Personalised recommendation : challenges and experimental issues

Chin, Jin Yao

2021

Chin, J. Y. (2021). Personalised recommendation : challenges and experimental issues.
Doctoral thesis, Nanyang Technological University, Singapore.
<https://hdl.handle.net/10356/154933>

<https://hdl.handle.net/10356/154933>

<https://doi.org/10.32657/10356/154933>

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0
International License (CC BY-NC 4.0).

Downloaded on 26 Apr 2023 17:32:24 SGT

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

SINGAPORE

PERSONALISED RECOMMENDATION: CHALLENGES AND EXPERIMENTAL ISSUES

CHIN JIN YAO

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

2021

PERSONALISED RECOMMENDATION: CHALLENGES AND EXPERIMENTAL ISSUES

CHIN JIN YAO

School of Computer Science and Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfilment of the requirements for the degree of
Doctor of Philosophy

2021

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

30/12/2021

.....

Date

ITU NTU NTU NTU NTU NTU NTU NTU
NTU NTU NTU NTU NTU NTU NTU NTU
ITU NTU NTU NTU NTU NTU NTU NTU
ITU NTU NTU NTU NTU NTU NTU NTU

Chin Jin Yao

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

30/12/2021

Date

NTU NTU NTU NTU NTU NTU NTU NTU
NTU NTU NTU NTU NTU NTU NTU NTU
NTU NTU NTU NTU NTU NTU NTU NTU
NTU NTU NTU NTU NTU NTU NTU NTU

Prof. Cong Gao

Authorship Attribution Statement

This thesis contains material from 1 paper published in the following peer-reviewed conference in which I am listed as an author.

Chapter 3 is published as **Jin Yao Chin**, Kaiqi Zhao, Shafiq Joty, and Gao Cong. ANR: Aspect-based Neural Recommender. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM'18, Pages 147–156, 2018*.

The contributions of the co-authors are as follows:

- I came up with the key idea, and formulated the problem after discussions with Dr. Kaiqi Zhao, Prof. Shafiq Joty, and Prof. Gao Cong.
- I designed and implemented the model, conducted all the experiments, and prepared the manuscript draft.
- Dr. Kaiqi Zhao, Prof. Shafiq Joty, and Prof. Gao Cong assisted in the revision of the manuscript drafts.

30/12/2021

.....
Date

ITU NTU NTU NTU NTU NTU NTU NTU
NTU NTU NTU NTU NTU NTU NTU NTU
ITU NTU NTU NTU NTU NTU NTU NTU
ITU NTU NTU NTU NTU NTU NTU NTU
.....

Chin Jin Yao

Acknowledgements

First and foremost, I would like to thank my supervisor **Prof. Cong Gao** for the opportunity to further my studies under his supervision. Prof. Cong has been extremely patient and encouraging, and his constructive feedback to all my doubts has helped to greatly improve the quality of my research.

I would also like to express my sincere gratitude to *Prof. Joty Shafiq, Dr. Zhao Kaiqi, Dr. Xie Miao*, and *Chen Yile*, for providing valuable suggestions and comments in the research work. I have benefited tremendously from discussions with the following researchers, namely Dr. Hai Zhen, Dr. Feng Shanshan, Dr. Anh Pham Nguyen Tuan, Dr. Liu Yiding, Dr. Li Xiucheng, and Dr. Tran Dang Quang Vinh. I am equally thankful to the following colleagues and friends for their discussion and inspiration. They include Dr. Feng Kaiyu, Dr. Guo Tao, Dr. Chen Zhida, Dr. Seyed Ali Majid Zonoozi, Dr. Li Jing, Dr. Yao Di, Kim Junghoon, and Chen Yue. Finally, I also appreciate the kind gesture from numerous researchers who have generously provided me with their datasets and helped to answer my research doubts.

Most importantly, I would like to thank my family and friends for their support throughout the years. This would not have been possible without their constant care and encouragement.

Summary

With the shift towards an increasingly digital lifestyle, recommender systems play a critical role in helping consumers to find the best product or service amongst a variety of options. Unsurprisingly, personalised recommendations have become part and parcel of our daily lives. For instance, recommender systems are widely adopted across various domains, including e-commerce platforms (e.g. *Amazon*, *eBay*, *Taobao*), location-based social networks (e.g. *Yelp*, *Foursquare*), and social media (e.g. *Facebook*, *Instagram*, *Twitter*). Arguably, both the importance and practicability of recommender systems have been a key driving force behind the sustained interest from both academia and industry. Nevertheless, there are various challenges and experimental issues which affect the predictive performance and/or robustness of a recommendation system.

In this dissertation, we propose novel hybrid models to overcome a longstanding challenge for personalised recommendation, i.e. the cold-start problem, by leveraging different types of content information in conjunction with recent advances in deep learning. Furthermore, we identify and examine challenges, as well as experimental issues, that persist in personalised recommendation. The contributions are summarised as follows:

- We propose a novel aspect-based neural recommender system which performs aspect-based representation learning for users and items, by designing an attention mechanism to focus on the relevant parts of these reviews while learning the representation of aspects on the task. Furthermore, we estimate aspect-level user and item importance in a joint manner using the idea of co-attention, which allows us to model the finer-grained interactions between users and items. To the best of our knowledge, we are the first to propose an end-to-end neural aspect-based recommender system which concurrently addresses the above-mentioned requirements. Extensive experiments have been conducted on 25 benchmark datasets from *Amazon* and *Yelp* to evaluate our proposed model against several state-of-the-art baselines such as DeepCoNN [168], D-Attn [119], and ALFM [25]. Furthermore, we investigate how the different components in our proposed model contribute to its effectiveness. In particular, we include a qualitative analysis of the aspects which are learned automatically by our model without any external supervision.

- We consider the problem of generating recommendations satisfactorily in *both* warm- and cold-start scenarios via a *single model*. To this end, we present an innovative hybrid model using our proposed neighbourhood-based *Collective Collaborative Filtering* (**CCF**) component, which aims to maximise the performance of a recommender system in the presence of a dynamic and constantly evolving environment. Evaluation procedures adopted in existing work for cold-start recommendation are examined critically, and a key finding is that the content information in commonly used datasets are often insufficient for differentiating one user (or item) from another. Furthermore, we adopt an evaluation framework which explicitly shows the model performance for both warm-start recommendation and the various cold-start scenarios, i.e. *user cold-start*, *item cold-start*, and *both user & item cold-start*. Experimental results on four datasets, including *ML-100K* & *ML-1M*, show that our proposed model consistently outperforms baseline methods for various cold-start scenarios, and achieves the best overall performance in all four datasets.
- We conduct a retrospective survey to provide an in-depth look at a fundamental but often neglected aspect of the evaluation procedure, i.e. the datasets themselves. To do so, we adopt a systematic and comprehensive approach to understand the datasets used for implicit feedback based top-K recommendation. We start by examining recent papers from several top-tier conferences to find out how different datasets have been utilised thus far. Next, we study the characteristics of these datasets to understand their similarities and differences. Finally, we conduct an empirical study to determine whether the choice of datasets used for evaluation can influence the observations and/or conclusions obtained. Our findings suggest that greater attention needs to be paid to the selection process of datasets used for evaluating recommender systems in order to improve the robustness of the obtained results.

In summary, we address both specific challenges, e.g. the cold-start problem, as well as broader experimental issues, e.g. the choice of datasets for empirical evaluation, in this dissertation. Our contributions bring about two clear benefits to the literature of recommendation systems: **(1)** We are able to better handle the cold-start problem by considering different types of content information with our newly proposed hybrid models, and **(2)** We are able to ensure the robustness of obtained experimental results by consciously selecting distinct datasets based on their characteristics.

Contents

Statement of Originality	i
Supervisor Declaration Statement	ii
Authorship Attribution Statement	iii
Acknowledgements	iv
Summary	v
List of Figures	xi
List of Tables	xiii
1 Introduction	1
1.1 Background	1
1.2 Challenges and Experimental Issues	3
1.3 Research Problems and Methodologies	4
1.3.1 Aspect-based Neural Recommender	4
1.3.2 Adaptive Framework for Seamless Warm- and Cold-start Recommendation	8
1.3.3 The Datasets Dilemma	10
1.4 Summary of Contributions	12
2 Literature Review	14
2.1 Overview of Recommendation Systems	14
2.1.1 General Recommendation	14
2.1.2 Collaborative Filtering	18
2.1.3 Cold-start Problem	19
2.2 Recommendation with Reviews	21
2.2.1 Deep Learning-based Recommender Systems	21

2.2.2	Aspect-based Recommendation Systems	22
2.2.3	Neural Attention & Co-Attention	24
2.3	Recommendation with User and Item Contents	25
2.3.1	End-to-end Hybrid Architectures	25
2.3.2	Mapping-based Hybrid Architectures	27
2.3.3	Meta-Learning	28
2.4	The Datasets Dilemma	29
2.4.1	Experimental Issues in Recommender Systems	29
2.4.2	Data Characteristics of Recommendation Datasets	30
2.5	Summary	32
3	Aspect-based Neural Recommender	34
3.1	Motivation and Problem Setting	35
3.1.1	Textual Reviews and the Cold-start Problem	35
3.1.2	Problem Setting	37
3.2	Overview of ANR	37
3.2.1	Embedding Layer	39
3.2.2	Aspect-based Representations	39
3.2.3	Aspect-level Importance	40
3.3	Aspect-based Representation Learning	41
3.4	Aspect Importance Estimation	44
3.5	Model Inference and Optimisation	46
3.5.1	Model Pre-training	47
3.5.2	Model Generalisation	47
3.6	Experimental Study	47
3.6.1	Datasets	48
3.6.2	Baseline Methods	48
3.6.3	Experimental Setup	50
3.6.4	Results and Discussion	51
3.6.5	Model Analysis	53
3.7	Summary	58

4	Adaptive Framework for Seamless Warm- and Cold-start Recommendation	59
4.1	Problem Setting	60
4.2	Proposed Model	60
4.2.1	Overall Model Architecture	61
4.2.2	Neighbourhood-based Collective CF	62
4.2.3	Model Inference and Optimisation	66
4.3	Experimental Set-up	67
4.3.1	Datasets	67
4.3.2	Baseline Methods	69
4.3.3	Evaluation Procedure	70
4.4	Experimental Results	72
4.4.1	Results and Discussion	73
4.4.2	Effects of Hyperparameters H and K	75
4.4.3	Task-specific Attention Weights	75
4.5	Summary	77
5	The Datasets Dilemma	79
5.1	Collection and Analysis	80
5.1.1	Paper and Dataset Collection	80
5.1.2	Dataset Usage Analysis	81
5.2	Datasets and their Characteristics	83
5.2.1	Dataset Characteristics	83
5.2.2	Similarities and Differences	86
5.3	Experiments and Results	90
5.3.1	Experimental Setup	91
5.3.2	Experimental Results	94
5.3.3	Recent Neural Recommendation Approaches	96
5.3.4	Key Findings	102
5.3.5	Suggestions	102
5.4	Summary	103

6	Conclusions and Future Work	104
6.1	Conclusions	104
6.2	Future Work	106
6.2.1	Explainable Recommendation	106
6.2.2	Domain-independent Aspect-based Recommendation	106
6.2.3	Datasets and their Characteristics	107
6.2.4	Summary of Future Work	107
	List of Publications	108
	References	109

List of Figures

1.1	Challenges and experimental issues for recommendation systems	3
1.2	Example of a textual review from Amazon	5
1.3	Illustration of various aspects for the restaurant domain	7
2.1	Illustration of a typical recommendation system	15
2.2	Recommendation with explicit feedback	16
2.3	Recommendation with implicit feedback	17
2.4	General architecture for deep learning-based recommender systems	22
2.5	Existing aspect-based recommendation systems	23
2.6	A generic end-to-end hybrid model	26
2.7	Degeneration of a generic end-to-end hybrid model	27
2.8	Illustration of the exploratory modelling approach	31
2.9	Relationships between different research problems and our contributions .	32
3.1	Users A and B giving the same rating to Restaurant X	35
3.2	Reviews written by Users A and B for Restaurant X	36
3.3	Overall architecture of the proposed ANR	38
3.4	Varied user preferences with respect to different items	40
3.5	Different appeal of the same item to different users	40
3.6	Joint and complex evaluation of interrelated aspects	41
3.7	Illustration of the aspect-based representation learning component	43
3.8	Aspect-level affinity matrix	45
3.9	Illustration of the user-item rating prediction component	46
3.10	Effect of the number of aspects across multiple datasets	53
3.11	Effect of the number of factors for h_1 and h_2	54

4.1	Overall model architecture	61
4.2	Neighbourhood-based Collective CF	63
4.3	Data partitioning	70
4.4	Overall MAE for ML-100K and Yelp	74
4.5	Overall RMSE for ML-100K and Yelp	74
4.6	Effect of hyperparameters K and H on the validation RMSE	76
4.7	Task-specific attention weights for ML-100K	77
5.1	Dataset usage patterns	81
5.2	Pairwise Euclidean distance between different datasets	88
5.3	Recall@10 of 4 recommendation algorithms for the 15 datasets	93
5.4	nDCG@10 of 4 recommendation algorithms for the 15 datasets	93
5.5	Recall@10 of 8 recommendation algorithms for the 15 datasets	98
5.6	nDCG@10 of 8 recommendation algorithms for the 15 datasets	99

List of Tables

3.1	Strengths and weaknesses of existing recommendation systems utilising textual reviews	34
3.2	Notations and their definitions	37
3.3	Statistical details for the datasets	49
3.4	Comparison with state-of-the-art recommender systems	52
3.5	Qualitative analysis of learned aspects	55
3.6	Ablation study for the proposed model	57
4.1	Statistical details for the datasets	67
4.2	Content information for the datasets	67
4.3	MAE & RMSE for ML-1M	72
4.4	MAE & RMSE for Douban (Movie)	72
5.1	Recent papers for implicit feedback based top-K recommendation	80
5.2	Combinations of datasets used together in 2 or more papers	82
5.3	Source and processing steps taken for the datasets	87
5.4	Dataset clusters obtained using k-means	89
5.5	Characteristics of all dataset clusters	90
5.6	Representative datasets for each dataset cluster	91

Chapter 1

Introduction

1.1 Background

As more users turn to online platforms for the consumption of various goods and services, recommendation systems become an increasingly important component for narrowing down the options in a personalised manner. Arguably, both the importance and practicability of recommender systems have been a key driving force behind the sustained interest from both academia and industry. This has brought about rapid growth in terms of the recommendation performance across a variety of application scenarios, ranging from e-commerce (e.g. Amazon, eBay) to social media (e.g. Facebook, Instagram). Furthermore, several decades of research have led the field of recommender systems from simple and widely known methods such as Weighted Matrix Factorization (**WMF**) [58] and Item k-Nearest Neighbour (**ItemKNN**) [115] to advanced and sophisticated algorithms such as Mult-VAE [82] and LightGCN [49].

Many successful recommendation systems are primarily based on the Collaborative Filtering (**CF**) technique [12], which analyses the historical user-item interaction data in order to learn user preferences and item profiles. In general, past interaction data can be categorised into two main types: **(1)** *Implicit feedback*, such as views, clicks, and purchases, as well as **(2)** *Explicit feedback*, such as ratings and textual reviews. However, regardless of the type of feedback collected and used within the recommendation system, a major limitation of CF techniques, such as the well-known Matrix Factorization (**MF**) [69], is its inability to provide reliable recommendations to users with few interactions, or to recommend items with limited interactions. As such, CF-based methods which rely solely on historical feedback are not directly applicable for users and items

which are relatively new to the system. This is often referred to as the ***cold-start problem***, which has been a long-standing challenge for most, if not all, recommender systems.

Due to the simplicity and effectiveness of MF, content information can be easily integrated without changing its learning framework. There are several hybrid models [2, 146, 147] which are built upon a probabilistic variant of MF [113], while other hybrid models [124, 125, 162, 57] tend to be a direct extension of the classic MF framework. Although hybrid models do alleviate the severity of the cold-start problem to some extent, there are several types of contents available. These include textual reviews, images, user demographics (e.g. the *age, gender, and occupation* for a user), as well as item attributes (e.g. the *release year, genre, and language* of a movie). As such, depending on the specific type of content information used, incorporating it within a CF-based framework does introduce its own set of unique challenges. For example, when dealing with textual contents such as user reviews, we find that **(1)** all not parts of the review are equally important, and **(2)** the same word can bear completely different sentiments depending on its context.

