), their personal and situational characteristics (Knijnenburg et al., 2012), and the nature of rating scales available to users on the quality of the users' ratings (Cosley et al., 2003).

On the other hand, sentiment analysis is a widely used technique to gauge the users' opinions and attitudes, for instance towards products and venues, from the textual user reviews (Han et al., 2011). Sentiment analysis not only predicts the polarity of user opinions (e.g. positive vs. negative) but can also provide a summary of the users' opinions of an item from their review text (i.e. we call this the sentiment property of reviews) (Han et al., 2011; López Barbosa et al., 2015). In fact, sentiment analysis has been adopted by many studies to adjust user ratings or provide extra features to enhance the performance of recommendation systems (Yang et al., 2013; Gurini et al., 2018; Wang et al., 2017a).

The integration of sentiment analysis into recommendation systems is still limited to adjusting the users' ratings to overcome their inconsistency. Instead, Lak and Turetken (2014) substituted the user' ratings with sentiment analysis but concluded that sentiment analysis was insufficient to replace the ratings in their experiments. Since then, sentiment analysis has seen a lot of attention in the literature cumulating in the development of advanced effective neural networks-based sentiment analysis of long and short texts (Kim, 2014; Baziotis et al., 2017). Indeed, previous sentiment analysis approaches mainly relied on human-crafted sentiment dictionaries (Mohammad et al., 2013; Kiritchenko et al., 2014), which are not necessarily sufficiently effective on the wide variety of words used in LBSNs (Baziotis et al., 2017).

Therefore, in this section, we argue that it is possible to replace the users' ratings by leveraging state-of-the-art sentiment analysers on the users' posted review text, thereby increasing the consistency of the user's preferences when making recommendations, as well as addressing the missing data of the explicit ratings in some existing review datasets (e.g. the IMDB and the Amazon datasets in Table 4.1). We integrate the obtained users' preference scores through sentiment analysis into the widely used MF, a classical model-based CF recommender (see Section 2.3.2), and GeoSoCa multi feature-based recommendation models (Zhang and Chow, 2015). The rest of this section is structured as follows. In Section 5.2.1, we describe the GeoSoCa model and the rating substitution strategy. Section 5.2.2 describes the sentiment analysis techniques that

we deploy. After that, we detail the setup for our experiments (Section 5.2.3). Then, in Section 5.2.4, we present our obtained results for evaluating the effectiveness of substituting ratings with sentiment scores.

5.2.1 Recommendation Model and Rating Substitution Strategy

In this section, we first recall the ranking-based recommendation problem that we discussed in Section 2.2.2, and the used notations (Section 5.2.1.1). Then, we introduce the two models (i.e. MF and GeoSoCa) that we use to evaluate the ability of sentiment analysis in capturing user preferences as a surrogate to user ratings. Finally, in Section 5.2.1.3, we describe the rating substitution strategy which we apply to MF and GeoSoCa.

5.2.1.1 Problem Statement

As we discussed in Section 2.2.2, the ranking-based recommendation task aims to rank highly items that users would like to interact with, based on the users' previous interactions and other sources of information. For instance, recall the introduced notations in Section 4.2, where given a set of users U and a set of items I (of size M and N, respectively), we encode the users' historical ratings in a matrix $R \in \mathbb{R}^{M \times N}$, where entries $r_{u,i} \in R$ can represent the previous item ratings (1..5) of user $u \in U$ of items $i \in I$. Item recommendation can then be described to accurately estimate the value $r_{u,i}$ for an item that the user has not previously interacted with, or to rank highly items that they would very likely interact with.

5.2.1.2 Ranking-based Recommendation Approaches

In this work, we examine the behaviour of two recommendation approaches, namely MF and GeoSoCa, and how they perform when we change the definition of $r_{u,i}$.

Matrix Factorization (MF). Recall the discussion of MF in Section 2.3.2. MF is a classic recommendation approach, which has been adopted by many recommendation model studies as a baseline (Lian et al., 2014; Zhao et al., 2016a; He et al., 2016). MF adopts singular value decomposition to learn latent semantic vectors p_u and q_i for user u and item i (discussed in Section 2.3.2), respectively on known ratings $r_{u,i} \in R$.

GeoSoCa. GeoSoCa (Zhang and Chow, 2015) is a popular venue recommendation approach. Venues are a specific type of items. Compared to MF, it encompasses three additional important sources of information in making improved recommendation, namely geography, social and item category (i.e. denoted by 'Geo', 'So' and 'Ca', respectively) information. Since

then, it has been frequently used and discussed in various studies (Manotumruksa et al., 2016a; Zhao et al., 2017, 2016b). The geographical and social influence features use the geographical distance and the users' social connections to measure the influence of different venues on users, respectively. The categorical influence estimates the users' preferences distribution over categories of venues (restaurants, bars, etc.). In particular, $pop_{cat,i}$ indicates the popularity of venue $i \in I$, which belongs to category $cat \in Cat$, where Cat denotes all venue categories. In computing all these three additional features, GeoSoCa makes use of the users' ratings to estimate the probability of user u visiting venue i (Zhang and Chow, 2015). Note that, following Zhang and Chow (2015), in our experiments, we also deploy both GeoSoCa and its components individually, i.e. Geo, So and Ca.

5.2.1.3 Rating Substitution Strategy

As argued in Section 5.2, the advent of accurate sentiment analysis approaches offers new opportunities for more refined recommendations. In particular, since the resulting sentiment classifiers can be formulated in a probabilistic manner, we assume that the users' preferences are indicated by the classifier's confidence, denoted as sentiment score $s_{u,i}$. This score captures the classifier confidence in a review text of user u on item i being positive. Indeed, our work examines if the sentiment score, $s_{u,i}$, can effectively replace the rating $r_{u,i}$ as an indicator of the users' preferences. We now describe our adaptations of MF and GeoSoCa. In MF, the sentiment-based MF approach replaces user ratings on items, $r_{u,i} \in R$, with sentiment scores $s_{u,i}$. In contrast, for GeoSoCa, we consider the substitution strategy on each component: In the geographical and social influence features, we replace the users' ratings $r_{u,i} \in R$ with $s_{u,i} \in R$. Moreover, different from the previous two features, in the categorical influence feature, we not only replace the users' ratings $r_{u,i} \in R$ with $s_{u,i}$, but we also modify the venue category popularity, $pop_{cat,i}$, as follows:

$$pop_{cat,i} = \sum_{u \in U} s_{u,i} \tag{5.1}$$

Therefore, we evaluate the ability of the sentiment scores to accurately capture the overall item popularity. In the next section, we discuss the sentiment classification approaches that we apply to calculate $s_{u,i}$.

5.2.2 Sentiment Classification Approaches

In this work, we classify the sentiment analysis approaches into three types, including dictionary-based, learned, and deep-learned approaches. We apply four approaches that represent all of the

Dataset Name	Number of Reviews
Training	600,000
Testing	200,000

Table 5.1: Dataset Summary for Sentiment Classification Usage

three categories, as well as a **Random** (Rand) classifier that matches the class distribution in its predictions, as an additional baseline.

1. **SentiWordNet-based Classifier (SWN).** The SentiWordNet-based classification approach, a dictionary-based technique, has been developed following the approach proposed by Guerini et al. (2013), which used the updated SentiWordNet3.0 dictionary (Baccianella et al., 2010). In addition, we use the 'geometric' weighting strategy that considers the word frequency to compute the prior polarity of each sentiment lexicon. The sentiment score is obtained by averaging the sentiment scores of words in each review text.

2. **SVM-based Classifier (SVM).** An SVM-based classifier is a representative learned sentiment classification model, which is optimised by learning and training on the available features. Following the experimental setup of Pang et al. (2002), we deploy an SVM-based classifier, which is a widely used classification technique (Moraes et al., 2013; Liu et al., 2013), using the labelled word frequency vector for each review, trained using a linear kernel.

3. **CNN-based Classifier (CNN).** We use a CNN-based classifier (Kim, 2014), a frequently used deep learning technique, for sentiment classification. We refer to Section 2.4.2 for the description of the CNN model. In addition, we also follow the 'CNN-Static' model setup in (Kim, 2014), which reported a good performance without the need for tuning the word embedding vectors.

4. **LSTM-based Classifier (LSTM).** We deploy an LSTM-based classifier (Baziotis et al., 2017), which is another deep learning-based classification approach. This classifier obtained the top performance in the sentiment classification competition in SemEval 2017. We described the LSTM model in Section 2.4.3. In particular, we follow the experimental model construction process and the configuration described by Baziotis et al. (2017).

5.2.3 Experimental Setup

In the following, we evaluate the sentiment classification approaches compared to the corresponding users' ratings of these items. Thereafter, we examine the difference in item recommendation effectiveness between models that leverage user ratings and those that use sentiment scores instead. Therefore, our experiments aim at answering the following research questions:

- **RQ 5.1:** Which sentiment analysis approaches exhibit the highest performance for the review sentiment classification?
- **RQ 5.2:** Can the sentiment scores, calculated by the top-performing sentiment classifiers, sufficiently capture the users' preferences so as to replace ratings for the purpose of effective item recommendation?

To address these research questions, we perform two experiments using the Yelp Challenge dataset Round 11 (Asghar, 2016). We use the Yelp dataset as it is a commonly available public dataset that fulfils our experimental requirements, i.e. to include geographical, social and category information, as well as user reviews.

5.2.3.1 Sentiment Classification

The statistics of the dataset extracted from Yelp for the sentiment classification experiments are shown in Table 5.1. In Yelp, all ratings are given using a 5-star rating scale (1 is poor, 5 is great). Following Koppel and Schler (2006), we label the polarity of each review according to the user's rating of the venue, which we regard as positive if the rating ≥ 4, and as negative if rating ≤ 2. Note that some existing studies (Shi and Liang, 2015; Kouadri et al., 2020) also mentioned that users frequently give inconsistent ratings. Indeed, some users tend to give high ratings (e.g. 4 and 5) instead of lower ratings (e.g. 1 and 2), while others are more conservative when giving high ratings. However, in this study, we simplify this problem setup by splitting the positive and negative ratings with a frequently used threshold (Koppel and Schler, 2006; Budhi et al., 2017), the rating 3, which is positioned in the middle of the given range (i.e. 1 to 5). Then we randomly select equal numbers of positive and negative reviews to construct the training and testing datasets, which also avoids the class bias phenomena. Moreover, as the CNN and LSTM approaches rely on trained word embedding models to model the review text and calculate the latent vector representations of reviews (see Sections 2.4.2 and 2.4.3), we use the remaining reviews (minus the reviews found in the Phoenix and Las Vegas city-based Yelp datasets, discussed below) in the Yelp dataset to train a word embedding model using the

Dataset	#Users	#Venues	#Reviews	Density
Phoenix	2,781	9,678	124,425	0.46%
Las Vegas	8,315	17,791	386,486	0.26%
Cross city	11,536	54,922	564,216	0.089%

Table 5.2: Datasets Summaries for the Venue Recommendation Task

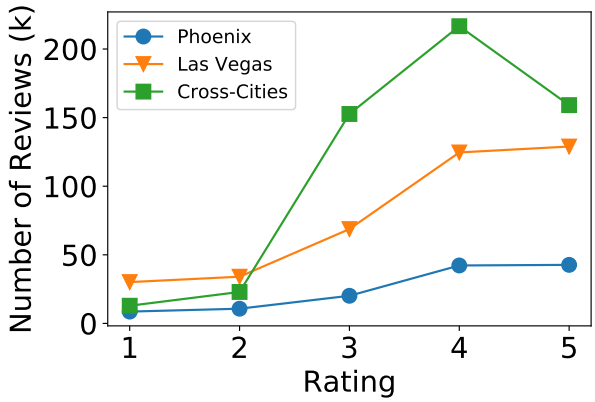


Figure 5.1: Ratings Distribution within the Datasets

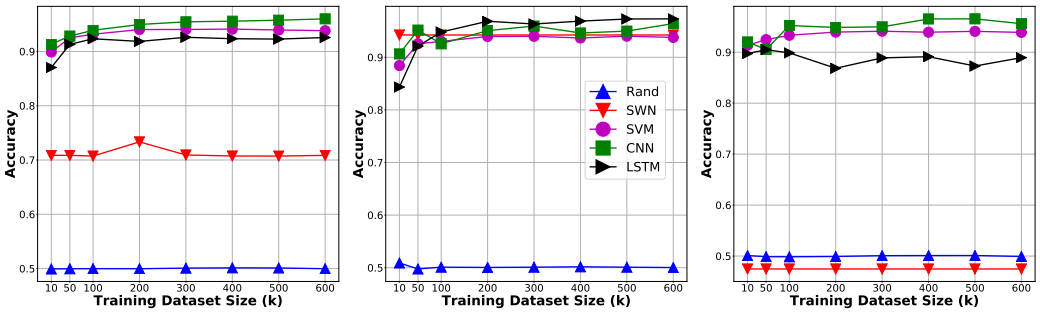
GenSim tool (Řehůřek and Sojka, 2010). For out-of-vocabulary words, we randomly initialise the embedding vectors, as suggested by Yang et al. (2018b).

We vary the size of the training dataset, from 10,000 to 600,000 reviews, to examine the stability and accuracy of the sentiment classification approaches. We use a 5-fold cross-validation setup on the training dataset before reporting the accuracy on the test datasets.

5.2.3.2 Recommendation Task

We use three subsets of the Yelp dataset to evaluate the performance differences between the rating-based and sentiment-based recommendation models. Unless otherwise stated, the sentiment scores are generated after the classifiers have been trained on 600,000 reviews. [1] Table 5.2 shows the statistics of the three Yelp-based datasets we use to evaluate the recommendation effectiveness. In particular, for generalisation purposes, we include two city-based datasets (namely Phoenix and Las Vegas) following other recent works (Yuan et al., 2016; Guo et al., 2017b), and one cross-city dataset. Indeed, we use these different Yelp subsets to obtain an overall understanding of the recommendation models' performances in different settings. To alleviate extreme sparseness, following Yuan et al. (2016), for each dataset, we remove users with

[1] As will be shown in Section 5.2.4, this is the best training setup in terms of sentiment classification accuracy.



(a) Overall Accuracy on comments. (b) Accuracy on Positive Comments. (c) Accuracy on Negative Comments.

Figure 5.2: Reviews sorted by different properties on the Yelp website.

less than 20 reviews and items with less than 5 visits. Figure 5.1 shows the ratings distribution of the three datasets. It is of note that for all three datasets, the number of positive reviews (ratings 4 & 5) outweighs the number of negative reviews (ratings 1 & 2) by quite a margin. Finally, experiments are conducted using a 5-fold cross-validation on each dataset, and evaluated for recommendation quality using Precision@5 & @10 and mean average precision (MAP).[2]

5.2.4 Results Analysis

We now present the experiments that address our two research questions, concerning the sentiment classification accuracy (Section 5.2.4.1) and the usefulness of sentiment classification as an effective proxy for ratings in recommendation (Section 5.2.4.2).

5.2.4.1 RQ 5.1: Opinion Classification

Figure 5.2 presents the classification accuracy of our sentiment classification approaches described in Section 5.2.2, while varying the amount of training review data. In particular, we show the overall accuracy (Figure 5.2(a)) as well as the accuracy on the positive and negative comments alone (Figure 5.2(b) & 5.2(c), respectively).

Figure 5.2(a) shows that our selected sentiment analysis approaches are divided into three groups with different classification performances – SVM, CNN and LSTM all exhibit similar top performances, followed by SWN with medium accuracy, and Rand with an (expected) low accuracy. Among the highest performing group (SVM, CNN and LSTM), CNN is the highest performer.

[2]The NDCG metric is not used since not all users will consistently use the rating scale (1-5).

Next, we consider the accuracy of the classifiers separately on the positive and negative classes. From Figures 5.2(b) & 5.2(c), we find that SVM, CNN and LSTM still provide a high accuracy (≥ 0.9) for both classes, while LSTM varies in accuracy across the classes. Indeed, LSTM surpasses CNN on the positive comments yet underperforms on the negative comments (indicating a higher false positive rate). Finally, since SWN exhibits a high accuracy on the positive comments but a low accuracy on the negative comments, it is mostly identifying comments as having a positive polarity and increases the likelihood of making false positive results.

Overall, in answer to research question RQ 5.1, we find that SVM, CNN and LSTM exhibit a high sentiment classification accuracy, with CNN outperforming all other techniques in terms of overall accuracy. In particular, LSTM performs better than SVM and CNN for positive comments, while CNN is more accurate than the other classifiers for negative comments.

5.2.4.2 RQ 5.2: Sentiment Classification in the MF and GeoSoCa Models

We now consider if the sentiment scores generated from the classifiers evaluated in Section 5.2.4.1 can be used for effective recommendation by the MF and GeoSoCa models. All results are presented in Table 5.3. Each column denotes the evaluation metric on the corresponding datasets. Each group of rows defines a particular recommendation approach: MF, GeoSoCa, or the latter's respective components (Geo, So, Ca). Each row in a group specifies the rating-based performance or the sentiment scores-based performances from the corresponding applied sentiment classification approaches. Finally, the rightmost column indicates the number of significant increases and decreases compared to the rating-based (baseline) model in that group of rows.

On analysing the general trends between MF, Geo|So|Ca and GeoSoCa, we find that the MF model exhibits a weak effectiveness for this ranking-based recommendation task. Indeed, the observed performances for the combined GeoSoCa approach are markedly higher (0.0029 vs. 0.0223 MAP).[3] Overall, the lower performance of MF is expected, as MF is intended as a rating prediction approach, rather than a ranking approach, where the objective is to rank highly the actual venues that the user visited. Using sentiment information shows some minor improvements, but none of the sentiment classifiers causes significant enhancements to this weak rating-based MF baseline. We also observe that the Rand baseline can give weak but comparable performances compared to other sentiment classification approaches. This observation is mainly caused by how the sentiment scores are used in these recommendation approaches. In the user-item interaction matrix R, due to the sparse interaction nature between users and items, many values in such a matrix are zero. A positive value, even calculated by a random base-

[3]The low absolute MAP values on this dataset are in line with other papers, e.g. (Liu et al., 2017).

		Phoenix (*100)			Las Vegas (*100)			Cross-City (*100)			Signf. #
		P@5	P@10	MAP	P@5	P@10	MAP	P@5	P@10	MAP	(↑ / ↓)
MF	Rating	0.12	**0.11**	**0.29**	0.01	0.02	0.12	0.02	0.01	0.05	I
	Rand	0.06	0.06	0.23	0.01	0.01	0.10	**0.05**	0.02	0.02	0 / 0
	SWN	**0.13**	**0.11**	0.28	**0.05**	**0.05**	**0.17**	0.02	0.02	0.05	0 / 0
	SVM	0.06	0.06	**0.29**	0.01	0.01	0.12	0.02	0.01	0.04	0 / 0
	CNN	0.08	0.08	0.27	0.01	0.01	0.11	0.02	**0.03**	**0.06**	0 / 0
	LSTM	0.04	0.03	0.25	0.03	0.02	0.11	0.04	0.02	**0.06**	0 / 0
Geo	Rating	0.67	0.73	0.83	**0.47**	0.43	**0.55**	0.78	0.74	1.02	I
	Rand	0.64	0.61	0.78	0.35	0.39	0.51	0.73	0.75	1.01	0 / 0
	SWN	**0.69**	0.69	0.86	0.44	**0.45**	**0.55**	0.73	0.77	1.03	0 / 0
	SVM	0.61	0.73	0.81	**0.47**	0.44	0.53	**0.76**	**0.79**	1.03	0 / 0
	CNN	0.66	0.73	0.82	0.45	0.44	**0.55**	0.75	**0.79**	**1.04**	0 / 0
	LSTM	0.67	**0.75**	0.84	0.45	**0.45**	**0.55**	0.75	0.78	**1.04**	0 / 0
So	Rating	3.38	2.88	1.95	**2.81**	**2.36**	1.76	2.97	**2.61**	1.53	I
	Rand	2.52↓	2.10↓	1.14↓	1.98↓	1.76↓	0.98↓	2.07↓	1.80↓	0.77↓	0 / 9
	SWN	2.73↓	2.36↓	1.74↓	2.15↓	1.82↓	1.36↓	2.15↓	1.89↓	1.28↓	0 / 9
	SVM	3.34	2.27↓	2.02	2.69	2.33	1.67	2.56↓	2.15↓	1.49	0/ 3
	CNN	3.39	2.87	2.06	2.70	2.33	1.70	2.84	2.49	1.57	0/ 0
	LSTM	**3.43**	**2.89**	**2.16**↑	2.71	2.34	**1.75**	**2.98**	2.54	**1.71**	1/ 0
Ca	Rating	3.51	**3.17**	**2.79**	2.54	2.27	**2.11**	**0.79**	**0.72**	0.54	I
	Rand	3.03↓	2.83	2.57	2.35	2.16	1.98↓	0.72	0.68	0.50	0/ 1
	SWN	1.88↓	2.14↓	2.72	0.72↓	0.85↓	2.01↓	0.49↓	0.55↓	0.54↓	0/ 9
	SVM	3.35	3.15	2.69	2.53	2.25	2.07	0.74	0.70	0.53	0/ 0
	CNN	**3.52**	**3.17**	2.77	2.51	2.26	2.07	0.78	0.71	0.54	0/ 0
	LSTM	3.50	3.15	2.78	**2.56**	2.26	2.10	0.78	**0.72**	**0.55**	0/ 0
GeoSoCa	Rating	3.68	**3.04**	2.23	2.64	2.09	1.51	3.92	3.17	**2.22**	I
	Rand	3.08↓	2.57↓	1.79↓	2.44	2.03	1.47	3.59	2.97	2.06	0/ 3
	SWN	1.32↓	1.35↓	1.39↓	0.52↓	0.58↓	1.06↓	1.83↓	1.75↓	1.44↓	0/ 9
	SVM	3.52	2.93	2.17	2.84	2.21	1.78↑	3.68↓	2.98↓	2.06	1/ 2
	CNN	3.62	2.90	2.18	2.86↑	2.29	1.79↑	3.71↓	3.02↓	2.09	2/ 2
	LSTM	**3.73**	2.96	**2.28**	**2.97**↑	**2.38**↑	**1.87**↑	**3.96**	**3.21**	2.15	3/ 0

Table 5.3: Recommendation performances of rating and sentiment-based approaches on three datasets (reported evaluation measures are *100). Using the t-test, statistically significant increases (resp. decreases) with respect to the corresponding rating-based baseline are indicated by ↑ (resp. ↓).

line (i.e. rand), can still indicate a recorded interaction between a corresponding user-item pair. Therefore, the comparable performances, when examined by the used P@5, P@10 and MAP metrics, in the use of sentiment scores between the Rand baseline and others (i.e. SWN, SVM, CNN and LSTM) are expected.

Next, we consider GeoSoCa and its components Geo|So|Ca for each dataset. For the geographical information, the rating and sentiment-based models provide statistically indistinguishable results (according to a paired t-test; p-value < 0.05), regardless of the sentiment classifi-

cation approach used. Next, for the social influence model (i.e. So), the distinction among the approaches is clear: SWN and Rand significantly degrade effectiveness compared to the rating-based baseline in 9 cases; the learned approach (SVM) significantly degrades effectiveness in 3 cases (P@10 for Phoenix and P@5 & P@10 for Cross-City); on the other hand, the deep-learned sentiment approaches (CNN and LSTM) are statistically indistinguishable from the corresponding rating-based baseline (only one significant increase on P@10: 1.95 $\rightarrow$ 2.16). Indeed, it is promising that the latest approaches (CNN and LSTM), which were shown to be the most effective sentiment classifiers in Section 5.2.4.1, also result in the recommendation models with the highest effectiveness, suggesting that they could be a suitable proxy for user ratings.

For the categorical information (i.e. Ca), recall that our substitution strategy replaces not only the users' preferences but also the aggregated popularity of the category for that user, as per Equation (5.1). On examining Table 5.3, the learned and deep learning sentiment approaches are able to provide comparable performances to the corresponding rating-based baseline. The same observation also holds with the social information-based model. Moreover, similar to the social information-based model, the recommendation effectiveness also aligns with the performances of sentiment classification, with CNN and LSTM providing the most effective results.

Finally, we consider the GeoSoCa model - where we observe that the combined use of the geographical, social and category features, when using the sentiment scores from CNN or LSTM, could still provide performances that cannot be statistically distinguished from those based on ratings (only 1 significant decrease). Moreover, in 5 cases there were actually significant increases in effectiveness by deploying CNN or LSTM. Therefore, in answer to research question RQ 5.2, we find that only the sentiment-based user preference scores from the state-of-the-art deep learning-based sentiment classification approaches (i.e. CNN and LSTM) can provide similar effectiveness to the rating-based models. This indicates that the learned sentiment information (i.e. the reviews' sentiment) can be a reliable surrogate for ratings as input to ranking-based recommendation techniques.

In this section, we have explored the performances of various sentiment analysis approaches at identifying the polarity of comments about items. In particular, we used the sentiment information as a surrogate to the users' explicit ratings for item recommendations. As we argued in the thesis statement (see Section 1.3), by inferring the sentiment property of reviews, review-based recommendation approaches can accurately estimate the users' preferences and obtain improved recommendation effectiveness. The experimental results of this study showed the effectiveness of the reviews' sentiment, compared to the users' explicit ratings, when applied to recommendation approaches. However, it is still unclear if the sentiment property of reviews can be effectively used to identify useful reviews, to improve the performances of review-based rec-

ommendation models (e.g. the DeepCoNN model discussed in Section 3.1). As a consequence, for the second study in the next section, we propose a new sentiment information-based attention mechanism that helps to identify user reviews that are more likely to enhance the accuracy of a review-based recommendation model.

5.3 Attention-based Sentiment Property Model for Rating Prediction

In this section, we present a study that further examines the effectiveness of using sentiment analysis within a review-based recommendation model to address a recommendation task. In particular, we apply the results from the sentiment analysis to the rating prediction task, which is a type of recommendation tasks and that has been frequently investigated by many well-cited review-based recommendation approaches (e.g. DeepCoNN (Zheng et al., 2017) and NARRE (Chen et al., 2018b)). Indeed, as we discussed in Section 2.2, rating prediction is a classical recommendation task (Ricci et al., 2011), where the recommendation system aims to accurately predict the user rating of an unseen item, so as to better estimate which items to recommend to a user. The predictions are typically based on the existing ratings by users. The rating prediction task remains a challenging and open problem. Indeed, the effectiveness of existing rating prediction-based recommendation systems is still limited, suffering from various types of challenges, including accuracy, data sparsity and the cold-start problem (Davagdorj et al., 2020; Zheng et al., 2017). Therefore, as we discussed in Section 3.1, many approaches have been proposed to leverage user reviews (Manotumruksa et al., 2016b; Wang et al., 2019d) – including the sentiment of the reviews (Lei et al., 2016; McAuley and Leskovec, 2013) – to improve the rating prediction accuracy. The users' reviews can enrich both user and item representations, while sentiment information is often useful for extracting the user preferences (Lei et al., 2016). However, not all reviews are useful to enhance the rating prediction performance, since they may convey varying actionable information about the users' preferences (Chen et al., 2018b). Recently, a number of approaches have made use of an attention mechanism to estimate the usefulness of reviews (Chen et al., 2018b; Seo et al., 2017). This attention mechanism focuses on the parts of review content that contribute to the rating prediction. While these existing approaches demonstrate that the attention mechanism can improve the rating prediction performance, they (i.e. the approaches of (Chen et al., 2018b; Seo et al., 2017)) do not leverage the sentiment information actually captured by the reviews.

Given the effectiveness of sentiment information in extracting the users' preferences, we

hypothesise that sentiment information should also be used in estimating the usefulness of reviews, so as to further improve the rating prediction performance. Indeed, reviews with a clear polarised sentiment (i.e. positive or negative) typically convey richer information about items and are more likely to influence the users' decision making when interacting with the corresponding items (Lei et al., 2016). In the literature, several approaches focused on leveraging the sentiment information as an additional feature to address the rating prediction task (Hyun et al., 2018; Wang et al., 2019d), while ignoring the advantage of a potential correlation between the sentiment polarity and the usefulness of reviews in the users' decision making process. In this section, we propose instead to directly leverage the sentiment scores of reviews (i.e. reviews' sentiment property) to address the aforementioned limitation. Recall the definition of the sentiment score from Section 5.2.1.3. The sentiment score of a review is estimated as the probability of the review having a clear positive or negative polarity as determined by a sentiment classifier. These sentiment scores are then used in a customised attention mechanism to identify informative reviews with rich user preferences. Hence, our resulting SentiAttn model assumes that reviews with clearly pronounced user preferences are useful for effective rating prediction. In addition, SentiAttn adds another attention mechanism (i.e. global attention (Luong et al., 2015)) to capture and model the importance of the parts of reviews that are likely to enhance the rating prediction performances. On the other hand, previous works on *neural architecture search* (Jiang et al., 2019b; Liu et al., 2019b) showed that fine-tuning the architecture of a neural model could have a marked positive impact on the model's performance. To leverage the advantage of fine-tuning the neural models' architectures, in this study, we propose a strategy where we first concatenate the users' and items' positive and negative reviews as input to SentiAttn, resulting in different SentiAttn architectures with various number of channels (e.g. if we concatenate all reviews for both users and items, then this leads to a single channel-based SentiAttn model). Next, we fine tune the architecture variants of our proposed SentiAttn model with different channel setups on the validation sets of two datasets from Yelp and Amazon, which are the two commonly used review-datasets (Catherine and Cohen, 2016; Ni et al., 2019; Zheng et al., 2017), so as to optimise the performances of SentiAttn on different datasets.

The rest of this section is structured as follows. In Section 5.3.1, we formally state the problem addressed and describe our proposed SentiAttn model and its rationale, as well as the proposed model architecture fine-tuning strategy to further enhance the model's performance. Section 5.3.2 introduces our research questions and describes our experimental setup. The obtained results are presented and analysed in Section 5.3.3.

Table 5.4: Review examples with sentiment information.

Positive and High Sentiment Score	
Rating: 5 **Sentiment Score:** 0.9726 **Category:** grocery and gourmet good	**Review 1:** This beverage is so delicious. I would like to order more in the future. I drink it to relax.
Positive but Low Sentiment Score	
Rating: 5 **Sentiment Score:** 0.1783 **Category:** grocery and gourmet good	**Review 2:** My husband insists on making his own yogurt and won't use any other starter. This assures the same consistency month after month.

5.3.1 The SentiAttn Model

In this section, we first recall the description of the rating prediction task introduced in Section 2.2 and the notations used. Next, we illustrate the motivation of using sentiment information to identify useful reviews and describe our proposed SentiAttn rating prediction model.

5.3.1.1 Task Definition

Recall from Section 2.2 that the rating prediction task aims to predict the ratings of unseen items. Consider a set of users U and items I (of size M and N, respectively). We also have the one-hot embedding vectors X_U and X_I, which map users and items to different randomly initialised vectors. User ratings can be encoded in a rating matrix $R \in \mathbb{R}^{M \times N}$, where entries $r_{u,i} \in R$ represent the previously observed ratings with a range from 1 to 5. In the rating prediction task, we aim to accurately predict the rating $r_{u,i}$ of an unseen item i for user u. Moreover, each rating $r_{u,i}$ is associated with a textual review $c_{u,i}$. In particular, different from the definition of the sentiment scores of reviews in Section 5.2.1.3, which simply calculate the probability of a review being positive, in this study, for each review $c_{u,i}$, the sentiment score $s_{u,i}$ indicates the probability of the review being polarised, i.e. being strongly positive or strongly negative.

5.3.1.2 Review Sentiment Information Analysis

To motivate the use of sentiment information in identifying useful reviews, we provide two illustrative review examples in Table 5.4. These two reviews are both positive and 5 star-rated. However, when we compare these two reviews, Review 1 better conveys the user's preferences, while Review 2 simply describes a personal event, making it hard for the model to capture the user's preferences. Therefore, Review 1 is deemed more useful than Review 2. In particular, the sentiment scores of Reviews 1 and 2 mirror their usefulness difference (Review 1 is scored 0.9726 as being strongly positive while Review 2 is scored 0.1783 only). Therefore, we propose to leverage the relationship between the sentiment scores and the usefulness of reviews in our proposed

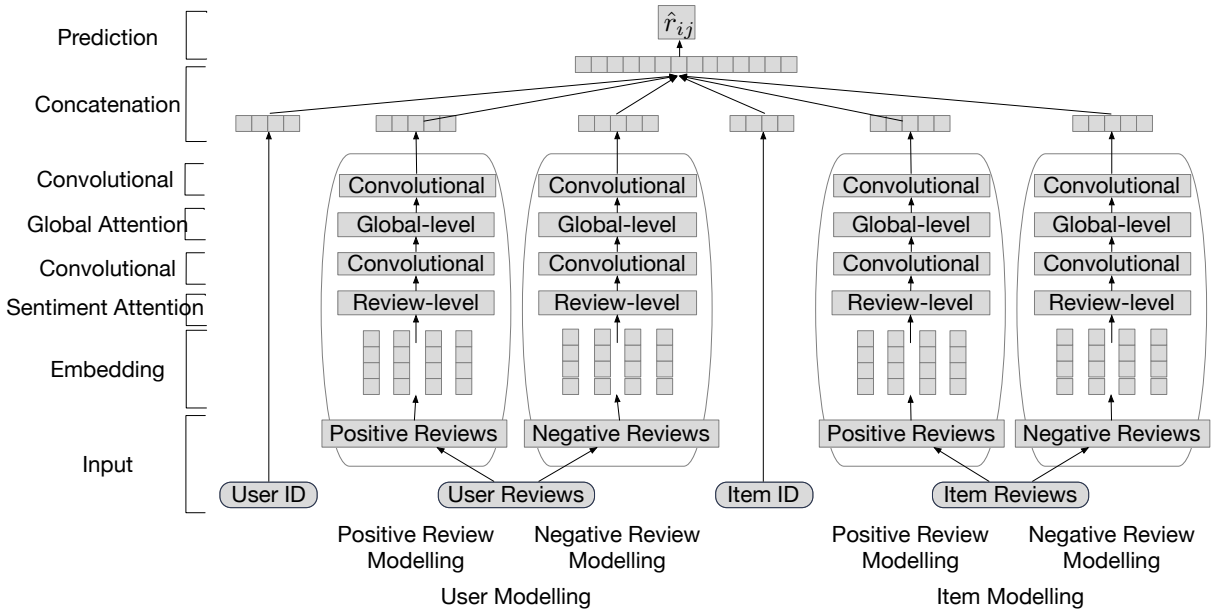


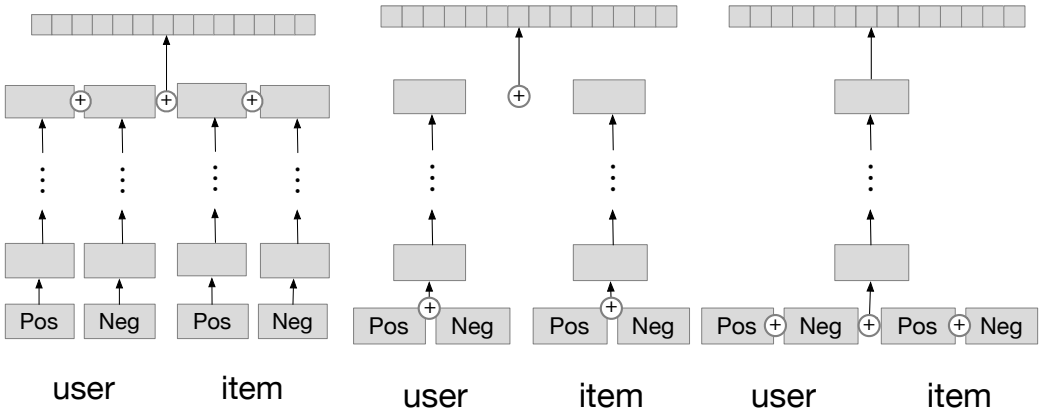
Figure 5.3: The architecture of our proposed SentiAttn model

SentiAttn model. Our model identifies useful reviews through the novel integration of sentiment information within an attention mechanism to improve the rating prediction performance.

5.3.1.3 Model Architecture

Our SentiAttn model aims to encode the review usefulness information through their sentiment scores. To achieve this, SentiAttn first uses a customised sentiment attention mechanism to embed the review usefulness information. After that, it also integrates another global attention mechanism (Luong et al., 2015) to capture the parts of reviews that are likely to enhance the rating prediction performance. Figure 5.3 illustrates the architecture of SentiAttn, which consists of eight layers from the input to the rating prediction layer, described further below:

- **Input & Embedding Layers:** In the input layer, users are represented by the reviews they have posted for items while items are represented by the reviews given by users. In particular, the input layer groups reviews into positive and negative reviews according to their corresponding rating values. If the rating $r_{u,i} \geq 4$, the review $c_{u,i}$ is positive, else, if the rating $r_{u,i} \leq 2$, the review $c_{u,i}$ is negative. A review $c_{u,i}$ with a rating of $r_{u,i} = 3$ or with no provided rating is classified as positive or negative according to a CNN-based binary sentiment classifier, which was experimentally validated to be an effective sentiment classifier in Section 5.2.4.1.Therefore, our SentiAttn model can be divided into



(a) Four Channels (b) Two Channels (c) One Channel

Figure 5.4: Architectures of the original SentiAttn four channels-based model and its variants (i.e. one and two channels-based).

four parallel networks (i.e. four channels), which model the positive and negative reviews for users and items.