As deep neural networks have been successfully applied to a large variety of tasks, such as natural language processing, computer vision, and speech recognition [41], many recently proposed recommender systems have also turned to deep learning. However, in most cases, they capitalise on the strong representation learning capabilities of neural networks, but fail to leverage these additional information sources in an effective and efficient manner. In this dissertation, we seek to develop novel hybrid models by leveraging recent advances in deep learning, e.g. neural attention mechanism, to fully utilise the different types of content information, and at the same time, overcome the limitations of existing proposals.

Nevertheless, while it is paramount to constantly develop new algorithms to advance the state-of-the-art, we believe that the evaluation procedure needs to be well-defined and robust so as to guarantee the validity of the obtained results. Very recently, several studies [30, 70, 132] have pinpointed worrisome problems that appear to undermine years of hard work within the recommendation systems community. Notably, [30] have discovered that many recently proposed methods are in fact not reproducible, and [132] noted that researchers often choose different datasets heuristically and there are in fact many seemingly trivial factors which can influence the recommendation performance.

To this end, we identify and critically examine several challenges in evaluating the performance of a recommender system on cold-start scenarios. Furthermore, we explore a specific problem that has long been overlooked by the research community: *How much do we really know about recommendation datasets?* In this dissertation, we conduct a retrospective survey on the datasets used for top-K recommendation and seek to shed light on whether the current practice of selecting datasets arbitrarily would lead to unintended (and perhaps, negative) consequences. To the best of our knowledge, we are the first to provide an extensive overview of recommendation datasets, which is a fundamental but often neglected aspect of the evaluation procedure.

1.2 Challenges and Experimental Issues

Figure 1.1: Challenges and experimental issues for recommendation systems

As we will see in greater detail later on, recommendation systems are often plagued by various challenges and experimental issues as illustrated in Figure 1.1.

In terms of the ‘*specific*’ challenges, we are interested in addressing the well-known *cold-start problem* for various reasons. First of all, we believe that making recommendations for relatively/completely new users and items are just as important as doing so for existing users and items. The ability to attract and retain such new users/items will be a key factor for any recommendation system to remain successful and relevant in the long

run. However, many existing works tend to focus on improving the recommendations generated for existing users/items. Next, for most products/services, there are often a wide variety of platforms to choose from. For example, if a new user is not satisfied with the recommendations provided by a particular platform, she could easily switch to a different platform. As such, the ability to handle cold-start situations involving new users and/or items could be the differentiating factor between two or more platforms having similar products/services. The research problems which will be presented subsequently in Sections 1.3.1 and 1.3.2 correspond to different formulations of the cold-start problem.

On the other hand, we have ‘*broader*’ experimental issues which go beyond specific recommendation frameworks and/or application scenarios. Some examples of such broader issues include the reproducibility/replicability of proposed methods, the proper selection and tuning of baseline methods, etc. The research problem described in Section 1.3.3 addresses one of these experimental issues, i.e. the choice of datasets for empirical evaluation. For more general application scenarios, e.g. top-K item recommendation based on implicit feedback, there could be numerous datasets to choose from. As we will show later on, there is a tendency for many existing works to select datasets arbitrarily. This becomes an issue when researchers try to compare their proposed method against different baseline methods empirically. Furthermore, the choice of datasets could affect the robustness of a proposed method and/or the validity of an empirical study. As such, we are motivated to look at this particular problem due to its great importance, as well as how it has been long overlooked by the research community.

1.3 Research Problems and Methodologies

1.3.1 Aspect-based Neural Recommender

Recent recommender systems have considered another valuable source of information which is readily available in many e-commerce and review websites such as *Amazon* and *Yelp*: Free-text reviews. More often than not, users provide an accompanying review to *explain* why they liked or disliked that particular product or service, i.e., the reasons contributing to the overall numerical rating. For example, a review may include the user’s opinions on the various *aspects* of an item, and these aspects are generally domain-dependent. As shown in Figure 1.2, when writing the review for a particular movie, the user considers different components such as the duration, cast, choreography, and

★★★★★ **Solid MI**

By [Zimmer](#) on September 2, 2018

Format: Blu-ray

No doubt one of if not the best movie released this year and, just my IMO, in the top 3 Mission films. However im not sure it is quite deserving of the high RT rating it received. It does drag a bit in the second act when Solomon Lane is introduced again. The film needed a truly great scene stealing villain IMO to compete with the great action and Cruise's stunts, and Lane just isnt that interesting. Much has been said of Cavill and his amazing moustache and he's decent but a bit wooden. Great physical presence though. Cruise is solid as usual. Really really enjoyed the first act and the action scenes toward the end were great. The score by Lorne Balfe might just be the best MI score yet. Should have cut the running time a bit tho

Figure 1.2: Example of a textual review from Amazon

music. In fact, reviews provide more than just an avenue for modelling the implicit user preferences or item properties. The rich semantic information in these reviews can be useful in helping us understand the multi-faceted process behind how users tend to rate items, i.e. the *key factors* which influence a user to prefer one item over the other.

1.3.1.1 Motivations

Owing to its superior representation learning capabilities, deep learning techniques have been widely used in recent state-of-the-art recommendation systems to construct latent user and item representations using the review contents. This includes models such as DeepCoNN [168], D-Attn [119], and TransNets [18], all of which are based on using Convolutional Neural Networks (CNNs) [66] to encode the user (and item) reviews into their corresponding latent embeddings. While these proposed methods have been shown to provide good predictive performance, their approach of simply inferring a single low-dimensional latent representation for each user (and item) would inherently be limited by its inability to capture the finer-grained interactions between users and items. In other words, a single low-dimensional latent representation may be inadequate for truly reflecting and understanding the complex dynamics in real-world recommendation systems.

Intuitively, not all parts of a review are equally important. For example, some parts of the review may be describing the plot of a movie, or even the storyline in a book, and such '*details*' may not be correlated with the overall user satisfaction. A common observation is that each part of the review tends to focus on a different facet of the user's

overall experience, such as the location of a restaurant, the attitude of its service staff, or even the taste of the dishes served in that restaurant. By focusing on these salient factors, we can better infer both the preferences of a specific user (E.g. User X prefers a restaurant with outdoor seating) and the properties of an item (E.g. Restaurant Y is famous for its seafood dishes).

However, to model the rich semantics of review contents, it is imperative to move beyond the surface-level word representations. Consider the following two sentences which contain the word ‘*long*’: (1) ”This laptop has a long battery life”, and (2) ”The laptop requires a long startup time”. It is evident that the word ‘*long*’ bears a positive sentiment towards the target aspect (or item property) in the first sentence, while the same word indicates a negative sentiment for the exact same item in the second sentence. Consequently, a flexible word representation scheme which is able to take into consideration such contextual information with respect to any given aspect would be desirable.

Additionally, different users may emphasise more on different aspects throughout their interactions with these items. For example, some user may like a particular ***restaurant*** for its *food*, while another user frequents the same restaurant due to its cosy *ambience*. Similarly, a user may prioritise the *storyline* when choosing a ***horror movie***, but pays more attention to the *cast* when evaluating an ***action movie***. Understandably, the importance of each aspect largely depends on both the user and item in question, and being able to capture such dynamic and fine-grained interactions between users and items would be invaluable in determining why some user may prefer an item over the other.

To summarise, we believe that each rating given by a user for some particular item can be viewed as the outcome of a multi-faceted decision-making process, and a good recommender system should aim to emulate this in order to provide reliable and interpretable recommendations.

1.3.1.2 Methodology

As we will see later on in our literature review (Chapter 2), there are two broad categories of recommender systems that have been shown to achieve state-of-the-art predictive performance. The first category consists of recommender systems which are based on deep learning techniques, such as Convolutional Neural Network (**CNN**) and Recurrent Neural Network (**RNN**). These models depend on the strong representation learning capabilities of neural networks to derive a suitable user (or item) representation from the

corresponding set of user (or item) reviews. However, they are often less intuitive and transparent, and most of them do not provide an explanation or justification for their recommendations.

Figure 1.3: Illustration of various aspects for the restaurant domain

On the other hand, another popular category of recommender systems stems from the modelling of aspects through various analyses of the associated textual reviews. An aspect can be described as a high-level semantic concept encompassing a specific facet of item properties for a given domain. As illustrated in Figure 1.3, the set of considered aspects in the domain of *restaurants* may include *price*, *quality*, *service*, *location*, etc. For instance, the ‘*service*’ aspect encompasses various attributes such as the staff of that particular restaurant, the waiting time, as well as whether reservation and/or valet parking would be available. In these existing aspect-based recommender systems, the aspects are usually extracted from the reviews using Sentiment Analysis (**SA**) tools, or learned using some probabilistic approach such as Latent Dirichlet Allocation (**LDA**). For models relying on SA tools, they are not self-contained and their performance largely depends on the quality of these SA tools. Furthermore, existing aspect-based recommender systems do not emphasise on how different parts of the reviews may contribute differently to the overall satisfaction, and they often fail to consider the varied aspect-level importance for both users and items.

As such, the goal of this study is to leverage the strengths of both deep learning-based and aspect-based recommender systems. In particular, we propose an *Aspect-based Neural Recommender* (**ANR**), which utilises recent advances in deep learning

such as the neural attention and co-attention mechanisms. First, we perform aspect-based representation learning for users and items by designing an attention mechanism to focus on the relevant parts of these reviews while learning the representation of aspects on the task. Next, we estimate aspect-level user and item importance in a joint manner by extending the idea of co-attention, which allows us to model the finer-grained interactions between users and items. Finally, the aspect-level representations and importance are combined in an effective and intuitive manner to obtain the overall numerical rating. Similar to existing deep learning-based recommendation systems, we are able to train our proposed model in an end-to-end fashion and capitalise on the superior representation learning capabilities of neural networks. At the same time, in contrast to existing aspect-based recommender systems, our proposed model do not require external supervision or upstream tools in order to obtain the aspects.

1.3.2 Adaptive Framework for Seamless Warm- and Cold-start Recommendation

Recommender systems are frequently deployed in dynamic and constantly evolving environments. For example, there are often new users joining the system, or new items might be added to the catalogue. Although textual reviews can be extremely helpful for modelling users and items with limited interactions, they are ultimately a form of *explicit feedback* which are not available for these completely new users and items. As such, in this study, we explore hybrid models which incorporate other forms of content information such as **user demographics** (e.g. the *age, gender, and occupation* for a user), as well as **item attributes** (e.g. the *release year, genre, and language* of a movie).

1.3.2.1 Motivations

Undoubtedly, the ability to attract and retain new users/items is a key factor for any recommendation system to remain successful and relevant. In practice, it can be rather challenging to separate the cold-start scenarios from the typical warm-start recommendations and determine the appropriate type of model to be used. For instance, a user who has rated just one or two movies might not have sufficient historical information to be modelled accurately by CF-based methods, and in this case, a content-based (**CB**) model could be a better choice. Furthermore, relying on different models for each possible scenario has several other implications, e.g. additional parameters and training

cost, as well as the maintainability of the recommendation system. As such, it is often desirable to have a ***single, unified model*** which is capable of handling *both* warm- and cold-start recommendation seamlessly. However, we argue that existing hybrid models are ***not*** equally adept at handling the different scenarios. More specifically, when generating recommendations for completely new users or items, they are either **(1)** fully content-driven, or **(2)** rely on pretrained CF representations of existing users and items.

In addition, we identify several challenges in evaluating the performance of a recommender system on cold-start scenarios. First, although recommendation systems rely mainly on the content information for cold-start situations, we find that the available user and/or item contents are usually lacking. Particularly, for many datasets (e.g. *ML-100K* & *ML-1M*) which are often used in existing work related to cold-start recommendation, we found that there are a considerable number of users (as well as items) which cannot be differentiated from one another based on their contents alone. For instance, 39.7% of the users in *ML-1M* have ‘*non-unique*’ contents, i.e. for these users, there is at least one other user with *identical* content information. This is a significant problem for cold-start scenarios, whereby the only information we have for a new user or item is its content. Hence, it is impossible to distinguish two (or more) users who have identical attributes and yet, different preferences. Next, we observe that most existing work either **(1)** do not perform evaluation explicitly on individual cold-start scenarios, i.e. *user cold-start*, *item cold-start*, and *both user & item cold-start*, or **(2)** solely compare with models using the same approach as their proposed method. For example, [124, 125] report a **single value** representing the model performance on *both* warm- and cold-start recommendation, which obfuscates its true performance for the cold-start scenarios.

1.3.2.2 Methodology

In this study, we present a novel hybrid model with an innovative *neighbourhood-based* component, which fuses the CF-based latent space together with the content-based space, and maximises the accuracy for *both* warm- and cold-start recommendation. Our proposed ***Collective Collaborative Filtering*** (**CCF**) component is designed to learn the collective preferences of a user (or item) neighbourhood with respect to the target user (or item). Based on our intuition that users/items with similar content information may have similar latent representations, we define the neighbourhood for a given user as the set of (existing) users having the most similar contents. The item neighbourhoods are defined analogously.

Our motivation for CCF is simple: We would like to draw on the collective preferences of similar users (or items) in order to learn a better representation for the target user (or item). First, this allows users and items with similar content information to share statistical strength with one another in the process. Furthermore, CCF is equally applicable to *both* warm- and cold-start scenarios by design, thus allowing new users and items to harness the collaborative information gleaned from existing users and items. Last but not least, the different representations (CF, CB, as well as our proposed CCF) are combined via a neural attention mechanism, thus allowing the system to generate recommendations satisfactorily for different scenarios via a single model.

CCF is based on an adaptation of the Multi-head Attention mechanism proposed in [141] which, to the best of our knowledge, has never been utilised in recommender systems for maximising the accuracy of *both* warm- and cold-start recommendation. We believe that the correlation between *user contents and user preferences* (or *item contents and item profiles*) cannot be easily expressed as a single function between the two modalities. For example, similarities in user preferences might be based on a subset of the user attributes, e.g. *age & occupation*. Hence, Multi-head Attention is employed to capture the subtle differences in the preferences with respect to changes in the content space via multiple perspectives (i.e. attention networks), and this inherently facilitates a better reflection of the complex relationships between the content space and the CF-based latent space.

1.3.3 The Datasets Dilemma

Recent studies have pinpointed several worrisome problems that appear to undermine years of hard work within the recommendation systems community. First, the authors in [30] have discovered that many recently proposed methods are in fact not reproducible, and for the remaining methods which could be reproduced, they are often outperformed by simpler but well-tuned baselines. Next, [70] has shown that the results obtained from evaluating item recommendation using a randomly sampled subset of candidate items, which is a *common practice* in recent research, can be inconsistent with results acquired using the complete set of candidate items. The former, also known as *sampled metrics*, is no more than a high-bias, low-variance estimator of its exact version, and results obtained using sampled metrics can be highly misleading as the bias is recommender-dependent. Very recently, [132] highlighted an important issue which is often the cause

of unreproducible evaluation and unfair comparison, i.e. the lack of any effective benchmark for evaluation. The authors noted that *researchers often choose different datasets heuristically*, and there are many seemingly trivial factors, e.g. data pre-processing and splitting strategies, evaluation metrics, etc., which can influence the recommendation performance.

1.3.3.1 Motivation and Methodology

Although the aforementioned studies have all brought attention to critical issues hampering measurable progress for recommendation in general, there might be other problems that we are oblivious to. In fact, we are aware of one such problem that has long been overlooked by the community: ***How much do we really know about recommendation datasets?***

To answer this question, we perform various analyses and experiments in this study to improve our understanding of the datasets used for evaluating *top-K item recommendation* in a systematic and comprehensive manner. Specifically, we adopt the following three steps: **(1)** First, we examined the proceedings of 5 top-tier conferences (*KDD*, *SIGIR*, *TheWebConf*, *WSDM*, and *RecSys*) for the past 5 years (2016 to 2020), and narrowed it down to 48 papers which address the problem of top-K item recommendation based on implicit feedback. There are a total of 45 publicly available datasets which have been used in one or more of these papers, and we performed various analyses to illustrate **how** different datasets have been utilised in recent papers. **(2)** After which, we introduce a set of data characteristics which can be used as a representation for each of these datasets, and at the same time, are easily obtained by analysing the (binary) user-item interaction matrix. We leveraged these characteristics to clearly understand **what** are the similarities as well as differences between various datasets. **(3)** Finally, we performed an empirical study using several representative methods, such as Weighted Matrix Factorization (**WMF**) [58], Item k-Nearest Neighbour (**ItemKNN**) [115], and Mult-VAE [82], to find out **if** the choice of datasets used for evaluating item recommendation could influence the observations and/or conclusions obtained.

In summary, this study undertakes a retrospective survey on the datasets used for top-K recommendation and seeks to shed light on whether the current practice of selecting datasets arbitrarily would lead to unintended (and perhaps, negative) consequences. To the best of our knowledge, we are the first to provide an extensive overview of recommendation datasets, which is a fundamental but often neglected aspect of the evaluation procedure.

1.4 Summary of Contributions

In this dissertation, we propose novel hybrid models to overcome a long-standing challenge for personalised recommendation, i.e. the *cold-start problem*, by leveraging different types of content information in conjunction with recent advances in deep learning. Furthermore, we identify and examine challenges, as well as experimental issues, that persist in personalised recommendation. The contributions are summarised as follows:

- We propose a novel aspect-based neural recommender system which performs aspect-based representation learning for users and items, by designing an attention mechanism to focus on the relevant parts of these reviews while learning the representation of aspects on the task. Furthermore, we estimate aspect-level user and item importance in a joint manner using the idea of co-attention, which allows us to model the finer-grained interactions between users and items. To the best of our knowledge, we are the first to propose an end-to-end neural aspect-based recommender system which concurrently addresses the above-mentioned requirements. Extensive experiments have been conducted on 25 benchmark datasets from *Amazon* and *Yelp* to evaluate our proposed model against several state-of-the-art baselines such as DeepCoNN [168], D-Attn [119], and ALFM [25]. Furthermore, we investigate how the different components in our proposed model contribute to its effectiveness. In particular, we include a qualitative analysis of the aspects which are learned automatically by our model without any external supervision.
- We consider the problem of generating recommendations satisfactorily in *both* warm- and cold-start scenarios via a *single model*. To this end, we present an innovative hybrid model using our proposed neighbourhood-based *Collective Collaborative Filtering* (**CCF**) component, which aims to maximise the performance of a recommender system in the presence of a dynamic and constantly evolving environment. Evaluation procedures adopted in existing work for cold-start recommendation are examined critically, and a key finding is that the content information in commonly used datasets are often insufficient for differentiating one user (or item) from another. Furthermore, we adopt an evaluation framework which explicitly shows the model performance for both warm-start recommendation and the various cold-start scenarios, i.e. *user cold-start*, *item cold-start*, and *both user & item cold-start*. Experimental results on four datasets, including *ML-100K* & *ML-1M*, show that our

proposed model consistently outperforms baseline methods for various cold-start scenarios, and achieves the best overall performance in all four datasets.

- We conduct a retrospective survey to provide an in-depth look at a fundamental but often neglected aspect of the evaluation procedure, i.e. the datasets themselves. To do so, we adopt a systematic and comprehensive approach to understand the datasets used for implicit feedback based top-K recommendation. We start by examining recent papers from several top-tier conferences to find out how different datasets have been utilised thus far. Next, we study the characteristics of these datasets to understand their similarities and differences. Finally, we conduct an empirical study to determine whether the choice of datasets used for evaluation can influence the observations and/or conclusions obtained. Our findings suggest that greater attention needs to be paid to the selection process of datasets used for evaluating recommender systems in order to improve the robustness of the obtained results.

Chapter 2

Literature Review

In this chapter, we first provide an overview of recommendation systems in Section 2.1. Specifically, we include a brief introduction to the Collaborative Filtering (**CF**) technique, and touch on a long-standing challenge for personalised recommendation, i.e. the *cold-start problem*. In Section 2.2, we look at how recommender systems can make use of textual reviews as a potential solution for the cold-start problem. Section 2.3 provides a review of how existing recommender systems utilise user and item contents as an alternative approach to solving the cold-start problem. Finally, Section 2.4 highlights several existing works which have expressed concern about the experimental issues present in recommendation systems, as well as a brief discussion on the data characteristics utilised in our retrospective study on the ‘*datasets dilemma*’.

2.1 Overview of Recommendation Systems

2.1.1 General Recommendation

With the shift towards an increasingly digital lifestyle, recommender systems play a critical role in helping consumers to find the best product or service amongst a variety of options. Unsurprisingly, personalised recommendations have become part and parcel of our daily lives. For instance, recommender systems are widely adopted across various domains, including e-commerce platforms (e.g. *Amazon*, *eBay*, *Taobao*), location-based social networks (e.g. *Yelp*, *Foursquare*), and social media (e.g. *Facebook*, *Instagram*, *Twitter*).

Figure 2.1: Illustration of a typical recommendation system

Broadly speaking, recommendation systems seek to capture the distinct preferences of different users, and leverage upon this information to generate items of interest for each user in a personalised manner. As illustrated in Figure 2.1, most, if not all, recommender systems are primarily driven by user feedback. The interactions between a recommender system and its users form a virtuous cycle, whereby (1) having more feedback from a particular user would allow the system to better understand his/her preferences, (2) a better understanding of a user's preferences lead to better recommendations for that user, and (3) better recommendations would in turn encourage that specific user to interact more with the recommender system. Naturally, a key consideration for designing any recommendation system would be the type of user feedback available. There are two main types of feedback (also known as interaction data) which are commonly utilised in existing recommender systems: (1) *Implicit feedback*, such as views, clicks, and purchases, as well as (2) *Explicit feedback*, such as ratings and textual reviews. While the goal remains the same, i.e. recommend items of interest to a given user, the problem formulation and evaluation procedure tends to be quite different.

2.1.1.1 Recommendation with Explicit Feedback

As mentioned previously, explicit feedback can include both numerical ratings as well as textual reviews. In addition, when considering explicit feedback, the recommendation process is typically formulated as a rating prediction problem. As shown in Figure 2.2, we have a $M \times N$ matrix $\mathbf{R}^1$ representing the interactions between a set of M users and

5	?	?	?
?	3	?	2
4	?	5	?
?	?	1	?

Figure 2.2: Recommendation with explicit feedback (*Best viewed in colour*)

the set of N items. For each user u , we would like to estimate the rating $\hat{r}_{u,i}$ for any unseen user-item pair, i.e. the unknown rating of a given user u towards an item i that he/she has not interacted with before. Put simply, given the observed entries (denoted with black in Figure 2.2), the goal of the recommender system is to recover the missing entries (denoted with red in Figure 2.2) as accurate as possible. Items with the highest predicted ratings are then recommended to each user.

The evaluation process for recommendation systems using explicit feedback tends to be rather straightforward as well. Predicted ratings are compared against ground-truth ratings using pointwise metrics such as Mean Absolute Error (**MAE**) and Root Mean Squared Error (**RMSE**). As compared to MAE, RMSE would penalise large prediction errors more severely. For pointwise metrics, a smaller error margin would imply that the recommender system performs better at recovering the missing entries, and consequently, be able to recommend the optimal set of items to the target user.

2.1.1.2 Recommendation with Implicit Feedback

However, a common argument against using explicit feedback in recommendation systems is that they tend to be harder to obtain in practice. More specifically, as its name

¹ Unless stated otherwise, we denote vectors with bold lower-cases, and bold upper-cases are reserved for matrices or high dimensional tensors.

Figure 2.3: Recommendation with implicit feedback (*Best viewed in colour*)

implies, explicit feedback requires active participation from the end users, and it could be rather challenging to encourage users to provide them consciously. Therefore, many recommendation systems have also considered using implicit feedback, such as views, clicks, purchases, etc. Implicit feedback are often collected ‘*in the background*’, i.e. they are gathered automatically when users are engaged with the platform. Nevertheless, we will see later on that there are certain disadvantages and/or pitfalls associated with implicit feedback.

As illustrated in Figure 2.3 (a), recommendation with implicit feedback is essentially based on a binary interaction matrix between the users and items. However, as we can see in Figure 2.3 (b), the goal in this case is to predict the probability of observing each missing entry. For each user, we can then rank the items that he/she has not interacted with according to their corresponding probabilities, and recommend the top-K items. When considering implicit feedback, the recommendation system is often evaluated using classification and/or ranking-based metrics. Classification metrics, such as Precision and Recall, simply verify whether the ground-truth item(s) can be found within the top-K list, whereas ranking-based metrics, such as Normalized Discounted Cumulative Gain (**nDCG**) and Mean Reciprocal Rank (**MRR**), take into consideration the position(s) of the ground-truth item(s) within the top-K list.

While it may seem like a better option to work with implicit feedback since they are relatively easy to obtain, there are various caveats associated with it. Due to the nature of implicit feedback, the observations (or the lack thereof) do not necessarily reflect preferences. For example, if a user didn't watch a particular movie, it is possible that **(1)** he/she dislikes that movie, or **(2)** he/she is not even aware of such a movie. On the other hand, even when we are aware that a user has interacted with an item, it is impossible to determine whether the user likes or dislikes it. For instance, a user may have purchased a watch, but it could have been bought as a gift for someone else. As such, the observation cannot be used as an indication of the user's preference for the watch.

2.1.1.3 Different Types of Recommendation Systems

Recommendation systems are used across various domains with distinctively different item types. For example, we can recommend books, music, Points of Interest (**POI**), etc. In addition, we might be recommending items to a group of users instead of a single user. This brings about various types of specialised recommender systems, e.g. sequential recommendation [46, 135, 154], POI recommendation [37, 161], group recommendation [15, 143, 114, 44], etc. Furthermore, they would often exploit problem-specific characteristics that are unique to their application scenarios as well as other forms of contextual information such as the user's current location, the time of the day, etc. For instance, POI recommendation systems tend to consider key factors such as spatial influence and temporal influence [90]. While specialised recommender systems are essential, we simply consider the general case of recommending the most suitable item for a given user and do not restrict ourselves to any specific item type in this dissertation.

2.1.2 Collaborative Filtering

A variety of recommender systems have been proposed over the years, but some of the most widely used models [115, 83, 113, 69, 51] are still based on the fairly simple concept of Collaborative Filtering (**CF**) [12]. The key idea behind CF is that people often get the best recommendations from someone with comparable taste/preference as themselves. CF generates personalised recommendation by leveraging information obtained from other like-minded users – the *filtering* process (i.e. recommending a subset of candidate items) is performed with the assistance of other users (i.e. in a *collaborative*

manner). Naturally, CF-based algorithms are predominantly based on analysing the historical interactions of the target user, as well as those of other similar users.

There are two main categories of CF-based systems: **(1)** Memory-based methods (e.g. neighbourhood-based algorithms), and **(2)** Model-based methods (e.g. latent factor models). We note that there are also several hybrid models, such as SVD++ [68] and Collaborative Memory Network (**CMN**) [34], which are essentially a combination of these two approaches.

Memory-based methods [54, 115, 83] leverage upon the historical interactions of the most similar users (or items), to perform recommendations. In order to identify these neighbouring users (or items), they make use of similarity measures such as Jaccard Similarity, Cosine Similarity, Pearson Correlation, etc. Memory-based methods such as User k-Nearest Neighbour (**UserKNN**) and Item k-Nearest Neighbour (**ItemKNN**) tend to be rather intuitive and allow the end user to understand why a certain recommendation has been generated. For example, when using a user-oriented approach, we can offer an explanation such as “We are recommending *Movie X* to you because other similar users (who have watched Movies Y and Z) enjoyed *Movie X* as well”.

Model-based methods are an alternative approach that has gained widespread popularity recently due to the continuous emergence of new machine learning and/or deep learning techniques. Amongst them, one of the most popular methods is the latent factor model. For example, the well-known Matrix Factorization (**MF**) [69] algorithm, and its variants [113, 72, 51], represent users and items in a shared low-dimensional latent space based on the observed user-item interaction data. Considering a d -dimensional latent space, a user u is represented as $\mathbf{v}_u \in \mathbb{R}^d$ and an item i can be represented by $\mathbf{v}_i \in \mathbb{R}^d$. Similar users are embedded close to one another, and similar items are embedded close to each other. The predicted user-item preference $\hat{r}_{u,i}$ can then be inferred via a function h , i.e. $\hat{r}_{u,i} = h(\mathbf{v}_u, \mathbf{v}_i)$, where $h(\cdot, \cdot)$ is usually the inner product between the input latent representations. User and item biases, as well as a global bias term, can be incorporated to improve its inductive bias [69]. Due to the simplicity and effectiveness of latent factor models such as MF, content information can be easily integrated without changing its learning framework.

2.1.3 Cold-start Problem

Notably, there are various challenges which affect most, if not all, recommendation systems in general. For example, recommender systems can suffer from popularity

bias [105, 101, 171, 169], whereby popular items (e.g. blockbuster movies) tend to get recommended over and over again, and less popular (or niche) products are less likely to be discovered. In addition, due to the collaborative nature of CF-based algorithms, they tend to be susceptible to shilling attacks [42, 71]. Shilling attacks can be characterised by fake user profiles which are introduced by an adversarial party in order to harvest recommendation outcomes towards a malicious desire. As typical recommender systems are unable to distinguish genuine user profiles from fake user profiles, shilling attacks can be detrimental to its predictive performance.

In this dissertation, our focus is more on a long-standing challenge for recommendation: The *cold-start problem* [117, 133, 104, 38, 89]. As described in [38], the cold-start problem can be further categorised into two distinct scenarios. In the first instance, the cold-start problem in the “*narrow sense*” refers to completely new users and items with *no historical information*. For these completely new users and items, CF-based algorithms are not applicable at all, since we would not be able to derive their representation (as in latent factor models) or determine its neighbouring users/items (as in neighbourhood-based models). In the second instance, the cold-start problem in the “*wide sense*” includes situations whereby we would like to generate recommendations for relatively new users and items with *limited historical information*. The key difference here is that CF-based methods can be applied for such users and items, but the performance may not be satisfactory as the system still lacks a good understanding of the preferences (properties) of these users (items).

Possible solutions to the cold-start problem include switching entirely to a content-based recommendation system [92, 107], as well as augmenting a CF-based approach with content information in order to obtain a hybrid model. As noted previously, depending on the application scenario, there could be several types of contents available. In Section 2.2, we look at how recommender systems can make use of textual reviews to combat the cold-start problem in the “*wide sense*”. However, textual reviews are ultimately a form of explicit feedback, which are not available for completely new users and items. Therefore, in Section 2.3, we provide a review of how existing recommendation systems utilise user and item contents to deal with the cold-start problem in the “*narrow sense*”. Accordingly, in this dissertation, we put forward two novel models to handle these two scenarios: **(1)** A neural recommendation system utilising textual reviews in Chapter 3, and **(2)** A hybrid model leveraging user and item contents in Chapter 4.

2.2 Recommendation with Reviews

Recent work [95, 84, 7, 4, 156, 6, 134, 168, 119, 18] have all shown the importance of using reviews to improve the performance and reliability of recommender systems. In this section, we focus on several key areas that are highly relevant to our study on aspect-based neural recommendation: **(1)** Deep Learning-based Recommender Systems, **(2)** Aspect-based Recommendation Systems, as well as **(3)** Neural Attention & Co-Attention.

2.2.1 Deep Learning-based Recommender Systems

In recent years, deep neural networks have been successfully applied to a large variety of tasks, such as natural language processing, computer vision, and speech recognition [41], while often achieving state-of-the-art performance in these domains. Many recently proposed recommender systems have also turned to various deep learning techniques in order to work with the textual information, such as the use of Denoising Autoencoders in [147, 77], Recurrent Neural Networks (**RNNs**) in [4, 6], and most notably, the use of Convolutional Neural Networks (**CNNs**) [66] in [65, 168, 18, 119, 21] due to its great success in many other natural language processing tasks [63, 29]. We provide an illustration of the general architecture for deep learning-based recommender systems in Figure 2.4.