The architecture of our SentiAttn model is flexible and can possibly have two additional variants (i.e. one channel or two channels). As shown in Figure 5.4, instead of modelling the polarised reviews for user and items individually, we can concatenate all the reviews for the user or the item, resulting in a SentiAttn model variant with two channels. Moreover, if we further concatenate all the reviews of the user and the item together, we can obtain the one channel SentiAttn variant. In particular, for each resulting channel, its review modelling pipeline remains the same as each individual channel depicted in Figure 5.3. It is of note that the one channel-based model variant can only be leveraged by a model that uses a value mapping-based predictor (e.g. the factorisation machine and the multilayer perceptron that we discussed in Section 3.1) and not an interaction-based predictor (e.g. the dot product function), which needs at least two inputs. In particular, we investigate which SentiAttn model variant exhibits the best performances on the used datasets. Next, following (Chen et al., 2018b; Zheng et al., 2017), in the embedding layer, we convert the reviews text into embedding vectors, denoted as X, which are then given as input to the next layer. Note that such an embedding process and the computation of the reviews' associated sentiment scores are the two most time-consuming stages of training SentiAttn. To address efficiency issues, we pre-calculate the embedding vectors of reviews as well as their sentiment scores – before the model training process – to significantly enhance the efficiency of SentiAttn and make it faster to train.

- **Sentiment Attention Layer:** In this layer, we customise a sentiment attention mechanism to encode the usefulness of reviews. Our sentiment attention mechanism is inspired by the dot-product attention function (Vaswani et al., 2017), which was discussed in Section 2.4.4. The dot-product attention function learns the importance (weight) of different embedding vectors. Then, it multiplies the resulting weighting vectors with the initial word embedding vectors to apply the attention mechanism. Unlike the dot-product attention function, our sentiment attention mechanism obtains the weighting vectors from the sentiment scores of the reviews, which indicate the reviews being polarised, not simply being positive like in Section 5.2.1.3.

  First, the reviews have been labelled as positive or negative in the previous layer. After that, we process these reviews with a given sentiment classifier and obtain the corresponding probability of the positive reviews being positive or the negative reviews being negative (denoted as $p_{u,i}$, which naturally ranges from 0 to 1). The corresponding sentiment scores for the positive reviews are $s_{u,i} = p_{u,i}$. Conversely, we use $s_{u,i} = 1 - p_{u,i}$ for the negative reviews. Hence, the sentiment score indicates the probability of a given review being polarised (positive or negative), and a review is deemed more useful if its sentiment score is closer to 1. Next, with a given review embedding vector X, and its sentiment score vector S, the converted vector X' is calculated as follows:

$$X' = ((SX^T)^T \oplus X) \tag{5.2}$$

  where $\oplus$ is a residual connector.

- **Convolutional Layer** Our SentiAttn model applies the convolution operation, as in (Seo et al., 2017; Zheng et al., 2017), on the latent vector X' with g neurons to generate feature vectors for the next layer. Each neuron applies the convolution operation to a sliding window t over latent vectors with width T. The convolution operation of neuron e is obtained as follows:

$$z_e = f(X'_{1:T} * K_e + b_e) \tag{5.3}$$

  where $f(.)$ indicates an activation function to filter the output of the convolution operation, $*$ is the convolution operator of neuron e on the corresponding window of vectors and b_e is a bias parameter (Kim et al., 2016). After applying the convolution operation, we apply the max pooling function over the output feature vectors, denoted as Z, and obtain the resulting vector o for each neuron (i.e. $o = max(z_1, z_2, ..., z_e^{T-t+1})$). Next, the outputs of the g neurons are concatenated together into the latent vector X_c.

- **Global Attention Layer:** Apart from the proposed sentiment attention layer above, we also use the global attention mechanism from (Luong et al., 2015). Accordingly, we add the global attention layer to SentiAttn to model the parts of review content that are likely to contribute to enhancing the rating prediction performances.

  In particular, the global attention mechanism considers all review embeddings as input and calculates the global attention score vector G of the embedding input:

$$G = \text{SoftMax}(W_g X_c) \tag{5.4}$$

  The embedding input X_c is then further weighted by the global attention score vector G as $X_g = (GX_c^T)^T$. In particular, after the global attention layer, we add another convolutional layer, which is the same as the one above the sentiment attention layer, to process the review embeddings. The outputs from the convolutional layer are used as the final latent feature vectors for each channel.

- **Concatenation & Prediction Layer:** In the concatenation layer, we concatenate the latent vectors from two groups of inputs: (1) the resulting latent feature vectors from the last convolutional layers of the review modelling channels; (2) the one-hot embedding vectors of each user and item. We refer to the concatenated vector as o.

  Next, in the prediction layer, we use a two-order factorisation machine (Rendle, 2012) as the rating predictor, which has the ability to capture the patterns inside of the data to improve the model's performance (Zhang et al., 2016). This predictor has also been widely used in the literature to address the rating prediction task (Chen et al., 2018c; Rendle et al., 2011; Cheng et al., 2014). Each predicted rating $\hat{r}_{u,i}$ is calculated as follows:

$$\hat{r}_{u,i} = w_0 + b_u + b_i + \left(\sum_{j=1}^{|o|} w_j o_j \right) + \left(\sum_{j=1}^{|o|} \sum_{k=j+1}^{|o|} o_j o_k \mathbf{w}_{j,k} \right) \tag{5.5}$$

  This equation has five summands: w_0 is the global bias parameter (Rendle, 2012); next, b_u and b_i correspond to the bias parameters for user u and item i, respectively; in the fourth summand, w_j models the weight of the j_{th} variable in o; the final summand models the interactions between pairs of variable vectors o_j and o_k in o, weighted by a factorised parameter $\mathbf{w}_{j,k} \approx \langle \mathbf{v}_j, \mathbf{v}_k \rangle$ as in (Rendle, 2012). SentiAttn is trained by minimising the prediction error between the true rating value $r_{u,i}$ and the predicted rating value $\hat{r}_{u,i}$ with the MSE function.

5.3.2 Experimental Setup

We now examine the performance of SentiAttn through experiments on two real-world datasets, in comparison to a number of classical and state-of-the-art rating prediction models. In particular, we further validate the performance of SentiAttn on the cold-start users (i.e. the users with few interaction records). Therefore, we aim to address the following three research questions:

1. **RQ 5.3:** Which architecture variant of the SentiAttn model (based on 1, 2 or 4 channels) performs the best on the two used datasets?

2. **RQ 5.4:** Does our SentiAttn model outperform other state-of-the-art models in addressing the rating prediction task and how much does it benefit from (i) the proposed sentiment attention mechanism and (ii) the global attention mechanism?

3. **RQ 5.5:** Does SentiAttn outperform the existing baselines when making rating predictions for cold-start users?

5.3.2.1 Datasets

To perform our experiments, we use two real-world datasets: (i) a Yelp[4] dataset, and (ii) an Amazon Product dataset.[5] This Yelp dataset contains a large number of user reviews on venues located in Phoenix, USA. The Amazon dataset contains user reviews on products among six categories.[6] These two datasets have also been used in previous studies (McAuley and Leskovec, 2013; Seo et al., 2017) and the Yelp dataset variant, which includes users' reviews on different cities, has also been used in our previous study (see Section 5.2.3.1 and 5.2.3.2). The statistics of these two datasets are provided in Table 5.5. Following a common setup (Chen et al., 2018b; Zheng et al., 2017), these two datasets are randomly divided into 80% training, 10% validation and 10% testing sets. Moreover, we denote those users with less than 5 reviews in the training dataset as the cold-start users. Table 5.5 shows that the Yelp dataset is more sparse than the Amazon dataset. This observation suggests that the data sparsity's influence might be amplified in a given model's performance when experimenting with the Yelp dataset. Moreover, as per the positive rating percentages in Table 5.5, it is of note that most user reviews are positive in both datasets.

[4] https://kaggle.com/c/yelp-recsys-2013

[5] http://jmcauley.ucsd.edu/data/amazon/

[6] 'amazon instant video', 'automotive', 'grocery and gourmet food', 'musical instruments', 'office products' and 'patio lawn and garden'

Table 5.5: Statistics of datasets.

Dataset	***Yelp***	***Amazon***
#Users	45,981	26,010
#Cold-start Users	33,306	7,874
#Items	11,537	16,514
#Reviews	229,907	285,644
% Density	0.043	0.066
% Positive ratings	67.88	81.24

5.3.2.2 Baselines and Evaluation Metrics

We compare our SentiAttn model to the following five baselines, which are classical recommendation approaches and widely cited review-based recommendation techniques. In particular, we recall the description of MF from Section 2.3.2, and the DeepCoNN and NARRE models, which have been illustrated in Section 3.1.

1. **MF** (Koren et al., 2009): MF is a widely used classical baseline, which characterises users and items with their rating pattern-based latent vectors.

2. **ConvMF** (Kim et al., 2016): ConvMF extends the latent feature vectors in MF with the embedding vector of reviews.

3. **DeepCoNN** (Zheng et al., 2017): DeepCoNN jointly models reviews to characterise users and items with latent vectors. This approach has been widely used as a strong review-based rating prediction model.

4. **D-Attn** (Seo et al., 2017): D-Attn is another review-based rating prediction model that includes two global and local attention mechanisms. D-Attn is another review-based rating prediction model. It includes two global and local attention mechanisms, to improve the explainability and rating prediction accuracy of a rating prediction model.

5. **NARRE** (Chen et al., 2018b): NARRE is a recent state-of-the-art attention-based rating prediction model. It weights reviews with its learned review usefulness estimates through the use of an attention mechanism.

Moreover, we examine the effectiveness of using our proposed sentiment attention mechanism in comparison to three further baselines derived from our SentiAttn model as follows:

- **Basic:** This variant removes both attention layers in the SentiAttn model, which results in a further baseline to compare with.
- **+Glb:** This variant is constructed by adding the global attention layer only to the *Basic* model.
- **+Sent:** This variant baseline integrates the sentiment attention layer to the *Basic* model.

As for the evaluation metrics, as we discussed in Section 2.2.1, we use Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) to measure the performances of SentiAttn and the baselines, which are the commonly used metrics to evaluate rating prediction models (Chen et al., 2018b; Zheng et al., 2017; Lei et al., 2016). In particular, to examine the statistical significance of the models' performances, we leverage both the paired t-test, with a significance level of $p < 0.05$, and the post-hoc Tukey Honest Significant Difference (HSD) (Sakai, 2018) test at $p < 0.05$ to account for the multiple comparisons with the t-tests.[7]

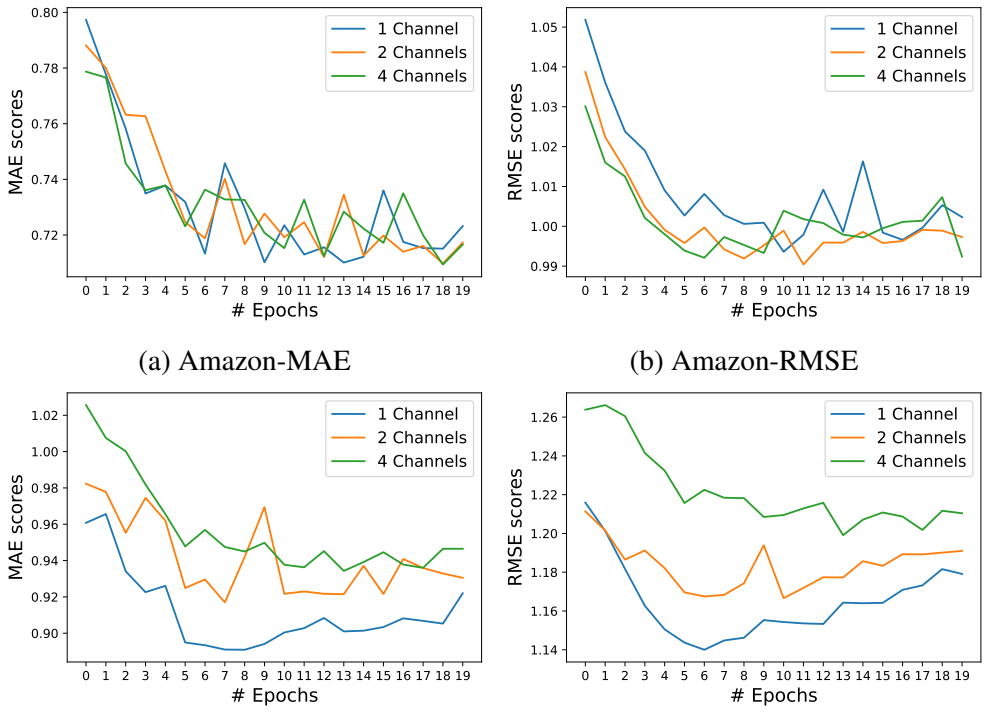
5.3.2.3 Model Setting

In the input layer, we use an existing CNN-based binary sentiment classifier to group reviews into positive and negative reviews, which we have shown to have a strong accuracy ($> 95\%$) for sentiment classification in Section 5.2.4.1. Other sentiment classifiers could have been used, but the investigation of such classifiers is beyond the scope of this thesis. For the used CNN-based binary sentiment classifier, we use the same experimental setup as in 5.2.3.1 and train it on 50,000 positive and 50,000 negative review instances that were sampled from a separate dataset, namely the Yelp Challenge Round 12 dataset.[8] Moreover, the classifier provides each review with its probability $p_{u,i}$ of carrying a strong polarised sentiment, so as to generate a sentiment score $s_{u,i}$ in the sentiment attention layer, as explained in Section 5.3.1.3. Next, in the embedding layer, we use the pre-trained GloVe (Pennington et al., 2014) word embedding dictionary,[9] similar to the experimental setup in Section 5.2.3.1, and map each word into an embedding vector with 100 dimensions. In the convolutional layer, following Seo et al. (2017), we set the kernel size to 100 and the activation function to ReLU. In particular, we use the Adam optimiser with a 10^{-4} learning rate. Moreover, it is of note that, to answer RQ 5.3, which investigates the performances of the considered SentiAttn architecture variants, we conduct experiments on

[7]Note that since RMSE is a non-linear aggregation of squared absolute error, a significance test cannot be conducted with this metric.

[8]https://www.yelp.com/dataset/challenge

[9]We also apply the pre-trained GloVe word embeddings within the baseline approaches, which ensures fair performance comparisons between approaches.



(a) Amazon-MAE (b) Amazon-RMSE

(c) Yelp-MAE (d) Yelp-RMSE

Figure 5.5: Performances of the SentiAttn model variants with different #channels on the validation sets.

the validation sets from the Yelp and Amazon datasets to select the best SentiAttn architecture – thereby mimicking the use of the validation sets for model selection.

5.3.3 Experimental Results

Next, in this section, we report and analyse our obtained results to answer each research question. In particular, in Section 5.3.3.1, we investigate the performance changes of our proposed SentiAttn model when varying its architectures. Then, in Section 5.3.3.2, with the optimised SentiAttn architecture, we examine its effectiveness in addressing the rating prediction task when compared to five baseline approaches. Finally, we further validate the performance of SentiAttn when applied to make rating predictions to cold-start users (Section 5.3.3.3).

5.3.3.1 RQ 5.3: Impact of Architecture Variants on Performances

We investigate which of the SentiAttn model architecture variants lead to the best rating prediction performances. In Figure 5.5, we report the performances of the three considered architecture

Table 5.6: Rating prediction accuracy; * denotes a significant differences in MAE with SentiAttn (w/ both the paired t-test and the Tukey HSD correction method, $p < 0.05$).

	All Users					**Cold-Start Users**			
	Yelp Dataset		**Amazon Dataset**			**Yelp Dataset**		**Amazon Dataset**	
	MAE	RMSE	MAE	RMSE		MAE	RMSE	MAE	RMSE
NMF	0.9866*	1.2630	0.8240*	1.0881	NMF	1.1690*	1.5025	0.9040*	1.1843
ConvMF	0.9748*	1.2329	0.7964*	1.0371	ConvMF	1.0785*	1.3812	0.8565*	1.1154
DeepCoNN	0.9247*	1.1885	0.7233*	0.9929	DeepCoNN	1.0462*	1.3506	0.7882*	1.0749
D-Attn	1.0040*	1.2106	0.8316*	1.0627	D-Attn	1.0154*	1.2394	0.8738*	1.1029
NARRE	0.9163*	1.1781	0.7065*	0.9783	NARRE	1.0289*	1.3481	0.7613*	1.0587
Basic	0.9084*	1.1769	0.7060*	0.9769	Basic	1.0003*	1.3602	0.7451*	1.0520
+Glb	0.8947	1.1734	0.6960	0.9723	+Glb	0.9867	1.3544	0.7253	1.0460
+Sent	0.8932	1.1476	0.6957	0.9685	+Sent	0.9817	1.2408	0.7190	1.0375
SentiAttn	**0.8888**	**1.1463**	**0.6841**	**0.9668**	SentiAttn	**0.9736**	**1.2327**	**0.7090**	**1.0273**

variants of our proposed SentiAttn model (namely the one, two and four channels-based SentiAttn architectures). Since we are using the MAE and RMSE error-based evaluation metrics, the lower the metrics' values, the higher is the model's rating prediction performance. First, we compare the performances of the SentiAttn variants on the Amazon dataset in Figure 5.5a and Figure 5.5b. For both MAE and RMSE, the three variants show similar trends and performances and are overall comparable. However, on the Yelp dataset, Figure 5.5c and Figure 5.5d show that the SentiAttn model with one-channel consistently outperforms both the original SentiAttn model with four channels and the two channels-based variant. These observed experimental results show different effectiveness of SentiAttn using various channel-based architectures when conducting experiments on different datasets. In particular, in the literature (e.g. the models in (Zheng et al., 2017; Lei et al., 2016)), the two channel-based architectures are the most commonly used. However, the two channel-based SentiAttn model gives the top performance on the Amazon dataset but performs the second-best on the Yelp dataset. Such distinct performances on different datasets indicate the necessity of fine-tuning the model architecture on a sample of the target datasets when addressing the rating prediction task. Therefore, in answer to RQ 5.3, we conclude that the one channel-based SentiAttn model performs the best on the Yelp dataset but the three model variants perform similarly on the Amazon dataset. Based on these results, we select the overall best variant model, namely the one channel-based SentiAttn model for the remaining experiments.

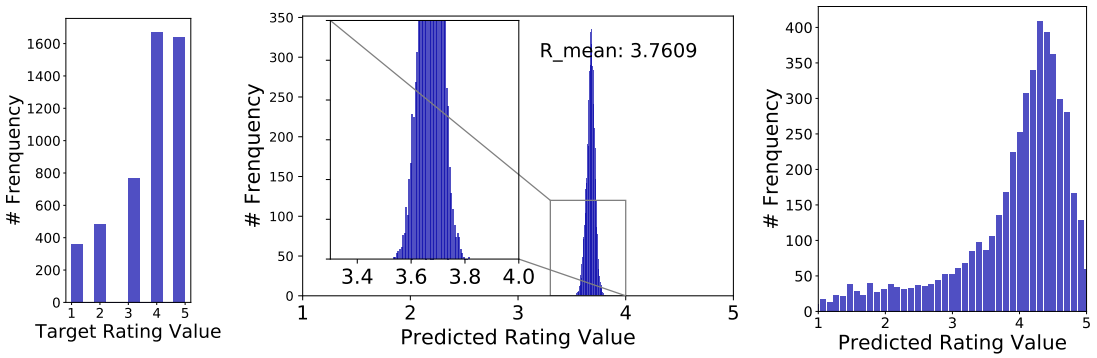
5.3.3.2 RQ 5.4: Comparison to the Baselines

Table 5.6 presents the prediction errors of both the rating prediction baseline models and SentiAttn. First, we consider the performance of our SentiAttn model in addressing the rating prediction task. In the obtained results for both the Yelp and Amazon datasets, SentiAttn significantly outperforms the baselines according to both the paired t-test and the Tukey HSD correction method. In particular, while D-Attn, NARRE and SentiAttn all use an attention mechanism to weight reviews with their estimated usefulness, our SentiAttn model, which relies on a novel sentiment attention and a global attention mechanism returns significantly smaller prediction errors in comparison to competitive baselines on both the Yelp and Amazon datasets.

We also evaluate the usefulness of the global attention layer and the proposed sentiment attention layer in SentiAttn by comparing the performances of SentiAttn with the Basic, +Glb, and +Sent models (introduced in Section 5.3.2.2). Table 5.6 shows that SentiAttn significantly outperforms (according to both the paired t-test and the Tukey HSD correction method, $p < 0.05$) the Basic model on both used datasets, which demonstrates the effectiveness of using the attention mechanisms. Moreover, the results show that the sentiment attention mechanism outperforms the global attention mechanism since it results in lower MAE and RMSE scores (0.8932 vs. 0.8947 (MAE) and 1.1476 vs. 1.1734 (RMSE) for +Sent vs. +Glb in Table 5.6). In particular, we observe that the sentiment attention mechanism is especially effective when measured by the RMSE scores. Indeed, the +Sent model outperforms both the Basic model and the +Glb model providing lower RMSE scores with wide margins on both used datasets. To answer RQ 5.4, we conclude that the obtained results empirically validate the effectiveness of our SentiAttn model in addressing the rating prediction task in comparison to strong baseline models. The results also show the effectiveness of using the sentiment attention mechanism – which weights the review input according to the corresponding review sentiment scores – thereby outperforming the global attention mechanism.

5.3.3.3 RQ 5.5: Cold-Start Users

We now evaluate the rating prediction performance of SentiAttn on cold-start users. As introduced in Section 5.3.2.1, we consider users in the training dataset with less than 5 reviews as cold-start users. The right hand side of Table 5.6 provides the rating prediction performances of SentiAttn and the various baseline models on both the Yelp and Amazon datasets for cold-start users. The results show that our SentiAttn model obtains a good cold-start performance by significantly outperforming all the strong baseline approaches from the literature on the Yelp and Amazon datasets.



(a) Target (b) D-Attn (c) SentiAttn

Figure 5.6: Cold-start user rating prediction performance comparison (D-Attn vs. SentiAttn) on the Yelp dataset.

Comparing the rating prediction results in Table 5.6 on the Yelp dataset, we note that as expected from the statistics of this dataset, the rating prediction performances of all models suffer from the cold-start problem. However, the cold-start problem appears to have only a small negative influence on the D-Attn model. To investigate the reasons behind the relative effectiveness of D-Attn in addressing the cold-start problem, we plot the predicted rating value frequency distribution of the cold-start users on the Yelp dataset using both D-Attn and our SentiAttn model in Figures 5.6b and 5.6c, respectively. These distributions are compared with the target rating distribution in Figure 5.6a. In Figure 5.6b of the D-Attn model, the predicted rating values shrink between around 3.55 and 3.80, which are all close to the average of the target rating value (i.e. $\overline{R} = 3.7609$). This distribution shows that the performance of D-Attn is less reliable in distinguishing the actual user preferences. On the contrary, in Figure 5.6c, the predicted rating value frequency distribution of our SentiAttn model ranges from 0 to 5 and its shape better aligns with the actual rating distribution of the Yelp dataset in Figure 5.6a.

We also compare the impact of using two attention mechanisms (namely the sentiment and global attention mechanisms) in addressing the cold-start problem. According to the results in Table 5.6, our sentiment attention mechanism outperforms the global attention mechanism in improving the rating prediction accuracy of the Basic model (e.g. 1.0003 $\rightarrow$ 0.9817 vs. 1.0003 $\rightarrow$ 0.9867 on the Yelp dataset) and lowers the variances of the rating prediction errors with a wider margin. For example, on the Yelp dataset, the Basic model benefits from using the global attention mechanism and lowers the RMSE score from 1.3602 to 1.3544. However, when applying the sentiment attention mechanism, the RMSE score of the Basic model is decreased from 1.3602 to 1.2408, indicating a higher improvement than when applying the global attention mechanism. Therefore, in answer to RQ 5.5, we find that our SentiAttn model is particularly

effective for the cold-start users compared with the five strong baselines from the literature. Our sentiment attention mechanism is also shown to be useful in improving the rating prediction accuracy, especially lowering the variance of the rating prediction errors for cold-start users. In particular, SentiAttn is more reliable than D-Attn in identifying user preferences, as illustrated by the predicted rating distributions. Therefore, according to the discussed experimental results, we have validated our proposed claim in our thesis statement (see Section 1.3), that by inferring the sentiment information from reviews, we could significantly improve the performances of a review-based recommendation approach.

5.4 Conclusions

In this chapter, we argued that the users' explicit ratings, a type of sentiment property, is not always readily available in public review datasets. In particular, to leverage the sentiment property in our proposed review-based recommendation framework, we proposed to estimate the reviews' sentiment property by applying sentiment classifiers on the reviews. Moreover, we conducted two consecutive studies to validate the effectiveness of the predicted sentiment property.

We first compared the learned sentiment scores to the ratings and then applied such scores to our proposed review-based recommendation model (SentiAttn). In particular, we also examined the performances of various sentiment analysis approaches at identifying the polarity of reviews. We found that CNN outperforms other approaches in terms of overall accuracy, while LSTM performs better in classifying positively labelled reviews (see Figure 5.2). Next we substituted the users' ratings with sentiment scores from state-of-the-art sentiment classification approaches (i.e. CNN and LSTM) to address the ranking-based recommendation task. We found that the resulting GeoSoCa-models were rarely significantly degraded with respect to the effectiveness and were actually seen to be significantly enhanced in several cases (see Table 5.3). These experimental results suggest that the sentiment scores calculated by the top-performing classifier can be used as an effective substitute for the users' explicit ratings in recommendation models.

Next, in the second part of this chapter, we focused on developing techniques to effectively leverage the sentiment information to enhance the performances of review-based recommenders. We proposed the SentiAttn model, which leverages a novel sentiment attention mechanism, to achieve this target. We first investigated the effect of using different architecture variants for our SentiAttn model and concluded on the necessity of fine-tuning the architecture of a rating prediction model on different datasets (see Section 5.3.3.1). Next, our results on two real-world datasets showed that SentiAttn significantly and consistently outperformed one classic

and four existing state-of-the-art rating prediction models. Moreover, the experimental results in Table 5.6 showed that the proposed sentiment attention mechanism outperforms the global attention layer in improving the rating prediction accuracy, resulting in a lower variance of the rating prediction errors. Furthermore, we showed that SentiAttn provides a significantly effective rating prediction accuracy, compared to the same group of baselines, and a reliable indication of user preferences for cold-start users (see Section 5.3.3.3).

In summary, in this chapter, we presented two consecutive studies that examined the effectiveness of the extracted sentiment property when leveraging sentiment property in a recommendation model. We have shown that the sentiment property can be useful as a replacement to the users' explicit ratings to improve the performance of review-based recommendation models. These empirical conclusions also validate the part of our thesis statement (see Section 1.3) that by inferring the sentiment properties from reviews, the recommendation model can collect accurate user preferences information, thereby increasing the recommendation effectiveness. Next, in Chapter 6, we argue that similar to the sentiment property, the reviews' helpfulness property is also not always readily available. Therefore, we aim to address the availability of reviews' helpfulness property with review helpfulness classification approaches and examine the effectiveness of the resulting helpfulness property in review-based recommendation models.

Chapter 6

Helpfulness Property Extraction and Integration

6.1 Introduction

In Chapter 5, we validated a part of our proposed thesis statement (Section 1.3), namely that by inferring the sentiment property of reviews, a review-based recommendation system can better estimate the users' preferences and obtain an improved recommendation performance. Next, in this chapter, we investigate the extraction and integration of the reviews' helpfulness property. Recall our proposed taxonomy of review properties from Section 4.4. The reviews' helpfulness property includes both the helpful votes as given by reviewers and the helpfulness scores estimated by a review helpfulness classifier on the reviews' text. However, as we argued in the thesis statement, similar to the ratings in the sentiment property, the reviews' helpful votes require human efforts to obtain, which makes them also frequently unavailable (e.g. the TripAdvisor, IMDB and Airbnb datasets in Table 4.1). Therefore, similar to the use of sentiment scores in addressing the extraction of the reviews' sentiment property, we aim to accurately estimate the helpfulness of reviews by leveraging effective binary classifiers.

A generic binary classifier focuses on modelling data with both positive and negative ground truth labels. However, in the review helpfulness classification, the helpful votes frequently contain only a few positive and many unlabelled instances. In particular, the unlabelled reviews may contain both helpful and unhelpful instances. This data incompleteness harms the accuracy of identifying the helpful and unhelpful reviews (Du Plessis et al., 2015). Therefore, this chapter focuses on accurately estimating the reviews' helpfulness from both the weakly labelled available helpful votes and the review text via weakly supervised classification techniques. Then, we

evaluate the effectiveness of the extracted reviews' helpfulness when applied in addressing the rating prediction task, following the evaluation of the estimated reviews' sentiment in Chapter 5.

The weakly supervised classifiers have been proposed to address classification with noisy, limited or imprecise data resources for a generic binary classification task (Zhou, 2017). Indeed, binary classification with incomplete positive instances and unlabelled instances can also be addressed by a weakly supervised learning process (Bao et al., 2018). Among various weakly supervised approaches, the Positive-Unlabelled learning approach (aka PU learning) is a popular solution in addressing cases where the data has few positive instances and many unlabelled instances, by leveraging the estimation of the class priors (Blanchard et al., 2010; Scott and Blanchard, 2009). However, as Du Plessis et al. (2015) argued, the class prior estimation-based solutions lead to a systematic estimation bias. Therefore, in this chapter, we propose to conduct binary weakly-supervised classification on data with incomplete positive instances and unlabelled instances without the aid of an estimated class prior for the unlabelled examples. Moreover, (Du Plessis et al., 2015) proposed to address the classifier bias problem of PU learning by applying different loss functions for the positive and unlabelled classes (an approach that we will refer to as **C-PU**). In this chapter, we also argue that using two customised loss functions for the positive and unlabelled data could help a classification model to fully leverage labelled and weakly labelled data, respectively. Hence, we follow (Du Plessis et al., 2015) by using different loss functions to different classes. Meanwhile, Ishida et al. (2018) proposed a classification approach that solely relies on the positively-labelled instances to generate a classifier according to the positivity or the confidence of the examples being positive. However, this approach filters out unlabelled examples, which can lead to a problem of information loss and, therefore, it can negatively impact classification performance. Inspired by earlier works on PU learning (Du Plessis et al., 2015; Ishida et al., 2018), we propose a negative confidence-aware weakly-supervised binary classification *correction* approach, **NCWS** (see Section 6.2.2), which considers both positive and unlabelled examples with corresponding customised loss functions. Note that by a classification correction approach, we refer to a technique that can be applied to various classifiers to adjust their classification results and improve their classification performances. In particular, we use an ordinary loss function for the positive class instead of the composite loss function used in (Du Plessis et al., 2015). Our approach is also different from (Ishida et al., 2018), which only uses the positive class for binary classification. For the negative class, instead of using an ordinary loss function for the unlabelled class, we design a customised loss function for the unlabelled class which considers the distinct probabilities of the unlabelled instances being negative (i.e. negative confidence). We leverage the properties of the unlabelled reviews (i.e. their age in the dataset since their creation time) to estimate their probabilities of

being negative. As we will explain later in Section 6.4.2, these properties are chosen based on their likely correlation with the negative class. Indeed, our proposed NCWS approach uses additional complementary information to the content and labels of instances (namely, the reviews' age) to infer the likelihood of the unlabelled reviews belonging to the unhelpful class.

In the review helpfulness classification task, modelling and representing reviews is key to the development of effective review helpfulness classifiers. Most existing approaches model the content of user reviews and the corresponding rating information, then make predictions on the review helpfulness labels (i.e. helpful or unhelpful). In this chapter, we reproduce many state-of-the-art review helpfulness classification approaches as baselines and refer to these approaches as the *basic* classifiers (see Section 6.4). We select the basic classifiers with the best performances and use them to evaluate the effectiveness of our proposed NCWS classification correction approach. Next, different from the used dot-product attention mechanism that we used in Section 5.3.1.3, we explore another technique for review property encoding, which is a filtering operation (detailed in Section 6.6). We use only those reviews that are predicted to be helpful in a review corpus as input to recommendation models. In particular, we extensively validate the effectiveness of our proposed NCWS approach in estimating the reviews' helpfulness property by investigating the extent to which accurately identifying additional helpful reviews can further enhance an existing review-based recommendation model. In particular, we use the DeepCoNN model proposed by Zheng et al. (2017), which is a representative state-of-the-art review-based rating prediction model (see the description of DeepCoNN in Section 3.1).

The remainder of this chapter is organised as follows: We state the tackled problem and the methodology underpinning our NCWS approach in Section 6.2. In Section 6.3, we list a number of review helpfulness classifiers. Next, in Section 6.4, we introduce our research questions and briefly describe the three used Yelp and Amazon-based datasets, which have also been previously used in the two studies of Chapter 5. We also describe the experimental setup for the basic classifiers, the baseline classification correction approaches and our NCWS approach, as well as the used evaluation metrics for performance comparison. In Section 6.5, we analyse the results from the experiments to answer the research questions. In Section 6.6, we show the effectiveness of using a filtering technique to implement the proposed review property encoding strategy. In particular, we demonstrate the effectiveness of the predicted reviews' helpfulness by our proposed NCWS approach by improving the accuracy of a rating prediction approach. Finally, we summarise the conclusions of this chapter in Section 6.7.

6.2 Methodology

First, we introduce the problem of a review helpfulness classification with limited positive and abundant unlabelled reviews, as well as the used notations (Section 6.2.1). Then, in Section 6.2.2, we illustrate our proposed NCWS approach and how we derive the loss functions for the positive and negative classes, respectively.

6.2.1 Problem Statement

The binary review helpfulness classification task consists in identifying the helpful reviews in a corpus of reviews using binary classification approaches. This problem consists of two main objects, the set of reviews $C = \{c_1, c_2, ..., c_N\}$ of size N and its corresponding class label set $Y = \{y_1, y_2, ..., y_N\}$ with label $y_i \in \{+1, -1\}$. Our objective is to obtain an unbiased and accurate classifier $g(x) \rightarrow \{+1, -1\}$, which can accurately identify the helpful and unhelpful instances by modelling the limited helpful and abundant unlabelled reviews.

In our scenario, a -1 label from the ground truth indicates that a review is unlabelled, rather than being unhelpful. The unlabelled reviews could be the result of a number of reasons, such as when the review is not yet old enough to have gained sufficient viewers to provide it with helpful votes, or when the user interface may have not yet shown the review to users (Liu et al., 2008). For this reason, review helpfulness classification can be seen as an example of classification with limited positive instances and many unlabelled instances. In particular, for the classification task we are addressing, many of the reviews are unlabelled and could be either positive or negative. The classical binary classifiers segregate positive and unlabeled instances and assume that all unlabelled instances are negative, ignoring whether the unlabeled reviews are positive. However, recall our argument that many unlabelled instances are actually positive but have not been fully labelled by users. Therefore, we expect a good classifier to identify more helpful reviews, including both labelled and unlabelled ones. Next, we define our NCWS approach, which leverages the age property of the unlabelled reviews during classification. We use the number of days d since the review has been posted (i.e. its age in days) in addition to the content of the review to infer confidence estimates about the unlabelled reviews being actually negative. We validate the reliability of using the age of reviews as an adequate instance property in Section 6.4.2.

6.2.2 Negative Confidence-aware Weakly Supervised Approach (NCWS)

As we explained in Section 6.1, weak supervision provides more data to the learner. In this work, we apply weak supervision to a classifier to address the limited positive instances and

the preponderance of unlabelled instances. Indeed, in the review helpfulness classification task, we might reasonably assume that some unlabelled newer reviews may in fact be positive, but have not yet experienced sufficient exposure to users to gain helpful votes; conversely older unlabelled reviews are less likely to be helpful.

To address the likelihood of the unlabelled instances are indeed positive, we propose a notion of *negativity*, the likelihood that an unlabelled instance belongs to the (latent) negative class. In review helpfulness classification, we assume that the likely negativity of an unlabelled review increases with its age (i.e. the time the review has been posted). This is motivated by the assumption that an old review has a higher probability to receive helpful votes (Liu et al., 2008; Lee et al., 2018a). Therefore, the longer time an unlabelled review has been posted, the more likely such review will be unhelpful.[1] Indeed, negativity is orthogonal to the notion of positivity – used by the PU learning approach of (Ishida et al., 2018) that only uses positive examples – which is the confidence in the positive examples actually belonging to the positive class. In contrast, our approach models the *unlabelled* reviews – i.e. *NCWS* – based on the age property of these reviews, which indicates the confidence that the reviews are indeed unhelpful. In the following, we use age to describe and illustrate how we generate the negativity scores during classification.

Firstly, let the negativity score $n(c) = p(y = -1|c)$ for each unlabelled review c be a function of the number of days since the review has been posted ($d(c)$):

$$n(c) = \frac{log(d(c)+1)}{log(max(d(C))+2)} \tag{6.1}$$

where $max(d(C))$ indicates the age in days of the oldest review in the review set C. Indeed, we argue that the longer that an unlabelled review has been posted, the more likely that the review will be unhelpful. We normalise the value of the review's number of days since it has been posted $d(c)$ into the range $(0,1)$. A higher negativity score for a reviews denotes a larger probability that the reviews are unhelpful for users. Furthermore, let π_+ and π_- indicate the class priors $p(y = +1)$ and $p(y = -1)$, respectively.