Generally, these methods try to capitalise on the strong representation learning capabilities of neural networks to learn latent feature representations from the reviews for both users and items. However, there are several noticeable drawbacks for these existing deep learning-based methods. Firstly, attempting to ‘*compress*’ all the available reviews for a user (or item) into a single latent representation may not be ideal. Besides the potential loss of useful information (due to the pooling techniques used in such models), there is an inherent risk of including the irrelevant parts of these reviews, resulting in a noisy and possibly inaccurate representation of the user (or item). Furthermore, the user representation is learned independently of the item representation, and vice versa. In fact, we noticed that the only interaction between users and items often occurs at the final prediction layer, where the learned user and item embeddings are used for the overall rating estimation using methods such as Factorization Machine (**FM**) [111] in [168, 18], Feedforward neural networks in [21], or simply via the inner product as in [119]. Finally, due to the black box nature of these models, it can be difficult to provide a convincing insight as to why the user rated an item in that particular manner.

Figure 2.4: General architecture for deep learning-based recommender systems

2.2.2 Aspect-based Recommendation Systems

Prior to the surge in utilising deep learning techniques for recommendation, a highly popular line of research focuses on either extracting or learning aspects from these textual reviews. We provide an illustration of these two types of aspect-based recommender systems in Figure 2.5.

As shown in Figure 2.5 (a), the first type of aspect-based recommendation systems such as EFM [165], TriRank [48], LRPPM [23], AGSG [45], and the recently proposed SULM [11], relies on external Sentiment Analysis (**SA**) tools [110, 164] to analyse the review contents and uncover the mentioned aspects together with their opinions and/or sentiments. To be more specific, they transform the reviews into $(aspect, sentiment)$ pairs or $(aspect, opinion, sentiment)$ triplets to be used as the inputs to their model. Besides the fact that they are not self-contained, the performance of such models largely

Figure 2.5: Existing aspect-based recommendation systems

depends on the quality of these SA tools, i.e. how well they are able to extract such information from these textual reviews. Furthermore, AGSG [45] is designed for Point-of-Interest (**POI**) recommendation, and it requires the location and category information associated with such items of interest.

An alternative type of aspect-based systems [32, 156, 25] automatically learn these aspects from the review contents, typically through the use of generative statistical models such as Latent Dirichlet Allocation (**LDA**) [13]. JMARS [32] and FLAME [156] are both integrated probabilistic frameworks which represent each aspect as a distribution over the words in the vocabulary, and we show the graphical model of FLAME [156] in Figure 2.5 (b) as an example. More recently, a newly proposed ALFM [25] includes an Aspect-aware Topic Model (**ATM**) which models each aspect as a multinomial distribution over the same set of K latent topics, each of which is defined as a multinomial distribution over the vocabulary. The output from ATM, i.e. the aspect-level user preferences and item characteristics, is subsequently used as part of their latent factor model for estimating the overall rating via the MF approach.

A key advantage to these aspect-based methods is that they are generally more transparent and intuitive, as most of them are capable of providing explanation in order to support their recommendations. However, existing aspect-based systems either **(1)** depend on external tools or input, or **(2)** does not emphasise on how different parts of the review may contribute differently to the overall satisfaction. Additionally, they fail to

consider the varied aspect-level importance for both users and items while taking into account the target user and item in question (as and when necessary).

2.2.3 Neural Attention & Co-Attention

In this section, we provide an overview of the key techniques used in our proposed model, namely **(1)** Neural Attention and **(2)** Neural Co-Attention. Loosely based on the idea of visual attention in humans, the neural attention mechanism is one of the most exciting developments in the field of deep learning, and has been successfully applied to a multitude of machine learning tasks such as machine translation and abstractive summarisation [5, 112, 141]. More recently, it has also been utilised in various recommender systems [119, 21, 137, 158, 22, 16, 55] for vastly different purposes. For example, in [21], the authors proposed using the attention mechanism to select ‘useful’ reviews to be used for deriving the user and item representations. As for the problem of group recommendation, [16] finds that the contribution of a user may vary depending on both the group of users and the target item, and uses neural attention to determine the relative importance of each user within a specific group with respect to the item of interest. In essence, it equips neural networks with the ability to focus on selective parts of the input, such as a certain region in an image or even specific words/sentences in a textual document.

For example, if we are trying to determine the suitability of some restaurant based on its *price*, not all words in its set of user reviews would be equally important. Almost instinctively, we would turn our attention to a subset of informative words in these reviews, such as *expensive*, *cheap*, *costly*, *affordable*, etc. This is the central idea behind how our proposed model is able to automatically derive the aspect-level representations from the corresponding textual contents using a fully data-driven approach. Basically, the model learns to identify a subset of vocabulary words which are highly relevant given some target aspect via the neural attention mechanism.

A closely related technique is neural co-attention [93, 160], which can be roughly described as a form of pairwise neural attention. In certain scenarios, it can be beneficial to jointly reason about the attention for a pair of related entities, such as between the image and question for the task of Visual Question Answering in [93]. The basic idea behind the neural co-attention mechanism is that the attention for one entity (e.g. image) is learned w.r.t. the representation(s) of the other entity (e.g. question), and vice versa.

For our model, we extend this particular idea of a two-way neural attention for the estimation of the aspect-level user and item importance, enhancing it with the ability to be aware of the current user-item pair. The *aspect-level **item** representations* are used as the context to influence the learning of *aspect-level **user** importance*, and conversely, the *aspect-level **item** importance* are conditioned on the *aspect-level **user** representations*. In other words, our proposed model takes into consideration the target item when inferring the importance of each aspect for the user, and vice versa.

2.3 Recommendation with User and Item Contents

As noted previously, textual reviews are ultimately a form of explicit feedback, which are not available for completely new users and items. Therefore, in this section, we provide a review of how existing recommendation systems utilise user and item contents in order to generate recommendations for such users and items. For example, user contents can be as simple as the **user demographics** (e.g. the *age, gender, and occupation* for a user), and **item attributes** (e.g. the *release year, genre, and language* of a movie) can be utilised as the content information for an item.

With user and item contents, it is possible to switch entirely to a content-based (**CB**) recommendation system. However, unlike CF-based methods, CB methods only capture ‘*individual preferences*’ as they do **not** consider other users/items when learning the user and item representations. Consequently, the performance of CB methods for in-matrix predictions (i.e. inferring the user-item preferences for existing users and items) often lags far behind CF-based methods.

Shortcomings of the aforementioned ‘*homogeneous models*’, i.e. purely CF-based or purely content-based, have propelled the proliferation of hybrid models, i.e. methods which utilise user-item interaction data in conjunction with the users and items’ contents. We are primarily interested in hybrid models designed for (and/or evaluated on) cold-start scenarios, hence we provide an overview of two common approaches used for constructing such hybrid models.

2.3.1 End-to-end Hybrid Architectures

Due to its simplicity and effectiveness, content information can be easily integrated into MF without changing its learning framework. Several pioneering hybrid models [2, 146,

147] are an augmentation of a probabilistic variant of MF [113], while recently proposed hybrid models [124, 125, 162, 57] tend to be a direct extension of the basic MF framework. In this dissertation, we consider the latter as the starting point for our proposed hybrid model in Chapter 4.

Figure 2.6: A generic end-to-end hybrid model

Figure 2.6 shows an example of an end-to-end hybrid model which augments the basic MF framework with user and item contents. Within the user network, the user representation $\mathbf{v}_u$ consists of two components, $\mathbf{v}_{u,CF}$ and $\mathbf{v}_{u,CB}$, which are learned based on the user’s historical feedback and content information, respectively. The item representation $\mathbf{v}_i$ can be obtained in a similar manner. Furthermore, unlike [2, 146, 147], recent hybrid models (e.g. [124, 125, 162, 57]) learn to combine the component vectors adaptively (often via a dedicated attention network). As such, they can fully utilise both the historical feedback as well as the content information for each user and item in an *individualised fashion*.

However, as illustrated in Figures 2.7 (a) and 2.7 (b), all aforementioned hybrid models do suffer from a critical drawback: The final user/item representation (i.e. $\mathbf{v}_u$ and $\mathbf{v}_i$) **degenerates** to the content-based part (i.e. $\mathbf{v}_{u,CB}$ and $\mathbf{v}_{i,CB}$) due to the absence of past interactions for a completely new user/item. The inability to leverage the invaluable collaborative information leads to a clear disadvantage for new users and items in such models, and motivates us to design an innovative *neighbourhood-based component* as a simple but effective solution to bridge this gap.

(a): Completely new user

(b): Completely new item

Figure 2.7: Degeneration of a generic end-to-end hybrid model

2.3.2 Mapping-based Hybrid Architectures

Albeit often overlooked, there is another category of hybrid models designed specifically for cold-start scenarios. [38, 139, 9, 8, 145, 10, 170] adopt a modular, mapping-based

approach, whereby CF-based user and item latent representations are learned in the *first step* through CF-based methods such as MF, and mapping (or transformation) functions from the content space to the latent CF space are derived separately in the *second step*.

However, these models suffer from the ‘*error superimposition*’ problem [170], whereby each step in the pipelined approach incurs a different error term, and this leads to an increase in the total error during inference. In addition, mapping-based models tend to assume a *one-to-one mapping* from the content space to the CF-based latent space, and simply learn a *single* mapping function from a content-based representation to the corresponding CF-based latent representation. Furthermore, the mapping function is usually shared across all users (or items). To provide ‘*individualised*’ transformation for each user and item, Heater [170] adopted a Mixture-of-Experts [120] structure in its mapping functions.

Even then, mapping-based methods are not self-contained due to their reliance on other models (e.g. MF) for deriving the pretrained CF-based representations in the first step. This could result in poor performance on datasets with high sparsity, whereby CF-based methods may fail to model the preferences satisfactorily. In addition, as noted in [170], several mapping-based models [116, 118, 76] are only capable of handling one-sided cold-start scenarios, i.e. new users or new items.

End-to-end hybrid models (Section 2.3.1) and the aforementioned mapping-based models are usually not compared with one another in previous work. As far as we know, we are the first to compare these two approaches empirically (in Chapter 4).

2.3.3 Meta-Learning

Other than these two directions, i.e. end-to-end hybrid models and mapping-based models, we note that there are several existing models (e.g. [140] and [73]) applying the idea of meta-learning [142, 74] to the problem of cold-start recommendation. [140] proposed two different architectures for using metric-based meta-learning to resolve the *item cold-start* problem (i.e. new tweets on Twitter), while [73] adopts an optimisation-based meta-learning approach for building a general recommender system. As [140] is designed for the one-sided cold-start scenario and [73] aims to adapt quickly to new users or items *given a few interactions*, we do not compare with them in our study.

2.4 The Datasets Dilemma

Nevertheless, while it is paramount to constantly develop new algorithms to advance the state-of-the-art, we believe that the evaluation procedure needs to be well-defined and robust so as to guarantee the validity of the obtained results. As we will see shortly in Section 2.4.1, recent studies have pinpointed several experimental issues in recommendation systems.

However, we believe that there remains a specific problem that has long been overlooked by the research community: *How much do we really know about recommendation datasets?* To be more specific, we observe that there is a tendency for many existing works to select datasets arbitrarily in their empirical evaluations. This becomes an issue when researchers try to compare their proposed model against different baseline methods empirically. For example, some researchers may try to utilise the same datasets that were used in the baseline methods. However, if Baseline 1 uses Datasets A and B, while Baseline 2 was evaluated on Datasets C and D, which datasets should we be using in our empirical evaluation? This could become extremely problematic when we consider more baseline methods, and at the same time, each of those baseline methods performed their evaluation on a disjoint set of datasets. Furthermore, it could be possible that a dataset used in an existing work is no longer available for various reasons. In that case, could we perform the experiments with a ‘*similar*’ but publicly available dataset? The key consideration here would be how similarities and differences between datasets can be measured in an objective manner. There are a variety of issues associated with the choice of datasets utilised in an empirical study, and we term the above-mentioned phenomenon as the “*datasets dilemma*”.

2.4.1 Experimental Issues in Recommender Systems

Lately, several studies have pointed out critical experimental issues related to recommender systems. First, the authors in [30] have discovered that many recently proposed methods (which are often based on deep learning) are in fact not reproducible. For the remaining methods which could be reproduced, they often fail to outperform simple, but fine-tuned, baseline methods such as ItemKNN [115] and RP3beta [106].

Next, [70] has shown that the results obtained from evaluating top-K item recommendation using a randomly sampled subset of candidate items, which is a *common*

practice in recent research, can be inconsistent with results acquired using the complete set of candidate items. The former, also known as *sampled metrics*, is no more than a high-bias, low-variance estimator of its exact version, and results obtained using sampled metrics can be highly misleading as the bias is recommender-dependent. Although ranking the relevant items together with a smaller set of random items helps to speed up the computation of metrics, it leads to an inaccurate evaluation and should be avoided.

Finally, [132] takes a broader view of the entire evaluation process and analyses the influence of different factors, e.g. utilised datasets, evaluation metrics, hyperparameter tuning strategies, etc., on the recommendation performance. Notably, they highlighted an important issue which is often the cause of unreproducible evaluation and unfair comparison, i.e. the lack of any effective benchmark for evaluation. Furthermore, the authors noted that *researchers often choose different datasets heuristically*, and there are many seemingly trivial factors, e.g. data pre-processing and splitting strategies, evaluation metrics, etc., which do influence the recommendation performance. To facilitate efficient, fair, and reproducible comparison, [132] constructed and released an evaluation benchmark with 6 datasets by considering their popularity and domain coverage. However, we argue that the popularity of a dataset is essentially a subjective measure. More specifically, the popularity of a dataset is unlikely to be correlated with its suitability for evaluating recommendation algorithms in general. For instance, many authors may be inclined to choose smaller datasets simply because the results can be obtained in a relatively shorter amount of time. In addition, we note that the different recommendation algorithms are evaluated using *sampled metrics* in [132]. However, as noted in [70], the observations obtained might not be completely accurate as the results tend to be biased.

2.4.2 Data Characteristics of Recommendation Datasets

In order to measure the similarities and differences between individual datasets, we have to be able to characterise them using a set of objective measures. Inspired by [1] and [31], our study (in Chapter 5) considers 5 different data characteristics which can be easily derived from the user-item interaction matrix in the *implicit feedback* setting.

The authors in [1] used an exploratory modelling (**EM**) approach to investigate the impact of different dataset characteristics on the performance of various recommendation algorithms for the *rating prediction* task. More recently, [31] adopted an identical approach in order to assess the *robustness* of several recommender algorithms against *shilling attacks*, using the same characteristics as [1].

Figure 2.8: Illustration of the exploratory modelling approach used in [1] and [31]

Although we utilise similar data characteristics as [1] and [31] in our study, there are several important differences: **(1)** We focus on the task of *implicit feedback* based top-K recommendation, while [1] and [31] consider rating prediction and robustness against shilling attacks, respectively, using *explicit feedback*. **(2)** We adopt a dataset-centric approach and conduct our empirical study on 15 distinctly different datasets, while both papers utilise 4 arbitrarily chosen datasets. **(3)** Most importantly, we evaluate various recommendation algorithms on the ‘*complete*’ datasets, i.e. just as it would be used in any other empirical study. On the other hand, as shown in Figure 2.8, these 2 papers utilise a random sampling approach to extract multiple submatrices from the original user-item interaction matrix. The data characteristics and recommendation performance are then derived from these submatrices in order to fit a linear regression model as part of their Exploratory Modelling (**EM**) approach. However, the characteristics as well as performance obtained via the submatrices could in reality end up being very different from that of the actual interaction matrix.

2.5 Summary

Figure 2.9: Relationships between different research problems and our contributions

To summarise, when it comes to addressing **challenges** such as the *cold-start problem*, we find that there are several shortcomings in existing literature.

Textual reviews can be used as a complementary information source to deal with the cold-start problem for relatively new users and items with *limited historical information*. However, existing work either fail to consider the importance of the different parts of the reviews, or they tend to depend on external tools or input. In **Chapter 3**, we put forward a novel hybrid model to fully utilise textual reviews while overcoming the drawbacks of existing approaches.

As an alternative information source, user and item contents (e.g. user demographics and item attributes) can be leveraged to generate recommendations for completely new users and items. However, existing work tend to be **(1)** completely content-driven and fail to make use of the historical interactions, or **(2)** only capable of handling one-sided cold-start scenarios, i.e. new users or new items. In **Chapter 4**, we introduce a new

hybrid model which is equally adept at handling both new and existing users/items. Furthermore, we critically examine the evaluation procedures adopted in existing work for cold-start recommendation.

As for **experimental issues** such as the *choice of datasets* used for empirical evaluation, we are not aware of any existing work which have thoroughly investigated this specific aspect of the overall evaluation process. Therefore, in **Chapter 5**, we adopt a systematic and comprehensive approach to understand the datasets used for implicit feedback based top-K recommendation.

Figure 2.9 illustrates the relationships between the different research problems and our contributions in the subsequent chapters.

Chapter 3

Aspect-based Neural Recommender

In this study, we present our proposed **Aspect-based Neural Recommender (ANR)**, a neural recommendation system which utilises textual reviews, and aims to capture the finer-grained interactions between users and items at an aspect-level. As noted previously in Sections 2.2.1 and 2.2.2, the two main approaches used in existing works are **(1) Deep Learning-based Recommender Systems**, and **(2) Aspect-based Recommendation Systems**. We summarise the strengths and weaknesses of these two approaches in Table 3.1.

Table 3.1: Strengths and weaknesses of existing recommendation systems utilising textual reviews

Approach	Deep Learning-based	Aspect-based
Strength	Strong representation learning capabilities of neural networks	Recommendations can be explained at an aspect-level Better reflects the thought process behind users' interaction with items
Weakness	Less transparent and intuitive Reviews are compressed into a single latent representation	Relies on upstream Sentiment Analysis (SA) tools Not self-contained

This chapter is published as **Jin Yao Chin**, Kaiqi Zhao, Shafiq Joty, and Gao Cong. ANR: Aspect-based Neural Recommender. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM'18, Pages 147–156, 2018*. [26]

3.1 Motivation and Problem Setting

3.1.1 Textual Reviews and the Cold-start Problem

As mentioned previously, textual reviews can be used as a complementary information source to deal with the *cold-start problem* for relatively new users and items with *limited historical information*. As a motivating example, let us consider two Users A and B who have given the same positive rating (e.g. 4 out of 5 stars) to a particular Restaurant X as illustrated in Figure 3.1.

Figure 3.1: Users A and B giving the same rating to Restaurant X

Assuming that both Users A and B have only provided a single rating, i.e. to Restaurant X in this case, the recommender system would be restricted to recommending the same set of candidate items to these two users. However, as shown in Figure 3.2, we might be able to know a lot more about them if we look at the reviews that they have written.

For example, despite finding some of its items being a little overpriced, User A might have given Restaurant X a positive rating as he really enjoyed the food. On the other hand, User B favours Restaurant X as it is easily accessible from his/her workplace, even though the service from the restaurant’s staff might be subpar. With these new information which we can extract from the textual reviews, the recommender system would be able to generate completely different, and much more personalised, recommendations for these two users.

In the remainder of this chapter, we first specify the problem setting and key notations used in Section 3.1.2. For our proposed model, we start with an overview of

Figure 3.2: Reviews written by Users A and B for Restaurant X

our architecture along with the motivations behind some of the key components in Section 3.2. Following which, Section 3.3 describes in detail our neural attention-based module for learning the aspect-level user and item representations. We show our neural co-attention-based module for dynamically inferring the aspect-level importance for any given user-item pair in Section 3.4. Finally, Section 3.5 depicts how the aspect-level representations and importance can be combined effectively to infer the overall rating.

We conduct an experimental study in Section 3.6 to evaluate our proposed model ANR against state-of-the-art baseline methods. First, we introduce the datasets and baseline methods in Sections 3.6.1 and 3.6.2, respectively. Next, Section 3.6.3 presents the experimental setup. In Section 3.6.4, we discuss the results obtained from the empirical evaluation. Last but not least, in Section 3.6.5, we examine the effects of key hyperparameters such as the number of aspects, as well as provide a qualitative analysis of the learned aspects together with a model ablation study.

Table 3.2: Notations and their definitions¹

Notation	Definition
$\mathcal{D}$	Corpus with Ratings & Reviews
$(u, i, r_{u,i}, d_{u,i})$	Complete User-Item Interaction
$r_{u,i}$	Rating from User u for Item i
$d_{u,i}$	Review from User u for Item i
D_u	User Document (Set of Reviews from User u)
D_i	Item Document (Set of Reviews for Item i)
$\mathcal{A}$	Set of K Aspects
$\mathbf{v}_a$	Embedding Vector for Aspect $a \in \mathcal{A}$
$\mathbf{W}_a$	Word Projection Matrix for Aspect $a \in \mathcal{A}$
$\mathbf{p}_{u,a}$	Latent Representation of User u for Aspect a
$\mathbf{q}_{i,a}$	Latent Representation of Item i for Aspect a
$\beta_{u,a}$	Importance of Aspect a for User u
$\beta_{i,a}$	Importance of Aspect a for Item i

3.1.2 Problem Setting

Considering a corpus of ratings and reviews $\mathcal{D}$, for a set of items $\mathcal{I}$ and a set of users $\mathcal{U}$, each user-item interaction can be represented as a tuple $(u, i, r_{u,i}, d_{u,i})$ where $r_{u,i}$ is a numerical rating denoting user u 's overall satisfaction towards the item i , and $d_{u,i}$ is the accompanying textual review. The primary objective is to estimate the rating $\hat{r}_{u,i}$ for any unseen user-item pair, i.e. the unknown rating of a given user u towards an item i that he/she has not interacted with before. Table 3.2 summarises the key notations used throughout the rest of this chapter.

3.2 Overview of ANR

Figure 3.3 shows the overall architecture of our proposed model. Similar to existing deep learning-based models [168, 119], we feed the user document D_u and item document D_i , i.e. the set of reviews written by the user u and the set of reviews written for item i , respectively, as the inputs to the network. Since the modelling process for users and items are identical, we focus on illustrating the process for a given user. It should be noted

¹ Unless stated otherwise, we denote vectors with bold lower-cases, and bold upper-cases are reserved for matrices or high dimensional tensors.

Figure 3.3: Overall architecture of the proposed ANR

that the construction of user and item documents is constrained to the set of reviews from the training split, i.e. they do **not** include any review from the validation or testing split.

3.2.1 Embedding Layer

First, the user document D_u is transformed into a matrix $\mathbf{M}_u \in \mathbb{R}^{n \times d}$ via an embedding layer, where n is the number of words in D_u , and d is the number of dimensions for each word embedding vector. Basically, the embedding layer performs a look-up operation in a shared embedding matrix $f : \mathcal{V} \rightarrow \mathbb{R}^d$ which maps each word in the vocabulary $\mathcal{V}$ to its corresponding d -dimensional vector. The embedding matrix can be initialised using word vectors that have been pre-trained on large corpora, such as *word2vec*² [97] or *GloVe*³ [108], which facilitates a better semantic representation of the user (and item) documents. Unlike topic modelling-based methods which rely on the *bag-of-words* assumption, the order and context of words is preserved in the embedded document.

3.2.2 Aspect-based Representations

For example, considering the domain of *restaurants*, the aspect set $\mathcal{A}$ could include aspects such as *price*, *quality*, *service*, *location*, etc. In other words, an *aspect* can be defined as a high-level semantic concept encompassing a specific facet of item properties for a given domain. For *restaurants*, the aspect *service* can encompass properties such as $\{\textit{staff}, \textit{waiting time}, \textit{reservation}, \textit{valet parking}, \dots\}$.

Given the embedded user document representation $\mathbf{M}_u$, our goal here is to derive a set of aspect-level user representations $\mathbf{P}_u = \{\mathbf{p}_{u,a} \mid a \in \mathcal{A}\}$ with respect to a set of K domain-dependent aspects, $\mathcal{A}$. Intuitively, the review $d_{u,i}$ describes user u 's opinions towards an item i based on this set (or possibly, subset) of aspects. Consequently, the user document D_u covers user u 's opinions towards $\mathcal{A}$ aggregated across all the items that he/she has previously interacted with. Similarly, the item document D_i describes the properties of an item i with respect to $\mathcal{A}$ aggregated across all the users that have reviewed it.

Our hypothesis is that given sufficient data, we can learn this set of aspects $\mathcal{A}$, as well as the aspect-level user (or item) representation for each aspect $a \in \mathcal{A}$, by learning to attend to a subset of aspect-related words within each user (or item) document. To this end, we propose a novel aspect-aware attention-based component for learning these aspect-level representations, and the details are presented in Section 3.3.

² <https://code.google.com/archive/p/word2vec/>

³ <https://nlp.stanford.edu/projects/glove/>

3.2.3 Aspect-level Importance

It is not uncommon for different users to have varied preferences for the different aspects of an item. We highlight 3 simple observations which makes it necessary for us to model the aspect-level user and item importance in such an interaction-specific manner.

Figure 3.4: Varied user preferences with respect to different items

(1) A user’s aspect-level preferences may vary with respect to the item of interest. For instance, as illustrated in Figure 3.4, a user may focus on the *price* and *aesthetics* when choosing a **mobile phone**, while he/she may be more concerned about the *performance* and *portability* when purchasing a **laptop**.

Figure 3.5: Different appeal of the same item to different users

(2) Likewise, the same item may appeal differently to two different users. To illustrate, as shown in Figure 3.5, some user may like a particular **restaurant** for its *food*, while another user frequents the same restaurant due to its cosy *ambiance*.

Figure 3.6: Joint and complex evaluation of interrelated aspects

(3) Additionally, these aspects are often not evaluated separately or independently. For example, consider the scenario presented in Figure 3.6. A user may be willing to overlook the steep *price* of a *mobile phone* if its *quality* and *performance* exceeds his/her expectations, even though the historical data may indicate that this user would generally prefer *cheaper mobile phones*.

As such, rather than having static user and item aspect importance, we believe that it is imperative to model the rich and complex interactions between users and items at the aspect level by dynamically estimating the user and item aspect importance for each user-item pair. To this end, we propose a novel co-attention-based component which is able to consider these crucial observations for recommendation, and the details are presented in Section 3.4.

3.3 Aspect-based Representation Learning

Before diving into the specific details of our proposed aspect-based representation learning approach, we highlight some vital intuitions which we aim to capture through this component.

Intuition 1: Not all words in a review (or document) are equally important, and the importance of each document word varies with respect to the aspect being considered. Generally, reviews tend to include opinions towards multiple aspects of the target item,

and we should be able to focus on specific sections of the review (or document) when learning the aspect-level representation for a given aspect.

Intuition 2: The sentimental polarity of the same word could be completely different for two different aspects in the same domain. For example, the word ‘*high*’ in the sentences “This phone has a high storage capacity” and “This camera captures high quality images” carries a positive sentiment towards the target aspect (or item property). On the other hand, considering the sentences “The price is way too high” and “This computer has extremely high power consumption”, the same word actually reflects a negative sentiment. In fact, many of these sentiment-bearing words tend to indicate a different polarity based on the aspect being considered, and this should be captured in the aspect-level representations.

Intuition 3: It has been well-established that aspect-related words (e.g. *price*, *taste*, *ambience*) and their sentiment-bearing words (e.g. *expensive*, *delicious*, *amazing*) are often in close proximity [56]. This implies that we can better infer the importance of a word within the document by looking at its surrounding words, i.e. by considering a local context window.

Now, we describe how the aspect-level user representation, i.e. $\mathbf{p}_{u,a}$, can be obtained for a user u and a given aspect $a \in \mathcal{A}$. Since all words in the vocabulary $\mathcal{V}$ share the same d -dimensional vector across the K aspects, we use an aspect-specific word projection matrix⁴ $\mathbf{W}_a \in \mathbb{R}^{d \times h_1}$ to allow variations in the word representations with respect to the target aspect a (**Intuition 2**). More formally,

$$\mathbf{M}_{u,a}[i] = \mathbf{M}_u[i] \mathbf{W}_a \quad (3.1)$$

where $\mathbf{M}_u[i]$ is the **original** d -dimensional word embedding for the i -th word in $\mathbf{M}_u$, $\mathbf{M}_{u,a}[i]$ is the **aspect-specific** word representation, and $\mathbf{M}_{u,a} \in \mathbb{R}^{n \times h_1}$ is the **aspect-specific** document embedding for user u and aspect a . The result of this projection is a tensor in $\mathbb{R}^{K \times n \times h_1}$ for K different aspects.

Each aspect $a \in \mathcal{A}$ is represented as an embedding vector $\mathbf{v}_a \in \mathbb{R}^{(c \times h_1)}$ with length $c \times h_1$, where c is a hyperparameter, which determines the width (in terms of the number of words) of the local context window (**Intuition 3**). To compute the importance of the

⁴ Note that h_1 is a hyperparameter, which allows the number of latent factors used for the aspect-level representations to be defined, without being constrained by the size of the original word embeddings.

i -th document word in this **aspect-specific** embedding subspace, we consider a local context window with it as the centre word:

$$\mathbf{z}_{u,a,i} = (\mathbf{M}_{u,a}[i - c/2]; \dots; \mathbf{M}_{u,a}[i]; \dots; \mathbf{M}_{u,a}[i + c/2]) \quad (3.2)$$

where $(\cdot; \cdot)$ is the concatenation operator. We calculate the attention score for the i -th word by taking the inner product followed by the softmax function:

$$\mathbf{attn}_{u,a}[i] = \text{softmax}(\mathbf{v}_a (\mathbf{z}_{u,a,i})^\top) \quad (3.3)$$

where $\text{softmax}(w_i) = \exp(w_i) / \sum_j \exp(w_j)$, and $\mathbf{attn}_{u,a}$ is the soft attention vector (i.e. a probability distribution) defined over the document words for user u with respect to aspect a . Intrinsically, the importance of the i -th word in the document depends on both

Figure 3.7: Illustration of the aspect-based representation learning component

the word itself and its surrounding words (**Intuition 3**). Taking into consideration the learned importance of each word in the document (**Intuition 1**), the aspect-level user representation can be derived based on the following weighted sum:

$$\mathbf{p}_{u,a} = \sum_{i=1}^n (\mathbf{attn}_{u,a}[i] \mathbf{M}_{u,a}[i]) \quad (3.4)$$

Figure 3.7 provides a simple illustration of the Aspect-based Representation Learning component, which transforms each document into its corresponding set of K aspect-level representations. The aspect-level item representation $\mathbf{q}_{i,a}$ for item i and aspect a can be obtained in a similar manner, following (3.1) to (3.4).

Additionally, for each aspect $a \in \mathcal{A}$, we share the aspect embedding vector $\mathbf{v}_a$ and aspect-specific word projection matrix $\mathbf{W}_a$ for users and items (i.e. the aspect-level user and item representations for each aspect a reside in the same aspect-specific feature space). Sharing aspect-specific parameters allows us to learn better mapping between aspects in user and item documents while reducing the number of trainable parameters in the model. We denote the set of parameters for the *Aspect-based Representation Learning* layer as $\Theta_{ARL} = \{\mathbf{v}_a, \mathbf{W}_a \mid a \in \mathcal{A}\}^5$.

3.4 Aspect Importance Estimation

A straightforward solution is that we could attempt to estimate the user and item aspect importance separately. However, this would result in ‘static’ user and item aspect importance, whereby the user aspect importance does not actually take into consideration the actual item of interest, and vice versa. In other words, the user (item) aspect importance remains fixed across all possible items (users), and would be suboptimal for any given user-item pair since it is not derived specifically for the user and item in question.

To this end, we propose learning the user and item aspect importance in a joint and interaction-specific manner. The aspect-level item representations are used as the context when learning the user aspect importance, and similarly, the user aspect-level representations can be used as the context when learning the item aspect importance. The output of this layer would be a K -dimensional vector indicating the importance of each aspect for the user, and a corresponding K -dimensional vector for the item.

In order to incorporate the aspect-level item representations when calculating the user aspect importance (and vice versa), we need to know how the target user and item matches at an aspect-level. First, using the aspect-level user representation $\mathbf{P}_u \in \mathbb{R}^{K \times h_1}$ and item representation $\mathbf{Q}_i \in \mathbb{R}^{K \times h_1}$, we can obtain an aspect-level affinity matrix $\mathbf{S}$ as follows:

$$\mathbf{S} = \phi(\mathbf{P}_u \mathbf{W}_s \mathbf{Q}_i^\top) \quad (3.5)$$

⁵ For each aspect a , both the aspect embedding vector $\mathbf{v}_a$ and the aspect-specific projection matrix $\mathbf{W}_a$ are initialised randomly using a uniform distribution $\mathcal{U}(-0.01, 0.01)$. Essentially, the aspect-based representation learning is performed in a data-driven manner without any external supervision.