Next, our NCWS classification approach builds a classifier, $g(c)$, by minimising the binary classification risk $R(g)$. Let the generic form of a classifier's risk $R(g)$ be as follows:

$$R(g) = E_{p(c,y)}\Big[\ell(y \cdot g(c))\Big] \tag{6.2}$$

where $E_{p(c,y)}$ indicates the expectation over $p(c,y)$ (i.e. the probability density of instance c for the corresponding label $y \in \{+1,-1\}$), while $\ell(\cdot)$ denotes the loss function of the classifier.

[1] We further validate the underlying assumption in our used datasets in Section 6.4.2.

Similar to the usage of *example positivity* in (Du Plessis et al., 2015), we leverage and incorporate the instance's negativity into the risk function (i.e. Equation (6.2)) as follows. First, we represent the risk function with the positive and negative prior probabilities as follows:

$$\begin{aligned} R(g) &= E_{p(c,y)}[\ell(yg(c))] \\ &= \sum_{y=\pm 1} \int \ell(yg(c))p(c|y)p(y)dc \\ &= \int \ell(g(c))p(c|y=+1)p(y=+1)dc + \int \ell(-g(c))p(c|y=-1)p(y=-1)dc \\ &= \pi_+ E_+\Big[\ell(g(c))\Big] + \pi_- E_-\Big[\ell(-g(c))\Big] \end{aligned} \tag{6.3}$$

The sum of the posterior probabilities of the two classes can be represented by the negative class-based posterior probability:

$$\pi_+ p(c|y=+1) + \pi_- p(c|y=-1) = p(c,y=+1) + p(c,y=-1) \tag{6.4}$$

$$= p(c) \tag{6.5}$$

$$= \frac{p(c,y=-1)}{p(y=-1|c)} \tag{6.6}$$

$$= \frac{\pi_- p(c|y=-1)}{n(c)} \tag{6.7}$$

Therefore, we can represent the positive part with the negative-based probabilities as follows:

$$\begin{aligned} \pi_+ p(c|y=+1) &= \frac{\pi_- p(c|y=-1)}{n(c)} - \pi_- p(c|y=-1) \\ &= \pi_- p(c|y=-1)(\frac{1-n(c)}{n(c)}) \end{aligned} \tag{6.8}$$

According to Equation (6.8), we can generate the positive summand of the risk function in Equation (6.3) as follows:

$$\begin{aligned} \pi_+ E_+\Big[l(g(c))\Big] &= \int \pi_+ p(c|y=+1)\ell(g(c))dc \\ &= \int \pi_- p(c|y=-1)(\frac{1-n(c)}{n(c)})\ell(g(c))dx \\ &= \pi_- E_-\Big[(\frac{1-n(c)}{n(c)})\ell(g(c))\Big] \end{aligned} \tag{6.9}$$

Recall from the discussion in Section 6.1. We then follow the strategy proposed by Du Plessis et al. (2015) in modelling the two classes with distinct loss or risk functions, which model the positive and unlabelled instances in different strategies. Indeed, for the positive class, we retain the original risk function (Equation (6.2)). As for the negative class, we combine Equations (6.3) and (6.9) as follows:

$$R(g) = \pi_{-}E_{-}\Big[(\frac{1-n(c)}{n(c)})\ell(g(c)) + \ell(-g(c))\Big] \tag{6.10}$$

Finally, we implement the risk function with the following objective function $J(g)$ for the positive and negative instances respectively:

$$J(g) = \begin{cases} min\sum_{i=1}^{n}\Big[\ell(g(c_i))\Big], & \text{if } y_i = 1 \\ min\sum_{i=1}^{n}\Big[(\frac{1-n(c)}{n(c))})\ell(g(c)) + \ell(-g(c))\Big], & \text{otherwise} \end{cases} \tag{6.11}$$

Thus far, we have formally introduced our NCWS classification correction approach with the aforementioned objective function (Equation (6.11)). Note that this is a general definition and can be applied to various generic binary classification approaches, and moreover, to various classification tasks for which a negativity score $n(x)$ can be defined (such as the age of review for review helpfulness). In this work, we apply NCWS to the loss function in two main classes of classifiers. The first class includes the SVM classifiers. SVM is a frequently used technique for classification (Kim et al., 2006; Moraes et al., 2013). The second class includes the neural network-based classifiers, specifically classifiers based on CNN (CNN was shown to be effective in addressing the classification task in Section 5.2.4.1) and BERT (Devlin et al., 2019). BERT has recently been shown to be effective in addressing the text classification task (Gao et al., 2019; González-Carvajal and Garrido-Merchán, 2020). Next, we introduce the used classifiers for review helpfulness classification.

6.3 Review Helpfulness Classification

In the following experiments, to demonstrate the generalisation of our NCWS approach, we use two families of classifiers. One is based on SVM along a set of hand-engineered features commonly used in the literature. We also use two neural network-based classifiers, namely CNN and BERT-based classifiers.

Table 6.1: Categorised hand-engineered features for SVM.

Structural Features	
LEN	The number of words included in each review.
NoS	The number of sentences contained in each review.
ASL	Average sentence length in each review.
PoQS	Percentage of question sentences in each review.
Structural	Combines all features in this structural feature category.
Lexical Feature	
UGR	Unigram, uses TF-IDF to generate a document feature vector for each review.
Syntactic Feature	
Syn	The percentage of nouns, adjectives and adverbs in each review.
Metadata Features	
Rating	The review's corresponding rating value
Age	The number of days since the review was posted (normalised using Eq. (6.1)).
ALL	This combines all hand-engineered features into one integrated feature.

6.3.1 SVM with Hand-Engineered Features

We use a support vector machine (SVM) to classify the users' posted reviews into helpful and unhelpful classes. Similar to (Kim et al., 2006), we use four groups of features in the SVM-based classifiers, namely Structural, Lexicon, Syntactic, and Metadata features. Table 6.1 lists the 10 applied features. To further enhance the basic SVM-based classifiers, we add another metadata feature, namely Age to the feature set. Age is a feature leveraged in our proposed NCWS approach as a key property of a review. Hence, it is added to the basic classifiers to provide additional insights into the performance of NCWS and also to make fairer comparisons. Finally, we combine all the features together into a single feature (ALL) that comprehensively considers all the review's information. For ease of notations, we denote by 'SVM-X', an SVM classifier that uses the list of features X (e.g. 'SVM-LEN' denotes the SVM classifier that is based on the LEN feature while SVM-ALL is the classifier that considers all features).

6.3.2 Neural Network (NN) Classifiers

We use CNN and BERT-based classifiers to experiment with the performance of state-of-the-art classifiers using neural-network-based approaches to validate the effectiveness of NCWS on

different classifier types. In our previous sentiment classification study (see Section 5.2.4.1), we observed a leading performance by the CNN model in identifying reviews with positive or negative sentiments. As for the BERT-based classifier, BERT is a pre-trained language model, which has been shown to outperform many other techniques when applied to address various classification tasks (e.g. patent classification (Lee and Hsiang, 2019) and news classification (González-Carvajal and Garrido-Merchán, 2020)). Therefore, we consider these two techniques as representative neural network classifiers and compare their performances to the classic SVM classifiers. We describe the implementation details of these two classifiers in Section 6.4.3.

6.4 Experimental Setup

In this section, we formulate our research questions, depict the used datasets and provide details on the experimental setup of the baselines, our NCWS approach and the evaluation metrics.

6.4.1 Research Questions

In the following, we evaluate the usefulness of our NCWS approach in comparison to other weakly supervised or other classification correction approaches from the literature (the *baselines*). First, we validate the effectiveness of the review helpfulness classification approaches introduced in Section 6.3, which are our *basic* classifiers:

RQ 6.1: How effective are the review helpfulness classification approaches? (Section 6.5.1)

Second, we evaluate our proposed NCWS approach when applied to those selected basic classifiers:

RQ 6.2: Can our proposed NCWS approach outperform other classification correction approaches on all three used datasets (i.e. Yelp and Amazon-based datasets) and can it improve the effectiveness when various classifiers with differing performances are used? (Section 6.5.2)

Finally, we demonstrate the usefulness of the review helpfulness models obtained using NCWS to in a recommendation scenario, as per our final research question:

RQ 6.3: Does an improved NCWS-based review helpfulness classifier benefit an existing state-of-the-art review-based rating prediction recommendation model? (Section 6.6)

6.4.2 Datasets

To address RQ 6.1 & RQ 6.2, we conduct experiments on three datasets from two data sources, namely the Yelp, and the Amazon, which have been used in our previous studies (see Sec-

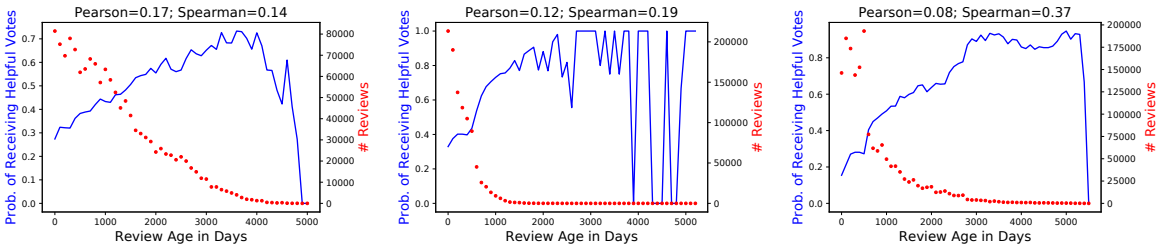
Table 6.2: Summary of the three used datasets.

Dataset	#Reviews	#Helpful	#Unlabelled	%Helpful
Yelp	1,373,587	621,112	770,780	45.21%
Kindle	982,619	407,019	575,600	41.42%
Electronics	1,689,188	633,154	1,056,034	37.48%

tions 5.2.3.2 and 5.3.2.1). We use the top categories from each data source with the largest amount of user reviews. Such a filtering strategy alleviates data sparseness in those categories with few user reviews, focusing instead on categories with a rich user feedback. For Yelp, we use the Yelp dataset challenge round 12.[2] After that, we collect reviews from the top three categories including 'restaurants', 'food', and 'nightlife'. For Amazon, we use a popular public Amazon review dataset (He and McAuley, 2016), which has been used in prior review helpfulness studies (Diaz and Ng, 2018; Malik and Hussain, 2018). We select reviews from two popular categories, namely 'Kindle' and 'Electronics' as our two Amazon datasets. Specifically, recall that we refer to the classification approaches we introduced in Section 6.3 as the basic classifiers. For these three datasets, following (Krishnamoorthy, 2015), we set the review helpfulness threshold to 1, i.e. we consider reviews to belong to the positive class if they have one or more helpful votes, otherwise, the reviews are regarded as unlabelled. Table 6.2 provides statistics for the used three datasets. We conduct a 5-fold cross-validation on the three used datasets. It is apparent from the datasets summary in Table 6.2 that these datasets are imbalanced, having all a smaller number of helpful reviews than that of the unlabelled reviews. Therefore, we balance the class distribution of the training dataset by applying a down-sampling strategy. Down-sampling is a well-known solution to address the class distribution imbalance when training binary classifiers (Sun et al., 2009). In particular, as introduced in Section 6.2, we take the age of reviews into account during the calculation of the review negativity by leveraging the assumption that an older review has a higher probability to have received helpful votes. To validate our assumption on the used datasets, in Figures 6.1a - 6.1c, we plot the age of reviews (in days) against the probability that reviews of that age have been labelled as helpful. In the same figures, we also plot the number of reviews of different ages (in red).

From Figures 6.1a - 6.1c, we observe that the Yelp, Kindle and Electronics datasets share similar helpful vote distributions across review ages. Indeed, on all three datasets, these plots appear to corroborate our assumption that there is a correlation between the number of received helpful votes by a review and the number of days the review has been posted, since the older

[2] https://www.yelp.com/dataset/challenge



(a) Yelp (b) Kindle (c) Electronics

Figure 6.1: The probability of obtaining helpful votes for reviews with different number of days (age) since a review was posted (blue lines) and the number of posted reviews of different ages (red dots), for the Yelp, Kindle and Electronics datasets.

reviews have a higher probability of receiving helpful votes than the younger reviews. In particular, we calculate the Pearson and Spearman correlations between the age of reviews (in days) and the helpful vote probability (see the top of Figure 6.1). According to the value of correlation scores, all three datasets exhibit positive Pearson and Spearman correlation scores.[3] This statistically validates our assumption that older reviews have a higher probability of receiving helpful votes. Note that for the Kindle dataset, reviews that are more than 2000 days old are rare (see the corresponding frequency plot using red dots), explaining the high variance in the corresponding helpful vote probabilities. In particular, we observe some sharp decreases in the probability of receiving helpful votes in the three figures (i.e. Figures 6.1a - 6.1c). Such probability changes are due to the missing reviews that have been posted for corresponding days (e.g. mere reviews have been posted for more than 5000 days in the Yelp dataset). As we can observe in Figures 6.1a and 6.1c, the probability of receiving helpful votes for reviews in the Yelp and Electronics datasets decreased to 0 due to the missing review samples that aged more than around 5000 days. Moreover, recall that the Kindle dataset has few review instances that are more than 2000 days old. Therefore, in Figure 6.1b, we can observe many share decreases in the probability of receiving helpful votes due to such missing reviews.

Armed with these examined correlations, the underlying assumption of NCWS is thus as follows: the longer a review has been posted and that remains unlabelled, the more likely that the review will be unhelpful. Indeed, new reviews without votes could be helpful, but have not had sufficient opportunity to be presented to users; on the other hand, older reviews have not received helpful votes despite being presented to users. Therefore, to evaluate the effectiveness of NCWS and its underlying assumption, we apply NCWS to two generic types of classifiers, namely the SVM models with the features introduced in Section 6.3 and the Neural Network

[3] All correlations are statistically significant according to the scipy implementations.

(NN)-based classifiers, namely CNN and BERT-based classifiers (introduced in Section 6.3.2). As mentioned in Section 6.3, we also use the age of reviews as an additional feature for the SVM classifier (denoted SVM-Age). Because of the observed correlations between the helpfulness and the age of reviews and the reliance of NCWS on the age property, such a basic baseline allows to evaluate if any improvement is the result of the use of the age feature itself, or whether it is due to NCWS itself. We also compare the resulting classification performances to the same classifiers but corrected using the competing methods from the literature, namely SVM-P, C-PU, and P-conf. We describe these classifiers and their corrected versions in the next section.

6.4.3 Classifiers

We use the three following so-called basic classifiers:

1. **SVM:** We implement the SVM model, which has been used in our previous sentiment classification task (see Section 5.2.2), with the LIBSVM (Chang and Lin, 2011) library. Moreover, we use the default setting for the parameter values with a penalty parameter C = 1.0 and the RBF kernel. We instantiate different SVM classifiers based on the features sets listed in Table 6.1.

2. **CNN:** Similar to the sentiment classification approaches in Section 5.2.2, we also consider CNN-based classifiers. However, as argued by (Wang et al., 2018b), a deeper CNN model is more effective with its multi-level abstraction of the semantic features from text. Therefore, in this chapter, we introduce a deeper CNN-based classifier, which consists of three vertically concatenated convolutional layers. Each layer has different convolutional filter sizes (namely $3 \times m$, $4 \times m$ and $5 \times m$ respectively, where m is the embedding size of a word). The output of the third convolutional layer is then fed into a linear prediction layer to predict the review helpfulness label. Moreover, we use the public pre-trained word2vec vectors from Glove (Pennington et al., 2014) with $m = 100$. In particular, we adopt the cross-entropy loss to train the CNN model.

3. **BERT:** In this chapter, we also consider a BERT-based classifier (Devlin et al., 2019). Recall the description of the BERT-based classification techniques from Section 6.3.2. They have been shown be top-performing in different text classification tasks (Lee and Hsiang, 2019; González-Carvajal and Garrido-Merchán, 2020). We implement a BERT-based classifier with a popular natural language processing architecture (from HuggingFace's Transformer (Wolf et al., 2019) library), which enables a quick implementation

of the BERT transformer to process text. In particular, we adopt the pre-trained BERT model (i.e. 'bert-base-uncased'). Next, following the setup of the CNN-based classifier, we again use a linear prediction layer to make the review helpfulness predictions and train the model by using the classic cross-entropy loss function. Both CNN and BERT-based classifiers are trained using batch size 100 for 10 epochs, using the Adam optimiser with a learning rate of 10^{-4}.

Classifier Correction Baselines: Our experiments also apply three existing correction approaches from the literature (namely SVM-P, C-PU and P-conf) as baselines to correct the basic classifiers:

1. **SVM-P:** The SVM-P approach applies a larger penalty value to the positive class than to the negative class according to the ratio of the number of positive examples versus the negative examples as suggested by Tang et al. (2009). We use LIBSVM (Chang and Lin, 2011) to apply the penalty values to different classes in the loss function. However, the experimental setup of SVM-P is different from other classification approaches in its training process. Indeed, the penalty values for different classes correspond to the class ratios of an unbalanced dataset. Therefore, we calculate the class ratio of the training dataset for SVM-P before down-sampling. Note that, the SVM-P approach is limited to the SVM-based classifiers.

2. **C-PU:** This approach was proposed by Du Plessis et al. (2015). C-PU has a similar methodology to our NCWS approach, applying different loss functions for the positive and unlabelled examples. However, unlike our approach, to adjust the basic classifiers, C-PU does not consider the negative confidence of the unlabelled instances. For SVM, following Du Plessis et al. (2015), we use the double hinge loss, $\ell_{dh}(z) = max(-z, max(0, \frac{1}{2} - \frac{1}{2}z))$, when applying C-PU. For the CNN and BERT-based classifiers, we directly integrate C-PU into the cross-entropy loss function.

3. **P-conf** (Ishida et al., 2018): P-conf learns a classifier only from the positively-labelled instances and leverages the probability of these instances to be positive. We apply the objective function of P-conf (Ishida et al., 2018) to all basic classifiers.

6.4.4 Evaluation Metrics

In this chapter, we aim to detect helpful reviews with weakly supervised binary classifiers in the review helpfulness classification task. Therefore, different from the use of the accuracy metric

in the sentiment classification (see Section 5.2.3.1), we use the F1 metric as the key metric to evaluate the performances of the classifiers in accurately classifying the reviews. Precision and recall are also reported to further examine the classification accuracy and the models' ability to identify positive examples in the corpus. Note that we focus on using the F1 metric, since, in this study, we aim to address the correction of review helpfulness classifiers by correctly identifying more helpful reviews. In particular, recall the discussion in Section 6.2.1 and our aim to improve the existing approaches by accurately identifying more helpful reviews, including both labelled and unlabelled ones. This objective means that we are not solely relying on the classification accuracy/precision but also on the performance in identifying helpful review instances (i.e. recall). Therefore, the F1 metric, which evaluates both the precision and recall performances of the classification approaches, is considered the main evaluation metric in this study. Meanwhile, we also report both the precision and recall performances of the corresponding classification approaches. The scores of such two metrics evaluate the models' effectiveness from two other perspectives allowing for a comprehensive classification performance analysis of the models.

It is of note that a number of studies have proposed approaches for the evaluation of PU learning. Claesen et al. (2015) proposed to use a ranking-based evaluation approach and set the threshold value to divide the positive and negative examples. Jain et al. (2017) proposed to evaluate the performance of PU learning-based classifiers with the aid of the class prior knowledge of the class distribution in the unlabelled dataset. However, the evaluation approaches of (Claesen et al., 2015; Jain et al., 2017) require the estimation of information such as the class threshold and the class prior, which causes a systematic estimation bias in the evaluation process (Du Plessis et al., 2015). Therefore, we resort to using the classical evaluation metrics we introduced above and rely only on the ground truth of the positive examples that we have.

6.5 Results Analysis

In this section, we present and analyse the results of our experiments and answer the first two research questions in Section 6.4.1. These research questions focus on identifying the review helpfulness classifiers with the best performances among the basic classifiers. Then, we aim to examine if we can better improve the classification performances of such classifiers with our proposed NCWS approach than leveraging other classification correction approaches, namely SVM-P, C-PU and P-conf.

6.5.1 RQ 6.1: Review Helpfulness Evaluation

To address RQ 6.1 and identify the best performing basic classifiers, we compare the effectiveness of various basic classifiers in distinguishing between helpful and unhelpful reviews using the F1 score. The results over the three used datasets are presented in Table 6.3.

According to the results in Table 6.3, we observe that, among the SVM-based classifiers, the SVM-LEN, SVM-Structural and SVM-ALL classifiers outperform other classification approaches and provide the best classification performances. They obtain the highest F1 scores across the 3 datasets ($>$0.6 on the Yelp and Electronics datasets and $>$0.45 on the Kindle dataset). Meanwhile, SVM-NoS and SVM-Age also obtain good classification performances on the Yelp and Electronics datasets (F1 scores $>$0.5). However, their classification performances decrease on the Kindle dataset ($<$0.35). In particular, by observing the good performances of the classifiers that deploy review length as a feature (i.e. SVM-LEN, SVM-NoS, SVM-Structure and SVM-ALL), we conclude that the length of a review is a useful feature for predicting review helpfulness. Furthermore, the inconsistent performances on three datasets of the SVM-Age classifier indicates that the age feature cannot by itself fully address the review helpfulness classification problem by leveraging the (weak) correlations between the age and the helpfulness of review that was validated in Section 6.4.2. Apart from these discussed classifiers, the remainder of the SVM-based classifiers each obtains high F1 scores on some but not all of the 3 datasets or exhibits bad performances across the 3 datasets. For example, the SVM-Rating classifier obtains good classification results on the Yelp and Electronics datasets but obtains a very low F1 score on the Kindle dataset. On the other hand, the CNN and BERT-based classifiers also provide good results with high F1 scores (>0.48). In particular, the BERT classifier obtains higher F1 scores than the CNN classifier across all 3 datasets. However, the best performing SVM-based approach (i.e. SVM-ALL) still outperforms these two NN-based approaches on all the 3 datasets.

Therefore, for RQ 6.1, we conclude that the basic classifiers that account for the length of a review, including the SVM-LEN, SVM-NoS, SVM-Structural and SVM-ALL classifiers, provide the best overall performances among the SVM-based classifiers. This conclusion highlights the effectiveness of taking review length into account in the review helpfulness classification task. This is in line with their good performances reported in the literature (Lu et al., 2010; Liu et al., 2007; Kim et al., 2006). Moreover, the age feature-based classifier (i.e. SVM-Age), which models our observed correlations between the age and the helpfulness of reviews, does not provide effective classification results on three datasets. This observation shows the necessity of considering additional text features, aside from the reviews' age, to classify the

Table 6.3: Performances of the basic classifiers.

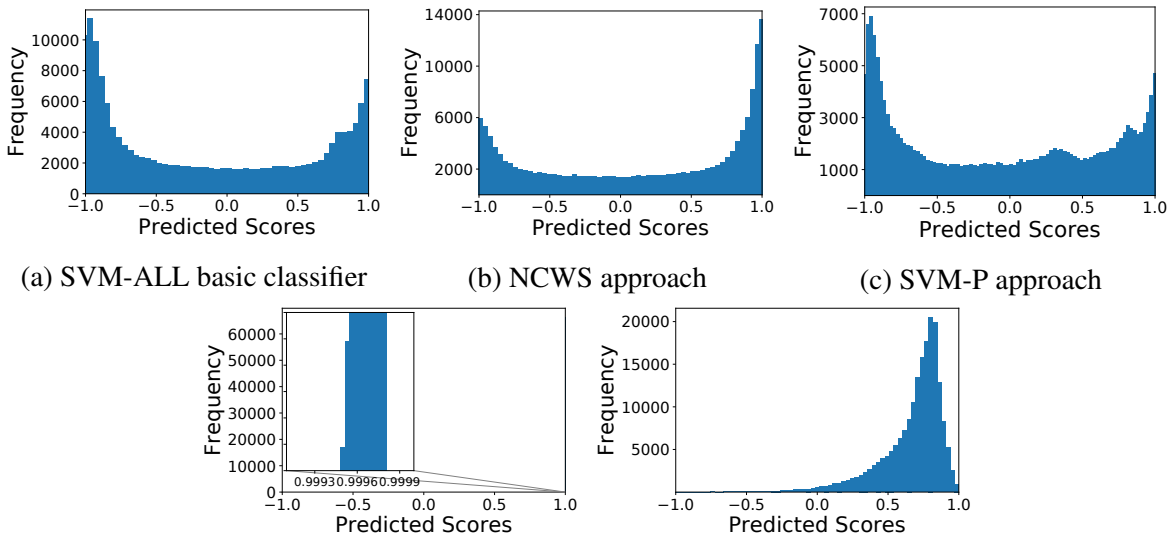
Basic Classifiers	Yelp	Kindle	Electronics
SVM-LEN	0.6069	0.4679	0.6047
SVM-NoS	0.5975	0.3362	0.5759
SVM-ASL	0.4931	0.0432	0.5315
SVM-PoQS	0.2153	0.0132	0.5287
SVM-Structural	0.6017	0.4576	0.6013
SVM-UGR	0.5344	0.0354	0.0925
SVM-Syn	0.0846	0.0086	0.0215
SVM-Rating	0.5439	0.0753	0.5081
SVM-Age	0.5428	0.3037	0.5451
SVM-ALL	**0.6340**	**0.5877**	**0.6336**
CNN	0.5018	0.4830	0.5103
BERT	0.6119	0.5618	0.5712

reviews' helpfulness. Furthermore, the NN-based approaches (i.e. the CNN and BERT-based classifiers) lead also to competitive review helpfulness classifiers (see Table 6.3). The good performances of the CNN-based classifier agree with its effectiveness in classifying the reviews' sentiment in Chapter 5. These results indicate the usefulness of applying a CNN-based classifier as well as the better performing BERT-based classifier in addressing the text classification tasks (e.g. the sentiment classification in Chapter 5 and the helpfulness classification in this Chapter). As a consequence, we select these aforementioned SVM-based (i.e. SVM-LEN, SVM-NoS, SVM-Structure, SVM-Age and SVM-ALL) and CNN-based and BERT-based classifiers as representatives of reasonable review helpfulness classification approaches and for evaluating the effectiveness of our proposed NCWS approach.

6.5.2 RQ 6.2: Classification Correction Evaluation

For RQ 6.2, we examine the effectiveness of our NCWS classification correction approach along with the best performing basic classifiers examined in the previous section. In particular, we compare NCWS to other existing classification correction approaches (namely SVM-P, C-PU and P-conf) from the literature, on three datasets, namely, the Yelp, Kindle and Electronics datasets. As mentioned in Section 6.1, we aim to address the problem of binary classification with incomplete positive instances and abundant unlabelled instances. Following the general weak supervision paradigm, the main objective of our NCWS approach is to help classifiers model the unlabelled instances by identifying further positive instances[4] from the many unla-

[4]These instances would have otherwise been assumed to be negative.



(a) SVM-ALL basic classifier (b) NCWS approach (c) SVM-P approach

(d) C-PU approach (e) P-conf approach

Figure 6.2: Frequency distribution of review helpfulness score predictions for the SVM-ALL basic classifier and the corresponding classification correction approaches applied to the SVM-ALL basic classifier.

belled instances, thereby improving the classifiers' performances.

First, we compare the differences between the classification results of the basic classifiers and the results after applying the corresponding classification correction approaches to the basic classifiers. Figures 6.2 plots the frequency distributions of the predicted scores of review instances (in the range -1 to 1), to illustrate the alteration of the classification results when NCWS and the classification correction baseline approaches are deployed. In particular, we show the classification results of the SVM-ALL classifier, the best performing classifier, as a representative example of the effects of applying NCWS and the various classification correction methods on the Kindle dataset. However, the observed trends remain consistent across all basic classifiers for the other two Yelp and Electronics datasets. In particular, Figure 6.2a shows the review helpfulness score predictions given by the basic SVM-ALL classifier, while Figure 6.2b shows the obtained predictions after correcting the classifier using our NCWS approach. It is clear from the figures that when NCWS is applied, a further number of reviews are classified as positive instances by the corrected SVM-ALL classifier. These results are in line with the objective of our NCWS approach to identify more positive examples from the unlabelled instances. Figures 6.2c, 6.2d and 6.2e show that after applying the classification correction baseline approaches (i.e. SVM-P, C-PU and P-conf) to the SVM-ALL classifier, the helpfulness score prediction distributions of the reviews change in different ways: while SVM-P

Table 6.4: Results of the classification correction approaches on the Yelp, Kindle and Electronics datasets. Statistically significant differences, according to the McNemar's test ($p < 0.05$), to the corresponding basic classifier are indicated by *.

		Yelp				Kindle				Electronics			
		Precision	**Recall**	**F1**		**Precision**	**Recall**	**F1**		**Precision**	**Recall**	**F1**	
SVM-LEN	basic	**0.5781**	0.6421	0.6069		0.5369	0.4243	0.4679		**0.5203**	0.7218	0.6047	
	SVM-P	0.5661	0.6791	0.6169		**0.5405**	0.3742	0.4288	*	0.5125	0.7373	0.6047	
	NCWS	0.5503	**0.7258**	**0.6256**	*	0.5294	**0.4890**	**0.5083**	*	0.5113	**0.7421**	**0.6054**	*
SVM-NoS	basic	**0.5597**	0.6437	0.5975		0.5370	0.2603	0.3362		**0.5268**	0.6351	0.5759	
	SVM-P	0.5418	0.6930	0.6081		**0.5377**	0.2421	0.3139		**0.5268**	0.6351	0.5759	
	NCWS	0.5315	**0.7330**	**0.6151**	*	0.5345	**0.2799**	**0.3443**	*	0.4777	**0.7468**	**0.5827**	*
SVM-Structural	basic	**0.5819**	0.6309	0.6017		0.5364	0.4103	0.4576		0.5386	0.6805	0.6013	
	SVM-P	0.5790	0.6378	0.6036		**0.5392**	0.3790	0.4343		**0.5462**	0.6621	0.5986	
	NCWS	0.5501	**0.7263**	**0.6252**	*	0.5267	**0.4883**	**0.5011**	*	0.5169	**0.7295**	**0.6051**	*
SVM-Age	basic	0.5569	0.5293	0.5428		0.6457	0.1986	0.3037		0.6012	0.4987	0.5451	
	SVM-P	**0.5925**	0.3990	0.4769		**0.7338**	0.0840	0.1508		**0.6598**	0.3336	0.4431	
	NCWS	0.5159	**0.7098**	**0.5975**	*	0.5910	**0.2526**	**0.3539**	*	0.5939	**0.5243**	**0.5569**	*
SVM-ALL	basic	**0.6023**	0.6253	0.6136		**0.5254**	0.6023	0.5612		**0.5975**	0.6619	0.6280	
	SVM-P	0.5651	0.6932	0.6226		0.5148	0.6237	0.5641		0.5472	0.6974	0.6132	
	NCWS	0.5751	**0.7063**	**0.6340**	*	0.4988	**0.7152**	**0.5877**	*	0.5730	**0.7086**	**0.6336**	*
CNN	basic	**0.5416**	0.4674	0.5018		**0.4969**	0.4699	0.4830		**0.4291**	0.6294	0.5103	
	SVM-P	-	-	-		-	-	-		-	-	-	
	NCWS	0.5291	**0.5254**	**0.5272**	*	0.4718	**0.5362**	**0.5019**	*	0.4082	**0.7624**	**0.5317**	*
BERT	basic	**0.5248**	0.7338	0.6119		**0.4783**	0.6807	0.5618		**0.5161**	0.6395	0.5712	
	SVM-P	-	-	-		-	-	-		-	-	-	
	NCWS	0.5034	**0.7927**	**0.6157**	*	0.4543	**0.7653**	**0.5701**	*	0.4923	**0.7127**	**0.5823**	*

adjusts the classification results of the basic SVM-ALL classifier by identifying further positive review instances, C-PU and P-conf completely change the frequency distribution of the original SVM-ALL classifier's score predictions. In particular, they classify most review instances as positive with C-PU squeezing most of the classification results into an extremely narrow range (i.e. between 0.9994 and 0.9998).

As a consequence, only NCWS and SVM-P are useful at correcting and enhancing the performance of the SVM-ALL classifier. Hence, in the following, we focus our experiments on the best performing basic classifiers – as identified in the conclusions of RQ 6.1 – and the corresponding NCWS and SVM-P corrected results, on the three datasets, to further examine their overall effectiveness.

Table 6.4 examines the effectiveness of SVM-P and NCWS after applying them to the SVM and NN-based classifiers on the three datasets. For the SVM-P approach, the F1 scores show that SVM-P can be helpful to all the basic classifiers – except SVM-Age – and enhances their classification performance with higher F1 scores. However, such effectiveness is not generalisable to all SVM-based approaches on the three datasets. For example, SVM-P negatively impacts the SVM-LEN classifier on the Kindle dataset with a lower F1 score (0.4679 $\rightarrow$ 0.4288). Meanwhile, our NCWS approach outperforms SVM-P by exhibiting higher F1 scores. In particular,

Table 6.5: The impact of NCWS on classification: number of unlabelled reviews that changed from being predicted negative to predicted positive by the application of NCWS; total number of reviews predicted negative by the basic classifiers; the percentage of reviews that changed is also shown.

	Yelp		Kindle		Electronics	
SVM-LEN	7559 / 90260	(8.3%)	10030 / 92138	(10.8%)	4587 / 126187	(3.6%)
SVM-NoS	15714 / 97301	(16.1%)	2563 / 104016	(2.4%)	31255 / 138253	(22.6%)
SVM-Structural	20534 / 95746	(21.4%)	6888 / 93687	(7.3%)	12562 / 136656	(9.1%)
SVM-Age	52797 / 160458	(32.9%)	9781 / 171451	(5.7%)	6778 / 232405	(2.9%)
SVM-ALL	23603 / 149574	(15.78%)	20235 / 101828	(19.8%)	21413 / 201927	(10.6%)
CNN	25204 / 171307	(14.7%)	35012 / 119640	(29.3%)	50397 / 152453	(33.1%)
BERT	12503 / 78451	(15.9%)	9217 / 68815	(13.3%)	10512 / 165227	(6.3%)

it can significantly and consistently enhance the basic SVM-based classifiers to yield higher F1 scores according to the McNemar's test ($p < 0.05$).

On the other hand, apart from these SVM-based classifiers, we further examine the effectiveness of NCWS[5] on the NN-based classifiers, which have been shown to be effective in many neural language processing tasks. Table 6.4 shows that significant and consistent improvements are observed after applying NCWS to the CNN and BERT-based NN classifiers on the 3 datasets. This demonstrates the effectiveness of NCWS when applied to the cross-entropy loss function. Moreover, when applied to the best performing approaches (i.e. SVM-LEN, SVM-Structural, SVM-ALL and BERT), NCWS can further increase the F1 scores of these classifiers. For example, NCWS improves SVM-ALL to obtain the overall best classification performance on the 3 datasets.

Table 6.5 shows how many unlabelled reviews were classified by the basic classifiers as negative but classified as positive by our NCWS approach. For example, in the Yelp dataset, the SVM-LEN basic classifier predicts 90,260 unlabelled examples as unhelpful while NCWS predicts 7,559 instances of these reviews as being helpful (i.e. an $\sim$8.3% increase). In general, about 5-30% of the unlabelled reviews are re-labelled as positive and identified as helpful by NCWS. These results align with our objective, discussed in Section 6.2.1, to identify more positive instances from the unlabelled corpus. Therefore, by analysing the results from the three datasets, we can now answer RQ 6.2: NCWS can successfully improve the performances of the basic classifiers while outperforming other classification correction approaches on the F1 evaluation metric. These results validate our assumption that the age of reviews is a reliable signal to infer which of the unlabelled reviews are actually unhelpful and that the longer an unlabelled review has been posted, the more likely this review is unhelpful. Moreover, since we are in-

[5]The SVM-P method is limited to SVM-based classifiers as introduced in Section 6.4.3.

vestigating review-based recommendation approaches in this thesis, to evaluate the helpfulness of the helpful-classified but unlabelled reviews, we aim to feed the classified helpful reviews to an existing review-based recommendation technique (i.e. DeepCoNN (Zheng et al., 2017)), to evaluate the effectiveness the classified helpful reviews. Note that this evaluation strategy does not directly answer the helpfulness of the positive-classified reviews for users. Therefore, a future corresponding crowdsourcing user study could be conducted to address such a concern. Then, in the next section, we show how the performance of a state-of-the-art review-based recommendation system can benefit from the predicted helpfulness of reviews and a filtering-based review property encoding approach.

6.6 Integration of the Helpfulness Property

This section aims to evaluate the effectiveness of the predicted reviews' helpfulness when applied to address the rating prediction task. We also aim to leverage the corresponding results to demonstrate the benefit of NCWS. This empirical study addresses RQ 6.3 stated in Section 6.4.1, by focusing on examining the usefulness of the identified helpful reviews in enhancing the review-based recommendation performance. Similar to the study conducted in Section 5.3, we apply NCWS in the context of a representative state-of-the-art review-based rating prediction model, namely DeepCoNN (Zheng et al., 2017), which has also been described in Section 3.1.

To integrate NCWS and encode the helpfulness property of reviews into DeepCoNN, we follow (Li et al., 2021a) and use a filtering-based property integration approach. Such filtering approach simply replaces the user and item review corpora with only those reviews that are predicted to be helpful. In doing so, we postulate that removing noisy or unlabelled reviews and replacing them with further positive reviews as identified by NCWS results into a more effective learned DeepCoNN model. We validate this through experiments on the Yelp dataset, which is a widely used recommendation dataset (Wang et al., 2019d) and has also been used in our previous studies (c.f. Section 5.2 and 5.3). Note that in this thesis, we aim to address the ranking-based recommendation task (see the problem statement in Section 4.2). Therefore, in the later chapter (i.e. Chapter 7), we will further evaluate the effectiveness of using the estimated reviews' helpfulness when applied in ranking-based and review-based recommendation models.

6.6.1 Experimental Setup

We compare different replacement strategies to assess the effectiveness of the corresponding review selection approaches as input for DeepCoNN. Such review selection approaches include:

- **(1) +Random:** randomly samples the same numbers of reviews as the predicted helpful reviews with the best corrected classifier (i.e. SVM-ALL) as input for DeepCoNN;
- **(2) +Basic:** selects helpful reviews with the basic SVM-ALL classifier, which had the best performance in review helpfulness classification in Section 6.5.1;
- **(3) +NCWS:** uses the predicted helpful reviews with a NCWS-corrected SVM-ALL classifier.
- Similarly, as additional comparative approaches, we use the corresponding (4) **+SVM-P**, (5) **+C-PU** and (6) **+P-conf** but with the SVM-P, C-PU and P-conf's predicted helpful reviews instead.

We apply these different review-selection strategies within the DeepCoNN recommendation model on the Yelp dataset, along with two baselines: DeepCoNN and a classic rating prediction approach, namely Matrix Factorisation (MF) (Sra and Dhillon, 2006), which was described in Section 2.3.2 and considers ratings, but not the text of the reviews.

Our experiments are conducted using a 5-fold cross validation, following as closely as possible the experimental setup of (Zheng et al., 2017) (with the same trained word embedding model, but with a different dataset and review selection strategies) to reproduce the original DeepCoNN model. We evaluate the rating prediction accuracy by using Mean Average Error (MAE) and Root Mean Square Error (RMSE), which were introduced in Section 2.2.1. For both metrics, smaller values are better. We use the paired t-test to determine significant differences of MAE.[6]

6.6.2 Results

Table 6.6 presents the MAE and RMSE scores of the DeepCoNN variants. In particular, the first group of rows are baselines, while the second group corresponds to approaches that make use of review helpfulness classification when filtering the set of reviews to use. On analysing the table, comparing the rating prediction error of DeepCoNN and MF using the MAE and RMSE metrics, we observe that DeepCoNN, which uses all reviews for item recommendation enables better representations of user preferences and item properties with lower rating prediction errors than MF. Indeed, recall that MF is only trained on ratings, while DeepCoNN has access to the text of the reviews. In relation to the helpful reviews, we observe that by filtering the reviews to include only those that are predicted to be helpful (c.f. DeepCoNN+Basic, +SVM-P and +NCWS), the rating prediction is improved (i.e. significantly reduced MAE and RMSE scores) compared to

[6]RMSE, which is a non-linear aggregation of squared absolute errors, is not suitable for significance testing.

Table 6.6: Recommendation results when using the predicted helpful reviews: Significant MAE improvements (t-test, $p < 0.05$) w.r.t. DeepCoNN, DeepCoNN+Basic & DeepCoNN+SVM-P are denoted by $\circ$, $\bullet$ & $*$, resp..

	MAE	RMSE
MF	1.1526	1.4345
DeepCoNN	0.8969	1.1798
+Random	0.9201 $\circ*$	1.2278
+Basic	0.8629 $\circ$	1.1012
+C-PU	0.8969 $\bullet*$	1.1798
+P-conf	0.8969 $\bullet*$	1.1798
+SVM-P	0.8597 $\circ$	1.0998
+NCWS	**0.8503** $\circ\bullet*$	**1.0954**

DeepCoNN. This implies that, among all the reviews considered by the DeepCoNN baseline, some are noisy, and removing these to focus upon the likely helpful reviews aids learning a more effective recommendation model. On the other hand, the +C-PU and +P-conf integrations have the same performances as DeepCoNN – indeed, this is expected from the results of Section 6.5.2, where P-Conf and C-PU were not effective in identifying helpful reviews.

Moreover, comparing the performances between the +Basic, +SVM-P and +NCWS integrations, we find that our proposed NCWS approach results in better rating predictions, exhibiting a significant 1.8% improvement in MAE and 0.27% improvement in RMSE, respectively, in comparison to the +Basic approach that uses the basic SVM-ALL classifier. In particular, +NCWS also shows a significantly better performance than +SVM-P, which indicates the benefit of using NCWS over SVM-P in identifying helpful reviews as input for the DeepCoNN model. Moreover, +NCWS is significantly more accurate than +Random, a variant that uses the same number of randomly sampled reviews. Hence, and in answer to RQ 6.3, we find that focusing on the likely helpful reviews, particularly those additional reviews found using our proposed NCWS classifier (see Table 6.5), allows the performance of the state-of-the-art DeepCoNN rating prediction approach to be significantly enhanced. This also validates the effectiveness of NCWS in identifying the reviews' helpfulness as well as the filtering-based review property encoding strategy. In particular, we show that the helpfulness of reviews estimated by an effective helpfulness classifier can significantly improve the performance of a recommendation approach. This observation validates our proposed claim in the thesis statement from Section 1.3: by inferring the helpfulness of properties of reviews, the recommendation approaches can obtain a better performance in estimating the users' preferences.

6.7 Conclusions

In the thesis statement (stated in Section 1.3), we postulated the usefulness of accurately inferring the helpfulness properties of reviews to a review-based recommender. Therefore, to accurately infer the reviews' helpfulness property, we proposed a novel weak supervised binary classification correction approach (NCWS). In particular, NCWS considers the confidence probabilities of the unlabelled examples as being negative. Using three datasets, we first examined the effectiveness of many basic review helpfulness classification approaches. In Table 6.3, we showed that the length feature-based SVM classifiers (i.e. SVM-LEN, SVM-NoS, SVM-Structure and SVM-ALL) and neural network-based approaches (i.e. the CNN and BERT-based classifiers) gave the best helpfulness classification performances. Next, we showed the effectiveness of our NCWS approach compared to several existing state-of-the-art classification correction approaches from the literature (see Table 6.4). In Table 6.5, we also illustrated how NCWS allows to increase the number of positive instances by 5–30% when integrated into various binary classifiers. These experimental results showed that by leveraging our proposed NCWS approach, we can accurately classify the reviews into helpful and unhelpful. Therefore, we could use the estimated reviews helpfulness with NCWS in our proposed review-based recommendation framework to address the availability of review attributes (see the challenges to address in Section 4.6). In particular, we further examined the effectiveness of the estimated reviews' helpfulness by our proposed NCWS approach when applied to a review-based recommendation model. In Table 6.6, we showed that by leveraging the predicted helpful reviews (i.e. the filtering-based property encoding strategy), we can significantly enhance the performance of a representative review-based recommendation model (namely DeepCoNN). These results indicated that the extract reviews' helpfulness could benefit the performance of review-based recommendation techniques.

Therefore, in summary, in this chapter, we have validated another part of our proposed thesis statement in Section 1.3, namely by inferring the reviews' helpfulness property, a review-based recommendation system can indeed better capture the users' preferences and obtain improved recommendation performances. In summary, in Chapters 5 and 6, we have addressed the extraction of the reviews' sentiment and helpfulness properties and showed their effectiveness in improving the accuracy of a representative review-based recommendation model (i.e. DeepCoNN). Then, with the improved availability of the review properties, starting from the next chapter, we focus on leveraging various inferred review properties from the reviews to address the user preference estimation. In particular, we aim to propose end-to-end review-based recommendation approaches to instantiate our proposed review-based recommendation framework, which takes the extracted review properties into account.

Chapter 7

Integrating Review Properties using an Attention Mechanism

7.1 Introduction

In Chapters 5 and 6, we improved the availability of the reviews' properties by accurately estimating the sentiment and helpfulness properties of reviews from the reviews' text via effective classifiers. In particular, the analysis of the experimental results in Chapters 5 and 6 showed that by identifying the reviews with strong sentiment or by only considering the helpful reviews deemed useful to users, we could significantly improve the recommendation accuracy of a review-based recommendation model. Next, recall the discussed challenges of implementing our proposed review-based recommendation framework in Section 4.6. We also need to address the estimation of the users' preferences for the review properties, so as to identify personalised useful reviews (see the users' preferences estimation discussed in Section 4.6.2). In Section 3.1, we have also explored many review-based recommendation approaches that aimed to incorporate the usefulness of reviews in making recommendations (Chen et al., 2018b; Bauman et al., 2017). Some models have used an attention mechanism to model the usefulness of reviews (Guan et al., 2019) or those portions of the textual content of reviews that contribute most to the recommendation performances (Chen et al., 2019). However, as we argued in Section 3.1, a limitation of such approaches is that they capture the usefulness of reviews by relying on historical data, which often do not generalise to reviews that are unseen by the trained model (Veličković et al., 2018). This limitation is commonly discussed as the out-of-distribution problem in the literature. Some unseen reviews could be written in different forms, styles, or languages compared to the learned historical reviews, making a different distribution of review

data and making it difficult for the attention-based model to estimate the exact usefulness of the reviews correctly.

To address the limited generalisability of using the attention mechanism, we propose to consider the *review properties* to model the usefulness of reviews. Indeed, by integrating such review properties into the review modelling process, a review-based recommendation model could also encapsulate the usefulness of reviews when capturing the users' preferences and items' attributes. Moreover, we have discussed in Sections 3.2, 3.3 and 3.4, how various review properties have been used as side/contextual information to enrich the user-item interactions when addressing recommendation tasks. However, it remains unclear how different review properties contribute to estimating the usefulness of their corresponding reviews for effective recommendation. As a consequence, in this thesis, we address this limitation by considering various review properties and examining their actual effectiveness in capturing the reviews' usefulness in addressing the recommendation task. According to the users' adoption of information framework, introduced in Chapter 1.1, each user follows a particular scheme or strategy in using different properties or aspects of the review information and hence each user makes different interaction decisions. Therefore, we argued that a model can better capture the users' preferences by learning how users use the reviews and by examining their preferences on different properties of the reviews (see Section 1.3).

In this chapter, we focus on addressing the personalised selection of reviews' properties by modeling the importance of different review properties in capturing the usefulness of reviews and then learning the users' preferences and the items' attributes. We propose a novel review property-based neural network model (RPRM), which instantiates our proposed review-based recommendation framework in Section 4.5, to effectively address the recommendation task. RPRM investigates the usage of review properties to model the usefulness of reviews and aims to enhance the ability of the recommendation model in capturing the usefulness of reviews and the users' adoption of information. In particular, RPRM uses six review properties, covering the four main types of review properties, introduced in Section 4.4, in recommendation scenarios to encode the usefulness of reviews. In particular, we consider the age of the reviews, the length of the reviews, the reviews' associated ratings, the number of helpful votes associated to the reviews, the probability of the reviews being helpful and the sentiment expressed in the reviews. Note that, in Chapters 5 and 6, we validated the usefulness of two such properties (i.e. the probability of the reviews being helpful and the sentiment expressed in the reviews) in improving the performances of review-based rating prediction models. However, different review properties can have various importance levels in capturing the usefulness of reviews for different users and items. Moreover, according to the aforementioned users' adoption of information framework

(Sussman and Siegal, 2003), users' tend to leverage distinct properties of reviews to consider the usefulness of different reviews, so as to support their choices. This suggests that users tend to prefer items whose associated useful reviews capture the same important properties as those the users prefer. Therefore, we also propose two loss functions and a negative sampling strategy that aim to reward the situation where a user and the interacted items agree on the most important properties while penalising the situation where the user disagrees with the negative sampled items on the most important properties. For example, the agreement will be high if a user considers longer reviews to be more useful and a given item's reviews usefulness is better captured by long reviews.

The remainder of this chapter is organised as follows: In Section 7.2, we recall the statement of the ranking-based recommendation task we are addressing in this thesis and its associated notations. Then, we describe our proposed review-based recommendation model (i.e. RPRM). RPRM is designed to address the ranking-based recommendation task by leveraging the reviews' associated properties. Next, we describe the experimental setup for our proposed RPRM model in Section 7.3. The experimental setup comprises the description of the research questions, the used datasets, the evaluation metrics, and the details of training our proposed RPRM and the baseline approaches. In Section 7.4, we discuss and analyse the obtained experimental results to answer the research questions. Moreover, in Section 7.5, we further analyse the users' preferences for the reviews' properties. In particular, we investigate the relationship between the importance of review properties between the users and their interacted items. Finally, we summarise the conclusions of this chapter in Section 7.6.

7.2 Methodology

We first briefly recall the top-n ranking-based recommendation task we aim to address in this thesis and its used notations (see Section 4.2). Next, we describe our proposed RPRM model, which leverages the reviews posted by users to enhance the recommendation task. In particular, RPRM instantiates our proposed review-based recommendation framework in Chapter 4.

7.2.1 Task Definition

As we introduced in Section 4.2, a ranking-based recommendation task aims to effectively rank items for users according to their preferences. The recommendation task involves connecting two key entity types, namely: the set of users $U = \{u_1, u_2, ..., u_N\}$ with size N and the set of items $I = \{i_1, i_2, ..., i_M\}$ with size M. To address the recommendation task, we aim to accurately

estimate the users' preferences on items so that we rank the items that a given user might find the most interesting in higher ranks. To do so, we investigate the use of the reviews that the users have posted on items, as well as their associated properties. Each user u or item i has an associated set of reviews, i.e. posted by that user, C_u, or posted on that item, C_i.

Furthermore, the reviews of a user or an item can be described using k review properties $\mathbf{P} = \{P_1, P_2, ..., P_k\}$. For example, to capture the preferences of user u for a given property P_1, we estimate the corresponding review property scores for each review in the review set of user u – i.e. $P_{1,u} = \{p_{1,1}, p_{1,2}, ..., p_{1,|C_u|}\}$, where $p_{1,t}$ is the property score of the t^{th} review of user u. For example, the geographical property ('near' vs. 'distant'), the length property ('long' vs. 'short'), or the age of reviews ('old' vs. 'recent') can all be mapped into a scalar in the interval [0, 1]. Indeed, for the length property, a longer review will have its length property score closer to 1 than other shorter reviews. Hence, similar to the methodology we used to encode the sentiment property in Chapter 5, the property scores depict the usefulness of reviews. In other words, the computed property scores enable the modelling of reviews from different perspectives and examine the relationship between the review usefulness and the review properties.

7.2.2 The RPRM Model

To address the top-n ranking-based recommendation task, we propose the novel Review Properties-based Recommendation Model (RPRM), which is a neural recommendation model that takes the reviews and their associated properties into account. In particular, following the review property encoding strategy used in Section 5.3.1.3, we use a dot-product attention mechanism to score and learn which review properties are more useful in describing the usefulness of reviews and thereby are important in making good recommendations. Note that the filtering-based review property encoding strategy that in Section 6.6 is also applicable. However, although the filtering-based strategy is effective according to an performance improvement obtained in Chapter 6, it might also lead to an information loss by ignoring part of the review data (Spicer et al., 2011). Figure 7.1 presents the architecture of our proposed RPRM model. In general, RPRM is a collaborative filtering-based framework, which models the interactions between users and items. It is of note that RPRM models both the users and items using the same neural network architecture (see User and Item Modelling in Figure 7.1). The RPRM architecture is organised into four layers, which we discuss in turn below:

1. The review property encoding layer, which combines the semantic textual representations of reviews obtained using BERT, a top-performing pre-trained language model (Devlin et al., 2019), with the properties of reviews (Section 7.2.2.1);

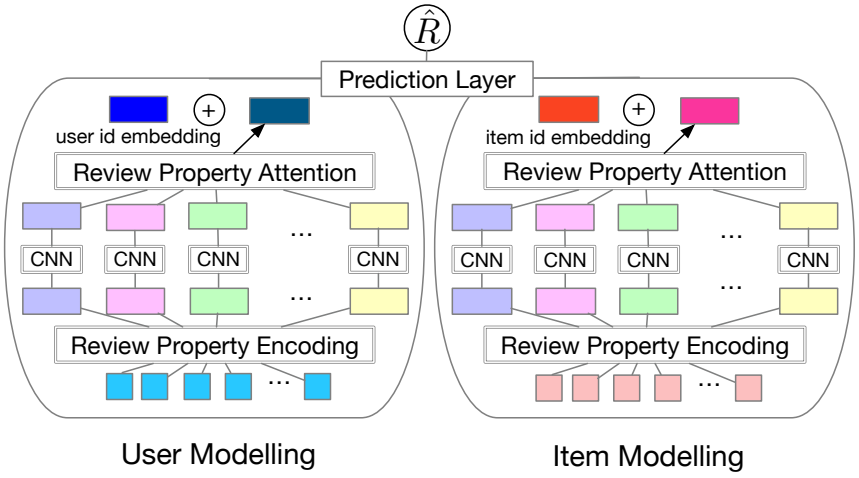


Figure 7.1: The Neural Network Architecture of RPRM.

2. The review embedding processing layer, which creates a low-dimensional representation of each review (Section 7.2.2.2);

3. The review property attention layer (Section 7.2.2.3), which identifies the properties of reviews that are more useful to represent the users' preferences and items' attributes;

4. We use the output from the last layer as input to the prediction layer, along with the identification embedding of a given user and item, to score the user's preferences on items (Section 7.2.2.4).

Later, in Section 7.2.3, we propose and discuss two new loss functions and a new negative sampling strategy to aid learning while encapsulating the properties of reviews.

7.2.2.1 Review Property Encoding Layer

RPRM first models the users' reviews and items' reviews. To process and summarise the semantic information of each review, we convert each review into a 768-sized embedding vector by using the pre-trained BERT model (Devlin et al., 2019). In particular, in Chapter 6, we have also showed the top performance of the BERT-based classifier in capturing the semantic feature of the review text in identifying helpful reviews. Therefore, in this chapter, we leverage the pre-trained BERT model to generate the review text representation. Next, in this layer, similar to the SentiAttn model in Chapter 5, RPRM encodes the embedding vectors of the reviews with various review properties through a dot-product function. The objective of encoding the review

latent vectors with different review properties is to model the usefulness of reviews from different perspectives (e.g. length, age, sentiment). Each review property can be represented by a list of normalised review property scores. These scores allow RPRM to focus on different reviews and encode the knowledge of the corresponding reviews' properties.

For example, by encoding the review length property, the model can capture how reviews with different lengths can have an influence on the recommendation outcome and how the length property of reviews is associated with the user/item representations. The encoding process of a given review property can be described as follows:

$$O_{u,P_1} = [X_1 P_{1,1}, X_2 P_{1,2}, ..., X_{|C_u|} P_{1,|C_u|}] \tag{7.1}$$

Following the notations we used in our proposed SentiAttn model in Section 5.3, $X_{1,...,|C_u|}$ indicates the embedding vectors of the reviews of user u and $|C_u|$ is the size of their review set. In particular, Equation (7.1) encodes the review property P_1 for user u. After encoding k review properties, for user u, we have $O_u = [O_{u,P_1}, ..., O_{u,P_k}]$.

In particular, as mentioned in Section 4.4, in this thesis, we use six commonly available review properties to describe the reviews from different perspectives:

1. **Age:** We calculate the number of days d since a review has been posted. Then, we compute the Age score of a review to be: $p = 1 - d/max(D)$, where $max(D)$ is the age of the oldest review in the collection. In this case, a recent review is considered more useful than an older review.

2. **Length:** The number of words that are included in a review.

3. **Ratings:** The ratings associated with the review (1-5 stars).

4. **Polar_Senti:** The Polar_Senti property indicates the probability of a given review being polarised (strongly positive or negative). We use the CNN-based classifier as in Chapter 5. Note that we showed that the CNN-based classifier is a strongly effective classifier with >95% classification accuracy (see Figure 5.2). We obtain the corresponding probabilities of the positive reviews being actually positive or the negative reviews being negative.[1]

5. **Helpful:** The number of helpful votes given by other users to a particular review.

[1] Recall that a review is positive (negative) if it has a rating ≥ 4 (≤ 2). The polarity of a 3-star review is predicted by the CNN classifier.

6. **Prob_Helpful:** We calculate the probability of a given review being helpful by using the best classification approach among those examined in Section 6.5.2. This classification approach is based on the SVM-ALL classifier, which considered various review text features (e.g. the reviews' length and reviews' age) and was then further enhanced by our proposed NCWS classification correction approach.

In addition, for the Length, Ratings and Helpful review properties, which have their property scores larger than 1, we apply the min-max normalisation to scale the property scores into [0, 1]. Note that, as we discussed in Section 4.4, we use these six review properties as typical review properties, which are commonly available in various datasets.

7.2.2.2 Review Processing Layer

After encoding the embedding vectors of the reviews with their review property scores, we use the convolutional operators, as in other review-based deep neural network approaches (Zheng et al., 2017; Chen et al., 2018b), to model the embedding vector of each review. The convolutional operators consist of m neurons, with the j^{th} neuron modelling the review embedding vector as follows:

$$Z_j = ReLU(V * K_j + b_j) \tag{7.2}$$

where V is the input vector (i.e. the encoded embedding vectors of the reviews), $*$ is the convolution operator with the j^{th} filter and b_j is a bias term. The ReLU activation function, a commonly used activation function (Nair and Hinton, 2010; Ramachandran et al., 2017), is applied to process the generated features.

Next, each neuron j applies a sliding window over the features Z with a max pooling function to then obtain the convolutional output o_j for the corresponding neuron. Therefore, for each review, we concatenate the convolutional output from the neurons and obtain the processed embedding vector for each review as follows:

$$O = [o_1, o_2, ..., o_m] \tag{7.3}$$

7.2.2.3 Review Property Attention Layer

In the review property encoding layer, RPRM converts the user/item modelling latent vectors into a set of latent vectors by considering various review properties. In this attention layer, the main objective is to observe which properties of reviews are more useful to represent the users' preferences and items' attributes. We hypothesise that the dot-product attention mechanism can

enhance the recommendation performance of RPRM. Moreover, each user or item is associated with a review property weighted vector ϕ_u or ϕ_i with size k, where k is the number of used review properties. For a given user u, the review property attention layer is defined as follows:

$$O'_u = \frac{\sum_{t=0}^{k} \phi_{u,t} O_{u,P_t}}{k} \tag{7.4}$$

7.2.2.4 Prediction Layer

In this layer, RPRM concatenates the processed review latent vectors with the identification embedding vector of users and items to make recommendations. The final prediction of the users' preferences on items can be computed as follows:

$$\hat{R}_{u,i} = (O'_u \oplus V_u) \odot (O'_i \oplus V_i) \tag{7.5}$$

where $\oplus$ is the concatenation operation, which combines the review embedding vector O', and the identification embedding vector V. Moreover, $\odot$ denotes the dot product of the resulting latent vectors to calculate the preference score $R_{u,i}$ of user u on item i. As we discussed in Section 3.1, the dot product, or in general the inner-product, has become the top choice in addressing the users' preferences estimation upon the available user/item feature vectors (Shi et al., 2018; Hyun et al., 2018; Cheng et al., 2018b; Chin et al., 2018). Therefore, we use the dot product function in our proposed RPRM model to address the final users' preferences prediction.

7.2.3 Model Learning

The RPRM model addresses a ranking-based recommendation task, i.e. for a given user, it ranks first those items likely to be of interest to the user. In particular, recall from Section 2.3.2 that a common and popular ranking scheme is to first apply the Bayesian Personalised Ranking (BPR) loss function (Rendle et al., 2009) to optimise the model by comparing the prediction scores for users U with the positive items I^+ and the negative items I^-. The positive items are those items the user has interacted with while the negative items are sampled from those items the users did not interact with thus far. In particular, the uniform sampling strategy is commonly used (Xu et al., 2016; Manotumruksa et al., 2017) to generate the negative items from the users' unseen items. We use this learning scheme as a basic setup of our proposed RPRM model.

7.2.3.1 Loss Functions Using the Importances of Review Properties

Aside from building upon the BPR's loss function and uniform sampling for generating negative items, we propose various novel learning schemes to enhance the recommendation effectiveness. According to the users' adoption of information framework (Sussman and Siegal, 2003) that we discussed in Section 3.5, users show distinct information processing behaviour. In particular, there is a relationship between the users' behaviour and the review properties. This suggests that users tend to prefer items whose associated useful reviews capture the same important properties as those the users prefer. Therefore, we propose two loss functions (i.e. $PropLoss_{uu}$ and $PropLoss_{ui}$) that reward the case where a user and the interacted items agree on the most important properties and penalise the case where the user disagrees with the negative sampled items on the most important properties. In particular, based on the users' adoption of information framework, we assume that users would prefer to process information from items that have similar importance scores on the review properties in capturing useful reviews. Leveraging the similarly scored items' properties by the used attention mechanism, users would exhibit a higher probability of interacting with these items than with other unknown items. Therefore, both of our proposed loss functions ensure that there is an agreement on the importance of review properties between the users and their interacted items (i.e. i^+).

However, $PropLoss_{uu}$ amplifies the disagreement on the importance of review properties between the users and the unseen (i^-) sampled items, while $PropLoss_{ui}$ amplifies the disagreement between the interacted items and the unseen (i^-) sampled items of users. These two loss functions are defined as follows:

$$PropLoss_{uu}(u, i^+, i^-) = Sim(\phi_u, \phi_{i^-}) - Sim(\phi_u, \phi_{i^+}) \tag{7.6}$$

$$PropLoss_{ui}(u, i^+, i^-) = Sim(\phi_{i^+}, \phi_{i^-}) - Sim(\phi_u, \phi_{i^+}) \tag{7.7}$$

where $Sim(.)$ is a function that measures the similarity between the weighted vectors of the review properties. Before applying the similarity function, we scale the weighting scores by dividing the scores by the sum of scores in each weighted vector to generate a discrete probability distribution of scores [0,1] over the review properties.

In particular, we use the Cosine similarity (Cos) function and the Kullback–Leibler (KL) divergence measure as the (dis)similarity functions, which have shown good performances in measuring latent vector similarities (Wang et al., 2019c). Note that, since KL is a divergence measure, we use the inverse of KL to compute similarity. Furthermore, we combine the Pro-

pLoss functions with the commonly-used BPR loss function as follows:

$$\mathscr{L} = \alpha \times BPR(u, i^+, i^-) + (1-\alpha) \times PropLoss(u, i^+, i^-) \tag{7.8}$$

where α controls the emphasis on the two loss functions.

7.2.3.2 A Sampling Strategy Using the Importances of Review Properties

To introduce a different technique in leveraging the agreement on the importances of the reviews' properties between the users and items, we also propose a novel negative sampling strategy, called *PropSample*, which models the agreement on the importance of review properties between the users' interacted items and the unseen items. We argue that if the same properties are important to two items (e.g. i_1 and i_2), but a particular user interacts with item i_1 but not with item i_2, then this user shows a clearer preference for item i_1 over item i_2. Therefore, we sample negative items from each user's unseen items by selecting items that have similar review properties to those items the user has already interacted with.

For a given positive item $i^+ \in I$, we again use a similarity function $Sim()$ to calculate the similarity on the paired property weighted vectors $\phi_{i,p}$ between the positive item i^+ and all negative (unseen) items (i.e. I^-). Note that the time complexity of calculating such a similarity between positive and negative items is linear. Indeed, firstly, the time complexity of calculating an individual pair of items is $O(n)$, which is proportionally to the size of the weighted vectors (i.e. the number of used properties in this study). In particular, since we are using six review properties, such time complexity is close to $O(1)$. Therefore, the calculation of the similarity between the positive and all negative items is in linear time complexity and increases proportionally to the number of negative instances. Next, similar to the loss functions, we normalise the similarity scores across all negative items into a probability distribution. This probability distribution gives the likelihood for sampling these items as a negative instance for learning.

7.3 Experimental Setup

In this section, we examine the performances of our proposed model and approaches on two real-world datasets (i.e. the Amazon and Yelp datasets), which we have previously used in Chapters 5 and 6. Moreover, we compare the performance of RPRM with one classical (i.e. BPR-MF, a classic collaborative filtering technique (see Section 2.3.2)) and five existing state-of-the-art recommendation approaches (i.e. DREAM, CASER, DeepCoNN, JRL and NARRE,

which were discussed in Chapter 3). In particular, we evaluate the performances of our proposed loss functions and negative sampling strategy in addressing the following research questions:

- **RQ 7.1:** Does RPRM outperform the recommendation baselines on the two used datasets?
- **RQ 7.2:** Do the proposed loss functions, $PropLoss_{uu}$ and $PropLoss_{ui}$, improve the recommendation performances of RPRM in comparison to the classical BPR loss function?
- **RQ 7.3:** Does the proposed negative sampling strategy, namely *PropSample*, further enhance the recommendation performance of RPRM compared to the uniform sampling strategy?

7.3.1 Datasets & Evaluation Metrics

For answering the aforementioned research questions, we use two real-world datasets, namely the Yelp dataset[2] and the Amazon Product dataset[3] (McAuley et al., 2015; He and McAuley, 2016) to examine the effectiveness of our RPRM model as well as our proposed loss functions and negative sampling strategy. These two datasets have been used in all of our previous experiments (see Chapters 5 and 6). The Yelp dataset focuses on the user reviews on their top popular category (i.e. 'restaurant'). For the Amazon dataset, we follow the experimental setup in Section 5.3.2.1, and include user reviews on products from six categories.[4] The use of various categories of the Amazon dataset allows to capture the users' preferences across different types of items/products.

In our experiments, we remove cold-start users and items from both datasets, as to ensure that each user and item have at least 5 associated reviews. The resulting Yelp dataset has 47k users, 16k items and 551k reviews; the Amazon dataset has 26k users, 16k items and 285k reviews. Then, following (Sachdeva and McAuley, 2020; Chen et al., 2018b) and our previous experimental setup in Section 5.3.2.1, the two datasets are divided into 80% training, 10% validation and 10% test sets in a time-sensitive manner. In particular, we ensure that the same data split ratio applies to the interactions of each user. Next, we measure the recommendation effectiveness by examining if the items interacted with by the users in the test sets are actually chosen for recommendation by the tested models. Hence, as discussed in Section 2.2.2, we compute the Precision and Recall metrics at different standard rank cutoff positions (namely, P@1, P@10,

[2] https://www.yelp.com/dataset

[3] http://jmcauley.ucsd.edu/data/amazon/

[4] 'amazon instant video', 'automotive', 'grocery and gourmet food', 'musical instruments', 'office products' and 'patio lawn and garden'

and R@10) as well as the Mean Average Precision (MAP) and Normalised Discounted Cumulative Gain (NDCG) metrics to examine the effectiveness of the evaluated recommendation approaches. To test statistical significance, we apply a paired t-test, with a significance level of $p < 0.05$, and use the post-hoc Tukey Honest Significant Difference (HSD) (Sakai, 2018) at $p < 0.05$ to account for the multiple comparisons with the t-tests. In the following, we describe the experimental setup of both RPRM and the used baselines.

7.3.2 Baseline Approaches

We use six baselines: one classical baseline and five existing state-of-the-art recommendation approaches:

- **BPR-MF** (Rendle et al., 2009): is a traditional and commonly used recommendation baseline that uses a pairwise ranking loss function (i.e. BPR) to learn the matrix factorised interactions between users and items. We have already described the MF model with the BPR-based loss function in Section 2.1.

Next, we consider two ranking-based recommendation models, which were discussed in Section 3.3. Both these models take the review property into account:

- **DREAM** (Yu et al., 2016): This model encodes the age property of the reviews and models the dynamic representations of the users' preferences with a recurrent neural network (see Section 2.4.3 for the description of the recurrent neural network).

- **CASER** (Tang and Wang, 2018): This model is a recent approach that sequentially models the implicit user historical interactions with convolutional neural networks. It is a state-of-the-art recommendation model (Guo et al., 2020) that encodes the age property of reviews.

Moreover, we consider three review-based recommendation models as baselines:

- **DeepCoNN** (Zheng et al., 2017): This model is a review-based recommendation model that jointly models users and items through a convolutional neural network (the description of the convolutional neural network can be found in Section 2.4.2). The DeepCoNN is a representative review-based recommendation technique and has been discussed in Chapters 5 and 6.

- **JRL** (Zhang et al., 2017): This model is a heterogeneous recommendation model that encodes various types of information resources including product images, review text and user ratings. In particular, we implement the JRL model by only using the review text.

- **NARRE** (Chen et al., 2018b): NARRE models users and items with two parallel neural networks, both of which include a convolutional layer and an attention layer to capture the usefulness of reviews. This model has also been used as a review-based recommendation baseline in Chapter 5.

7.3.3 Model Training

We implement our proposed RPRM model and the NN-based baseline approaches (namely DREAM, CASER, DeepCoNN, JRL and NARRE) using the PyTorch framework (Paszke et al., 2019). For the setup of RPRM, in the review processing layer, as introduced in Section 7.2.2, we use the pre-trained BERT model (Devlin et al., 2019) to convert each review into a 768-sized latent vector. However, since BERT is limited to encoding a maximum of 512 tokens, we limit the maximum review length to using the first 512 tokens. Next, in the review property encoding layer, we use the SVM-ALL classifier that improved by our proposed NCWS classification correction approach in Chapter 6. Such a corrected classifier generates the 'Prob_Helpful' property scores, which estimates the probability of the reviews being helpful. In particular, we follow the method from Chapter 6 in training the NCWS model using reviews from the 'food' and 'nightlife' categories of the Yelp Challenge dataset round 12[5] and on the Kindle reviews from Amazon[6]. We use NCWS from Chapter 6 to predict the 'Prob_Helpful' property scores of the reviews in both the Yelp and Amazon datasets. Similarly, to generate the 'Polar_Senti' review property scores, we use a CNN-based binary sentiment classifier (Kim, 2014), which has been shown to have a strong classification accuracy (>95%) (see Chapter 5). Similarly, we then train it on 50,000 positive and 50,000 negative sentiment reviews that are sampled from the Yelp Challenge dataset round 12 to conduct sentiment classification. We label the polarity of each review according to the user's posted ratings, which we label as positive if the rating ≥ 4, and negative if the rating ≤ 2. This CNN classifier provides each review with its probability of carrying a strong polarised sentiment. Finally, when training our proposed RPRM model, we apply early-stopping and use the Adam optimiser (Kingma and Ba, 2015) with a $5e^{-4}$ and $1e^{-3}$ learning rates for the Yelp and Amazon datasets, respectively. These learning rates are selected after tuning the model on the validation set, varying the learning rates between $1e^{-5}$ and $1e^{-3}$.

To ensure a fair comparison, we also apply early-stopping on all baseline approaches. Moreover, since we use a pre-trained BERT model to convert the reviews into embedding vectors

[5]We use different Yelp dataset rounds, different categories & removed reviews that belong to 'restaurant' from 'food' and 'nightlife', to avoid overlaps between the NCWS and RPRM evaluation settings.

[6]Again, we use a different Amazon review category for training NCWS to avoid any overlap with the RPRM evaluation.

for our proposed RPRM model, we also extend the DeepCoNN and NARRE baselines by using the BERT-encoded review embedding vectors. In particular, for DeepCoNN, we concatenate all reviews given by/to a single user/item and form a user/item review document. Similar to RPRM, for both DeepCoNN and NARRE, we limit the maximum length of the user/item document in the DeepCoNN model, and the maximum tokens of each review in NARRE, to 512 tokens. We then fine tune every baseline model with learning rates in $[1e^{-3}, 1e^{-4}, 1e^{-5}]$ and compare our approaches with the settings that exhibited the best performances on the validation set. We evaluate the various components of our proposed RPRM model incrementally. First, we capture the effectiveness of using the review properties in a review-based recommendation model. Next, we remove the review property encoding layer in RPRM, denoted as 'No-Prop', to examine its effectiveness on our datasets. Next, we also examine the effectiveness of using each single review property from the included properties in Section 7.2.2 (i.e. the reviews' ages, lengths, sentiments, ratings, helpfulness as judged by the users and helpfulness as predicted by a helpfulness classifier). Therefore, we apply each single review property in the review property attention layer of RPRM to evaluate their effectiveness in identifying the usefulness of reviews. We denote the resulting recommendation models with the name of the corresponding review properties (e.g. 'Age' for using the Age property). Next, we examine the effectiveness of our proposed RPRM's learning schemes, namely the two loss functions ($PropLoss_{uu}$ and $PropLoss_{ui}$) and the negative sampling strategy $PropSample$. In particular, we compare their performances to the commonly used BPR loss function and the uniform sampling, respectively.

7.4 Results and Analysis

We present and analyse the results of our experiments to answer the research questions in Section 7.3. Our experiments focus on investigating the performance of RPRM as well as the effectiveness of our proposed loss functions and negative sampling strategies in comparison to six strong baselines from the literature.

7.4.1 RQ 7.1: Review Property-based Model Evaluation

To answer RQ 7.1, we first examine the performances of the six baselines and compare them to the performances of our proposed RPRM model. In particular, we integrate each review property separately, before combining all of them together in the full RPRM model. The results on the two used datasets are presented in Table 7.1.