 Figure 3.8: Aspect-level affinity matrix (*Best viewed in colour*)

where $\mathbf{W}_s \in \mathbb{R}^{h_1 \times h_1}$ is a learnable weight matrix, $\phi(x) = \max(0, x)$ is the ReLU function, and each entry in the affinity matrix $\mathbf{S} \in \mathbb{R}^{K \times K}$ denotes the *affinity* (or *shared similarity*) between the corresponding pair of aspect-level user and item representations. We provide the illustration of an affinity matrix with K aspects in Figure 3.8.

Next, following [93], we use the affinity matrix $\mathbf{S}$ as a *feature* to estimate the aspect-level user and item importance:

$$\mathbf{H}_u = \phi(\mathbf{P}_u \mathbf{W}_x + \mathbf{S}^\top (\mathbf{Q}_i \mathbf{W}_y)), \beta_u = \text{softmax}(\mathbf{H}_u \mathbf{v}_x) \quad (3.6)$$

$$\mathbf{H}_i = \phi(\mathbf{Q}_i \mathbf{W}_y + \mathbf{S} (\mathbf{P}_u \mathbf{W}_x)), \beta_i = \text{softmax}(\mathbf{H}_i \mathbf{v}_y) \quad (3.7)$$

where $\mathbf{W}_x, \mathbf{W}_y \in \mathbb{R}^{h_1 \times h_2}$ and $\mathbf{v}_x, \mathbf{v}_y \in \mathbb{R}^{h_2}$ are the learnable parameters. $\beta_u \in \mathbb{R}^K$ and $\beta_i \in \mathbb{R}^K$ are the estimated aspect importance over the set of K aspects $\mathcal{A}$ for user u and item i , respectively. Basically, we consider both the user representation $\mathbf{P}_u$ and the item representation $\mathbf{Q}_i$ when computing $\mathbf{H}_u$ and $\mathbf{H}_i$. Considering the fact that the aspect-level representations may be substantially different between (1) users and items, (2) two different users, and (3) two different items, we find that these subsequent non-linear transformations improve the model performance by allowing it to better estimate the pairwise aspect-level importance for any given user-item pair.

As highlighted previously, we specifically designed this component to take into consideration the target user and item, enabling such estimation of the aspect-level importance to be personalised for both the user and item. We denote the set of parameters for the *Aspect Importance Estimation* layer as $\Theta_{AIE} = \{\mathbf{W}_s, \mathbf{W}_x, \mathbf{W}_y, \mathbf{v}_x, \mathbf{v}_y\}$.

Figure 3.9: Illustration of the User-Item Rating Prediction component (*Best viewed in colour*)

3.5 Model Inference and Optimisation

We now describe the *User-Item Rating Prediction* component shown in Figure 3.3. By combining the user and item aspect-level representations $\mathbf{P}_u$, $\mathbf{Q}_i$ with the aspect importance β_u , β_i , the overall rating for any user-item pair can be inferred as follows:

$$\hat{r}_{u,i} = \sum_{a \in \mathcal{A}} \overbrace{(\beta_{u,a} \cdot \beta_{i,a})}^{\text{Aspect Importance}} \cdot \underbrace{(\mathbf{p}_{u,a} (\mathbf{q}_{i,a})^\top)}_{\text{Aspect-based Representations}} + b_u + b_i + b_0 \quad (3.8)$$

where b_u , b_i , b_0 are the user, item, and global bias (as in traditional latent factor models), respectively. A graphical illustration of how (3.8) effectively combines the user and item aspect-level representations and importance is shown in Figure 3.9. The model optimisation process can be viewed as a regression problem and the complete set of model parameters $\Theta = \{\Theta_{ARL}, \Theta_{AIE}, \mathbf{b}_u, \mathbf{b}_i, b_0\}$ can be learned using the backpropagation technique with the standard Mean Squared Error (**MSE**) as the loss function.

3.5.1 Model Pre-training

It has been shown in [36] that the performance of neural networks can be rather sensitive to how the parameters are initialised. For our proposed model, the *Aspect Importance Estimation* component is fully based on the outputs $\mathbf{P}_u, \mathbf{Q}_i$ from the previous *Aspect-based Representation Learning* layer, i.e. it implicitly relies on the set of parameters $\Theta_{ARL} = \{\mathbf{v}_a, \mathbf{W}_a \mid a \in \mathcal{A}\}$. As such, we employed a *pre-training phase* using a simplified model to obtain a good initialisation for Θ_{ARL} . We replaced the *Aspect Importance Estimation* component with two feed-forward neural networks, for users and items, respectively. The user (item) network takes as input the concatenation of the aspect-level user (item) representations, and produces an abstract user (item) representation. These abstract user and item representations are then concatenated and used for predicting the overall rating $\hat{r}_{u,i}$ via another feed-forward layer. This simplified model does not consider the aspect-level interactions between users and items, and it is trained in a similar fashion using the backpropagation method with the MSE loss function.

3.5.2 Model Generalisation

Many existing work have found that deep learning models tend to suffer from overfitting. In order to improve the generalisation performance, we adopt the dropout technique [127], which is widely used in existing neural models for recommendation [168, 119, 18, 21]. For each aspect-level representation, which is a h_1 -dimensional vector of latent factors, ρ percent of this vector is dropped out at random during the training phase. Additionally, we apply L_2 regularisation to the user and item biases in (3.8).

3.6 Experimental Study

As part of our experimental study, we evaluate our proposed model against several state-of-the-art baseline methods using publicly available datasets from *Yelp* and *Amazon*. In this section, we describe the datasets used, introduce the baseline methods, elaborate on the experimental setup, and present the experimental results. Additionally, we perform an in-depth analysis of our proposed model ANR.

3.6.1 Datasets

For *Yelp*, we use the latest version (*Round 11*) of the Yelp Dataset Challenge⁶, which contains ratings and reviews for local businesses across 4 countries. As for *Amazon*, we use the Amazon Product Reviews⁷ from [96, 47] which has already been organised into **24** individual product categories.

For the Yelp dataset and 3 of the larger datasets from Amazon (i.e. Books, Electronics, and Clothing, Shoes & Jewelry), we randomly sub-sampled 5,000,000 user-item interactions for the experiments. Following which, similar to [168, 119, 18, 21], we randomly partitioned each of these **25** datasets into training, validation, and testing sets using the ratio 80:10:10. Following [95, 168, 18], we directly use these datasets as they are. Specifically, we have chosen not to adopt the ‘*5-core setting*’ used in [119, 21, 25], whereby there are at least 5 ratings/reviews for each user and item, as it trivialises the problem of data sparsity which is inevitable in real-world recommendation systems.

Table 3.3 shows the statistics of the datasets used. It should be noted that the average sparsity of these datasets is 99.9985%, and it would be challenging for any recommender system to perform sufficiently well on these datasets. Additionally, the average number of ratings (or equivalently, reviews) per user and item (across all datasets) are 1.91 and 12.12, respectively.

3.6.2 Baseline Methods

We compare our proposed method against 3 state-of-the-art baseline methods, which utilise review information to improve the overall recommendation performance.

(1) Deep Cooperative Neural Networks (**DeepCoNN**) [168]: This is a state-of-the-art neural recommendation model which derives latent user and item representations from their corresponding reviews using a convolutional architecture. The user and item representations are concatenated and used as the input to a Factorization Machine (**FM**) [111] for the overall rating prediction. It has been shown from extensive empirical evaluations that DeepCoNN far outperforms *classic* recommendation methods such as Matrix Factorization (**MF**) [69], Latent Dirichlet Allocation (**LDA**) [13], and Hidden Factors as Topics (**HFT**) [95].

⁶ <https://www.yelp.com/dataset/challenge>

⁷ <http://jmcauley.ucsd.edu/data/amazon/>

Table 3.3: Statistical details for the datasets

Dataset	Users	Items	Ratings
Amazon Instant Video	348,665	22,083	499,667
Apps for Android	1,135,316	56,841	2,424,812
Automotive	710,163	279,269	1,193,219
Baby	448,895	59,005	823,549
Beauty	1,014,152	224,878	1,794,288
Books	2,370,327	1,068,230	5,000,000
CDs & Vinyl	1,338,741	445,885	3,454,125
Cell Phones & Accessories	1,886,723	284,794	3,014,598
Clothing, Shoes & Jewelry	2,417,497	926,060	5,000,000
Digital Music	399,571	225,461	725,103
Electronics	2,586,767	362,819	5,000,000
Grocery & Gourmet Food	642,408	150,567	1,151,829
Health & Personal Care	1,546,374	229,078	2,638,255
Home & Kitchen	2,118,130	368,247	3,800,692
Kindle Store	1,189,641	394,742	2,944,055
Movies & TV	1,765,998	187,426	4,241,131
Musical Instruments	280,758	74,731	433,834
Office Products	749,514	116,666	1,069,322
Patio, Lawn And Garden	588,559	95,824	853,064
Pet Supplies	624,250	93,917	1,103,110
Sports And Outdoors	1,667,978	425,034	2,887,105
Tools & Home Improvement	1,012,104	232,744	1,693,910
Toys & Games	1,127,969	294,840	1,998,854
Video Games	689,357	47,562	1,177,239
Yelp (2018)	1,144,046	174,013	5,000,000

(2) Dual Attention-based Model (**D-Attn**) [119]: Similar to DeepCoNN, D-Attn relies on Convolutional Neural Networks (**CNNs**) to learn the user and item representations. The key difference is that prior to the convolutional layer, D-Attn incorporates local and global attention-based modules for selecting locally and globally informative words from the reviews, respectively. However, instead of a FM, D-Attn simply uses the inner product of the user and item representations for the rating prediction.

(3) Aspect-aware Latent Factor Model (**ALFM**) [25]: ALFM is a state-of-the-art aspect-based recommendation system which does **not** rely on external sentiment analysis tools. The authors designed an Aspect-aware Topic Model (**ATM**) to represent each aspect $a \in \mathcal{A}$ as a distribution over latent topics based on the review contents. The output from ATM is then combined with ALFM, which associates latent factors with the same set of aspects $\mathcal{A}$ by using the MF approach on the ratings.

It should be noted that all three baseline methods have been proposed *very recently*, and amongst them, have been shown to outperform many other highly competitive recommendation methods [146, 84, 147, 134, 18].

3.6.3 Experimental Setup

First, all reviews are tokenised using NLTK⁸ and we retain the 50,000 most frequent words to be used as the vocabulary $\mathcal{V}$ for each dataset.

For ALFM, we use the code provided by the authors, and follow the hyperparameter settings and optimisation method as reported in the paper. Both the number of aspects and latent topics used in ALFM are set to 5. Although [25] only uses 5 latent factors for their model comparison, their hyperparameter study found that more latent factors generally leads to better performance. As such, we used the validation set to select the optimal number of latent factors amongst $\{5, 10, 15, 20, 25\}$ for each dataset.

We implemented the neural recommendation models, i.e. DeepCoNN, D-Attn, as well as our proposed method, using PyTorch⁹. We set the length of input user and item documents, i.e. $|D_u|$ and $|D_i|$, to 500. Our model and DeepCoNN use 300- d word embeddings trained on Google News [97], while D-Attn uses 100- d word embeddings trained on Wikipedia using *GloVe* [108] (We tried using the same 300- d embeddings for D-Attn, but it consistently degrades its performance across multiple datasets). We reuse

⁸ <https://www.nltk.org/>

⁹ <https://pytorch.org/>

the settings reported in [168, 119] for hyperparameters such as the number and size of convolutional filters, the number of factors used for the fully connected layers, and the activation functions. For DeepCoNN, we set the dropout rate to 0.5 and the number of factors used in the FM as 10, based on a grid search using the validation sets as these values were not specified in the paper. For fair comparison with ALFM, we use the same number of aspects in our model, i.e. $|\mathcal{A}| = K = 5$. The remaining hyperparameters for ANR, such as the width of the local context window c , number of latent factors h_1, h_2 , and dropout rate ρ are set as 3, 10, 50, and 0.5, respectively. All 3 neural models are trained using Adam [67], using an initial learning rate of 0.002, a batch size of 128 and the MSE loss.

Following [168, 119], we use the standard Mean Squared Error (**MSE**) as the evaluation metric. All the experiments are repeated 5 times, and we report the (average) test MSE obtained when the validation MSE is the lowest.

3.6.4 Results and Discussion

Table 3.4 shows the results from our experiments on all 25 datasets. We observe that ANR achieves statistically significant improvement over all 3 state-of-the-art baseline methods, based on the paired sample t-test using results from 5 separate runs for each model.

Next, we note that aspect-aware recommendation methods such as ALFM and ANR consistently outperforms DeepCoNN and D-Attn. We believe that this can be attributed to the fact that DeepCoNN and D-Attn ‘compresses’ the user (and item) documents into a single representation (i.e. vector), and consequently, the only ‘interaction’ between users and items occurs at the prediction layer, i.e. when using the user and item representations for predicting the overall rating. In other words, they are unable to capture the multi-faceted decision-making process involved in these user-item interactions. Both DeepCoNN and D-Attn have a similar model architecture due to their use of the convolutional layer as the encoder, and it may seem like D-Attn would perform better with its additional local and global attention-based modules. However, D-Attn was previously evaluated using the much denser 5-core setting [119], and it seems to underperform due to the data sparsity which is evident in our experimental setup.

Finally, although ALFM attempts to utilise the review contents in their framework, they do so using a topic modelling approach. One major drawback is that the proposed

Table 3.4: Comparison with state-of-the-art baseline methods in terms of the Mean Squared Error (The best result for each dataset is indicated in bold). All reported improvements over baseline methods are statistically significant with $p\text{-value} < 0.01$ based on the paired sample t-test.

Dataset	D-Attn	DeepCoNN	ALFM	ANR	Improvement (%)		
	(a)	(b)	(c)	(d)	(d) vs. (a)	(d) vs. (b)	(d) vs. (c)
Amazon Instant Video	1.213	1.178	1.075	1.009	16.83	14.36	6.13
Apps for Android	1.637	1.593	1.555	1.412	13.73	11.34	9.14
Automotive	1.411	1.349	1.257	1.188	15.76	11.91	5.43
Baby	1.507	1.442	1.359	1.258	16.51	12.73	7.44
Beauty	1.609	1.566	1.466	1.386	13.89	11.48	5.46
Books	1.122	1.089	1.055	0.976	12.94	10.30	7.43
CDs & Vinyl	1.014	0.980	0.956	0.914	9.93	6.81	4.46
Cell Phones & Accessories	2.083	2.040	1.787	1.689	18.92	17.23	5.50
Clothing, Shoes & Jewelry	1.491	1.430	1.316	1.266	15.09	11.48	3.78
Digital Music	0.775	0.749	0.725	0.688	11.22	8.12	5.07
Electronics	1.744	1.659	1.563	1.445	17.10	12.89	7.50
Grocery & Gourmet Food	1.386	1.345	1.284	1.187	14.42	11.76	7.57
Health & Personal Care	1.612	1.545	1.466	1.356	15.91	12.23	7.49
Home & Kitchen	1.575	1.508	1.443	1.317	16.38	12.69	8.76
Kindle Store	0.949	0.905	0.870	0.834	12.08	7.81	4.10
Movies & TV	1.246	1.207	1.193	1.112	10.75	7.88	6.80
Musical Instruments	1.224	1.160	1.072	1.034	15.51	10.81	3.49
Office Products	1.650	1.569	1.474	1.337	18.98	14.79	9.30
Patio, Lawn & Garden	1.696	1.622	1.510	1.403	17.30	13.51	7.09
Pet Supplies	1.628	1.565	1.485	1.377	15.41	12.05	7.28
Sports & Outdoors	1.354	1.300	1.221	1.137	16.04	12.55	6.86
Tools & Home Improvement	1.474	1.429	1.348	1.230	16.51	13.93	8.74
Toys & Games	1.298	1.227	1.131	1.075	17.16	12.34	4.88
Video Games	1.533	1.498	1.383	1.292	15.72	13.72	6.57
Yelp (2018)	1.691	1.669	1.614	1.527	9.68	8.49	5.42
Average	1.437	1.385	1.304	1.218	14.95	11.73	6.47

Aspect-aware Topic Model (**ATM**) of ALFM does not consider the rating information when inferring the user and item preferences from the reviews; and the review contents are not utilised when ALFM learns the latent user and item representations using the MF approach. Put differently, unlike our proposed method, ALFM uses the review contents and rating information *separately*.