First, we compare the performance of the RPRM model without using the review property

Table 7.1: Recommendation performances when using review properties in RPRM. Significant differences w.r.t. 'No-Prop' are indicated by '*' (according to both the paired t-test and the Tukey HSD correction method, $p < 0.05$). 1/2/3 denote a significant difference according to both tests w.r.t. to the indicated approach. ↑ indicates that the corresponding approach is significantly outperformed by RPRM on all ranking metrics according to both tests.

Dataset			**Amazon**		
Model	**P@1**	**P@10**	**R@10**	**MAP**	**NDCG@10**
↑BPR-MF	0.0053*	0.0034*	0.0301*	0.0111*	0.0163*
↑DREAM	0.0052*	0.0043*	0.0291*	0.0106*	0.0151*
↑CASER	0.0093*	0.0060*	0.0499*	0.0239*	0.0315*
↑1 DeepCoNN	0.0053*^{2,3}	0.0037*^{2,3}	0.0343*^{2,3}	0.0119*^{2,3}	0.0157*^{2,3}
↑2 JRL	0.0041*^{1,3}	0.0031*^{1,3}	0.0310*^{1,3}	0.0092*^{1,3}	0.0123*^{1,3}
↑3 NARRE	0.0175*^{1,2}	0.0066*^{1,2}	0.0588*^{1,2}	0.0279*^{1,2}	0.0365*^{1,2}
↑No-Prop	0.0208	0.0088	0.0805	0.0357	0.0467
Age	0.0215*	0.0089	0.0820*	0.0372*	0.0489*
Length	0.0214	0.0089	0.0815*	0.0364*	0.0477*
Helpful	0.0218*	0.0089	0.0817*	0.0365*	0.0483*
Prob-Helpful	0.0214	0.0093	0.0852*	0.0376*	0.0495*
Ratings	0.0206	0.0087	0.0795*	0.0352	0.0460
Polar-Senti	0.0211	0.0086	0.0783*	0.0355	0.0466
RPRM	**0.0223***	**0.0095***	**0.0865***	**0.0378***	**0.0496***
Dataset			**Yelp**		
Model	**P@1**	**P@10**	**R@10**	**MAP**	**NDCG@10**
↑BPR-MF	0.0101*	0.0058*	0.0391*	0.0145*	0.0217*
↑DREAM	0.0083*	0.0065*	0.0469*	0.0155*	0.0229*
↑CASER	0.0111*	0.0083*	0.0571*	0.0229*	0.0333*
↑1 DeepCoNN	0.0054*^{2,3}	0.0025*^{3}	0.0173*^{2,3}	0.0072*^{2,3}	0.0115*^{2,3}
↑2 JRL	0.0043*^{1,3}	0.0021*^{3}	0.0135*^{1,3}	0.0061*^{1,3}	0.0093*^{1,3}
↑3 NARRE	0.0137*^{1,2}	0.0087*^{1,2}	0.0605*^{1,2}	0.0228*^{1,2}	0.0332*^{1,2}
↑No-Prop	0.0153	0.0099	0.0745	0.0260	0.0387
Age	0.0157	0.0105*	0.0756*	0.0267*	0.0415*
Length	0.0159*	0.0101	0.0726*	0.0262	0.0403
Helpful	0.0151	0.0100	0.0719*	0.0255	0.0394
Prob-Helpful	0.0152	0.0103	0.0750	0.0264	0.0409
Ratings	0.0160*	0.0102	0.0730*	0.0264	0.0411
Polar-Senti	0.0155	0.0102	0.0738	0.0262	0.0401
RPRM	**0.0161***	**0.0104**	**0.0761***	**0.0271***	**0.0421***

encoding layer (namely No-Prop) to the baseline approaches from Table 7.1. We observe that No-Prop significantly outperforms all baseline approaches, including the state-of-the-art recommendation approaches (namely CASER, DeepCoNN and NARRE), according to both the paired t-test and the Tukey HSD correction method regardless of whether they use any review information. In particular, we focus on DeepCoNN, JRL and NARRE, which make use of review information. Both DeepCoNN and JRL exhibit weak recommendation performances on the two used datasets with low precision, recall and MAP scores, which are lower than the traditional BPR-MF approach. The BPR-MF approach is a strong baseline and was shown recently to outperform various state-of-the-art recommendation approaches from the literature (Rendle et al., 2020). Among these three baselines, NARRE significantly outperforms both the Deep-

CoNN and JRL approaches according to both the paired t-test and the Tukey HSD correction method on the two datasets with higher evaluation scores. However, NARRE is significantly outperformed by our No-Prop variant (according to both the paired t-test and the Tukey HSD correction method), despite No-Prop having a simpler structure than NARRE. The effectiveness of this simple review-based recommendation approach is consistent with the conclusions in (Sachdeva and McAuley, 2020). Moreover, by comparing No-Prop and DeepCoNN, we note that the only architecture difference between these two models is that No-Prop integrates the identification embedding vectors of users and items. The observed significantly enhanced performances of No-Prop over DeepCoNN on all used metrics (paired t-test and Tukey HSD correction method) demonstrate the benefits of using such embedding vectors to model the users' preferences and items' attributes. In summary, we find that No-Prop significantly outperforms all baseline approaches. In particular, the use of the embedding vectors, which model the users' preferences and items' attributes, explains the superior performances of both the No-Prop and NARRE models in comparison to other baseline approaches.

Next, we evaluate the effectiveness of integrating different review properties to the basic No-Prop approach to model the usefulness of reviews. The results from Table 7.1 show that in general, the review properties can significantly improve the performances of No-Prop on both used datasets according to both the paired t-test and the Tukey HSD correction method. In particular, we observe that the 'Age' and 'Prob_Helpful' review properties are the two most effective properties among the six review properties we tested in capturing the usefulness of reviews and improving the recommendation effectiveness of No-Prop. The other review property-based approaches show different performances on the two datasets. For example, the 'Helpful' property enhances the performances of No-Prop on the Amazon dataset, but decreases its performances on the Yelp dataset. Moreover, the 'Ratings' property improves the recommendation performances of No-Prop on the Yelp dataset but not on the Amazon dataset.

Therefore, these results suggest that it is more effective to selectively apply the right review properties in the recommendation model to assess the usefulness of the reviews and to leverage them in the made recommendations, which is one of the main underlying ideas of our proposed review-based recommendation framework (see Chapter 4). In particular, these results indicate the necessity of understanding the importance of different review properties on different datasets or recommendation applications. Therefore, next, we evaluate the performance of our proposed RPRM model, which integrates all six review properties and appropriately scores (or weights) the importance of different reviews' properties. The observed results for RPRM from Table 7.1 show that the RPRM model provides the best recommendation effectiveness on the two used datasets. Moreover, the observed performances significantly outperform both No-Prop

Table 7.2: Impact of the model's learning schemes on RPRM. Statistically significant differences with respect to 'RPRM-basic' are indicated by '*' (according to both the paired t-test and the Tukey HSD correction method, $p < 0.05$).

Dataset	**Amazon**					**Yelp**				
Model	**P@1**	**P@10**	**R@10**	**MAP**	**NDCG**	**P@1**	**P@10**	**R@10**	**MAP**	**NDCG**
RPRM-basic	0.0223	0.0095	0.0865	0.0378	0.0496	0.0161	0.0104	0.0761	0.0271	0.0421
PropLoss$_{uu}$-KL	0.0217	0.0094	0.0867	0.0381	0.0499	0.0163	0.0106	0.0772*	0.0281*	0.0434*
PropLoss$_{uu}$-Cos	0.0211*	0.0093	0.0853*	0.0369*	0.0484*	0.0154*	0.0105	0.0764	0.0271	0.0423
PropLoss$_{ui}$-KL	**0.0226**	**0.0097**	**0.0894***	**0.0385***	**0.0507***	**0.0175***	**0.0107**	**0.0788***	**0.0288***	**0.0442***
PropLoss$_{ui}$-Cos	0.0211*	0.0093	0.0859*	0.0382	0.0496	0.0165	0.0105	0.0772*	0.0278*	0.0430
PropSample-KL	0.0225	0.0093	0.0863	**0.0385***	0.0503*	0.0163	0.0102	0.0738*	0.0268	0.0419
PropSample-Cos	0.0220	0.0093	0.0855	0.0376	0.0493	0.0166	0.0105	0.0767*	0.0276	0.0424
PropLoss$_{ui}$-KL +*PropSample*-KL	0.0210*	0.0095	0.0871*	0.0373	0.0489	0.0162	0.0100	0.0740*	0.0266	0.0415
PropLoss$_{ui}$-KL +*PropSample*-Cos	0.0211*	0.0094	0.0863	0.0372*	0.0486*	0.0159	0.0105	0.0770*	0.0273	0.0425

and all the baseline approaches, including the existing state-of-the-art recommendation models (namely, NARRE, CASER and DeepCoNN), according to both the paired t-test and the Tukey HSD correction method. These results demonstrate the benefits of using all review properties and weighting their importance for capturing the useful reviews and their leverage in recommendation.

Therefore, in answering RQ 7.1, we conclude that different review properties show distinct effectiveness levels in enhancing the performance of a review-based recommendation model. Among the six used review properties, the 'Age' and 'Prob_Helpful' properties are the most effective, and consistently enhance the effectiveness of the No-Prop recommendation model. In particular, we identified the consistent effectiveness of leveraging the 'Prob_Helpful' property to address rating prediction and ranking-based recommendation tasks. However, we also showed the inconsistent effectiveness of applying the sentiment property of reviews to these two tasks (i.e. effective when used in addressing the rating prediction but not the ranking task). Furthermore, by integrating all six review properties and weighting their importance in the full RPRM model, we observe that RPRM achieves the best performance among all evaluated approaches on both used datasets. Our results also validate our hypothesis in Section 7.2.2.3, namely that weighting the importance of review properties with a dot-product attention mechanism can enhance the recommendation performances.

7.4.2 RQ 7.2: Effectiveness of the Proposed Loss Functions

To answer RQ 7.2, we examine the impact of using our proposed loss functions (namely *PropLoss*$_{uu}$ and *PropLoss*$_{ui}$) using two different (dis)similarity approaches (namely the KL divergence and Cosine similarity). We also compare the RPRM model that uses our proposed loss functions

with the same model using a standard ranking scheme, namely the BPR loss function and a uniform sampling strategy for generating negative items (i.e. RPRM-basic). By exploring different combinations, we have four possible model learning setups, i.e. $PropLoss_{uu}$ with KL or Cosine and $PropLoss_{ui}$ with KL or Cosine. Table 7.2 presents the obtained experimental results on the two Amazon and Yelp datasets for these four model learning setups. First, from Table 7.2, we observe that our proposed two loss functions can consistently improve the performance of the basic RPRM with the exception of the $PropLoss_{uu}$-Cos model setup on the Amazon dataset. In particular, by comparing the evaluation performances of the PropLoss-based approaches with that of the basic RPRM, we observe that $PropLoss_{ui}$-KL improves upon the recommendation performances of RPRM-basic with significantly higher MAP and NDCG scores according to both the paired t-test and the Tukey HSD correction method on the two used datasets: (MAP: 0.0378 $\rightarrow$ 0.0385) and (NDCG: 0.0496 $\rightarrow$ 0.0507) on the Amazon dataset, and (MAP: 0.0271 $\rightarrow$ 0.0288) and (NDCG: 0.0421 $\rightarrow$ 0.0442) on the Yelp dataset, which is significant according to both the paired t-test and the Tukey HSD correction method at $p < 0.05$.

Next, we compare the impact of integrating the two proposed loss functions in turn into RPRM. From the results in Table 7.2, we observe that the $PropLoss_{ui}$-based approaches outperform the $PropLoss_{uu}$-based approaches on both datasets. This observation suggests that, in terms of setting the importance of the used review properties, it is more effective to amplify the disagreement between the users' interacted items and the negatively sampled items than that between users and the negatively sampled items. Our results also demonstrate that leveraging the users' adoption of information framework is a promising approach. Finally, by examining the effectiveness of the two similarity measurement approaches, we observe that both the KL and Cosine (dis)similarity-based approaches can outperform the RPRM-basic model when applied with the $PropLoss_{ui}$ approaches. However, KL is consistently more effective on both datasets in comparison to the Cosine similarity method. Moreover, the $PropLoss_{uu} - KL$ significantly outperform RPRM-basic according to both the paired t-test and the Tukey HSD correction method. This result overall demonstrates the effectiveness of modelling the divergence between the weighting vectors of the review properties on our used datasets.

After analysing the results from Table 7.2, we now answer RQ 7.2: our proposed loss functions $PropLoss_{uu}$ and $PropLoss_{ui}$ can both improve the performances of RPRM-basic. Moreover, $PropLoss_{ui}$ shows a higher effectiveness than $PropLoss_{uu}$ in enhancing the recommendation performance. We conclude that the divergence between the weighted vectors of the review properties using the KL divergence measure, $PropLoss_{ui}$, can enhance the RPRM-basic model and gives the best overall recommendation performances.

7.4.3 RQ 7.3: Effectiveness of the Proposed Negative Sampling Strategy

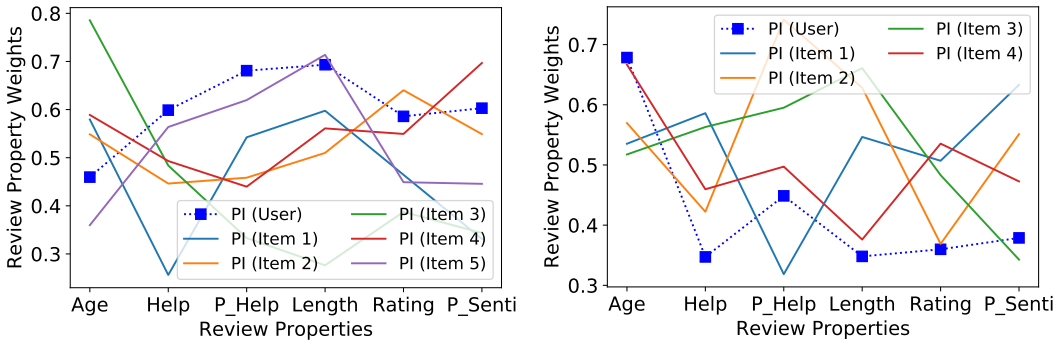
We now examine the effectiveness of our proposed negative sampling strategy (*PropSample*), so as to answer RQ 7.3. From Table 7.2, we observe that *PropSample* does not consistently outperform the RPRM-basic model when using the same similarity approach. In particular, the *PropSample* model with the KL divergence can improve the recommendation performance of RPRM-basic on the Amazon dataset but not on the Yelp dataset. On the other hand, the *PropSample* model that uses the Cosine similarity can improve the recommendation performance of RPRM-basic on the Yelp dataset but not on the Amazon dataset. Next, we investigate combining the *PropSample* negative sampling strategy with the best performing loss function, namely $PropLoss_{ui}$-KL (see the last two rows of Table 7.2). We observe that the combination of both *PropSample* and $PropLoss_{ui}$-KL with RPRM-basic does not lead to a better performance than when solely using $PropLoss_{ui}$-KL. These results might be caused by the fact that both $PropLoss_{ui}$ and *PropSample* similarly capture the importance of the review properties between the users' interacted items and the unseen items. Furthermore, *PropSample* only considers the agreement between the users' interacted (positive) and the unseen (negative) items on the important reviews' properties, which might not be sufficient to sample useful negative items.

Therefore, to answer RQ 7.3, we conclude that *PropSample* can enhance RPRM if an adequate similarity measure is applied on each used dataset. In addition, by comparing the performances of the *Prop -Sample* and *PropLoss* approaches, our results show that the *PropLoss* loss function has more impact on the recommendation effectiveness than the negative sampling strategy, suggesting that it is more important to capture the reviews' properties importance between the users and their interacted or unseen items.

7.5 Users' Property Preferences

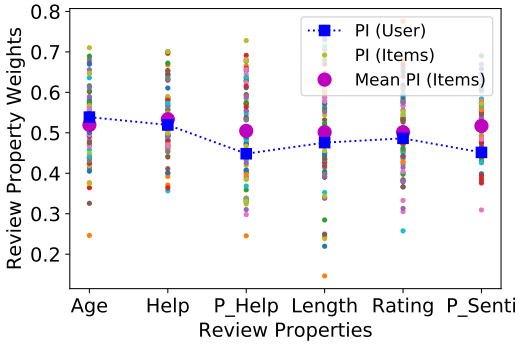
The users' personal views on the usefulness of reviews, as discussed in the users' adoption of information framework is one of the main arguments underlying our proposed RPRM model as well as the key motivation of this thesis (see Section 1.2). In Section 7.4, we showed the effectiveness of modelling the agreement between the users and items in terms of the reviews' properties. Therefore, in this section, we use three randomly selected users to illustrate the users' preferences on different review properties and the agreement on the importance of review properties between the users and their interacted items.

To this end, Figure 7.2(a)-(c) plots the learned RPRM property importance scores for the review properties of three randomly selected users, say A, B & C, as well as their interacted



(a) User A (few interactions).

(b) User B (few interactions).



(c) User C (many interactions).

Figure 7.2: The properties' importance scores of reviews for randomly selected users and their interacted items. 'Help', 'P_Help', 'P_Senti' are the abbreviations of 'Helpful', 'Prob_Helpful' & 'Polar_Senti', resp. 'PI' refers to the Properties' Importance scores.

items. The users' property importance preferences are shown using a blue dashed line with square markers; their interacted items in the test set are also shown (solid lines in Figures 7.2a and 7.2b and dots in different colours in Figure 7.2c) from the Yelp dataset. In particular, we select users A & B as example users that have few interactions with items, to illustrate the importance scores on the review properties between the target users and their interacted items. Indeed, we select users with few interactions so as to be able to visually plot all these items in a figure. From Figure 7.2a, we observe that user A shows stronger preferences for the 'Length' property (i.e. A prefers longer reviews) and that the 'Length' property is also highly weighted when determining the usefulness of reviews associated to that user's interacted items. On the other hand, for user B (Figure 7.2b), the 'Age' property is an important review property (i.e. B prefers recent reviews) to capture the review usefulness, which is similar to the high weights on 'Age' for the interacted items.

Next, since users A and B have few item interactions, they might not accurately reflect the

behaviour of the general user population in using different reviews. Therefore, we plot the learned importance scores of the review properties for a third user, C, who has interacted with a higher number of items. We also plot the mean importance scores of the review properties of his/her interacted items in Figure 7.2c. From Figure 7.2c, we observe that the importance scores on the review properties of user C are close to the mean importance scores on the review properties of his/her interacted items, especially on the two most important review properties ('Age' and 'Helpful'). This tells us that the 'Age' and 'Helpful' properties are important properties to observe the usefulness of reviews for both user C and his/her interacted items.

The above figures provide further evidence that users and their interacted items agree on the important review properties. We envisage that an online platform could leverage these weighting scores to customise the review presentation to different users according to their preferences for different review properties. For example, it is better to present recent reviews to user B than to user A, while user A would prefer to see longer reviews so as to obtain more information about the items' features. Our proposed RPRM model can learn the importance of the review properties to identify useful reviews and enables making review presentation decisions.

7.6 Conclusions

After effectively extracting two useful review properties (i.e. the reviews' sentiment and helpfulness properties) in Chapters 5 and 6, in this chapter, we aimed to address another main challenge in our proposed recommendation framework, namely the estimation of users' preferences. To address the estimation of users' preferences, we proposed the review-based RPRM model (see Section 7.2.2), which leverages the attention mechanism to model the importance of different review properties in capturing the usefulness of reviews thereby enhancing the recommendation performance. In particular, inspired by the users' adoption of information framework (Sussman and Siegal, 2003), in Section 7.3.3, we further proposed two new loss functions and a negative sampling strategy that account for the usefulness of the review properties.

The experimental results in Table 7.1 showed that RPRM consistently and significantly outperformed six strong existing recommendation models across the Yelp and Amazon datasets. In particular, when an individual review property drives the weights of the reviews' usefulness, the results in Table 7.1 showed that different review properties showed different levels of effectiveness in identifying useful reviews to improve the models' recommendation performances. The reviews' age and Prob_Helpful properties were shown to be the two most effective among the six used review properties. Next, among the loss functions and negative sampling strategies, we

showed that leveraging the KL divergence measure-based loss function (i.e. $PropLoss_{ui} - KL$) together with a uniform sampling, can enhance RPRM-basic and yields the best overall recommendation performances (see Table 7.2). These results validate the effectiveness of leveraging the attention mechanism to model the importance of the reviews' properties when identifying useful reviews. They also show that we could indeed obtain more accurate recommendations by leveraging the personalised usefulness of reviews among users through the reviews' associated properties. Hence, these results validate a key claim in our thesis statement (see Section 1.3). However, we argue that the attention mechanism is not the only method that can be applied to address the users' preferences estimation to identify personalised useful reviews. Therefore, in Chapter 8, we propose another review-based recommendation model that instantiates our proposed review-based recommendation framework in Chapter 4 using a different technique.

Chapter 8

Integrating Review Properties using Bandit Algorithms

8.1 Introduction

In Chapter 7, we instantiated our proposed review-based recommendation framework in Chapter 4 by introducing a novel review-based recommendation model, called RPRM. In particular, to address the estimation of the users' preferences for review properties and items (see Section 4.6.2), RPRM leveraged the attention mechanism to estimate the personalised importance of the review properties for different users. RPRM showed effective recommendation results. However, we argue that we can apply different techniques, aside from an attention mechanism, to effectively estimate the users' preferences for the review' properties, so as to obtain enhanced recommendation performances. Therefore, in this chapter, we investigate a different technique to address the choice of reviews' properties, thereby instantiating differently our proposed review-based recommendation framework in Chapter 4.

Recall from the users' adoption of information framework, which we discussed in Section 3.5, that different types of users prefer reviews exhibiting different properties. In particular, according to the results in Table 7.1 from Chapter 7, we showed that not every review property is effective when applied to model the usefulness of reviews in the recommendation task. This observation aligns with the users' adoption of information framework, which states that users selectively focus on specific review properties to identify useful reviews. The attention mechanism can effectively learn the importance of different review properties and linearly combine the resulting features when modelling the users/item features (see Section 7.4.1). However, the attention mechanism cannot model the users' behaviour, whereby they focus on specific review

properties. Therefore, in this chapter, we argue that, instead of using an attention mechanism, it is beneficial to model the users' choice in using some specific review properties to estimate the usefulness of reviews. In particular, we propose to characterise such selection behaviour as a multi-armed bandit problem. Indeed, the identification of those review properties that are most preferred by a given user can be characterised as an exploration vs. exploitation dilemma (Auer et al., 2002). Typically, a contextual bandit search algorithm relies on the context to estimate the rewards for selecting different arms. In particular, in the thesis statement (see Section 1.3), we also argued that users present distinct personalised preferences on the use of reviews and when interacting with items. To capture the users' differences, we propose to leverage the users' posted reviews, since they convey rich preferences information (Antelmi, 2019; Jin et al., 2013), as the context used by the contextual bandit search algorithm, so as to effectively estimate the value of each review property for a given user. In particular, recall the used attention mechanism from Section 7.2.2.3, which relied on the resulting recommendation outcome to estimate the importance of various review properties. In this chapter, different from the attention mechanism, using the users' posted reviews as the context in the bandit search algorithms, we can also model the users' differences according to the features of their posted reviews.

In the literature, the recommendation task has been frequently converted into a bandit problem (Xu et al., 2020; Li et al., 2010; Cañamares et al., 2019) to understand the users' preferences when interacting with different items. Many studies (Li et al., 2010; Xu et al., 2020) applied different strategies to estimate the value of selecting different items to recommend to users. However, for a multi-armed bandit problem with k arms, the difficulty of finding the optimal arm increases as the value of k increases (Cicirello and Smith, 2005). Therefore, if a recommendation model tackles the recommendation task as a bandit problem, its performance will be limited when applied to a large dataset with numerous items.

In this chapter, instead of considering the selection of items as a bandit problem like in (Xu et al., 2020; Li et al., 2010; Cañamares et al., 2019), we capture the users' preferences on reviews using six common review properties, such as the review length, age and helpfulness, as in Chapter 7. Then, different from using the dot-product attention mechanism in Chapter 7, we convert the learning of the users' preferences on reviews encoded with such a small pre-defined number of review properties into a bandit problem. This helps our approach to provide stable and consistent recommendation performances on both small and large datasets. Hence, we propose a novel deep neural network (NN) model (called BanditProp) to address the introduced bandit problem and the recommendation task. In particular, BanditProp is a hard parameter sharing-based multi-task learning (MTL) NN model (Ruder, 2017). In this MTL model, each task corresponds to the use of reviews encoded with one particular review property to model the

reviews. As a consequence, with k review properties, BanditProp has k sub-networks to extract and generate the feature vectors of reviews corresponding to every review property. Using the generated feature vectors through the modelling of reviews encoded with k review properties, for a user-item pair, the model generates k scores that estimate the user's preferences on the item given a particular review property. To the best of our knowledge, BanditProp is novel in investigating the use of both MTL and the bandit algorithms to estimate the users' preferences on using various review properties to examine the reviews' usefulness, so as to enhance the performance of a recommendation system. We also explore various bandit algorithms and propose a new neural contextual bandit algorithm (i.e. ConvBandit) to estimate the payoffs of selecting the k scores and their corresponding review properties so as to improve the recommendation performance of the proposed model.

The remainder of this chapter is organised as follows: In Section 8.2, we recall again the ranking-based recommendation task that we aim to address in this thesis as well as its notations. Next, we discuss the proposed review modelling networks in the BanditProp model that model reviews encoded with various review properties (Section 8.2.2). Then, in Section 8.2.3, we describe the arm selection layer in our proposed BanditProp model, which considers various bandit algorithms to selectively use the reviews' features encoded with different reviews' properties. After that, we introduce the experimental setup in Section 8.3, which presents the research questions to answer in this chapter, the used datasets, the baseline approaches and the details of the models' settings. Next, we present and analyse the experimental results in Section 8.4, and answer to the proposed research questions. Finally, we summarise the conclusions of this chapter in Section 8.6.

8.2 Methodology

In this section, we recall again the ranking-based recommendation task stated in Section 4.2 and the commonly available properties associated to the reviews (Section 8.2.1). Next, we describe the review modelling in our proposed BanditProp multi-task learning recommendation model (Section 8.2.2). We also introduce how BanditProp estimates the users' preferences on the learned features from using different review properties as a bandit problem (Section 8.2.3). In particular, we describe various bandit algorithms used to address the bandit problem.

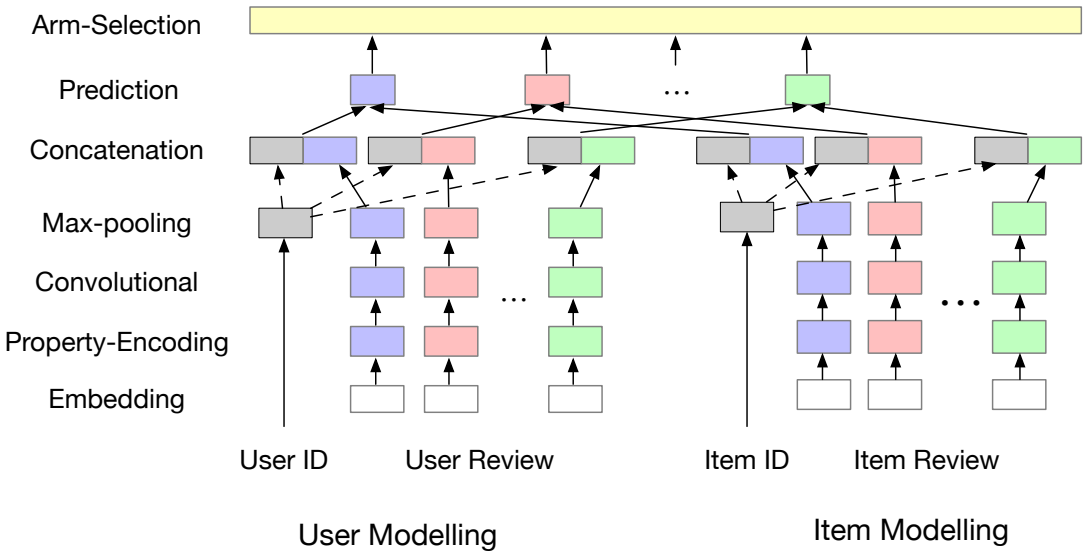


Figure 8.1: The structure of the BanditProp model

8.2.1 Problem Statement

The ranking-based recommendation task we are addressing involves two main entities: the set of users $U = \{u_1, u_2, ..., u_N\}$ with size N and the set of items $I = \{i_1, i_2, ..., i_M\}$ with size M. To address this task, we leverage the reviews posted by users on items to learn their preferences. For a given user u or item i, there is a set of corresponding reviews C_u or C_i. Furthermore, each review c can be additionally encoded with k review properties $\mathbf{P} = \{P_1, P_2, ..., P_k\}$ to depict the usefulness of reviews from different perspectives. In particular, for a given review property, e.g. P_k, such a review c posted by user u or on item i, has an associated property score $P^u_{k,c}$ or $P^i_{k,c}$. Note that, as in Section 7.2.2.1, we map the property scores into scalars in the range of [0..1] through an adequate function to avoid using review property scores in different scales.

8.2.2 Review Modelling in BanditProp

To address the introduced recommendation task aside from using the RPRM model in Chapter 7, in this chapter, we propose a recommendation model (i.e. BanditProp) that learns the users' preferences on items from reviews additionally encoded with their properties. BanditProp is a neural network model, which captures the interaction between the user-item pairs. Figure 8.1 presents the architecture of BanditProp. Its architecture involves two duplicated neural networks to learn the features of the user and the item, respectively. We use the user modelling neural network as an example to illustrate the network structure. In the BanditProp's input data, for

a given user, we have their ID and associated reviews. In particular, we use two separated networks to learn the information conveyed by the user/item ID and reviews, respectively. To use the user ID, we leverage the one-hot encoding technique to encode the general features of a given user. In the review modelling network, in order to capture the users' preferences on k different review properties, we use k parallel neural networks to model the reviews additionally encoded with k different review properties. Each network captures features from reviews based on a given review property. Then, each of the k parallel neural networks consists of five layers from the embedding layer to the concatenation layer. Each such a network outputs a latent vector to represent the learned features, which is concatenated with the user embedding. Next, after the interaction between the k pairs of the resulting latent vectors of the given user and item, we obtain k scores, which indicate the user's preferences on the item using the reviews embedded with k different review properties. This architecture can also be seen as a multi-task learning network that uses the shared user/item embeddings among k different review property modelling networks. In the following, we further describe the details of the review modelling neural networks.

8.2.2.1 Review Modelling Networks

In the review modelling networks of BanditProp, there are k parallel neural networks that learn features from the reviews additionally encoded with k different review properties, Recall from the description in Section 4.4 that in this thesis, we use six commonly available review properties, namely age, length, rating, Polar_Senti, Helpful and Prob_Helpful, to describe the reviews from different aspects. These properties are again described in Table 8.1. The network that models the reviews, additionally encoded with each of the six review properties, consists of five layers from the embedding to the concatenation layer. In the following, since the parallel review property modelling networks have the same structure, we use the network that focuses on the reviews additionally encoded with the length property to illustrate the five layers of the networks.

First, in the embedding layer, similar to the RPRM model introduced in Chapter 7, we convert each review into the embedding vector X by using the pre-trained BERT model (Devlin et al., 2019), which is a widely used language modelling approach. Next, in the property-encoding layer, we embed the scores of the length property of reviews into the converted review embedding vectors. For example, for a given review c posted by user u and the length property P_k, we calculate the length property score $p^u_{k,c}$ for review c as described in Table 8.1.

This review property score indicates the importance of review c according to the length

Properties	Descriptions
Age	The min-max normalised number of days d since a review has been posted.
Length	The min-max normalise number of words that are included in a review.
Ratings	The min-max normalised ratings associated with the review.
Polar_Senti	A CNN classifier (Kim, 2014) is used to calculate the corresponding probabilities of the positive reviews being actually positive or the negative reviews being negative.
Helpful	The min-max normalised number of helpful votes for a given review.
Prob_Helpful	The estimated probability of a review being helpful by using a review helpfulness classifier (Wang et al., 2020b).

Table 8.1: Summary of the review properties.

property P_k. In particular, according to the motivation of the length property, a longer review will obtain a higher length property score than a shorter review. Afterwards, the integration of the review property scores to the review embedding vector X is following the sentiment integration technique (i.e. the sentiment attention mechanism) in Chapter 5 and performed by the dot-product operation as follows:

$$O_{u,P_k} = [X_1 \cdot p^u_{k,1}, X_2 \cdot p^u_{k,2}, ..., X_{|C_u|} \cdot p^u_{k,|C_u|}] \tag{8.1}$$

where P_k refers to the k_{th} property and the property scores are applied to user u's $|C_u|$ reviews. As a consequence, BanditProp can extract and learn features from the property score-encoded latent vectors for each of the given review property (e.g. the length property). Moreover, by using k review properties, BanditProp can comprehensively capture features from reviews using different review properties.

After the property encoding layer, similar to the introduced RPRM model in Chapter 7, the review property-encoded latent vector O_{u,P_k} is used as input to the convolutional layer and then the max-pooling layer. In the convolutional layer, given the textual nature of users' posted reviews, we apply the one-dimensional convolutional operator to capture the features from the reviews. Then, we use the max-pooling operation to filter the resulting feature vectors. Next, we concatenate the learned embedding vector from the review modelling network and the embedding of the corresponding user into a latent vector. This latent vector indicates the learned feature from the user by looking at both their general preferences on items and their learned

preferences from the reviews using a specific review property (e.g. length of reviews). Afterwards, for a given user u or item i with k review properties, we obtain k resulting latent vectors ($O'_{u,P}$ and $O'_{i,P}$, $P \in \mathbf{P}$) for both user u and item i. We denote the parameters included in these review modelling networks as ϕ where ϕ_{P_k} indicates the parameters in the network that models the k_{th} review property P_k. Next, for the review modelling network of the reviews additionally encoded with a given review property P_k, we apply the dot-product operation on the learned latent vectors between user u and item i (i.e. $o''_{u,i,k} = O'_{u,P_k} \cdot O'_{i,P_k}$) to conduct users' preference estimation. This results in k scores that estimate the user's preferences on items for each of the k review properties.

8.2.3 Arm Selection Layer in BanditProp

Building on the prediction layer, we now formulate a bandit problem to imitate the users' behaviours of using different types of reviews. In particular, this bandit problem selectively applies the computed scores $o''_{u,i,k}$ from the modelling of k review properties. Bandit problems are typically formulated in terms of the arms ($\mathscr{A}$), the reward ($\mathscr{R}$) given to those arms during learning and any state recorded between successive turns $\mathscr{S}$. In the arm selection layer, different from the use of the attention mechanism in Chapter 7, we propose to consider the selection of the computed scores $o''_{U,I,k}$ of all users U and items I on the k review properties from the review modelling networks as a bandit problem. The target is to enhance BanditProp's performances with a corresponding optimised rewards $\mathscr{R}$ by selecting the scores $o''_{U,I,k}$ (i.e. the arms $\mathscr{A}$) to estimate users' preferences on items. In particular, the main difference between using the attention mechanism and the bandit algorithms is that the attention mechanism calculates a linear weight over using every review property. In contrast, the bandit algorithms consider selecting each user's most useful review properties. In the real scenario of users interacting with different review properties, users will be less likely to use every review property to search for valuable reviews. Therefore, we expect BanditProp, which uses the bandit algorithms, to outperform RPRM, which considers the attention mechanism. Next, we formulate the focused bandit problem as follows:

- The arms $\mathscr{A}$ correspond to the k review properties $\mathbf{P}$.
- The reward $\mathscr{R}$ follows a multinomial distribution and is defined as the ratio of times that the model ranks the positive item higher than the randomly sampled negative item in Γ positive-negative item pairs by using the feature score of a given review property. The

Algorithm 1: Model learning for user u by using the default greedy search.

$R_{u,\mathbf{P}} \leftarrow 0$; // Accumulated reward ;
$N_{u,\mathbf{P}} \leftarrow 0$; // Count the selection of arms ;
$n \leftarrow 0$;
repeat
  for $P_k \in \boldsymbol{P}$ **do**
    $r_{u,P_k} \leftarrow$ the reward by using property p;
    Compute gradients of ϕ_{P_k};
    Freeze the gradients of $\phi_{\mathbf{P}} - \phi_{P_k}$;
    Update the parameters;
  end
  for $i \leftarrow 0$ **to** n **do**
    $P \leftarrow max(R_{u,P_k}/N_{u,P_k})$; // Equation (8.2);
    $R_{u,P_k} \leftarrow R_{u,P_k} + r_{u,P_k}$;
    $N_{u,P_k} \leftarrow N_{u,P_k} + 1$;
  end
  $n \leftarrow n+1$;
until *Convergence*;

resulting rewards' values are the elements of a set $\{0, \frac{1}{\Gamma}, ..., 1\}$. For example, after conducting an arm selection, if the model ranks the positive item higher than the negative item six times in ten positive-negative item pairs, the reward is 0.6.

- The state $\mathscr{S}$ is the review-based feature vector of a given user. As discussed in the users' adoption of information framework (Sussman and Siegal, 2003; Filieri and McLeay, 2014) in Section 3.5, there is a relationship between the users' type and their preferred properties of reviews. In particular, in Chapter 7, we experimentally showed that there is a potential relatedness between the properties of the users' posted reviews and the users' types. Therefore, we propose to leverage the users' posted reviews as context to approximate the users' type so as to support the prediction of the users' preferred review properties. Note that, the contextual bandit algorithms extend the multi-armed bandit algorithms by leveraging the state $\mathscr{S}$ to construct their algorithms and predict the users' preferences on the review properties.

To effectively address the estimation of users' preferences for selecting the reviews properties, we propose a customised learning algorithm, which is described in Algorithm 1. In Algorithm 1, we introduce how we separately train the parallel review modelling networks and the bandit approaches. Note that we use the greedy search as an example in Algorithm 1, but

it can be replaced with any other bandit algorithm. For the bandit approaches, in this chapter, we consider two types of bandit algorithms, namely multi-armed bandit (MAB) and neural contextual bandit algorithms, to address our introduced bandit problem. We now introduce these algorithms:

8.2.3.1 Multi-Armed Bandit Approaches

In general, a multi-armed bandit can be described as a tuple $\langle \mathscr{A}, \mathscr{R} \rangle$ that indicates actions and rewards, respectively. In this chapter, we include three types of multi-armed bandit (MAB) algorithms, namely greedy search-based Kuleshov and Precup (2000), UCB-based Kaufmann et al. (2012) and Thompson Sampling (TS)-based Russo et al. (2018) algorithms. The greedy search-based approaches, namely greedy search, ε-greedy search and decayed ε-greedy search, are commonly used MAB algorithms to address a bandit problem (Kuleshov and Precup, 2000). They exploit the averaged historical accumulated payoffs of selecting the corresponding review properties. The corresponding arm selection strategy is formulated as follows:

$$a_t^* = \arg\max_{a \in \mathscr{A}} \left[\frac{1}{N(a)} \sum_{t=0}^{T} r_t(a_t = a) \right] \tag{8.2}$$

However, the greedy search-based algorithms differ in conducting the exploration of the arm selections. Greedy search solely relies on the exploitation of historical payoffs (i.e. Equation (8.2)). Note that we later use greedy search as the default bandit algorithm in our proposed BanditProp model. Meanwhile, ε-greedy search (Kuleshov and Precup, 2000) uses a fixed exploration factor ε to indicate the probability of selecting random arms. On the other hand, the decayed ε-greedy (i.e. DE-greedy) search (Kuleshov and Precup, 2000) function downgrades the value of the exploration factor ε over time. In this chapter, we use the following decay function:

$$\varepsilon_t = \frac{1}{log(t)} \tag{8.3}$$

Note that, since the comparison of different decay functions is beyond current scope of this chapter, we use the above decay function as a representative method to implement the DE-greedy search.

Apart from the greedy search-based algorithms, we consider another family of bandit algorithms (i.e. UCB search (Kaufmann et al., 2012)). Different from the greedy search algorithms, the UCB search algorithms estimate the values of arms by capturing the upper confidence bounds of selecting arms. The upper confidence bound $\hat{U}$ is defined as the uncertainty in the payoffs

of selecting an arm. If an arm has been frequently selected, it has a lower uncertainty in the payoffs than selecting other arms. Therefore, the system tends to select arms with higher uncertainty in the payoffs to capture the values of selecting these arms. We first consider a classic UCB algorithm (i.e. UCB1), which can be formulated as follows:

$$a_t^* = \underset{a \in \mathscr{A}}{\arg\max} \left[\frac{1}{N(a)} \sum_{t=0}^{T} r_t(a_t = a) + \sqrt{\frac{2\log(T)}{N(a)}} \right] \tag{8.4}$$

The UCB1 strategy estimates the value of arms according to the averaged accumulated rewards (left part of Equation (8.4)) like the greedy search-based approaches and the uncertainty score (right part of Equation (8.4)). Aside from the UCB1 strategy, we also include another UCB algorithm (i.e. Bayes-UCB (Kaufmann et al., 2012)) that leverages the prior knowledge of the arms' values to estimate the posterior values of arms. We assume that the prior knowledge of arms can be modelled by the beta distribution, which includes two shape parameters $A(a)$ and $B(a)$. At each iteration, if arm a is selected, $A(a)$ and $B(a)$ are updated by $A_{t+1}(a) = A_t(a) + r_{t,a}$ and $B_{t+1}(a) = B_t(a) + (1 - r_{t,a})$, respectively. The Bayes-UCB strategy estimates the values of arms as:

$$a_t^* = \underset{a \in \mathscr{A}}{\arg\max} \left[\frac{A_t(a)}{A_t(a) + B_t(a)} + c\sigma\Big(A_t(a), B_t(a)\Big) \right] \tag{8.5}$$

where $c\sigma(A_t(a), B_t(a))$ is the standard deviation of the beta distribution and c determines the size of the confidence interval.

Moreover, similar to the Bayes-UCB strategy, we also consider the TS algorithm, which also relies on the beta distribution as the prior knowledge. However, the Bayes-UCB and TS algorithms are different in estimating the values of selecting arms. The TS algorithm follows the idea of probability matching (Shanks et al., 2002), which is a decision strategy that relies on the defined prior distribution. At each time step, the system samples the posterior probability of each arm and selects the arm that maximises the expected reward $\mathbb{E}$. This process can be interpreted with the following equation:

$$a_t^* = \mathbb{E}\left[\underset{a \in \mathscr{A}}{\arg\max} \Big(a | Beta(\alpha_t(a), \beta_t(a)) \Big) \right] \tag{8.6}$$

Furthermore, given that we are using multinomial rewards to examine the value of selecting arms, we also consider Multinomial TS (Mul-TS) (Riou and Honda, 2020), a state-of-the-art bandit algorithm, as another bandit approach to address our bandit problem. Mul-TS is designed to address bandit problems that use multinomial rewards. At each round of the bandit selection,

for each arm a, Mul-TS leverages the Dirichlet distributions $Dir(\beta_0^a, \beta_1^a, .., \beta_\Gamma^a)$ with dimension $\Gamma+1$ to generate samples L_a. Essentially, L_a refers to the likelihood of obtaining specific values of multinomial rewards after selecting a corresponding arm. After that, the arm selection can be described by the following equation:

$$a_t^* = \arg\max_{a \in \mathscr{A}} \left[(0, \frac{1}{\Gamma}, \frac{2}{\Gamma}, ..., 1)^\top L_a \right] \tag{8.7}$$

Then, the parameters of the used Dirichlet distribution are updated with: $\beta_\gamma^{a_t^*} := \beta_\gamma^{a_t^*} + 1$, where $\frac{\gamma}{\Gamma}$ is the observed reward. On the other hand, by comparing the implementations of the considered multi-armed bandit approaches in this chapter, we observe that the greedy search-based approaches and the UCB1 strategy rely on the accumulated historical rewards to estimate the values of arms, while the Bayes-UCB, TS and Mul-TS algorithms focus on constructing functional estimators to predict the values of arms. Given such a difference, the Bayes-UCB and TS-based approaches require an additional effort in learning the estimators, which makes it challenging for them to effectively estimate the values of arms. However, by comparing TS and Mul-TS, we expect that Mul-TS should outperform TS by capturing the multinomial nature of the reward distribution.

8.2.3.2 Neural Contextual Bandit Approaches

Neural contextual bandit algorithms extend the MAB algorithms by leveraging the state information $\mathscr{S}$ to support the estimation of the arms' values. According to recent developments in neural contextual bandits (Zahavy and Mannor, 2019; Riquelme et al., 2018), neural contextual bandit approaches are considered to be the current state-of-the-art and generally have a good performance. Therefore, in this chapter, we consider the neural contextual bandit approaches, which model the arms, states and rewards (i.e. $\langle \mathscr{A}, \mathscr{S}, \mathscr{R} \rangle$) through a neural network. The action-value function is formulated by the expected reward, which considers both the state and action (i.e. $Q(s,a) = \mathbb{E}(r|s,a)$).

As mentioned above, we propose to use the review-based feature vectors X of the users as the state $\mathscr{S}$. We first construct a linear neural contextual bandit algorithm (LinBandit) Zhou et al. (2020a). It approximates the value of arms with the following equation:

$$Approx^a_{LinBandit} = W_L^a \sigma \Big(W_{L-1}^a \sigma(...\sigma(W_1[X \oplus X^a])) \Big) \tag{8.8}$$

where X indicates the feature vector of the reviews posted by a given user. X^a is the review

feature vector encoded with the information of arm a. The latter corresponds to the feature vector O_{u,P_k} within Equation (7.1) with review property P. $\oplus$ refers to the residual connector. In particular, LinBandit assumes that the values of arms can be linearly approximated from the corresponding context or state $\mathscr{S}$.

Next, we observe that the review feature vectors X of the arms are extracted from the textual documents. Given that the convolutional operation can effectively capture the local features of documents (Kim, 2014; Kim et al., 2016), we postulate that the convolutional operation can outperform the linear operation in better capturing information in the review-based feature vectors X. Therefore, in this chapter, differently from LinBandit, which uses a linear approximation function, we propose another neural contextual bandit algorithm (i.e. ConvBandit) that uses the convolutional operation-based approximation function to estimate the reward of selecting various arms. Its reward approximation function is:

$$Approx^{a}_{ConvBandit} = \sigma\left[f_1^a * (g_1^a(\sigma(f_0^a * g_0^a[X \oplus X^a])))\right] \tag{8.9}$$

where $\sigma(.)$ and $*$ are the activation function and the convolutional operation, respectively. Note that for both LinBandit and ConvBandit, we first train the approximation functions by minimising Mean Squared Error (MSE) between their predicted values for the given arms and the averaged historical rewards (i.e. $\frac{1}{N(a)}\sum_{t=0}^{T} r_t(a_t = a)$). Next, similar to NeuralUCB (Zhou et al., 2020a), while using the neural approaches for the arm value's estimation, we also leverage the upper confidence bound as in Equation (8.4) to implement the arm exploration. Therefore, the arm selection of both LinBandit and ConvBandit are implemented as follows:

$$a_t^* = \underset{a \in \mathscr{A}}{\arg\max} \left[Approx^a + \sqrt{\frac{2\log(T)}{N(a)}} \right] \tag{8.10}$$

In the following, we describe the experiments we apply to examine the effectiveness of our proposed techniques in this section.

8.3 Experimental Setup

We first introduce our research questions, which examine the effectiveness of BanditProp. Finally, we describe the two used datasets as well as the baseline approaches. Next, we describe in detail the settings used to train and deploy both BanditProp and the baseline approaches.

In order to evaluate our proposed BanditProp model, which uses various bandit algorithms,

we aim to answer the following research questions:

1. **RQ 8.1:** Does BanditProp, which uses a multi-task learning framework with a default greedy search algorithm outperform existing state-of-the-art baseline approaches, especially our proposed RPRM model in Chapter 7?

2. **RQ 8.2:** Which multi-armed bandit algorithm performs the best when applied to the BanditProp model on the two used datasets?

3. **RQ 8.3:** Do the neural contextual bandit algorithms outperform the multi-armed bandit ones when applied to the BanditProp model?

8.3.1 Datasets

We follow the experimental setup in Chapter 7 and use the same two real-world datasets (i.e. the Yelp (round 13)[1] and the Amazon[2] (He and McAuley, 2016) datasets). Recall the description of these two datasets from Chapter 7. Following the dataset filtering strategy, described in Section 7.3.1, for the Yelp dataset, we use the users' reviews on the top category of venues (i.e. restaurant). Moreover, for the Amazon dataset, we include the users' reviews from six categories[3]. We use one category in Yelp and six categories in Amazon in order to capture the users' preferences on properties and items under two different system interfaces and conditions. Moreover, as we discussed in Section 4.4, we use these two datasets because of their included rich review attributes (e.g. timestamp and helpfulness vote), allowing to extract the corresponding review properties, so as to capture the users' preferences on these review properties. We also filter the datasets to alleviate the datasets' sparseness problem as in (Sachdeva and McAuley, 2020) and in Section 7.3.1. The filtering operation results in all the users and items in both datasets having at least 5 associated reviews. After the filtering step, the resulting Yelp and Amazon datasets, have 47k users / 16k items / 551k reviews and 26k users / 16k items / 285k reviews, respectively. In order to evaluate the performances of our proposed approaches and baselines on the two datasets, we further split both datasets into 80% training, 10% validation and 10% testing sets. Then, after training and fine-tuning a given model or baseline on the training and validation sets, we evaluate their performances on the test sets. The recommenders' performances are measured using the Precision (P@1 and P@5), Recall (R@10), Mean Average Precision (MAP)

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

[2] http://jmcauley.ucsd.edu/data/amazon/

[3] 'amazon instant video', 'automotive', 'grocery and gourmet food', 'musical instruments', 'office products' and 'patio lawn and garden'

and Normalised Discounted Cumulative Gains (NDCG@10) metrics. Note that'@k' indicates the cutoff position at 'k' for the corresponding metric.

8.3.2 Baseline Approaches

We evaluate our proposed BanditProp model on the two used datasets in comparison to six strong state-of-the-art baselines. In this chapter, we focus on examining if the newly proposed BanditProp model can outperform the RPRM model in Chapter 7 as well as the other baseline techniques (i.e. BRP-MF, DREAM, CASER, JRL, DeepCoNN and NARRE). In the following, we recall the description of the first five baseline approaches from Chapter 7 and briefly describe the RPRM model.

1. *BPR-MF (Rendle et al., 2009)* is a classical recommendation model, which estimates the users' preferences on items by leveraging the interactions between the decomposed user-item feature vectors.

2. *DREAM (Yu et al., 2016)* is a sequential recommender, which leverages the ages of reviews and estimates the users' preferences on items by considering sequential dependencies in user-item interactions.

3. *CASER (Tang and Wang, 2018)* is a state-of-the-art sequential recommender. It uses the convolutional neural network to model the sequential interaction between a given user and the items.

4. *JRL (Zhang et al., 2017)* is a heterogeneous recommender that estimates the users' preferences from various types of users' explicit feedback (e.g. reviews and ratings). We implement the JRL model by only using the users' posted reviews.

5. *DeepCoNN (Zheng et al., 2017)* is a popular review-based recommendation baseline (Chen et al., 2018b; Wang et al., 2020b). It uses the interaction between the review feature vectors of the user-item pairs to estimate the users' preferences.

6. *NARRE (Chen et al., 2018b)* is a state-of-the-art review-based recommender. It ranks the users' interest in items, using both the one-hot embedding and the review feature vectors for a given user-item pair. It also leverages an attention mechanism to observe useful review features.

7. *RPRM* is the recommender we proposed in Chapter 7, which built upon the users' adoption of information framework. It used various review properties to describe the reviews from different perspectives. In particular, it leverages the attention mechanism to model the users' interactions with a given type of reviews so as to estimate the users' tendency in using the review data and to improve the recommendation accuracy.

8.3.3 Models' Settings

We deploy our proposed BanditProp model and the baseline approaches using the PyTorch framework (Paszke et al., 2019). As for the BanditProp's setup, we first follow the review processing setup in Chapter 7 by limiting the review length to 512 tokens. Next, we apply the pretained-BERT model to each review and use the representation vector of the '[CLS]' token to convert the review tokens into a 768-sized latent vector. Similarly, for the DeepCoNN, NARRE and RPRM baseline approaches, we also use the review embeddings, calculated by BERT (Devlin et al., 2019) as input to these models. Next, in the property-encoding layer, we compute the scores of the review properties as described in Table 8.1. For the 'Polar_Senti' and 'Prob_Helpful', we use a trained CNN-based classifier and a review helpfulness classifier, namely Negative Confidence-aware Weakly Supervised (NCWS), which we investigated in Chapters 5 and 6, respectively. We leverage these two techniques to estimate the probabilities of the reviews being sentimentally polarised (i.e. strongly positive or negative) or being helpful. Note that, following the experimental setup in Chapter 7, for the CNN-based sentiment classifier, we train it on 100,000 reviews, which include half positive and half negative reviews that are sampled from the Yelp Challenge dataset round 12. As for the NCWS classifier, we train two classifiers on the reviews from the Yelp Challenge dataset round 12 and the Amazon kindle datasets, respectively. These two classifiers are used to generate the probability of each of the collected reviews from the same website, being helpful. In particular, in the ground truth, we consider one review as helpful if it has at least one helpful vote. Furthermore, for the bandit problem tackled in this chapter, to generate the multinomial rewards and speed up the training, we follow Yang et al. (2018a) and set the number of randomly sampled positive-negative item pairs Γ to 10. For the ε-greedy search algorithm, as in (dos Santos Mignon and de Azevedo da Rocha, 2017), we set the ε to 0.1 to indicate the probability of selecting random arms. On the other hand, in the Bayesian UCB algorithm of Equation (8.5), we set the value of c to 3 to control the size of the confidence interval. For training the BanditProp model, we apply the early-stopping strategy with a maximum 200 epochs and use the Adam optimiser (Kingma and Ba, 2015) with a $1e^{-4}$ learning rate. The same strategy is also used to train the baseline approaches.

Table 8.2: Recommendation performances of BanditProp and baseline approaches. 1/2/3/4/5 denote a significant difference w.r.t. BPR-MF, DREAM, NARRE, RPRM and Bandit-Prop, respectively, on NDCG@10 according to a paired t-test with the Tukey HSD correction ($p < 0.05$).

	Dataset		**Amazon**				
	Model		**P@1**	**P@5**	**R@10**	**MAP**	**NDCG@10**
1	**BPR-MF**	2,3,4,5	0.00538	0.00431	0.03013	0.01118	0.01633
2	**DREAM**	1,3,4,5	0.00523	0.00358	0.02914	0.01067	0.01506
–	**CASER**	1,2,3,4,5	0.00934	0.00720	0.04991	0.02392	0.03150
–	**DeepCoNN**	2,3,4,5	0.00534	0.00451	0.03431	0.01196	0.01575
–	**JRL**	1,2,3,4,5	0.00417	0.00384	0.03102	0.00923	0.01238
3	**NARRE**	1,2,4,5	0.01753	0.01024	0.05885	0.02797	0.03654
4	**RPRM**	1,2,3,5	0.02238	0.01270	0.08657	0.03784	0.04966
5	**BanditProp**	1,2,3,4	**0.02534**	**0.06456**	**0.09481**	**0.04197**	**0.05536**
	Dataset		**Yelp**				
	Model		**P@1**	**P@5**	**R@10**	**MAP**	**NDCG@10**
1	**BPR-MF**	2,3,4,5	0.01014	0.00656	0.03919	0.01455	0.02176
2	**DREAM**	1,3,4,5	0.00838	0.00723	0.04692	0.01555	0.02292
–	**CASER**	1,2,3,4,5	0.01112	0.00937	0.05710	0.02291	0.03333
–	**DeepCoNN**	2,3,4,5	0.00549	0.00310	0.01739	0.00726	0.01154
–	**JRL**	1,2,3,4,5	0.00437	0.00294	0.01355	0.00615	0.00928
3	**NARRE**	1,2,4,5	0.01375	0.00994	0.06057	0.02289	0.03324
4	**RPRM**	1,2,3,5	0.01618	0.01221	0.07611	0.02713	0.04205
5	**BanditProp**	1,2,3,4	**0.01760**	**0.01326**	**0.08144**	**0.02952**	**0.04486**

8.4 Results analysis

In this section, we evaluate and analyse the performance of our proposed BanditProp recommendation model on the Yelp and Amazon datasets, and answer our three research questions. In particular, we analyse the effectiveness of improving the BanditProp's performance after the application of various bandit algorithms to it.

8.4.1 RQ 8.1: BanditProp Performance Evaluation

Our first research question aims to determine if our BanditProp model – with a default greedy search algorithm – can outperform the existing state-of-the-art recommendation. In particular, we further aim to investigate if the BanditProp using the bandit algorithms can outperform the RPRM proposed in Chapter 7 using the attention mechanism. Table 8.2 presents the performances of BanditProp with a default greedy search algorithm in comparison to 7 baselines. Following Sun et al. (2020a); Manotumruksa and Yilmaz (2020), statistical significance differences among the various approaches on the NDCG@10 metric are assessed using the paired

t-test with the Tukey HSD correction method ($p < 0.05$) for multitude comparisons.

Recall the result analysis from Section 7.4.1 about the comparison among the first five baseline approaches, we observe that the DeepCoNN and JRL models, which solely rely on the reviews' text as input, provide less competitive recommendation performances than other approaches. In particular, the JRL model is significantly outperformed by the classical BPR-MF recommender. Essentially, the BPR-MF, DeepCoNN and JRL approaches differ in their used input data. Indeed, BPR-MF leverages the randomly initialised and fine-tuned users/items embeddings, while both DeepCoNN and JRL rely on the embedding of textual reviews to calculate the users/items embeddings, to estimate the users' preferences on items. Moreover, the NARRE model, which uses both the users/items embeddings and the review feature embeddings of both users and items, significantly outperforms BPR-MF, DeepCoNN and JRL on the two datasets. Similarly, by comparing the performances of the DREAM and CASER models, we observe that CASER significantly outperforms the DREAM model. CASER and DREAM are both sequential recommenders that leverage the age property of reviews. However, differently from DREAM, CASER leverages the users/items embeddings. This result supports the use of the users/items embeddings in BanditProp.

Next, recall that the RPRM model can significantly outperform all the considered baselines on two datasets. This result demonstrates that RPRM, using the attention mechanism, is effective at capturing the users' preferences by leveraging various review properties, and predicting the users' interests in items. We now turn our attention to the performance of our proposed BanditProp model. The experimental results in Table 8.2 show that just by using the default greedy search algorithm, BanditProp significantly outperforms all the baseline approaches including RPRM. Given the difference between RPRM and BanditProp is their use of review properties, these results also support our argument for developing a model encompassing multi-task learning and a bandit search strategy that captures the users' preferences from their interaction with different types of reviews. Therefore, in answering RQ 8.1, we conclude that our proposed BanditProp model can significantly outperform the existing state-of-the-art recommendation baselines. In particular, we also showed the effectiveness of using the bandit search strategy (i.e. BanditProp) in comparison to using the attention mechanism (i.e. in RPRM) to model the personalised usefulness of reviews' properties. Therefore, we have demonstrated the effectiveness of two main components in the BanditProp model, namely (1) the users/items embeddings; (2) the multi-task learning component and the corresponding bandit search strategy-based model architecture, as motivated the need to capture users' behaviour with the review sorting interfaces of various platforms.

Table 8.3: Recommendation performances of BanditProp using various bandit algorithms. ↑ denotes a significant difference w.r.t. RPRM on all ranking metrics according to a paired t-test with the Tukey HSD multiple testing correction ($p < 0.05$). Similarly ∘ and •, respectively, denote significant differences using the same test w.r.t. BanditProp using the default greedy bandit algorithm and ConvBandit.

	Dataset	**Amazon**				
	Model	**P@1**	**P@5**	**R@10**	**MAP**	**NDCG@10**
–	**RPRM**	0.02238∘•	0.01270∘•	0.08657∘•	0.03784∘•	0.04966∘•
	Multi-armed Bandit Algorithms					
↑	**greedy**	0.02534•	0.01423•	0.09481•	0.04197•	0.05536•
↑	**ε-greedy**	0.02545•	0.01447•	0.09521•	0.04258∘•	0.05603∘•
↑	**DE-greedy**	0.02534•	0.01437•	0.09413∘•	0.04237•	0.05568•
↑	**UCB**	0.02484•	0.01428•	0.09315∘•	0.04156•	0.05488∘•
–	**Bayes-UCB**	0.02491•	0.01404•	0.09135∘•	0.04070∘•	0.05416∘•
–	**TS**	0.02461∘•	0.01386∘•	0.09193∘•	0.04122∘•	0.05448∘•
↑	**Mul-TS**	0.02526•	0.01423•	0.09298∘•	0.04190•	0.05473∘•
	Neural Contextual Bandit Algorithms					
↑	**LinBandit**	0.02572•	0.01476∘•	0.09553∘•	0.04285∘•	0.05603∘•
↑	**ConvBandit**	**0.02637**∘	**0.01558**∘	**0.09675**∘	**0.04386**∘	**0.05723**∘
	Dataset	**Yelp**				
	Model	**P@1**	**P@5**	**R@10**	**MAP**	**NDCG@10**
–	**RPRM**	0.01618∘•	0.01221∘•	0.07611∘•	0.02713∘•	0.04205∘•
	Multi-armed Bandit Algorithms					
↑	**greedy**	0.01760•	0.01326•	0.08144•	0.02952•	0.04486•
↑	**ε-greedy**	0.01773•	0.01325•	0.08144•	0.02936•	0.04431•
↑	**DE-greedy**	0.01773•	0.01345•	0.08146•	0.02957•	0.04490•
↑	**UCB**	0.01680∘•	0.01267∘•	0.07943∘•	0.02800∘•	0.04281∘•
–	**Bayes-UCB**	0.01656∘•	0.01252∘•	0.07859∘•	0.02790∘•	0.04227∘•
–	**TS**	0.01637∘•	0.01229∘•	0.07472∘•	0.02725∘•	0.04189∘•
↑	**Mul-TS**	0.01728•	0.01302•	0.07878∘•	0.02849∘•	0.04311∘•
	Neural Contextual Bandit Algorithms					
↑	**LinBandit**	0.01781•	0.01344•	0.08121•	0.02946•	0.04437•
↑	**ConvBandit**	**0.01825**∘	**0.01408**∘	**0.08183**∘	**0.03015**∘	**0.04539**∘

8.4.2 RQ 8.2: Effectiveness of the MAB Algorithms

To answer RQ 8.2, we investigate the effectiveness of using various multi-armed bandit (MAB) algorithms when applied to our proposed BanditProp model. We introduced three types of MAB algorithms in Section 8.2.3, namely the greedy search-based, UCB-based and TS-based approaches. Recall that in Section 8.2.3, the main difference between the MAB algorithms consists in whether a bandit algorithm uses the averaged historical rewards or a functional estimator to approximate the value of arms. The greedy search-based approaches and the UCB1 algo-

rithm rely on the aggregated historical rewards and adopt different strategies to conduct arm exploration (e.g. selecting random arms with a fixed probability ε for ε-greedy). On the other hand, the Bayes-UCB and TS-based (namely TS and Mul-TS) algorithms construct functional estimators, that are trained according to the historical rewards, to predict the values of arms.

Table 8.3 presents the performances of using these three types of bandit algorithms in our BanditProp model. Note that the algorithm denoted by 'greedy' in Table 8.3 corresponds to the default bandit greedy search strategy used in the BanditProp model in Table 8.2. First, we investigate the effectiveness of using the greedy search-based algorithms. We observe that the ε-greedy search approach is the best performing among the greedy-search approaches on the Amazon dataset. It also significantly outperforms the default greedy search on MAP and NDCG@10. On the Yelp dataset, the DE-greedy search algorithm outperforms the greedy and ε-greedy search algorithms. However, overall, we observe that the various greedy search-based approaches perform comparably with no significant differences between these approaches on several evaluation metrics.

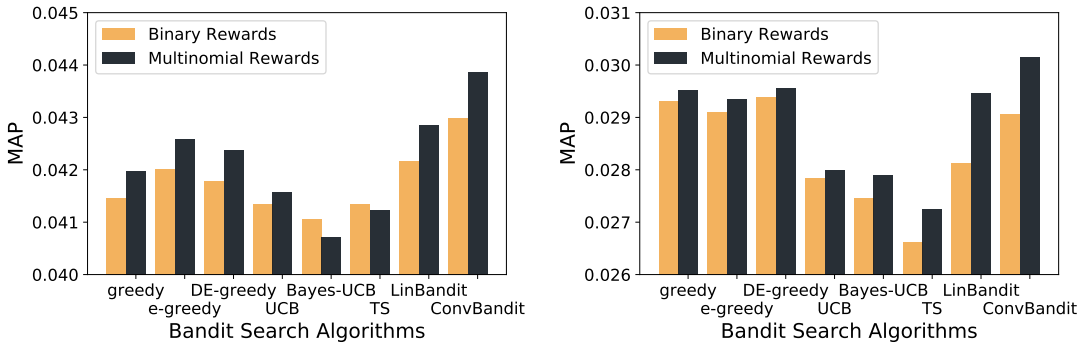
In addition, we observe that all the greedy search approaches as well as the UCB1 algorithm, which rely on the accumulated historical rewards, significantly outperform the RPRM, from Chapter 7 and using the attention mechanism, on all ranking metrics. These results indicate the benefits of leveraging the accumulated historical rewards for estimating the values of arms, so as to make effective recommendations. Next, we evaluate the performances of using the Bayes-UCB and TS-based approaches (namely TS and Mul-TS), which use functional estimators to approximate the values of arms, within the BanditProp model. The reported results in Table 8.3 show that the Bayes-UCB and TS approaches are significantly outperformed by the default greedy search algorithm on most of the ranking metrics. This observation indicates that it is difficult for the functional estimator-based algorithms (e.g. the Bayes-UCB and TS-based approaches) to effectively approximate the values of arms and improve the recommendation performances when used within BanditProp. However, by comparing the performances of TS and Mul-TS, the obtained results in Table 8.3 show that Mul-TS outperforms TS while achieving a significantly indistinguishable performance in comparison to the default greedy search algorithm on many ranking metrics. This indicates the advantage of modelling the multinomial rewards with a Dirichlet distribution instead of the beta distribution used within TS. Therefore, in answering RQ 8.2, we conclude that the multi-armed bandit algorithms that rely on the accumulated rewards to approximate the values of arms (namely the greedy-based and UCB1 approaches) are more effective than the approaches that use the functional estimators (namely the Bayes-UCB and TS-based approaches) in predicting the arms' values within our proposed BanditProp model (i.e. in modelling users' selection of leveraging different review properties

to identify useful reviews). Moreover, we also observe the benefit of using a bandit approach (namely Mul-TS) that models the multinomial rewards with the Dirichlet distribution within BanditProp.

8.4.3 RQ 8.3: Using the Contextual Information

To answer RQ 8.3, we investigate whether a neural contextual bandit algorithm that encapsulates the features extracted from the reviews (namely the state information $\mathscr{S}$ that not conveyed by the MAB algorithms), can lead to effective recommendation performances along BanditProp. In particular, we also aim to investigate the impact of using the state information to learn users' differences from their posted reviews on the BanditProp's performance. Moreover, we examine the performance of our proposed ConvBandit algorithm in comparison to the LinBandit algorithm, which is a state-of-the-art neural contextual bandit algorithm. Recall that in Section 8.2.3, ConvBandit differs from LinBandit, that used a linear operation, by using the convolutional operation to model the state information $\mathscr{S}$. Table 8.3 presents the obtained experimental results on the two used datasets.

First, we observe that BanditProp using either LinBandit or ConvBandit still significantly outperforms the RPRM model, that we introduced in Chapter 7. This validates our argument in Section 8.1 that, instead of using an attention mechanism (as in the RPRM model), it is beneficial to model the selection of using various review properties to estimate the usefulness of reviews. After that, comparing the performances of ConvBandit and Linbandit, we observe that ConvBandit significantly outperforms LinBandit on all ranking metrics across the two used datasets. Next, we compare the performances between our proposed ConvBandit approach and the multi-armed bandit algorithms. The difference between the neural contextual bandit approaches (e.g. ConvBandit and LinBandit) and the MAB algorithms is that the neural contextual bandit algorithms consider the action, state and reward instead of only the action and reward used by the MAB algorithms (as denoted in Section 8.2.3). We observe that on both datasets, ConvBandit significantly outperforms all the MAB algorithms on all of the ranking metrics. This observation shows that it is useful, for a bandit approach, to leverage the feature vectors of reviews as context to approximate the values of arms within BanditProp. In particular, as previously described in Section 8.2.3, the Bayes-UCB, TS-based, LinBandit and ConvBandit approaches all use functional estimators to approximate the values of arms. The significantly better performance achieved by ConvBandit, in comparison to both the Bayes-UCB and TS-based approaches, further demonstrates that it is useful to leverage the feature embeddings of reviews as context when estimating the arms' values to improve the BanditProp's performance. Therefore,



(a) Amazon dataset

(b) Yelp Dataset

Figure 8.2: Effectiveness comparison between using the binary rewards or the multinomial rewards on two datasets.

in answering RQ 8.3, we conclude that ConvBandit significantly outperforms the existing state-of-the-art LinBandit and Mul-TS bandit algorithms as well as various classic bandit algorithms on both the Amazon and Yelp datasets when applied to our proposed BanditProp model. In addition, the significantly better performance of ConvBandit, in comparison to other functional estimator-based bandit approaches without the context information, further supports the added value of the used context information within ConvBandit. In particular, recall that BanditProp leverages the context information, encoded users' features from their posted reviews, to model the users' distinct preferences in using various review properties. The experimental results also further examine the effectiveness of modelling users' differences to learn their selection of reviews' properties in comparison to using the attention mechanism in our proposed RPRM model (see Chapter 7).

8.5 Multinomial versus Binary Rewards

The strategy used to leverage the collected rewards after selecting the arms plays an important role in addressing a bandit problem (Agarwal et al., 2019). In Section 8.2.3, we used the multinomial rewards in various bandit algorithms. The multinomial rewards are computed according to the ratios of the correctly ranked positive-negative item pairs. Alternatively, we can also compute binary rewards by applying a conditional function to the multinomial rewards (i.e. the reward is 1 if the multinomial reward is greater than 0.5, otherwise 0). In this chapter, for the used bandit approaches, we argued that using the multinomial rewards can better capture the values of selecting different arms than the binary rewards. However, in the existing literature, the

binary rewards are still frequently used in many bandit approaches for addressing a wide range of bandit problems (Guo and Yu, 2018; Kveton et al., 2019; Bouneffouf et al., 2017). Therefore, in this section, we experimentally compare the effectiveness of using either the multinomial or binary rewards in various bandit approaches within the BanditProp model on two datasets. Figure 8.2 presents the recommendation performance differences between the use of binary or multinomial rewards for various bandit approaches within BanditProp. Note that, as an illustration, we use the MAP ranking metric when measuring the performance differences of various approaches.[4] However, we observed similar trends when using other ranking metrics.

In Figure 8.2, we observe that the multinomial rewards outperform the binary rewards when used within most bandit approaches with the exceptions of Bayes-UCB and TS on Amazon. This shows the effectiveness of using the multinomial rewards to measure the values of arms when addressing our bandit problem. Moreover, for the Bayes-UCB and TS algorithms, the binary rewards do not consistently enhance the performances of both Bayes-UCB and TS on the two used datasets (i.e. the multinomial rewards are a better choice for both approaches on Yelp). Therefore, from the obtained results, we conclude that to address our tackled bandit problem,the multinomial rewards are more effective than the binary rewards when used in various bandit approaches within the BanditProp model.

8.6 Conclusions

In this chapter, we argued that instead of using the attention mechanism as in the RPRM model introduced in Chapter 7, we can instead apply different effective techniques to estimate the personalised usefulness of reviews. In particular, we argued that the attention mechanism ignores the users' features when addressing the estimation of the users' preferences on the reviews' properties to identify useful reviews (see Section 8.1). Therefore, this chapter proposed a novel recommendation model, BanditProp (see Section 8.2), which encompasses multi-task learning and a bandit search strategy to estimate the users' preferences on the reviews' properties, thereby enhancing the recommendation performances. The experimental results in Table 8.2 showed that BanditProp, using a default greedy search algorithm, can significantly and consistently outperform one classical and six existing state-of-the-art baseline approaches over two datasets (i.e. the Yelp and Amazon datasets). In particular, by showing the effectiveness of BanditProp, in comparison to RPRM, we showed that a bandit algorithm is more effective

[4]Mul-TS is not included, since it is designed to model the multinomial rewards, which is not appropriate approach for binary rewards.

than using the attention mechanism to estimate the users' preferences over the reviews' properties. Next, we compared the effectiveness of different multi-armed bandit algorithms when applied to the BanditProp model. In Table 8.3, we showed that among the multi-armed bandit algorithms, the accumulated reward-based algorithms (namely the greedy-based and UCB1 approaches) outperformed the functional estimator-based algorithms (namely the Bayes-UCB and TS-based approaches), when applied to BanditProp. In particular, the experimental results and the analysis in Section 8.5 showed the effectiveness of modelling the multinomial rewards compared to only modelling the binary rewards within BanditProp. Next, the experimental results showed that our proposed contextual bandit algorithm (ConvBandit) significantly outperformed other bandit algorithms on the two review datasets (i.e. the Yelp and Amazon datasets) when applied to BanditProp (see Table 8.3). The effectiveness of the BanditProp model again validated our claim in the thesis statement (see Section 1.3) that by distinguishing among the personalised usefulness of reviews, a review-based recommendation model can effectively apply personalised recommendations and obtain more accurate recommendations.

Thus far, we have shown the effectiveness of our proposed BanditProp model, which instantiated our proposed review-based recommendation framework in Chapter 4. In particular, starting from Chapter 5 until Chapter 8, we have proposed various techniques to address the challenges (i.e. the availability of review attributes and the estimation of users' preferences) within such a framework. Therefore, in the next chapter, we summarise the main contributions and conclusions of this thesis and also discuss possible future directions.