3.6.5 Model Analysis

In this section, we examine the effects of key hyperparameters on the model performance. Furthermore, we provide a glimpse of the inner workings of our model via a qualitative analysis of the learned aspects and an ablation study.

Figure 3.10: Effect of the number of aspects across multiple datasets

3.6.5.1 Number of Aspects

Figure 3.10 illustrates the effect of varying the number of aspects between 2 and 8 for our model across multiple datasets. We notice that the optimal number of aspects varies across the different datasets, and most likely depends on the characteristics of the review contents for any given dataset. In general, we observe that reasonably good performance can be obtained using around 4 to 6 aspects. Additionally, we hypothesise that changing the total number of aspects only affects the granularity of each individual aspect, i.e. numerous *fine-grained aspects* versus a handful of *broader aspects*. As such, varying the number of aspects (within a reasonable range) would have rather little impact on the overall model performance.

(a): CDs & Vinyl

(b): Toys & Games

(c): Video Games

Figure 3.11: Effect of the number of factors for h_1 and h_2

3.6.5.2 Number of Factors for h_1 and h_2

We investigate the model’s sensitivity to the number of factors used for h_1 and h_2 . The 3-D figures in Figure 3.11 show the performance of our model by varying h_1 from 5 to 50 and h_2 from 10 to 100, for different datasets. Recall that h_1 determines the number of latent factors used for the aspect-level user and item representations (i.e. $|\mathbf{p}_{u,a}|$ & $|\mathbf{q}_{i,a}|$), while h_2 defines the size of the hidden layers used for estimating the user and aspect importance (i.e. β_u & β_i) based on the affinity matrix $\mathbf{S}$.

First, it does not require a large number of latent factors to encode the user and item representations at an aspect-level and the model performance does not improve when h_1 is greater than 15. However, as shown in Figure (a), the performance may degrade if insufficient latent factors are used for $\mathbf{p}_{u,a}$ and $\mathbf{q}_{i,a}$. Next, we find that the number of hidden factors used for estimating the aspect importance has a much lesser impact on the overall performance, and our choice of setting h_2 to 50 should suffice for most datasets.

Table 3.5: Top 10 words for each aspect in the *Video Games* dataset. The ‘aspect labels’ in the first row are added based on our interpretation of the top words for that aspect.

Price	Family	Negative	Gameplay	Graphics
works	son	bad	lot	bought
recommend	new	little	hours	pretty
well	highly	horrible	bit	still
buy	story	waste	couple	graphics
bought	favourite	hard	characters	much
awesome	part	boring	stars	think
price	character	terrible	course	work
loves	daughter	frustrating	minutes	recommend
worth	controller	difficult	side	cool
purchase	characters	disappointed	fan	nice

3.6.5.3 Qualitative Analysis of Learned Aspects

In Section 3.3, we described the process of obtaining the aspect-level representations by learning to attend to a subset of aspect-related words within the corresponding document. The soft attention vector $\mathbf{attn}_{u,a}$ in (3.3) can also be viewed as a probability distribution

over the vocabulary $\mathcal{V}$, for a user $u \in \mathcal{U}$ and an aspect $a \in \mathcal{A}$. As such, we can calculate the *importance* of each word $z \in \mathcal{V}$ w.r.t. the user u and an aspect a as follows:

$$\psi_{z,u,a} = \sum_{i=1}^{|D_u|} \text{attn}_{u,a}[i] \text{ if } (D_u[i] = z) \quad (3.9)$$

where $D_u[i]$ refers to the i -th word in the user document. The *importance* of word z for aspect a can then be computed as:

$$\psi_{z,a} = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \psi_{z,u,a} + \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \psi_{z,i,a} \quad (3.10)$$

Following [95], the ‘background’ distribution of a word z is defined as:

$$b_z = \left(\sum_{a \in \mathcal{A}} \psi_{z,a} \right) / |\mathcal{A}| \quad (3.11)$$

where $|\mathcal{A}|$ is the number of aspects, and we semantically represent each aspect a using its top words based on $(\psi_{z,a} - b_z)$.

The aspects learned by our model for the **Video Games** dataset are shown in Table 3.5. We find that each aspect does cover a rather specific and meaningful facet of item properties for the particular domain, and reflects the *different factors* that contribute to the overall rating of these user-item interactions. Given that the optimal number of aspects can be rather different for each dataset, the quality of these learned aspects could potentially improve if we consider a different number of aspects.

3.6.5.4 Ablation Study

We perform an ablation study to analyse how different components in our proposed model contribute to the overall performance and hopefully, justify some of our architectural decisions. The ‘*baseline*’ of this discussion refers to the complete model as described in Sections 3.2 to 3.5, using the hyperparameter settings stated in Section 3.6.3, and we compare it with its five variants:

- **Simplified Model:** Instead of modelling the aspect-level interactions, the aspect-level user (and item) representations are concatenated and put through a hidden layer to obtain the final user (and item) representation. Similar to the baseline methods DeepCoNN and D-Attn, the only interaction between users and items occurs at the final prediction layer, i.e. when using the user and item embeddings to derive the overall rating.

- **No Pre-training:** We forgo the pre-training phase for Θ_{ARL} , i.e. the set of parameters for *Aspect-based Representation Learning* layer, to validate its effectiveness.
- **Shared Projection Layer:** Rather than having aspect-specific projection matrices, we constrain the model by having only a single projection matrix which is shared across all aspects. Basically, each word has the exact same representation across all aspects, and we make use of this model variant to verify **Intuition 2**.
- **Uniform Aspect Importance:** We replace $\beta_{u,a} \cdot \beta_{i,a}$ in (3.8) with $1/K$, i.e. all aspects are assumed to be *equally important*. This model variant helps to validate the necessity of learning both user and item importance at the aspect-level.
- **Main Diagonal:** The main diagonal of the affinity matrix $\mathbf{S}$ obtained using (3.5) is used to replace $\beta_{u,a} \cdot \beta_{i,a}$. Note that the main diagonal only captures the relationship between corresponding pairs of aspects, i.e. aspect a (User) and aspect a (Item) for each $a \in \mathcal{A}$.

 Table 3.6: Comparison of model variants for the *Toys & Games* and *Video Games* datasets

Setup	Toys & Games	Video Games
<i>Baseline</i>	1.069	1.278
Simplified Model	1.173	1.495
No Pre-training	1.123	1.354
Shared Projection Layer	1.122	1.349
Uniform Aspect Importance	1.106	1.310
Main Diagonal	1.108	1.315

The results of the ablation study for the *Toys & Games* and *Video Games* datasets are shown in Table 3.6. First, we observe that the lack of aspect-level interactions in the *Simplified Model* leads to large performance degradation on both datasets. For the *Aspect-based Representation Learning* layer, we find that the pre-training phase does provide a better starting point for learning the user and item aspect importance. Additionally, allowing variations in the word representations through the aspect-specific projection layer leads to better overall performance, supporting our **Intuition 2**. Finally, results from the last 2 model variants highlight the need for dynamically adapting the user and item aspect importance for each user-item pair, and show that modelling such a fine-grained interaction between users and items can improve the rating prediction accuracy through a better reflection of the complex decision-making process.

3.7 Summary

In this chapter, we have presented **Aspect-based Neural Recommender (ANR)**, an aspect-based recommender system which is based on recently proposed deep learning techniques. ANR includes an aspect-aware representation learning component and an aspect importance estimator, that are based on the ideas of neural attention and co-attention, respectively. Our proposed model concurrently address several shortcomings of existing recommender systems, and a thorough experimental study on 25 benchmark datasets from *Amazon* and *Yelp* shows that ANR significantly outperforms recently proposed state-of-the-art baselines such as DeepCoNN, D-Attn and ALFM. Last but not least, we have demonstrated how textual reviews can be utilised at a finer-grained aspect-level to complement historical interactions in order to improve both the robustness as well as the predictive performance of a recommendation system.

Chapter 4

Adaptive Framework for Seamless Warm- and Cold-start Recommendation

While textual reviews can be highly beneficial for any recommendation system, they might not always be readily available. For example, they are not applicable for completely new users and items. Furthermore, considerable effort is required from end users to provide the reviews consciously, and such a feature might not even be available on certain platforms.

In this chapter, we explore hybrid models which incorporate other forms of content information such as **user demographics** (e.g. the *age, gender, and occupation* for a user), as well as **item attributes** (e.g. the *release year, genre, and language* of a movie). Our proposed *Collective Collaborative Filtering* (**CCF**) model is a hybrid recommender system which aims to maximise the performance for both warm- and cold-start recommendation.

We first outline the problem setting and define the recommendation tasks in Section 4.1. Next, our proposed model CCF is presented in Section 4.2. Following which, in Section 4.3, we describe the datasets and baseline methods used for our empirical evaluation, as well as a discussion about the evaluation procedure. Notably, we highlight and examine key challenges for both commonly used datasets and evaluation procedures adopted in existing work. Finally, in Section 4.4, we present the experimental results, examine the effects of key hyperparameters, and visualise the task-specific attention weights which are learned adaptively for each recommendation scenario.

4.1 Problem Setting

Considering a collection of ratings $\mathcal{R}$, for a set of users $\mathcal{U}$ and a set of items $\mathcal{I}$, each user-item interaction can be represented as a tuple $(u, i, r_{u,i})$, where $r_{u,i}$ is a numerical rating reflecting user u 's preference towards an item i . The primary objective of any recommender system is to estimate the rating $\hat{r}_{u,i}$ for any unseen user-item pair. Additionally, we assume that each user u is associated with user contents $\mathbf{x}_u \in \mathbb{R}^{d_{\text{user}}}$, such as her *age*, *occupation*, etc. Similarly, there are item contents $\mathbf{x}_i \in \mathbb{R}^{d_{\text{item}}}$ available for each item i , for instance the *release year* and *genre* of a movie.

We account for two different groups of users: (1) Existing (or *warm-start*) users $\mathcal{U}_E$ with at least one interaction in $\mathcal{R}$, and (2) New (or *cold-start*) users $\mathcal{U}_N$. Note that, $\mathcal{U}_E \cup \mathcal{U}_N = \mathcal{U}$ and $\mathcal{U}_E \cap \mathcal{U}_N = \emptyset$. Existing items $\mathcal{I}_E$ and new items $\mathcal{I}_N$ are defined identically. Based on $\mathcal{U}_E$, $\mathcal{U}_N$, $\mathcal{I}_E$, and $\mathcal{I}_N$, we specify *four* different recommendation tasks:

Task 1 (In-matrix): The standard problem of predicting the user-item preference of an existing user towards an existing item;

Task 2 (User Cold-start): Infer the user-item preference from a new user towards an existing item;

Task 3 (Item Cold-start): For an existing user, determine her preference for a new item;

Task 4 (Both Cold-start): Predict the user-item preference of a new user for a new item.

The most challenging and often neglected situation is perhaps Task 4, whereby both the user and item are new and do not have any historical data. We find that most existing work consider a subset of these four recommendation tasks, instead of evaluating their proposed framework on all possible scenarios.

4.2 Proposed Model

We first provide an overview of the hybrid model architecture in Section 4.2.1. Following which, in Section 4.2.2, we illustrate the *neighbourhood-based* Collective Collaborative Filtering (**CCF**) module, which is central to our model. Finally, we will go through the process of deriving the user-item preference, as well as the model optimisation details in Section 4.2.3.

Figure 4.1: Overall model architecture

4.2.1 Overall Model Architecture

The overall model architecture is shown in Figure 4.1. The inputs to the model are the user u , item i , as well as the user contents $\mathbf{x}_u$ and the item contents $\mathbf{x}_i$. Unlike existing hybrid models, we have an additional Collective CF component in both the user and item networks, which take as inputs the user neighbourhood $\mathcal{N}_u$ and the item neighbourhood $\mathcal{N}_i$, respectively. The learned representations from the Collective CF components can benefit both warm- and cold-start recommendation, and we present the details in Section 4.2.2. The final objective is to provide an accurate estimation of the user-item preference $\hat{r}_{u,i}$ for any given user-item pair. Since the user and item networks have identical structures, we simply describe the representation learning process for a given user.

CF-based Representation. First, user u is represented as a one-hot vector $\mathbf{z}_u \in \{0, 1\}^{|\mathcal{U}|}$, which is then transformed into a dense low-dimensional vector $\mathbf{v}_{u,CF} \in \mathbb{R}^d$ using a user embedding matrix $\mathbf{E}_u \in \mathbb{R}^{|\mathcal{U}| \times d}$. The CF-based representation $\mathbf{v}_{u,CF}$ captures the underlying preferences of the user u based on the historical data, such that users with more items in common (i.e. similar preferences) tend to be embedded closer to each other.

Content-based Representation. Next, we elaborate on the function f_{user} in Figure 4.1, which converts the user contents $\mathbf{x}_u$ to a dense vector $\mathbf{v}_{u,CB}$. Suppose we have

the following user contents:

$$\mathbf{x}_u = [\mathbf{x}_{u,1}; \mathbf{x}_{u,2}; \dots; \mathbf{x}_{u,m}; \dots; \mathbf{x}_{u,M}] \quad (4.1)$$

where $[\cdot; \cdot]$ is the concatenation operator, and M is the number of user attributes. For simplicity, similar to [124, 125, 162], we consider (and hence, preprocess) all attributes to be categorical features.

$$\mathbf{e}_{u,m} = (\mathbf{F}_m \mathbf{x}_{u,m}) / c_{u,m} \quad (4.2)$$

where $\mathbf{F}_m \in \mathbb{R}^{d_f \times d_m}$ is the feature embedding matrix for the m -th user attribute, d_f is the embedding size of each feature value, $c_{u,m}$ is the number of values that user u has for the m -th user attribute, and $\mathbf{e}_{u,m} \in \mathbb{R}^{d_f}$ is the low-dimensional feature representation. After concatenating the embedded user attributes, i.e. $\mathbf{e}_u = [\mathbf{e}_{u,1}; \dots; \mathbf{e}_{u,m}; \dots; \mathbf{e}_{u,M}]$, we transform $\mathbf{e}_u$ into a fixed size representation $\mathbf{v}_{u,\text{CB}}$ with d dimensions:

$$\mathbf{v}_{u,\text{CB}} = \mathbf{W}_u \mathbf{e}_u \quad (4.3)$$

where $\mathbf{W}_u \in \mathbb{R}^{d \times (d_f \times M)}$ is a matrix to project the embedded user contents to a d -dimensional dense vector $\mathbf{v}_{u,\text{CB}}$. Equations 4.1 to 4.3, constitute the user feature extractor f_{user} in Figure 4.1. We denote the model parameters for f_{user} as $\Theta_{f_{\text{user}}} = \{\mathbf{F}_1, \mathbf{F}_2, \dots, \mathbf{F}_M, \mathbf{W}_u\}$.

With $\mathbf{v}_{u,\text{CB}}$ residing in the same latent space as the CF-based representation, $\mathbf{v}_{u,\text{CF}}$, most existing work represent a user using $\mathbf{v}_u = \mathbf{v}_{u,\text{CF}} + \mathbf{v}_{u,\text{CB}}$. However, for a new user u , her user representation degenerates to the content-based part, i.e. $\mathbf{v}_u = \mathbf{v}_{u,\text{CB}}$. Consequently, new users are merely defined by the content-based part, and do not benefit from the collaborative information that can be extracted from the historical data. This motivates us to incorporate an additional component vector, $\mathbf{v}_{u,\text{CCF}}$, which captures the collective preferences of the user neighbourhood $\mathcal{N}_u$ with respect to our target user u .

4.2.2 Neighbourhood-based Collective CF

Before we delve into our proposed Collective Collaborative Filtering (**CCF**) module which is illustrated in Figure 4.2, we first describe the process of acquiring the user neighbourhood $\mathcal{N}_u$. Following which, we highlight the intuitions behind our proposed idea.

Neighbourhood. The user neighbourhood $\mathcal{N}_u$ is defined as the set of K existing users $\{v \mid v \in \mathcal{U}_E\}$ having the highest similarity with the user u in terms of their contents.