Chapter 9

Conclusions and Future Work

9.1 Contributions and Conclusions

This thesis addressed the challenges of leveraging reviews and the reviews' associated properties when making ranking-based recommendations. In particular, as motivated by the users' adoption of information framework, we postulated that reviews are not equally valuable for different users. By leveraging the users' personal views on the usefulness of reviews, we argued that a recommendation model can make more accurate recommendations. Therefore, in Chapter 4, we proposed a review-based recommendation framework, which comprises four components, namely the data collection, the review property extraction, the review property encoding and the user preferences estimation. In particular, we summarised the two main challenges within such framework into (i) the *availability of review attributes* and (ii) the *estimation of users' preferences*. The *availability of review attributes* is concerned with the extraction of the reviews' properties by leveraging their corresponding attributes. This allows to tackle the problems of attribute unavailability and insufficient labels. The *estimation of users' preferences* is related to the effective usage of the extracted reviews' properties to accurately estimate the users' personal views about considering different review properties to identify useful reviews. This allows to provide good approximations of the users' item preferences. In particular, we discuss our main contributions and conclusions in addressing these two challenges as follows:

- **Conclusion 1: Effective Recommendations Using the Estimated Reviews' Sentiment:** We showed that the use of the reviews' sentiment can be a reliable surrogate of the users' explicit ratings for making effective recommendations (see Section 5.2). In particular, the review' sentiment is extracted by leveraging existing sentiment classifiers. The Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) classifiers per-

formed the best in comparison to other classifiers, including the SentiWordNet and Support Vector Machine (SVM) approaches (see Figure 5.2). Next, we evaluated the effectiveness of the predicted sentiment property of reviews by replacing the users' explicit ratings in two widely-cited recommendation approaches (namely MF and GeoSoCa) to address the Top-N ranking-based recommendation task. The experimental results in Table 5.3 showed similar effectiveness when using the reviews' sentiment property – estimated by using the deep learning-based sentiment classification approaches (i.e. the CNN and LSTM models) – to the users' explicit ratings when applied to ranking-based recommendation models.

- **Conclusion 2: Improved Rating Prediction Accuracy by Using Reviews' Sentiment through a Sentiment Attention Mechanism:** We further evaluated the estimated reviews' sentiment when used in addressing the review-based rating prediction task (see Section 5.3). In particular, we encoded the sentiment scores into our proposed review-based recommendation model, namely the SentiAttn model, through a customised sentiment attention mechanism. According to the estimated reviews' sentiment property, the sentiment attention mechanism weights the usefulness of reviews within the SentiAttn model. Compared to state-of-the-art rating prediction models, our experimental results showed that SentiAttn can significantly outperform the baseline approaches according to both the paired t-test and the Tukey HSD correction method ($p < 0.05$). In particular, to evaluate the effectiveness of our proposed sentiment attention mechanism, we compared the variants of SentiAttn (i.e. SentiAttn uses either the proposed sentiment attention mechanism (i.e. +Sent) or the global attention layer (i.e. +Glb) or neither of them (i.e. Basic)). According to the experimental results in Table 5.6, the sentiment attention mechanism outperformed the global attention mechanism with lower MAE and RMSE scores (0.8932 vs. 0.8947 (MAE) and 1.1476 vs. 1.1734 (RMSE) for +Sent vs. +Glb) as well as a significantly higher accuracy when compared to the Basic model. These results showed the effectiveness of leveraging the reviews' sentiment for addressing the review-based rating prediction task via a novel sentiment attention mechanism.

- **Conclusion 3: Accurate Helpfulness Classification of Reviews by a Novel Weakly Supervised Binary Classification Correction Method:** We introduced a novel weakly supervised binary classification correction approach (NCWS), which effectively addressed the review helpfulness classification with insufficient helpful votes of reviews (see Chapter 6). NCWS can be applied to many state-of-the-art review helpfulness classifiers, including the SVM, CNN and BERT-based classifiers, so as to improve their performances.

In particular, NWCS leverages the positive unlabelled learning algorithm as well as a novel negative confidence-based loss function to model the unlabelled instances. First, we evaluated the helpfulness classification performances of existing classification approaches. The results in Table 6.3 showed that the Support Vector Machine (SVM)-based classifiers that consider the length-based features (e.g. number of words and the number of sentences), the CNN and the BERT-based classifiers provide the best performances on the three used datasets (namely the Yelp, Amazon Kindle and Amazon Electronics datasets). In particular, the SVM classifier that considered all features performed the best. After that, we showed that NCWS can significantly improve the classification accuracy of many top-performing review helpfulness classifiers in comparison to existing classification correction approaches (i.e. SVM-P (Tang et al., 2009), C-PU (Du Plessis et al., 2015) and P-conf (Ishida et al., 2018)). In particular, in Figure 6.2, we illustrated that only NCWS and SVM-P were useful at correcting and enhancing the best performing classifiers. NCWS also outperformed SVM-P by correcting the classifiers and provided higher F1 scores (see Table 6.4).

- **Conclusion 4: Improved Rating Prediction Accuracy by Leveraging the learned Reviews' Helpfulness through a Filtering-based Mechanism:** We investigated the effectiveness of applying the estimated helpfulness of reviews in a review-based recommendation model to address the rating prediction task (Section 6.6). In particular, the helpfulness of reviews is estimated by a corrected SVM-based classifier (i.e. the SVM-classifier that leveraged all considered features). The classification correction was made by our proposed NCWS approach. We examined the performances of applying the estimated reviews' helpfulness to a widely cited review-based rating prediction model (i.e. DeepCoNN) by filtering out reviews that were predicted to be unhelpful. The results in Table 6.6 showed that by leveraging the helpful reviews identified by our proposed NCWS classifier, we can significantly enhance the performance of the DeepCoNN. These results also validate the effectiveness of encoding the reviews' helpfulness into a review-based recommendation model.

- **Conclusion 5: Improved Ranking-based Recommendation Performances by Using Various Reviews' Properties:** We showed that review properties have varying improvements when used in our proposed RPRM ranking-based recommendation model (see Chapter 7). Note that RPRM instantiated our proposed review-based recommendation framework in Chapter 4. RPRM utilises the reviews' properties to identify useful reviews and generate the resulting features for making recommendations. In Table 7.1, we showed

the experimental results on the Amazon and Yelp datasets. We observed that the reviews' age and the reviews' helpfulness estimated by a classifier (i.e. the SVM-ALL corrected by NCWS) are the two most effective review properties among the six studied review properties in identifying the usefulness of reviews and improving the recommendation effectiveness of RPRM. The other review properties showed distinct effects on different datasets. For example, the users' ratings improved the recommendation performances of RPRM on the Yelp dataset but not on the Amazon dataset. These results suggest the necessity of selectively applying suitable review properties in the recommendation model to assess the reviews' usefulness for enhanced recommendations.

- **Conclusion 6: Improved Ranking-based Recommendation Performance by Modelling the Importance of Different Review Properties:** Our proposed RPRM model in Chapter 7, leveraged an attention mechanism to model the importance of various review properties in identifying useful reviews. RPRM showed strong top-N ranking-based recommendation performances. In particular, the experimental results in Table 7.1 showed that RPRM significantly outperformed state-of-the-art review-based recommendation models on the Amazon and Yelp datasets. Inspired by the users' adoption of information framework, we proposed two loss functions and one negative sampling strategy that model the agreement on the importance of review properties between the users and items. The results in Table 7.2 showed that these two loss functions can both improve the performance of RPRM. In particular, the loss function that leveraged the calculated differences between users and items with the KL divergence measure led to the best performances. Moreover, the proposed sampling strategy can also enhance RPRM if an adequate similarity measure is applied. These results showed the effectiveness of modelling the importance of different review properties to improve the models' recommendation performances.

- **Conclusion 7: Improved Ranking-based Recommendation Performance by Modelling the Users' Behaviour for Selecting Review Properties that Identify Useful Reviews:** We showed the effectiveness of converting the selection of the properties of the useful reviews that better capture the users' preferences into a bandit problem. In particular, we introduced a novel review-based recommendation model, called BanditProp in Chapter 8, which can integrate various bandit algorithms to predict the users' preferred review properties. This model showed strong recommendation performances. Indeed, the experimental results in Table 8.2 showed that the BanditProp model, which used a default greedy search algorithm, can significantly outperform the baseline approaches, including the RPRM model. These results showed a better performance when using the

bandit algorithms than when using the attention mechanism. In addition, we compared the use of various existing bandit algorithms when integrated into BanditProp. In Table 8.3, we showed that the multi-armed bandit algorithms that use the accumulated rewards to approximate the value of arms (i.e. the greedy search-based and UCB1 approaches) typically outperform other functional estimator-based algorithms (i.e. the Bayes-UCB and TS-based algorithms). We further proposed leveraging the users' posted reviews as the contextual information within a proposed contextual bandit algorithm (called ConvBandit) to enhance the effectiveness of estimating the users' preferred review properties when making recommendations. We further experimentally demonstrated the effectiveness of ConvBandit in comparison to various existing bandit approaches (e.g. the greedy search and UCB search algorithms) in addressing our bandit problem. Indeed, the experimental results in Table 8.3 showed that ConvBandit enhanced BanditProp and outperformed other bandit algorithms in yielding the best recommendation performances.

Next, based on the experimental results from Chapters 5 to 8, we validate our thesis statement that we introduced in Section 1.3. This thesis stated that we could improve the effectiveness of a review-based recommendation approach by leveraging various reviews' associated properties. These review properties include the reviews' age, length, sentiment, rating, helpfulness as judged by users and helpfulness as estimated by a helpfulness classifier. Moreover, a review-based recommendation model can obtain more accurate recommendations by distinguishing among the users' preferences over different review properties to examine the usefulness of reviews. In the following, we discuss the corresponding experimental results and observations that validate our proposed thesis statement.

- **Claim 1:** *By inferring the sentiment property from the users' posted reviews, the recommendation model can accurately capture the users' preferences and obtain an improved recommendation effectiveness.* The experiments of the two studies in Chapter 5 validated this claim by first showing that the estimated review sentiment can be a reliable surrogate for the reviews' ratings in the two used recommendation models (i.e. MF and GeoSoCa) (see Table 5.3). Then, in Table 5.6, we also showed that our proposed review-based recommendation model (i.e. SentiAttn) when it incorporates the estimated sentiment property of reviews can significantly outperform other state-of-the-art rating prediction models, according to both the paired t-test and the Tukey HSD correction method ($p < 0.05$).

- **Claim 2:** *By leveraging the inferred helpfulness property from the users' posted reviews, a review-based recommendation model can improve the recommendation performances by*

correctly identifying the reviews deemed helpful. Our experiments in Chapter 6 validated this claim by showing that, when using the predicted helpful reviews as input, a state-of-the-art review-based recommendation approach (i.e. DeepCoNN) can obtain a significantly lower rating prediction error (see Table 6.6). The review helpfulness prediction is addressed by a corrected SVM-based classifier (i.e. the SVM-classifier that leveraged all considered features). The classification correction was made by our proposed weakly supervised classification correction approach (i.e. NCWS). The resulting classifier outperformed the leading review helpfulness classifiers in the literature (see Table 6.4).

- **Claim 3:** *The review-based recommendation approaches could attain more effective recommendation by leveraging various available review properties.* We argue that we have validated this claim in Chapter 7, where we encoded six commonly available review properties, namely the reviews' age, length, sentiment, rating, helpfulness as judged by users (helpful) and helpfulness as estimated by a helpfulness classifier (Prob_helpful), in our proposed review-based recommendation approach (i.e. RPRM). These six review properties showed distinct effectiveness in improving the recommendation performances of RPRM. In particular, the reviews' age and Prob_helpful are the two most effective review properties on our examined datasets (i.e. the Yelp and Amazon datasets).

- **Claim 4:** *By distinguishing among the users' preferences on the usefulness of reviews examined through various available review properties, the review-based recommendation model could obtain improved recommendation effectiveness.* In Chapters 7 and 8, we instantiated our proposed review-based recommendation framework with two novel review-based recommendation models (i.e. RPRM and BanditProp). They modelled the users' choices of considering different review properties when examining the usefulness of reviews by using the attention mechanism and multi-armed bandit algorithms in RPRM and BanditProp, respectively. RPRM and BanditProp both significantly outperformed the state-of-the-art review-based recommendation techniques, according to both the paired t-test and the Tukey HSD correction method ($p < 0.05$) (see Table 7.1 and 8.3). Therefore, we concluded that modelling the users' preferences of using different review properties when examining the usefulness of reviews can indeed yielding strong recommendation performances, which validates our proposed claim.

In summary, we argue that we have validated every claim of our posed thesis statement in Section 1.3 using public available review datasets. Indeed, we showed that our proposed review-based recommendation framework, which used available review properties, yielding strong rec-

ommendation performances. The used six review properties (i.e. the reviews' age, length, sentiment, ratings, helpfulness as judged by users and helpfulness as estimated by a helpfulness classifier) can enhance the recommenders' performances with varying levels of effectiveness. In particular, our proposed review-based recommenders obtained enhanced performances by distinguishing the users' preferences on the usefulness of reviews examined by different review properties. Next, we describe some future research directions for review-based recommendation models in Section 9.2.

9.2 Directions for Future Work

In this section, we discuss possible future directions that could benefit the performances of review-based recommendation techniques.

Explore techniques to encode the reviews' properties as additional information: In Chapters 5 and 6, we proposed to integrate the reviews' sentiment and helpfulness properties via a dot-product attention mechanism and a filtering technique, respectively. However, recently, many different types of feature integration techniques have been shown to be useful in the recommendation literature. For example, these techniques include using kernel functions to model the temporal effects of the users' historical interactions (Wang et al., 2020a); feature encoding with an autoencoder technique (Shen et al., 2020; Karamanolakis et al., 2018; Wu et al., 2020; Chen and de Rijke, 2018); or the one-hot embedding technique to encode the categorical features (Kang et al., 2021). Therefore, it would be interesting to replace our used attention mechanism and filtering techniques for encoding the reviews' properties with other feature encoding techniques to further enhance the review-based recommendation models.

Modelling additional review properties to enrich the knowledge of reviews as per different reviews' characteristics: In this thesis, we have considered six commonly available review properties to model the users' preferences on the usefulness of reviews. These review properties include the reviews' age, length, sentiment, rating, helpfulness as judged by the users and helpfulness as predicted by a review helpfulness classifier to enhance the recommendation performance. Recall from the discussion in Section 4.4 that we proposed a taxonomy of review properties, which includes the lexical, sentiment, helpfulness and event-based classes. However, the review properties are not limited to these six properties. Some other possible review properties are the part-of-speech of the review text, the review location, and the review text's writing style. Therefore, it would be interesting to investigate the impact of considering additional review properties on the performances of review-based recommendation approaches.

Levearging state-of-the-art text modelling techniques to enhance the performances of the proposed review-based recommendation approaches: In this thesis, we leveraged the word embedding techniques (i.e. the GloVe embedding (Pennington et al., 2014) or the pre-trained BERT model (Devlin et al., 2019)) to represent the reviews' texts to model the users' preferences and items' features within the reviews' texts. However, many natural language modelling techniques have recently outperformed the BERT model in addressing various downstream tasks (e.g. the text classification (Adhikari et al., 2019) and translation (Yang et al., 2020) tasks). These techniques include the XLNet model (Yang et al., 2019), the RoBERTa model (Liu et al., 2019d), which is an optimised BERT pre-training technique and the BertGCN model (Lin et al., 2021), which combined the BERT and GCN techniques. Therefore, it is worth to investigate the effectiveness of applying different language modelling techniques to model the reviews' texts and improve the review-based recommenders' performances.

9.3 Concluding Remarks

This thesis has addressed a challenging task: the review-based recommendation task. In particular, this thesis contributed to the progress of the review-based recommendation task by proposing effective techniques based on the use of the reviews' associated properties. However, in Section 9.2, we identified a number of interesting challenges that still remain open in this area. This work provides a solid motivation and the groundwork for exploring these further research directions in the future. We remain of the opinion that using the reviews' associated properties to leverage the available reviews effectively will continue to benefit the future development of review-based recommendation approaches.

Bibliography

Adhikari, A., Ram, A., Tang, R., and Lin, J. (2019). DocBERT: BERT for document classification. *CoRR*, abs/1904.08398.

Adomavicius, G. and Tuzhilin, A. (2005). 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(6):734–749.

Agarwal, R., Liang, C., Schuurmans, D., and Norouzi, M. (2019). Learning to generalize from sparse and underspecified rewards. In *Proc. of ICLR*.

Aghakhani, N., Oh, O., Gregg, D. G., and Karimi, J. (2020). Online review consistency matters: An elaboration likelihood model perspective. *Information Systems Frontiers*, pages 1–15.

Almahairi, A., Kastner, K., Cho, K., and Courville, A. (2015). Learning distributed representations from reviews for collaborative filtering. In *Proc. of RecSys*.

Antelmi, A. (2019). Towards an exhaustive framework for online social networks user behaviour modelling. In *Proc. of UMAP*.

Asghar, N. (2016). Yelp dataset challenge: Review rating prediction. *CoRR*, abs/1605.05362.

Auer, P., Cesa-Bianchi, N., and Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2).

Baccianella, S., Esuli, A., and Sebastiani, F. (2010). SentiWordNet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining. In *Proc. of LREC*.

Bahdanau, D., Cho, K. H., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In *Proc. of ICLR*.

Bai, Y., Li, Y., and Wang, L. (2021). A joint summarization and pre-trained model for review-based recommendation. *Information*, 12(6):223.

Baltrunas, L. and Ricci, F. (2009). Item weighting techniques for collaborative filtering. In *Knowledge Discovery Enhanced with Semantic and Social Information*, pages 109–126. Springer.

Bao, H., Niu, G., and Sugiyama, M. (2018). Classification from pairwise similarity and unlabeled data. In *Proc. of ICML*.

Baral, R., Zhu, X., Iyengar, S., and Li, T. (2018). Reel: Review aware explanation of location recommendation. In *Proc. of UMAP*.

Bartels, R. H. and Golub, G. H. (1969). The simplex method of linear programming using lu decomposition. *Communications of the ACM*, 12(5):266–268.

Bauman, K., Liu, B., and Tuzhilin, A. (2017). Aspect based recommendations: Recommending items with the most valuable aspects based on user reviews. In *Proc. of SIGKDD*.

Baziotis, C., Pelekis, N., and Doulkeridis, C. (2017). Datastories at SemEval-2017 task 4: Deep LSTM with attention for message-level and topic-based sentiment analysis. In *Proc. of SemEval*.

Beheshti, A., Yakhchi, S., Mousaeirad, S., Ghafari, S. M., Goluguri, S. R., and Edrisi, M. A. (2020). Towards cognitive recommender systems. *Algorithms*, 13(8):176.

Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828.

Bengio, Y., Simard, P., and Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166.

Bharadhwaj, H., Park, H., and Lim, B. Y. (2018). Recgan: recurrent generative adversarial networks for recommendation systems. In *Proc. of RecSys*.

Billsus, D. and Pazzani, M. J. (2000). User modeling for adaptive news access. *User modeling and user-adapted interaction*, 10:147–180.

Bird, S. (2006). Nltk: the natural language toolkit. In *Proc. of COLING/ACL*.

BL, V. (2019). Typicality-based collaborative filtering for book recommendation. *Expert Systems*, 36.

Blanchard, G., Lee, G., and Scott, C. (2010). Semi-supervised novelty detection. *Journal of Machine Learning Research*, 11:2973–3009.

Boag, S. (2017). *Conscious, Preconscious, and Unconscious*, pages 1–8. Springer International Publishing.

Bobadilla, J., Ortega, F., Hernando, A., and Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-based systems*, 46:109–132.

Bouneffouf, D., Rish, I., Cecchi, G. A., and Féraud, R. (2017). Context attentive bandits: Contextual bandit with restricted context. In *Proc. of IJCAI*.

Breese, J. S., Heckerman, D., and Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. In *Proc. of UAI*.

Broderick, T., Boyd, N., Wibisono, A., Wilson, A. C., and Jordan, M. I. (2013). Streaming variational bayes. In *Proc. of NeurIPS*.

Budhi, G. S., Chiong, R., Pranata, I., and Hu, Z. (2017). Predicting rating polarity through automatic classification of review texts. In *Proc. of ICBDA*.

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12(4):331–370.

Cañamares, R., Redondo, M., and Castells, P. (2019). Multi-armed recommender system bandit ensembles. In *Proc. of RecSys*.

Catherine, R. and Cohen, W. (2016). Personalized recommendations using knowledge graphs: A probabilistic logic programming approach. In *Proc. of RecSys*.

Chan, H. P., Chen, W., and King, I. (2020). A unified dual-view model for review summarization and sentiment classification with inconsistency loss. In *Proc. of SIGIR*.

Chan, Y. Y. and Ngai, E. W. (2011). Conceptualising electronic word of mouth activity. *Marketing Intelligence & Planning*.

Chang, C.-C. and Lin, C.-J. (2011). Libsvm: a library for support vector machines. *ACM transactions on intelligent systems and technology*, 2(3).

Chen, C., Yang, Y., Zhou, J., Li, X., and Bao, F. (2018a). Cross-domain review helpfulness prediction based on convolutional neural networks with auxiliary domain discriminators. In *Proc. of NAACL*.

Chen, C., Zhang, M., Liu, Y., and Ma, S. (2018b). Neural attentional rating regression with review-level explanations. In *Proc. of WWW*.

Chen, C., Zhao, P., Li, L., Zhou, J., Li, X., and Qiu, M. (2017a). Locally connected deep learning framework for industrial-scale recommender systems. In *Proc. of WWW*.

Chen, J., Zhang, C., and Niu, Z. (2016). Identifying helpful online reviews with word embedding features. In *Proc. of KSEM*.

Chen, J., Zhang, H., He, X., Nie, L., Liu, W., and Chua, T.-S. (2017b). Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In *Proc. of SIGIR*.

Chen, L., Chen, G., and Wang, F. (2015). Recommender systems based on user reviews: the state of the art. *User Modeling and User-Adapted Interaction*, 25(2):99–154.

Chen, L., Liu, Y., Zheng, Z., and Yu, P. (2018c). Heterogeneous neural attentive factorization machine for rating prediction. In *Proc. of CIKM*.

Chen, P. and Li, J. (2019). A recurrent model with self-attention for product repurchase recommendation. In *Proc. of ICMAI*.

Chen, X., Chen, H., Xu, H., Zhang, Y., Cao, Y., Qin, Z., and Zha, H. (2019). Personalized fashion recommendation with visual explanations based on multimodal attention network: Towards visually explainable recommendation. In *Proc. of SIGIR*.

Chen, X., Xu, H., Zhang, Y., Tang, J., Cao, Y., Qin, Z., and Zha, H. (2018d). Sequential recommendation with user memory networks. In *Proc. of WSDM*.

Chen, Y. and de Rijke, M. (2018). A collective variational autoencoder for top-n recommendation with side information. In *Proc. of DLRS*.

Chen, Y., Wang, Y., Zhao, X., Yin, H., Markov, I., and Rijke, M. D. (2020). Local variational feature-based similarity models for recommending top-n new items. *ACM Transactions on Information Systems*, 38(2):1–33.

Cheng, C., Xia, F., Zhang, T., King, I., and Lyu, M. R. (2014). Gradient boosting factorization machines. In *Proc. of RecSys*.

Cheng, H.-T., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhye, H., Anderson, G., Corrado, G., Chai, W., Ispir, M., et al. (2016). Wide & deep learning for recommender systems. In *Proc. of DLRS*.

Cheng, Z., Ding, Y., He, X., Zhu, L., Song, X., and Kankanhalli, M. S. (2018a). Aˆ 3ncf: An adaptive aspect attention model for rating prediction. In *Proc. of IJCAI*.

Cheng, Z., Ding, Y., Zhu, L., and Kankanhalli, M. (2018b). Aspect-aware latent factor model: Rating prediction with ratings and reviews. In *Proc. of WWW*.

Cheung, C. M.-Y., Sia, C.-L., and Kuan, K. K. (2012). Is this review believable? a study of factors affecting the credibility of online consumer reviews from an elm perspective. *Journal of the Association for Information Systems*, 13(8).

Chin, J. Y., Zhao, K., Joty, S., and Cong, G. (2018). ANR: Aspect-based neural recommender. In *Proc. of CIKM*.

Cho, K., van Merrienboer, B., Gülçehre, Ç., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proc. of EMNLP*.

Chu, Y., Huang, F., Wang, H., Li, G., and Song, X. (2017). Short-term recommendation with recurrent neural networks. In *Proc. of ICMA*.

Cicirello, V. A. and Smith, S. F. (2005). The max *K*-armed bandit: A new model of exploration applied to search heuristic selection. In *Proc.of AAAI*.

Cilimkovic, M. (2015). Neural networks and back propagation algorithm. *Institute of Technology Blanchardstown*, 15:1–12.

Claesen, M., Davis, J., De Smet, F., and De Moor, B. (2015). Assessing binary classifiers using only positive and unlabeled data. *CoRR*, abs/1504.06837.

Cosley, D., Lam, S. K., Albert, I., Konstan, J. A., and Riedl, J. (2003). Is seeing believing?: how recommender system interfaces affect users' opinions. In *Proc. of SIGCHI*.

Covington, P., Adams, J., and Sargin, E. (2016). Deep neural networks for youtube recommendations. In *Proc. of RecSys*.

Craswell, N., Zoeter, O., Taylor, M., and Ramsey, B. (2008). An experimental comparison of click position-bias models. In *Proc. of WSDM*.

Cremonesi, P., Koren, Y., and Turrin, R. (2010). Performance of recommender algorithms on top-n recommendation tasks. In *Proc. of RecSys*.

Dang, C. N., Moreno-García, M. N., and Prieta, F. D. l. (2021). An approach to integrating sentiment analysis into recommender systems. *Sensors*, 21(16):5666.

Davagdorj, K., Park, K. H., and Ryu, K. H. (2020). A collaborative filtering recommendation system for rating prediction. In *Proc. of IIHMSP*.

Da'u, A. and Salim, N. (2019). Sentiment-aware deep recommender system with neural attention networks. *IEEE Access*, 7:45472–45484.

Deng, L. and Yu, D. (2014). Deep learning: methods and applications. *Foundations and trends in signal processing*, 7(3–4):197–387.

Deshpande, M. and Karypis, G. (2004). Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems*, 22:143–177.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2019). BERT: pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL-HLT*.

Devooght, R. and Bersini, H. (2017). Long and short-term recommendations with recurrent neural networks. In *Proc. of UMAP*.

Diaz, G. O. and Ng, V. (2018). Modeling and prediction of online product review helpfulness: A survey. In *Proc. of ACL*.

Do, M.-P. T., Nguyen, D., and Nguyen, L. (2010). Model-based approach for collaborative filtering. In *Proc. of ICPS*.

Dogo, E., Afolabi, O., Nwulu, N., Twala, B., and Aigbavboa, C. (2018). A comparative analysis of gradient descent-based optimization algorithms on convolutional neural networks. In *Proc. of CTEMS*.

Dong, X., Ni, J., Cheng, W., Chen, Z., Zong, B., Song, D., Liu, Y., Chen, H., and de Melo, G. (2020). Asymmetrical hierarchical networks with attentive interactions for interpretable review-based recommendation. In *Proc. of AAAI*.

dos Santos Mignon, A. and de Azevedo da Rocha, R. L. (2017). An adaptive implementation of ε-greedy in reinforcement learning. In *Proc. of ANT*.

Du, Y., Sutton-Charani, N., Ranwez, S., and Ranwez, V. (2021). EBCR: Empirical bayes concordance ratio method to improve similarity measurement in memory-based collaborative filtering. *Plos one*, 16.

Du Plessis, M., Niu, G., and Sugiyama, M. (2015). Convex formulation for learning from positive and unlabeled data. In *Proc. of ICML*.

Dziugaite, G. K. and Roy, D. M. (2015). Neural network matrix factorization. *CoRR*, abs/1511.06443.

Ekstrand, M. D., Riedl, J. T., and Konstan, J. A. (2011). *Collaborative filtering recommender systems*. Now Publishers Incorporated.

Elman, J. L. (1990). Finding structure in time. *Cognitive science*, 14(2):179–211.

Fan, W., Ma, Y., Li, Q., He, Y., Zhao, E., Tang, J., and Yin, D. (2019). Graph neural networks for social recommendation. In *Proc. of WWW*.

Fang, X. and Zhan, J. (2015). Sentiment analysis using product review data. *Journal of Big Data*, 2(1).

Filieri, R. and McLeay, F. (2014). E-WOM and accommodation: An analysis of the factors that influence travelers' adoption of information from online reviews. *Journal of Travel Research*, 53(1).

Fu, M., Qu, H., Moges, D., and Lu, L. (2018). Attention based collaborative filtering. *Neurocomputing*, 311:88–98.

Fu, W., Peng, Z., Wang, S., Xu, Y., and Li, J. (2019). Deeply fusing reviews and contents for cold start users in cross-domain recommendation systems. In *Proc. of AAAI*.

Gagniuc, P. A. (2017). *Markov chains: from theory to implementation and experimentation*. John Wiley & Sons.

Gao, J., Lin, Y., Wang, Y., Wang, X., Yang, Z., He, Y., and Chu, X. (2020). Set-sequence-graph: A multi-view approach towards exploiting reviews for recommendation. In *Proc. of CIKM*.

Gao, Z., Feng, A., Song, X., and Wu, X. (2019). Target-dependent sentiment classification with bert. *IEEE Access*, 7:154290–154299.

Garcin, F., Dimitrakakis, C., and Faltings, B. (2013). Personalized news recommendation with context trees. In *Proc. of RecSys*.

Gers, F. A., Schmidhuber, J., and Cummins, F. (2000). Learning to forget: Continual prediction with lstm. *Neural Computation*, 12(10):2451–2471.

Gong, S., Ye, H., and Tan, H. (2009). Combining memory-based and model-based collaborative filtering in recommender system. In *Proc. of PACCS*.

González-Carvajal, S. and Garrido-Merchán, E. C. (2020). Comparing BERT against traditional machine learning text classification. *CoRR*, abs/2005.13012.

Guan, X., Cheng, Z., He, X., Zhang, Y., Zhu, Z., Peng, Q., and Chua, T.-S. (2019). Attentive aspect modeling for review-aware recommendation. *ACM Transactions on Information Systems*, 37(3):28.

Guan, Z., Chen, L., Zhao, W., Zheng, Y., Tan, S., and Cai, D. (2016). Weakly-supervised deep learning for customer review sentiment classification. In *Proc. of IJCAI*.

Guerini, M., Gatti, L., and Turchi, M. (2013). Sentiment analysis: How to derive prior polarities from SentiWordNet. In *Proc. of EMNLP*.

Guha, S., Rastogi, R., and Shim, K. (2000). Rock: A robust clustering algorithm for categorical attributes. *Information systems*, 25(5):345–366.

Gunawardana, A. and Shani, G. (2015). Evaluating recommender systems. In *Recommender systems handbook*, pages 265–308. Springer.

Guo, D. and Yu, A. J. (2018). Why so gloomy? A bayesian explanation of human pessimism bias in the multi-armed bandit task. In *Proc. of NeurIPS*.

Guo, H., Tang, R., Ye, Y., Li, Z., and He, X. (2017a). DeepFM: A factorization-machine based neural network for ctr prediction. In *Proc. of IJCAI*.

Guo, Q., Sun, Z., Zhang, J., Chen, Q., and Theng, Y.-L. (2017b). Aspect-aware point-of-interest recommendation with geo-social influence. In *Proc. of UMAP*.

Guo, Q., Sun, Z., Zhang, J., and Theng, Y.-L. (2020). An attentional recurrent neural network for personalized next location recommendation. In *Proc. of AAAI*.

Gurini, D. F., Gasparetti, F., Micarelli, A., and Sansonetti, G. (2018). Temporal people-to-people recommendation on social networks with sentiment-based matrix factorization. *Future Generation Computer Systems*, 78(P1).

Hahn, A., Judd, C. M., Hirsh, H. K., and Blair, I. V. (2014). Awareness of implicit attitudes. *Journal of Experimental Psychology: General*, 143(3):1369.

Han, J., Pei, J., and Kamber, M. (2011). *Data Mining: Concepts and Techniques*. Morgan Kaufmann.

He, J., Li, X., Liao, L., Song, D., and Cheung, W. K. (2016). Inferring a personalized next point-of-interest recommendation model with latent behavior patterns. In *Proc. of AAAI*.

He, R. and McAuley, J. (2016). Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proc. of WWW*.

He, X., Chen, T., Kan, M.-Y., and Chen, X. (2015). Trirank: Review-aware explainable recommendation by modeling aspects. In *Proc. of CIKM*.

He, X., Du, X., Wang, X., Tian, F., Tang, J., and Chua, T.-S. (2018). Outer product-based neural collaborative filtering. In *Proc. of IJCAI*.

He, X., Liao, L., Zhang, H., Nie, L., Hu, X., and Chua, T.-S. (2017). Neural collaborative filtering. In *Proc. of WWW*.

Hidasi, B., Karatzoglou, A., Baltrunas, L., and Tikk, D. (2015). Session-based recommendations with recurrent neural networks. In *Proc. of ICLR*.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Hong, H., Xu, D., Wang, G. A., and Fan, W. (2017). Understanding the determinants of online review helpfulness: A meta-analytic investigation. *Decision Support Systems*, 102:1–11.

Hong, J.-W., Hong, A. J., and Kim, S. R. (2021). Exploring implicit and explicit attitudes of employees' authentic organizational loyalty. *Frontiers in Psychology*, 12.

Hu, L., Sun, A., and Liu, Y. (2014). Your neighbors affect your ratings: on geographical neighborhood influence to rating prediction. In *Proc. of SIGIR*.

Hu, R. and Pu, P. (2013). Exploring relations between personality and user rating behaviors. In *Proc. of Workshop on Emotions and Personality in Personalized Services (EMPIRE at UMAP)*.

Huang, A. H., Chen, K., Yen, D. C., and Tran, T. P. (2015). A study of factors that contribute to online review helpfulness. *Computers in Human Behavior*, 48:17–27.

Huang, J., Meng, Y., Guo, F., Ji, H., and Han, J. (2020). Weakly-supervised aspect-based sentiment analysis via joint aspect-sentiment topic embedding. In *Proc. of EMNLP*.

Hutto, C. and Gilbert, E. (2014). Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proc. of AAAI*.

Hyun, D., Park, C., Yang, M.-C., Song, I., Lee, J.-T., and Yu, H. (2018). Review sentiment-guided scalable deep recommender system. In *Proc. of SIGIR*.

Ishida, T., Niu, G., and Sugiyama, M. (2018). Binary classification from positive-confidence data. In *Proc. of NeurIPS*.

Jain, S., White, M., and Radivojac, P. (2017). Recovering true classifier performance in positive-unlabeled learning. In *Proc. of AAAI*.

Järvelin, K. and Kekäläinen, J. (2002). Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems*, 20(4):422–446.

Jiang, L., Cheng, Y., Yang, L., Li, J., Yan, H., and Wang, X. (2019a). A trust-based collaborative filtering algorithm for e-commerce recommendation system. *Journal of Ambient Intelligence and Humanized Computing*, 10(8):3023–3034.

Jiang, Y., Hu, C., Xiao, T., Zhang, C., and Zhu, J. (2019b). Improved differentiable architecture search for language modeling and named entity recognition. In *Proc. of EMNLP*.

Jin, L., Chen, Y., Wang, T., Hui, P., and Vasilakos, A. V. (2013). Understanding user behavior in online social networks: a survey. *IEEE Communications Magazine*, 51(9).

Jin, W., Ho, H. H., and Srihari, R. K. (2009). OpinionMiner: a novel machine learning system for web opinion mining and extraction. In *Proc. of SIGKDD*.

Joachims, T. (1996). A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. Technical report, Carnegie-mellon univ pittsburgh pa dept of computer science.

Kanagal, B., Ahmed, A., Pandey, S., Josifovski, V., Yuan, J., and Pueyo, L. G. (2012). Supercharging recommender systems using taxonomies for learning user purchase behavior. *Proc. of VLDB Endow.*, 5(10):956–967.

Kang, H., Yoo, S. J., and Han, D. (2012). Senti-lexicon and improved naïve bayes algorithms for sentiment analysis of restaurant reviews. *Expert Systems with Applications*, 39(5):6000–6010.

Kang, W.-C., Cheng, D. Z., Yao, T., Yi, X., Chen, T., Hong, L., and Chi, E. H. (2021). Learning to embed categorical features without embedding tables for recommendation. In *Proc. of SIGKDD*.

Kang, W.-C. and McAuley, J. (2018). Self-attentive sequential recommendation. In *Proc. of ICDM*.

Karamanolakis, G., Cherian, K. R., Narayan, A. R., Yuan, J., Tang, D., and Jebara, T. (2018). Item recommendation with variational autoencoders and heterogeneous priors. In *Proc. of DLRS*.

Karypis, G., Han, E.-H., and Kumar, V. (1999). Chameleon: Hierarchical clustering using dynamic modeling. *Computer*, 32(8):68–75.

Kaufman, L. and Rousseeuw, P. J. (2009). *Finding groups in data: an introduction to cluster analysis*, volume 344. John Wiley & Sons.

Kaufmann, E., Cappé, O., and Garivier, A. (2012). On bayesian upper confidence bounds for bandit problems. In *Proc. of AISTATS*.

Kharitonov, E. (2016). *Using interaction data for improving the offline and online evaluation of search engines*. PhD thesis, University of Glasgow.

Kim, D., Park, C., Oh, J., Lee, S., and Yu, H. (2016). Convolutional matrix factorization for document context-aware recommendation. In *Proc. of RecSys*.

Kim, S.-M., Pantel, P., Chklovski, T., and Pennacchiotti, M. (2006). Automatically assessing review helpfulness. In *Proc. of EMNLP*.

Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proc. of EMNLP*.

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In *Proc. of ICLR*.

Kiritchenko, S., Zhu, X., and Mohammad, S. M. (2014). Sentiment analysis of short informal texts. *Journal of Artificial Intelligence Research*, 50.

Knijnenburg, B. P., Willemsen, M. C., Gantner, Z., Soncu, H., and Newell, C. (2012). Explaining the user experience of recommender systems. *User Modeling and User-Adapted Interaction*, 22(4):441–504.

Koppel, M. and Schler, J. (2006). The importance of neutral examples for learning sentiment. *Computational Intelligence*, 22(2).

Koren, Y. (2009). Collaborative filtering with temporal dynamics. In *Proc. of SIGKDD*.

Koren, Y., Bell, R. M., and Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37.

Korfiatis, N., García-Bariocanal, E., and Sánchez-Alonso, S. (2012). Evaluating content quality and helpfulness of online product reviews: The interplay of review helpfulness vs. review content. *Electronic Commerce Research and Applications*, 11(3):205–217.

Kouadri, W. M., Ouziri, M., Benbernou, S., Echihabi, K., Palpanas, T., and Amor, I. B. (2020). Quality of sentiment analysis tools: The reasons of inconsistency. *Proc. of the VLDB Endowment*, 14(4):668–681.

Krishnamoorthy, S. (2015). Linguistic features for review helpfulness prediction. *Expert Systems with Applications*, 42(7).

Kuleshov, V. and Precup, D. (2000). Algorithms for multi-armed bandit problems. *Machine Learning Research*.

Kumar, S., De, K., and Roy, P. P. (2020). Movie recommendation system using sentiment analysis from microblogging data. *IEEE Transactions on Computational Social Systems*, 7(4):915–923.

Kveton, B., Szepesvári, C., Vaswani, S., Wen, Z., Lattimore, T., and Ghavamzadeh, M. (2019). Garbage in, reward out: Bootstrapping exploration in multi-armed bandits. In *Proc. of ICML*.

Lak, P. and Turetken, O. (2014). Star ratings versus sentiment analysis–a comparison of explicit and implicit measures of opinions. In *Proc. of HICSS*.

Lee, J. and Hsiang, J. (2019). PatentBERT: Patent classification with fine-tuning a pre-trained BERT model. *CoRR*, abs/1906.02124.

Lee, P., Hu, Y., and Lu, K. (2018a). Assessing the helpfulness of online hotel reviews: A classification-based approach. *Telematics and Informatics*, 35(2).

Lee, P.-J., Hu, Y.-H., and Lu, K.-T. (2018b). Assessing the helpfulness of online hotel reviews: A classification-based approach. *Telematics and Informatics*, 35(2):436–445.

Lei, X., Qian, X., and Zhao, G. (2016). Rating prediction based on social sentiment from textual reviews. *IEEE Transactions on Multimedia*, 18(9).

Li, C., Niu, X., Luo, X., Chen, Z., and Quan, C. (2019a). A review-driven neural model for sequential recommendation. In *Proc. of IJCAI*.

Li, C., Quan, C., Peng, L., Qi, Y., Deng, Y., and Wu, L. (2019b). A capsule network for recommendation and explaining what you like and dislike. In *Proc. of SIGIR*.

Li, J., Wang, Y., and McAuley, J. (2020). Time interval aware self-attention for sequential recommendation. In *Proc. of WSDM*.

Li, L., Chu, W., Langford, J., and Schapire, R. E. (2010). A contextual-bandit approach to personalized news article recommendation. In *Proc. of WWW*.

Li, L., Chu, W., Langford, J., and Wang, X. (2011). Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proc. of WSDM*.

Li, P., Wang, Z., Ren, Z., Bing, L., and Lam, W. (2017). Neural rating regression with abstractive tips generation for recommendation. In *Proc. of SIGIR*.

Li, Q., Li, X., Lee, B., and Kim, J. (2021a). A hybrid cnn-based review helpfulness filtering model for improving e-commerce recommendation service. *Applied Sciences*, 11(18):8613.

Li, Y., Deng, X., Ba, S., Myers, W. R., Brenneman, W. A., Lange, S. J., Zink, R., and Jin, R. (2021b). Cluster-based data filtering for manufacturing big data systems. *Journal of Quality Technology*, pages 1–13.

Li, Y.-M. and Li, T.-Y. (2013). Deriving market intelligence from microblogs. *Decision Support Systems*, 55(1):206–217.

Lian, D., Ge, Y., Zhang, F., Yuan, N. J., Xie, X., Zhou, T., and Rui, Y. (2018a). Scalable content-aware collaborative filtering for location recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 30(6).

Lian, D., Zhao, C., Xie, X., Sun, G., Chen, E., and Rui, Y. (2014). GeoMF: joint geographical modeling and matrix factorization for point-of-interest recommendation. In *Proc. of SIGKDD*.

Lian, J., Zhou, X., Zhang, F., Chen, Z., Xie, X., and Sun, G. (2018b). xDeepFM: Combining explicit and implicit feature interactions for recommender systems. In *Proc. of KDD*.

Lin, Y., Meng, Y., Sun, X., Han, Q., Kuang, K., Li, J., and Wu, F. (2021). BertGCN: Transductive text classification by combining GCN and BERT. *CoRR*, abs/2105.05727.

Lin, Z., Yang, W., Zhang, Y., Wang, H., and Tang, Y. (2018). MulAttenRec: A multi-level attention-based model for recommendation. In *Proc. of ICONIP*.

Lippmann, R. (1987). An introduction to computing with neural nets. *IEEE Assp magazine*, 4(2):4–22.

Liu, D., Li, J., Du, B., Chang, J., and Gao, R. (2019a). Daml: Dual attention mutual learning between ratings and reviews for item recommendation. In *Proc. of SIGKDD*.

Liu, H., Simonyan, K., and Yang, Y. (2019b). DARTS: differentiable architecture search. In *Proc. of ICLR*.

Liu, H., Wang, W., Xu, H., Peng, Q., and Jiao, P. (2020a). Neural unified review recommendation with cross attention. In *Proc. of SIGIR*.

Liu, H., Wang, Y., Peng, Q., Wu, F., Gan, L., Pan, L., and Jiao, P. (2020b). Hybrid neural recommendation with joint deep representation learning of ratings and reviews. *Neurocomputing*, 374:77–85.

Liu, H., Wu, F., Wang, W., Wang, X., Jiao, P., Wu, C., and Xie, X. (2019c). NRPA: neural recommendation with personalized attention. In *Proc. of SIGIR*.

Liu, J., Cao, Y., Lin, C.-Y., Huang, Y., and Zhou, M. (2007). Low-quality product review detection in opinion summarization. In *Proc. of EMNLP*.

Liu, J., Wu, C., and Wang, J. (2018a). Gated recurrent units based neural network for time heterogeneous feedback recommendation. *Information Sciences*, 423:50–65.

Liu, N. N., Meng, X., Liu, C., and Yang, Q. (2011). Wisdom of the better few: cold start recommendation via representative based rating elicitation. In *Proc. of RecSys*.

Liu, Q., Wu, S., Wang, D., Li, Z., and Wang, L. (2016). Context-aware sequential recommendation. In *Proc. of ICDM*.

Liu, Q., Zeng, Y., Mokhosi, R., and Zhang, H. (2018b). STAMP: short-term attention/memory priority model for session-based recommendation. In *Proc. of SIGKDD*.

Liu, S., Li, F., Li, F., Cheng, X., and Shen, H. (2013). Adaptive co-training svm for sentiment classification on tweets. In *Proc. of CIKM*.

Liu, S., Ounis, I., Macdonald, C., and Meng, Z. (2020c). A heterogeneous graph neural model for cold-start recommendation. In *Proc. of SIGIR*.

Liu, T., Fang, S., Zhao, Y., Wang, P., and Zhang, J. (2015). Implementation of training convolutional neural networks. *CoRR*, abs/1506.01195.

Liu, Y., Huang, X., An, A., and Yu, X. (2008). Modeling and predicting the helpfulness of online reviews. In *Proc. of ICDM*.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019d). RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Liu, Y., Pham, T.-A. N., Cong, G., and Yuan, Q. (2017). An experimental evaluation of point-of-interest recommendation in location-based social networks. In *Proc. of VLDB*.

Liu, Y.-Y., Yang, B., Pei, H.-B., and Huang, J. (2020d). Neural explainable recommender model based on attributes and reviews. *Journal of Computer Science and Technology*, 35(6):1446–1460.

Liu, Z., Yuan, B., and Ma, Y. (2021). A multi-task dual attention deep recommendation model using ratings and review helpfulness. *Applied Intelligence*, pages 1–13.

López Barbosa, R. R., Sánchez-Alonso, S., and Sicilia-Urban, M. A. (2015). Evaluating hotels rating prediction based on sentiment analysis services. *Aslib Journal of Information Management*, 67(4).

Lu, Y., Tsaparas, P., Ntoulas, A., and Polanyi, L. (2010). Exploiting social context for review quality prediction. In *Proc. of WWW*.

Luong, T., Pham, H., and Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. In *Proc. of EMNLP*.

Lv, Y., Zheng, Y., Wei, F., Wang, C., and Wang, C. (2020). AICF: Attention-based item collaborative filtering. *Advanced Engineering Informatics*, 44:101090.

Ma, H., Yang, H., Lyu, M. R., and King, I. (2008). SoRec: social recommendation using probabilistic matrix factorization. In *Proc. of CIKM*.

MacQueen, J. et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proc. of Berkeley symposium on mathematical statistics and probability*.

Malik, M. and Hussain, A. (2018). An analysis of review content and reviewer variables that contribute to review helpfulness. *Information Processing & Management*, 54(1).

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J. R., Bethard, S., and McClosky, D. (2014). The stanford corenlp natural language processing toolkit. In *Proc. of ACL*.

Manotumruksa, J. (2021). Effective neural architectures for context-aware venue recommendation. In *ACM SIGIR Forum*, volume 53.

Manotumruksa, J., Macdonald, C., and Ounis, I. (2016a). Modelling user preferences using word embeddings for context-aware venue recommendation. In *Neu-IR: The SIGIR 2016 Workshop on Neural Information Retrieval*.

Manotumruksa, J., Macdonald, C., and Ounis, I. (2016b). Regularising factorised models for venue recommendation using friends and their comments. In *Proc. of CIKM*.

Manotumruksa, J., Macdonald, C., and Ounis, I. (2017). A personalised ranking framework with multiple sampling criteria for venue recommendation. In *Proc. of CIKM*.

Manotumruksa, J., Macdonald, C., and Ounis, I. (2018). A contextual attention recurrent architecture for context-aware venue recommendation. In *Proc. of SIGIR*.

Manotumruksa, J. and Yilmaz, E. (2020). Sequential-based adversarial optimisation for personalised top-n item recommendation. In *Proc. of SIGIR*.

Mao, Y., Shi, X., Shang, M.-S., and Zhang, Y. (2018). TCR: Temporal-cnn for reviews based recommendation system. In *Proc. of ICDLT*.

Massimo, D. and Ricci, F. (2018). Harnessing a generalised user behaviour model for next-poi recommendation. In *Proc. of RecSys*.

McAuley, J. and Leskovec, J. (2013). Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proc. of RecSys*.

McAuley, J., Pandey, R., and Leskovec, J. (2015). Inferring networks of substitutable and complementary products. In *Proc. of SIGKDD*.

Medhat, W., Hassan, A., and Korashy, H. (2014). Sentiment analysis algorithms and applications: A survey. *Ain Shams engineering journal*, 5(4):1093–1113.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Proc. of NeurIPS*.

Miller, G. A. (1995). WordNet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Miyahara, K. and Pazzani, M. J. (2002). Improvement of collaborative filtering with the simple bayesian classifier. *Information Processing Society of Japan*, 43(11).

Moghaddam, S., Jamali, M., and Ester, M. (2012). ETF: extended tensor factorization model for personalizing prediction of review helpfulness. In *Proc. of WSDM*.

Mohammad, S., Dunne, C., and Dorr, B. (2009). Generating high-coverage semantic orientation lexicons from overtly marked words and a thesaurus. In *Proc. of EMNLP*.

Mohammad, S., Kiritchenko, S., and Zhu, X.-D. (2013). NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. In *Proc. of SemEval*.

Moraes, R., Valiati, J. F., and Neto, W. P. G. (2013). Document-level sentiment classification: An empirical comparison between svm and ann. *Expert Systems with Applications*, 40(2):621–633.

Mustafa, N., Ibrahim, A. O., Ahmed, A., and Abdullah, A. (2017). Collaborative filtering: Techniques and applications. In *Proc of ICCCCEE*.

Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proc. of ICML*.

Nepal, S., Paris, C., and Bista, S. K. (2012). SRec: a social behaviour based recommender for online communities. In *Proc. of UMAP Workshops*.

Ng, R. T. and Han, J. (2002). CLARANS: A method for clustering objects for spatial data mining. *IEEE transactions on knowledge and data engineering*, 14(5):1003–1016.

Ni, J., Li, J., and McAuley, J. (2019). Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proc. of EMNLP-IJCNLP*.

Ning, X., Desrosiers, C., and Karypis, G. (2015). A comprehensive survey of neighborhood-based recommendation methods. In *Recommender Systems Handbook*, pages 37–76. Springer.

Ortigosa-Hernández, J., Rodríguez, J. D., Alzate, L., Lucania, M., Inza, I., and Lozano, J. A. (2012). Approaching sentiment analysis by using semi-supervised learning of multi-dimensional classifiers. *Neurocomputing*, 92:98–115.

O'Shea, K. and Nash, R. (2015). An introduction to convolutional neural networks. *CoRR*, abs/1511.08458.

Ou, C., Buschek, D., Eiband, M., and Butz, A. (2021). Modeling web browsing behavior across tabs and websites with tracking and prediction on the client side. *CoRR*, abs/2103.04694.

O'Mahony, M. P., Hurley, N. J., and Silvestre, G. C. (2003). An evaluation of the performance of collaborative filtering. In *Proc. of AICS*.

Paice, C. D. (2018). Lexical analysis of textual data. In *Encyclopedia of Database Systems, Second Edition*. Springer.

Pang, B. and Lee, L. (2005). Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proc. of ACL*.

Pang, B., Lee, L., and Vaithyanathan, S. (2002). Thumbs up?: sentiment classification using machine learning techniques. In *Proc. of ACL*.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. (2019). PyTorch: An imperative style, high-performance deep learning library. In *Proc. of NeurIPS*.

Paul, D., Sarkar, S., Chelliah, M., Kalyan, C., and Sinai Nadkarni, P. P. (2017). Recommendation of high quality representative reviews in e-commerce. In *Proc. of RecSys*.

Pennington, J., Socher, R., and Manning, C. D. (2014). GloVe: Global vectors for word representation. In *Proc. of EMNLP*.

Pessiot, J.-F., Truong, T.-V., Usunier, N., Amini, M.-R., and Gallinari, P. (2007). Learning to rank for collaborative filtering. In *Proc. of ICEIS*.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. In *Proc. of NAACL-HLT*.

Petty, R. E. and Cacioppo, J. T. (2012). *Communication and persuasion: Central and peripheral routes to attitude change*. Springer Science & Business Media.

Qahri-Saremi, H. and Montazemi, A. R. (2019). Factors affecting the adoption of an electronic word of mouth message: A meta-analysis. *Journal of Management Information Systems*, 36(3).

Qiu, G., Liu, B., Bu, J., and Chen, C. (2011). Opinion word expansion and target extraction through double propagation. *Computational linguistics*, 37(1):9–27.

Qiu, X., Sun, T., Xu, Y., Shao, Y., Dai, N., and Huang, X. (2020). Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, pages 1–26.

Ramachandran, P., Zoph, B., and Le, Q. V. (2017). Searching for activation functions. In *Proc. of ICLR*.

Řehůřek, R. and Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. In *Proc. of the LREC 2010 Workshop on New Challenges for NLP Frameworks*.

Rendle, S. (2010). Factorization machines. In *Proc. of ICDM*.

Rendle, S. (2012). Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology*, 3(3).

Rendle, S., Freudenthaler, C., Gantner, Z., and Schmidt-Thieme, L. (2009). BPR: bayesian personalized ranking from implicit feedback. In *Proc. of UAI*.

Rendle, S., Freudenthaler, C., and Schmidt-Thieme, L. (2010). Factorizing personalized markov chains for next-basket recommendation. In *Proc. of WWW*.

Rendle, S., Gantner, Z., Freudenthaler, C., and Schmidt-Thieme, L. (2011). Fast context-aware recommendations with factorization machines. In *Proc. of SIGIR*.

Rendle, S., Krichene, W., Zhang, L., and Anderson, J. (2020). Neural collaborative filtering vs. matrix factorization revisited. In *Proc. of RecSys*.

Ricci, F., Rokach, L., and Shapira, B. (2011). *Introduction to Recommender Systems Handbook*. Springer.

Ricci, F., Rokach, L., and Shapira, B. (2015). Recommender systems: Introduction and challenges. In *Recommender Systems Handbook*, pages 1–34. Springer.

Riou, C. and Honda, J. (2020). Bandit algorithms based on thompson sampling for bounded reward distributions. In *Proc. of ALT*.

Riquelme, C., Tucker, G., and Snoek, J. (2018). Deep bayesian bandits showdown: An empirical comparison of bayesian deep networks for thompson sampling. In *Proc. of ICLR*.

Rojas, R. (1996). The backpropagation algorithm. In *Neural networks*, pages 149–182. Springer.

Ruder, S. (2017). An overview of multi-task learning in deep neural networks. *CoRR*, abs/1706.05098.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, 323(6088):533–536.

Russo, D., Roy, B. V., Kazerouni, A., Osband, I., and Wen, Z. (2018). A tutorial on thompson sampling. *Foundations and Trends in Machine Learning*, 11(1).

Saad, D. (1998). Online algorithms and stochastic approximations. *Online Learning*, 5:6–3.

Sachdeva, N. and McAuley, J. (2020). How useful are reviews for recommendation? A critical review and potential improvements. In *Proc. of SIGIR*.

Sahoo, N., Singh, P. V., and Mukhopadhyay, T. (2012). A hidden markov model for collaborative filtering. *MIS quarterly*, pages 1329–1356.

Said, A. and Bellogín, A. (2018). Coherence and inconsistencies in rating behavior: estimating the magic barrier of recommender systems. *User Modeling and User-Adapted Interaction*, 28(2):97–125.

Said, A., Jain, B. J., Narr, S., and Plumbaum, T. (2012). Users and noise: The magic barrier of recommender systems. In *Proc. of UMAP*.

Sakai, T. (2018). Laboratory experiments in information retrieval. *The Information Retrieval Series*, 40.

Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proc. of WWW*.

Schafer, J. B., Frankowski, D., Herlocker, J. L., and Sen, S. (2007). Collaborative filtering recommender systems. In *The Adaptive Web, Methods and Strategies of Web Personalization*.

Scott, C. and Blanchard, G. (2009). Novelty detection: Unlabeled data definitely help. In *Proc. of AISTATS*.

Şenel, L. K., Utlu, I., Yücesoy, V., Koc, A., and Cukur, T. (2018). Semantic structure and interpretability of word embeddings. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(10):1769–1779.

Seo, S., Huang, J., Yang, H., and Liu, Y. (2017). Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In *Proc. of RecSys*.

Shamir, M., Kaushal, A., Reddy, K., Vaishnaw, K., and Singh, M. (2021). Review based recommendations with human-like reasons. In *Proc. of CODS-COMAD*.

Shani, G. and Gunawardana, A. (2011). Evaluating recommendation systems. In *Recommender systems handbook*, pages 257–297. Springer.

Shanks, D. R., Tunney, R. J., and McCarthy, J. D. (2002). A re-examination of probability matching and rational choice. *Behavioral Decision Making*, 15(3).

Shardanand, U. and Maes, P. (1995). Social information filtering: Algorithms for automating "word of mouth". In *Proc. of SIGCHI*.

Sharma, S. and Sharma, S. (2017). Activation functions in neural networks. *Towards Data Science*, 6(12):310–316.

Shen, T., Jia, J., Li, Y., Wang, H., and Chen, B. (2020). Enhancing music recommendation with social media content: an attentive multimodal autoencoder approach. In *Proc. of IJCNN*.

Sherstinsky, A. (2020). Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Physica D: Nonlinear Phenomena*, 404:132306.

Shi, C., Hu, B., Zhao, W. X., and Philip, S. Y. (2018). Heterogeneous information network embedding for recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 31(2):357–370.

Shi, X. and Liang, X. (2015). Resolving inconsistent ratings and reviews on commercial webs based on support vector machines. In *Proc. of ICSSSM*.

Sinha, R. R., Swearingen, K., et al. (2001). Comparing recommendations made by online systems and friends. *DELOS*, 106.

Song, Q., Chang, S., and Hu, X. (2019). Coupled variational recurrent collaborative filtering. In *Proc. of SIGKDD*.

Spicer, R. A., Bera, S., De Bera, S., Spicer, T. E., Srivastava, G., Mehrotra, R., Mehrotra, N., and Yang, J. (2011). Why do foliar physiognomic climate estimates sometimes differ from those observed? insights from taphonomic information loss and a clamp case study from the ganges delta. *Palaeogeography, Palaeoclimatology, Palaeoecology*, 302(3-4).

Sra, S. and Dhillon, I. S. (2006). Generalized nonnegative matrix approximations with bregman divergences. In *Proc. of NeurIPS*.

Steck, H. (2013). Evaluation of recommendations: rating-prediction and ranking. In *Proc. of RecSys*.

Su, X. and Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009.

Sun, J., Zhang, Y., Guo, W., Guo, H., Tang, R., He, X., Ma, C., and Coates, M. (2020a). Neighbor interaction aware graph convolution networks for recommendation. In *Proc. of SIGIR*.

Sun, P., Wu, L., Zhang, K., Fu, Y., Hong, R., and Wang, M. (2020b). Dual learning for explainable recommendation: Towards unifying user preference prediction and review generation. In *Proc. of WWW*.

Sun, X., Han, M., and Feng, J. (2019). Helpfulness of online reviews: Examining review informativeness and classification thresholds by search products and experience products. *Decision Support Systems*, 124.

Sun, Y., Wong, A. K., and Kamel, M. S. (2009). Classification of imbalanced data: A review. *International Journal of Pattern Recognition and Artificial Intelligence*, 23(04).

Sussman, S. W. and Siegal, W. S. (2003). Informational influence in organizations: An integrated approach to knowledge adoption. *Information systems research*, 14(1):47–65.

Tang, J., Gao, H., Hu, X., and Liu, H. (2013). Context-aware review helpfulness rating prediction. In *Proc. of RecSys*.

Tang, J. and Wang, K. (2018). Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proc. of WSDM*.

Tang, L. R., Jang, S. S., and Morrison, A. (2012). Dual-route communication of destination websites. *Tourism Management*, 33(1):38–49.

Tang, Y., Zhang, Y.-Q., Chawla, N. V., and Krasser, S. (2009). SVMs modeling for highly imbalanced classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(1).

Taud, H. and Mas, J. (2018). Multilayer perceptron (MLP). In *Geomatic Approaches for Modeling Land Change Scenarios*, pages 451–455. Springer.

Tay, Y., Luu, A. T., and Hui, S. C. (2018). Multi-pointer co-attention networks for recommendation. In *Proc. of SIGKDD*.

Taylor, W. L. (1953). "Cloze procedure": A new tool for measuring readability. *Journalism quarterly*, 30(4):415–433.

Thorat, P. B., Goudar, R., and Barve, S. (2015). Survey on collaborative filtering, content-based filtering and hybrid recommendation system. *International Journal of Computer Applications*, 110(4):31–36.

Tsur, O. and Rappoport, A. (2009). RevRank: A fully unsupervised algorithm for selecting the most helpful book reviews. In *Proc. of ICWSM*.

Ungar, L. H. and Foster, D. P. (1998). Clustering methods for collaborative filtering. In *Proc. of AAAI*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In *Proc. of NeurIPS*.

Velàsquez, J. D. and Palade, V. (2007). Building a knowledge base for implementing a web-based computerized recommendation system. *International Journal on Artificial Intelligence Tools*, 16(05):793–828.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. (2018). Graph Attention Networks. In *Proc. of ICLR*.

Villatel, K., Smirnova, E., Mary, J., and Preux, P. (2018). Recurrent neural networks for long and short-term sequential recommendation. *CoRR*, abs/1807.09142.

Volkovs, M. and Yu, G. W. (2015). Effective latent models for binary feedback in recommender systems. In *Proc. of SIGIR*.

Wang, C., Zhang, M., Ma, W., Liu, Y., and Ma, S. (2020a). Make it a chorus: knowledge-and time-aware item modeling for sequential recommendation. In *Proc. of SIGIR*.

Wang, H., Fu, Y., Wang, Q., Yin, H., Du, C., and Xiong, H. (2017a). A location-sentiment-aware recommender system for both home-town and out-of-town users. In *Proc. of SIGKDD*.

Wang, H., Shao, N., and Lian, D. (2019a). Adversarial binary collaborative filtering for implicit feedback. In *Proc. of AAAI*.

Wang, H., Wang, N., and Yeung, D.-Y. (2015a). Collaborative deep learning for recommender systems. In *Proc. of SIGKDD*.

Wang, J., Feng, Y., Naghizade, E., Rashidi, L., Lim, K. H., and Lee, K. (2018a). Happiness is a choice: sentiment and activity-aware location recommendation. In *Companion Proc. of WWW*.

Wang, J., Liu, Q., Liu, Z., and Wu, S. (2019b). Towards accurate and interpretable sequential prediction: A cnn & attention-based feature extractor. In *Proc. of CIKM*.

Wang, L., Yang, Y., Min, R., and Chakradhar, S. (2017b). Accelerating deep neural network training with inconsistent stochastic gradient descent. *Neural Networks*, 93:219–229.

Wang, S., Huang, M., Deng, Z., et al. (2018b). Densely connected cnn with multi-scale feature attention for text classification. In *Proc. of IJCAI*.

Wang, X., Fang, A., Ounis, I., and Macdonald, C. (2019c). Evaluating similarity metrics for latent twitter topics. In *Proc. of ECIR*.

Wang, X., Ounis, I., and Macdonald, C. (2019d). Comparison of sentiment analysis and user ratings in venue recommendation. In *Proc. of ECIR*.

Wang, X., Ounis, I., and Macdonald, C. (2020b). Negative confidence-aware weakly supervised binary classification for effective review helpfulness classification. In *Proc. of CIKM*.

Wang, X., Ounis, I., and Macdonald, C. (2021a). Leveraging review properties for effective recommendation. In *Proc. of WWW*.

Wang, X., Ounis, I., and Macdonald, C. (2022). Effective rating prediction using an attention-based user review sentiment model. In *Proc. of ECIR*.

Wang, X., Wen, J., Luo, F., Zhou, W., and Ren, H. (2015b). Personalized recommendation system based on support vector machine and particle swarm optimization. In *Proc. of KSEM*.

Wang, X., Zhang, R., Sun, Y., and Qi, J. (2021b). Combating selection biases in recommender systems with a few unbiased ratings. In *Proc. of WSDM*.

Wang, Y., Liang, D., Charlin, L., and Blei, D. M. (2020c). Causal inference for recommender systems. In *Proc. of RecSys*.

Wang, Y., Wang, M., and Xu, W. (2018c). A sentiment-enhanced hybrid recommender system for movie recommendation: a big data analytics framework. *Wireless Communications and Mobile Computing*, 2018.

Wang, Z., Yu, X., Feng, N., and Wang, Z. (2014). An improved collaborative movie recommendation system using computational intelligence. *Journal of Visual Languages & Computing*, 25(6):667–675.

Wei, J., He, J., Chen, K., Zhou, Y., and Tang, Z. (2017). Collaborative filtering and deep learning based recommendation system for cold start items. *Expert Systems with Applications*, 69:29–39.

Werbos, P. J. (1990). Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., and Brew, J. (2019). Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Wu, C. and Shaffer, D. R. (1987). Susceptibility to persuasive appeals as a function of source credibility and prior experience with the attitude object. *Journal of personality and social psychology*, 52(4):677.

Wu, C., Wu, F., Qi, T., and Huang, Y. (2021). FeedRec: News feed recommendation with various user feedbacks. In *Proc. of KDD*.

Wu, C.-Y., Ahmed, A., Beutel, A., Smola, A. J., and Jing, H. (2017). Recurrent recommender networks. In *Proc. of WSDM*.

Wu, S., Ren, W., Yu, C., Chen, G., Zhang, D., and Zhu, J. (2016). Personal recommendation using deep recurrent neural networks in netease. In *Proc. of ICDE*.

Wu, Y., Macdonald, C., and Ounis, I. (2020). A hybrid conditional variational autoencoder model for personalised top-n recommendation. In *Proc. of ICTIR*.

Xia, Z., Dong, Y., and Xing, G. (2006). Support vector machines for collaborative filtering. In *Proc. of ACM-SE*.

Xing, S., Wang, Q., Zhao, X., Li, T., et al. (2019). A hierarchical attention model for rating prediction by leveraging user and product reviews. *Neurocomputing*, 332:417–427.

Xu, C., Zhao, P., Liu, Y., Xu, J., S. Sheng, V. S. S., Cui, Z., Zhou, X., and Xiong, H. (2019). Recurrent convolutional neural network for sequential recommendation. In *Proc. of WWW*.

Xu, X., Dong, F., Li, Y., He, S., and Li, X. (2020). Contextual-bandit based personalized recommendation with time-varying user interests. In *Proc. of AAAI*.

Xu, Z., Chen, C., Lukasiewicz, T., Miao, Y., and Meng, X. (2016). Tag-aware personalized recommendation using a deep-semantic similarity model with negative sampling. In *Proc. of CIKM*.

Xu, Z., Zeng, H., and Ai, Q. (2021). Understanding the effectiveness of reviews in e-commerce top-n recommendation. In *Proc. of ICTIR*.

Xue, H.-J., Dai, X., Zhang, J., Huang, S., and Chen, J. (2017). Deep matrix factorization models for recommender systems. In *Proc. of IJCAI*.

Yan, A., Cheng, S., Kang, W.-C., Wan, M., and McAuley, J. (2019). CosRec: 2D convolutional neural networks for sequential recommendation. In *Proc. of CIKM*.

Yang, D., Zhang, D., Yu, Z., and Wang, Z. (2013). A sentiment-enhanced personalized location recommendation system. In *Proc. of ACMHT*.

Yang, J., Wang, M., Zhou, H., Zhao, C., Zhang, W., Yu, Y., and Li, L. (2020). Towards making the most of bert in neural machine translation. In *Proc. of AAAI*.

Yang, L., Cui, Y., Xuan, Y., Wang, C., Belongie, S. J., and Estrin, D. (2018a). Unbiased offline recommender evaluation for missing-not-at-random implicit feedback. In *Proc. of RecSys*.

Yang, X., Guo, Y., Liu, Y., and Steck, H. (2014). A survey of collaborative filtering based social recommender systems. *Computer communications*, 41:1–10.

Yang, X., Macdonald, C., and Ounis, I. (2018b). Using word embeddings in Twitter election classification. *Information Retrieval Journal*, 21(2-3).

Yang, Y., Yan, Y., Qiu, M., and Bao, F. (2015). Semantic analysis and helpfulness prediction of text for online product reviews. In *Proc. of ACL*.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., and Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Ye, W., Wang, S., Chen, X., Wang, X., Qin, Z., and Yin, D. (2020). Time matters: Sequential recommendation with complex temporal information. In *Proc. of SIGIR*.

Yilmaz, E., Aslam, J. A., and Robertson, S. (2008). A new rank correlation coefficient for information retrieval. In *Proc. of SIGIR*.

You, J., Wang, Y., Pal, A., Eksombatchai, P., Rosenburg, C., and Leskovec, J. (2019). Hierarchical temporal convolutional networks for dynamic recommender systems. In *Proc. of WWW*.

Yu, F., Liu, Q., Wu, S., Wang, L., and Tan, T. (2016). A dynamic recurrent model for next basket recommendation. In *Proc. of SIGIR*.

Yu, X., Ren, X., Sun, Y., Sturt, B., Khandelwal, U., Gu, Q., Norick, B., and Han, J. (2013). Recommendation in heterogeneous information networks with implicit user feedback. In *Proc. of RecSys*.

Yuan, F., Jose, J. M., Guo, G., Chen, L., Yu, H., and Alkhawaldeh, R. S. (2016). Joint geo-spatial preference and pairwise ranking for point-of-interest recommendation. In *Proc. of ICTAI*.

Yue, Y., Patel, R., and Roehrig, H. (2010). Beyond position bias: Examining result attractiveness as a source of presentation bias in clickthrough data. In *Proc. of WWW*.

Yun, Y., Hooshyar, D., Jo, J., and Lim, H. (2018). Developing a hybrid collaborative filtering recommendation system with opinion mining on purchase review. *Journal of Information Science*, 44(3):331–344.

Zahavy, T. and Mannor, S. (2019). Deep neural linear bandits: Overcoming catastrophic forgetting through likelihood matching. *CoRR*, abs/1901.08612.

Zeng, J., He, X., Tang, H., and Wen, J. (2019). A next location predicting approach based on a recurrent neural network and self-attention. In *Proc. of COLLABORATECOM*.

Zhang, H. (2004). The optimality of naive bayes. In *Proc. of AAAI*.

Zhang, J.-D. and Chow, C.-Y. (2015). GeoSoCa: Exploiting geographical, social and categorical correlations for point-of-interest recommendations. In *Proc. of SIGIR*.

Zhang, K., Qian, H., Liu, Q., Zhang, Z., Zhou, J., Ma, J., and Chen, E. (2021). SIFN: A sentiment-aware interactive fusion network for review-based item recommendation. *CoRR*, abs/2108.08022.

Zhang, L., Liu, B., Lim, S. H., and O'Brien-Strain, E. (2010). Extracting and ranking product features in opinion documents. In *Proc. of COLING*.

Zhang, S., Yao, L., Sun, A., and Tay, Y. (2019). Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys*, 52(1):1–38.

Zhang, T., Ramakrishnan, R., and Livny, M. (1996). Birch: an efficient data clustering method for very large databases. *ACM sigmod record*, 25(2):103–114.

Zhang, W., Du, T., and Wang, J. (2016). Deep learning over multi-field categorical data. In *Proc. of ECIR*.

Zhang, Y., Ai, Q., Chen, X., and Croft, W. B. (2017). Joint representation learning for top-n recommendation with heterogeneous information sources. In *Proc. of CIKM*.

Zhang, Y., Dai, H., Xu, C., Feng, J., Wang, T., Bian, J., Wang, B., and Liu, T.-Y. (2014a). Sequential click prediction for sponsored search with recurrent neural networks. In *Proc. of AAAI*.

Zhang, Y., Lai, G., Zhang, M., Zhang, Y., Liu, Y., and Ma, S. (2014b). Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *Proc. of SIGIR*.

Zhang, Y., Zhang, H., Zhang, M., Liu, Y., and Ma, S. (2014c). Do users rate or review? boost phrase-level sentiment labeling with review-level sentiment classification. In *Proc. of SIGIR*.

Zhang, Z. and Nasraoui, O. (2007). Efficient hybrid web recommendations based on markov clickstream models and implicit search. In *Proc. of WI*.

Zhao, G., Qian, X., and Xie, X. (2016a). User-service rating prediction by exploring social users' rating behaviors. *Transactions on multimedia*, 18(3).

Zhao, S., Zhao, T., King, I., and Lyu, M. R. (2017). Geo-teaser: Geo-temporal sequential embedding rank for point-of-interest recommendation. In *Proc. of WWW*.

Zhao, S., Zhao, T., Yang, H., Lyu, M. R., and King, I. (2016b). Stellar: Spatial-temporal latent ranking for successive point-of-interest recommendation. In *Proc. of AAAI*.

Zhao, X., Zhu, Z., Zhang, Y., and Caverlee, J. (2020). Improving the estimation of tail ratings in recommender system with multi-latent representations. In *Proc. of WSDM*.

Zheng, L., Noroozi, V., and Yu, P. S. (2017). Joint deep modeling of users and items using reviews for recommendation. In *Proc. of WSDM*.

Zheng, Q., Tian, X., Jiang, N., and Yang, M. (2019). Layer-wise learning based stochastic gradient descent method for the optimization of deep convolutional neural network. *Journal of Intelligent & Fuzzy Systems*, 37(4):5641–5654.

Zhou, C., Bai, J., Song, J., Liu, X., Zhao, Z., Chen, X., and Gao, J. (2018). Atrank: An attention-based user behavior modeling framework for recommendation. In *Proc. of AAAI*.

Zhou, D., Li, L., and Gu, Q. (2020a). Neural contextual bandits with upper confidence bound-based exploration. In *Proc. of ICML*.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., and Sun, M. (2020b). Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81.

Zhou, Z.-H. (2017). A brief introduction to weakly supervised learning. *National Science Review*, 5(1):44–53.

Zhu, Q., Zhou, X., Song, Z., Tan, J., and Guo, L. (2019). DAN: Deep attention neural network for news recommendation. In *Proc. of AAAI*.

Ziani, A., Azizi, N., Schwab, D., Aldwairi, M., Chekkai, N., Zenakhra, D., and Cheriguene, S. (2017). Recommender system through sentiment analysis. In *Proc. of ICATS*.