 Figure 4.2: Neighbourhood-based Collective CF (*User Side*)

The cosine similarity between $\mathbf{x}_u$ and $\mathbf{x}_v$, i.e. the input user contents for users u and v , respectively, is used as the similarity measure. Item neighbourhoods are obtained in an identical manner.

Intuition 1: Just like the mapping-based approach introduced in Section 2.3.2, we believe that there exists a correlation between the content space and the CF latent space. However, rather than simply learning a direct transformation from a user’s contents to her CF-based representation, we turn to other existing users with similar contents. This is in fact reminiscent of an alternative CF approach: *Neighbourhood-based models*, such as [115, 83]. While traditional neighbourhood-based CF models rely on a predefined similarity measure, we would like to learn the similarity between a user and her neighbours

as part of the representation learning process.

Intuition 2: With the exception of [170], existing mapping-based approaches learn a *single* mapping between the content space and the CF space across all users. This is unrealistic as the relationship between the users' contents and their preferences can be rather convoluted. For example, the content space is often much more complex, e.g. due to a large number of attributes, as compared to the CF space. Furthermore, we hypothesise that the mapping is often based on a subset of the attributes, e.g. *age & occupation*, or *education level & occupation*, as opposed to considering the similarity across all attributes. However, since it is not feasible to enumerate all possible attributes-to-preferences combinations, a model which can emulate such a process automatically would be desirable.

Now, we describe how $\mathbf{v}_{u,\text{CCF}}$ can be obtained using the CCF module shown in Figure 4.2. Our proposed CCF module is in fact an adaptation of the **Multi-head Attention** introduced in [141]. However, to the best of our knowledge, we are the first to propose using Multi-head Attention for improving *both* warm- and cold-start recommendation.

The inputs to the CCF module are $\mathbf{v}_{u,\text{CB}}$, $\{\mathbf{v}_{v,\text{CB}} \mid v \in \mathcal{N}_u\}$, and $\{\mathbf{v}_{v,\text{CF}} \mid v \in \mathcal{N}_u\}$, which are the CB representation of the target user u , the set of CB representations of her neighbours, and the set of CF-based representations of her neighbours, respectively. Following the convention used in [141], $\mathbf{v}_{u,\text{CB}}$ is also known as the query, while $\{\mathbf{v}_{v,\text{CB}} \mid v \in \mathcal{N}_u\}$ and $\{\mathbf{v}_{v,\text{CF}} \mid v \in \mathcal{N}_u\}$ are the keys and values, respectively. Note that all input vectors are d -dimensional. As the name implies, the Multi-head Attention is essentially an ensemble of H independent attention networks. With reference to Figure 4.2, we first show the process behind a single attention network, followed by how the representations from all H networks are combined into a single fixed size representation.

The current attention network is denoted as h , and the parameters belonging to the h -th attention network are indicated using the superscript h . Each attention network starts off by performing its own projection of the **query**, **keys**, and **values** via the weight matrices $\mathbf{W}_q^h \in \mathbb{R}^{d' \times d}$, $\mathbf{W}_k^h \in \mathbb{R}^{d' \times d}$, and $\mathbf{W}_v^h \in \mathbb{R}^{d' \times d}$, respectively.

Next, the scaled dot-product is used to derive the probability distribution over the neighbours by taking the scaled dot product followed by the softmax function:

$$\boldsymbol{\alpha}_u^h = \text{softmax}((\mathbf{v}_{u,\text{CB}}^h \mathbf{W}_{\text{CB}}^h) / \sqrt{d'}) \quad (4.4)$$

where $\mathbf{v}_{u,\text{CB}}^h \in \mathbb{R}^{d'}$ is the projected CB representation of the user u , $\mathbf{V}_{\text{CB}}^h \in \mathbb{R}^{d' \times K}$ is the matrix denoting the projected CB representations of users $\{v \in \mathcal{N}_u\}$, and $\text{softmax}(w_i) = \exp(w_i) / \sum_j \exp(w_j)$. The soft attention vector $\boldsymbol{\alpha}_u^h \in \mathbb{R}^K$ reflects the importance of each neighbouring user v in the user neighbourhood $\mathcal{N}_u$, under the h -th attention network. Basically, $\boldsymbol{\alpha}_u^h$ is calculated based on a learned similarity measure between user u 's content information and the contents of neighbouring users $\{v \in \mathcal{N}_u\}$.

We can aggregate the projected CF-based representations of neighbouring users, i.e. $\mathbf{V}_{\text{CF}}^h \in \mathbb{R}^{d' \times K}$, w.r.t. the learned importance of each user v based on the following weighted sum:

$$\mathbf{v}_u^h = \sum_{k=1}^K (\boldsymbol{\alpha}_u^h[k] \mathbf{V}_{\text{CF}}^h[k]) \quad (4.5)$$

where $\boldsymbol{\alpha}_u^h[k]$ is the importance of the k -th neighbouring user, $\mathbf{V}_{\text{CF}}^h[k]$ is the corresponding CF-based vector, and $\mathbf{v}_u^h \in \mathbb{R}^{d'}$ captures the *collective* preferences of the user neighbourhood $\mathcal{N}_u$ with respect to user u under the h -th attention network.

The outputs of the H networks are a set of d' -dimensional vectors, i.e. $\{\mathbf{v}_u^1, \dots, \mathbf{v}_u^H\}$. Finally, these vectors are concatenated and transformed into the d -dimensional vector, $\mathbf{v}_{u,\text{CCF}}$, using a projection matrix $\mathbf{W}_o \in \mathbb{R}^{d \times (d' \times H)}$. The parameters for the user CCF module are defined as $\boldsymbol{\Theta}_{\text{userCCF}} = \{\{\mathbf{W}_q^h, \mathbf{W}_k^h, \mathbf{W}_v^h\}_{h=1}^H, \mathbf{W}_o\}$.

Cold-start Recommendation. Before we describe how the additional $\mathbf{v}_{u,\text{CCF}}$ bridges the gap between existing and new users (as well as items), we provide a short discussion on how the Multi-head Attention reflects our intuitions.

Intrinsically, each attention network is an independent mapping from the content space to the CF space. With its own projection matrices, each network can focus on a different content subspace (and/or CF subspace). This corresponds well with **Intuition 2**, and within each attention network, we have a learned similarity measure between the user u and her neighbours $\{v \in \mathcal{N}_u\}$ (**Intuition 1**).

For cold-start scenarios with new users (or items), as we only rely on the content-based representation of u , in addition to the content-based and CF-based representations of the K existing users in $\mathcal{N}_u$, we are *always* able to obtain $\mathbf{v}_{u,\text{CCF}}$ even if u happens to be a new user. Thus, based on the learned content-based similarities between u and her neighbours, we can fully utilise the collaborative information gleaned from these existing users to derive a better representation for u .

4.2.3 Model Inference and Optimisation

Finally, to combine the vectors $\mathbf{v}_{u,CF}$, $\mathbf{v}_{u,CB}$, and $\mathbf{v}_{u,CCF}$, in an adaptive manner [124, 125, 162], we simply use an attention network g_{user} :

$$\gamma_u = \sum_* \exp(g(\mathbf{v}_{u,*})) \quad (4.6)$$

$$\delta_{u,*} = \exp(g(\mathbf{v}_{u,*})) / \gamma_u \quad (4.7)$$

$$\begin{aligned} \mathbf{v}_u &= g_{\text{user}}(\mathbf{v}_{u,CF}, \mathbf{v}_{u,CB}, \mathbf{v}_{u,CCF}) \\ &= \sum_* \delta_{u,*} \mathbf{v}_{u,*} \end{aligned} \quad (4.8)$$

where $* \in \{CF, CB, CCF\}$, $\delta_{u,*}$ is the weight of the corresponding representation, and $g(x) = (\mathbf{h}_u)^\top (\mathbf{H}_u(x))$ is a feed-forward layer with learnable weights $\mathbf{H}_u \in \mathbb{R}^{d_a \times d}$ and $\mathbf{h}_u \in \mathbb{R}^{d_a}$. $\Theta_{g_{\text{user}}}$ are the parameters for the user attention network g_{user} .

With the final user vector $\mathbf{v}_u$ and item vector $\mathbf{v}_i$, the preference for any user-item pair can be inferred as follows:

$$\hat{r}_{u,i} = (\mathbf{v}_u)^\top \mathbf{v}_i + b_u + b_i + b_0 \quad (4.9)$$

where b_u , b_i , b_0 are the user, item, and global bias, respectively. Since the model is end-to-end differentiable, the full set of model parameters $\Theta = \{\mathbf{E}_u, \Theta_{f_{\text{user}}}, \Theta_{\text{userCCF}}, \Theta_{g_{\text{user}}}, \mathbf{E}_i, \Theta_{f_{\text{item}}}, \Theta_{\text{itemCCF}}, \Theta_{g_{\text{item}}}, \mathbf{b}_u, \mathbf{b}_i, b_0\}$ can be updated using gradient descent. We would like to point out that our proposed method is relatively lightweight; In fact, the only additional parameters (as compared to typical hybrid models such as [124, 162]) are Θ_{userCCF} and Θ_{itemCCF} . Model parameters Θ are learned by minimising the Mean Squared Error (**MSE**) between the actual ratings $\{r_{u,i}\}$ and the predicted ratings $\{\hat{r}_{u,i}\}$.

Cold Sampling. A crucial observation made by [124] is that new users (and items) are never encountered during the training phase. Hence, we adopted their proposed ‘*cold sampling*’ technique, whereby a hyperparameter β controls the percentage of users and items which are ‘dropped out’ in each batch. For example, with $\beta = 0.1$, 10% of the users in each batch will set as ‘new users’ by masking their CF-based representations with randomly initialised vectors. This enables the learning of adequate representations, as well as attention weights, for new users and/or items without actually encountering them during the training phase.

4.3 Experimental Set-up

In this section, we evaluate the performance of our proposed method CCF against various state-of-the-art baseline methods. First of all, Section 4.3.1 describes the datasets used in our empirical evaluation. In particular, in Section 4.3.1.1, we highlight an issue with the quality of the content information available in these datasets. Next, we introduce the baseline methods in Section 4.3.2. Finally, we end this section with a discussion about the evaluation procedure in Section 4.3.3.

Table 4.1: Statistical details for the datasets

Dataset	Users	Items	Ratings ¹	Density
ML-100K	943	1,682	100,000	0.063
ML-1M	6,040	3,706	1,000,209	0.044
Douban (Movie)	13,342	12,673	1,066,014	0.0063
Yelp (2020) ²	203,449	42,171	2,989,284	0.00035

Table 4.2: Content information for the datasets

Dataset	Type	Contents
ML-100K & ML-1M	User	Age, Gender, Occupation, <i>Location</i>
	Item	Genres, Year, <i>Actors, Directors, Languages, Duration</i>
Douban (Movie)	User	Friends, Groups
	Item	Actors, Directors, Genres
Yelp (2020)	Item	Categories, City, State, Opening Hours, <i>Others</i> ³

4.3.1 Datasets

As shown in Table 4.1, we consider four datasets which are relatively different from one another statistically. For instance, the *MovieLens* datasets are much denser, while the largest *Yelp* dataset is in fact very sparse. To facilitate evaluation of cold-start scenarios, i.e. Tasks 2 to 4, we are mainly interested in datasets with both user contents and item contents. The only exception is *Yelp*, which does not contain user contents. We note that

¹ All ratings are in $\{1, 2, \dots, 5\}$.

² The Yelp (2020) dataset was retrieved from <https://www.yelp.com/dataset> on 10/06/2020.

³ There are 19 one-hot attributes, e.g. Outdoor Seating, Wi-Fi, etc., and 4 multi-hot attributes.

due to issues such as user privacy, publicly available datasets with user contents are in fact *uncommon*. As such, many existing works perform their evaluation based on the two *MovieLens*⁴ datasets having both user and item contents, i.e. *ML-100K* and *ML-1M*. Table 4.2 presents the content information available for each dataset.

For *ML-100K* and *ML-1M*, *Location* consists of two attributes, *State* and *County*, which we derived based on the users' zip codes. Similarly, we collected information about the *Actors*, *Directors*, *Languages*, and *Duration* of these movies using IMDbPY⁵. The 'quality' of the user and item contents is often overlooked by existing work, and we will elaborate on its importance in Section 4.3.1.1.

Although *ML-100K* and *ML-1M* are stable benchmark datasets, they are relatively old and do not have that many users or items. Thus, we also consider *Yelp*, and a recent *Douban (Movie)* dataset crawled and preprocessed by the authors in [159, 122]. All user and item attributes in *Douban (Movie)* are anonymised.

4.3.1.1 Quality of Content Information

We find that the content information in commonly used datasets, such as *ML-100K* and *ML-1M*, are in fact **less than ideal** for complementing the historical feedback. For example, existing work such as [124, 125, 162] consider using just the *age*, *gender*, and *occupation* as the content information for the *MovieLens* users. However, if we consider the *ML-1M* dataset which has 6,040 users, the number of possible attribute values for *age*, *gender*, and *occupation* are 7, 2, and 21, respectively. A simple calculation shows that the total number of attribute combinations is at most $7 \times 2 \times 21 = 294$, a value which is much lesser than the number of users. This leads to a serious issue which, as far as we know, has never been discussed in existing work using the *MovieLens* datasets: *Many users are **indistinguishable** based on their content information alone.*

Its impact on cold-start recommendation cannot be underestimated as the only information associated with new users and items are their content information. While we try to include as many attributes as possible, such as using the location information derived from the zip codes, it only lessens the impact of this issue as there are still a considerable number of users who are indistinguishable from each other. Admittedly, many items also suffer from this problem. For instance, if we utilise just the *year* and *genres* for *ML-1M*,

⁴ <https://grouplens.org/datasets/movielens/>

⁵ <https://github.com/alberanid/imdbpy>

there are 86 movies released in 1998 categorised under the ‘drama’ *genre*. As such, we had to augment the items’ contents using another data source, e.g. IMDb for the *MovieLens* datasets. Even then, this may not be feasible for all datasets, e.g. *Douban (Movie)* which has anonymised attributes.

4.3.2 Baseline Methods

We compare our proposed method against state-of-the-art baseline methods, which can be categorised into four different approaches:

(1) ***CF-based methods***: **MF** [69] and **NeuMF** [51] learn user preferences and item profiles using a collaborative filter. While MF uses the dot product as the interaction function, NeuMF is a neural variation of MF which captures both linear and non-linear interactions between users and items.

(2) ***Content-based models***: We compare with basic CB models such as Linear Regression (**LR**) and Factorization Machine (**FM**) [111], as well as the recently proposed **AutoInt** [126]. For both existing and new users/items, these models rely only on the associated contents to derive the user-item preference. By using Multi-head Attention [141], AutoInt has outperformed many other well-known CB models such as *DeepFM* [43] and *xDeepFM* [80].

(3) ***End-to-end hybrid models***: Recent hybrid models such as **ACCM** [124] and **SAIN** [162] learn separate component vectors from the historical feedback and content information, and combine these two sources of information adaptively to derive the final user and item representation. In addition, SAIN uses Multi-head Attention [141] to model the higher-order interactions across both user and item attributes in order to produce better content-based user and item representations. However, due to the absence of historical feedback for a new user or item, these models can only utilise the content information.

(4) ***Mapping-based hybrid models***: In addition, we consider two recently proposed methods, **CB2CF** [10] and **Heater** [170], which adopts the mapping-based approach discussed in Section 2.3.2. While these models specifically target the cold-start scenarios, the modular design results in several drawbacks: (1) The two independent steps lead to separate error terms which can cause an increase in the total inference error, and (2) Their dependence on a separate model for generating pretrained CF-based

representations may precipitate poor performance in certain settings, e.g. datasets with high sparsity.

We find that end-to-end hybrid models are usually not compared with mapping-based hybrid models. As such, we are the first to compare these two approaches with one another empirically.

Figure 4.3: Data partitioning

4.3.3 Evaluation Procedure

For each dataset, the collection of ratings $\mathcal{R}$ is first divided into four partitions as illustrated in Figure 4.3. By holding out 20% of the users to form $\mathcal{U}_N$ and 20% of the items to form $\mathcal{I}_N$, we get the partitions P2, P3, and P4, which corresponds to Task 2 (User Cold), Task 3 (Item Cold), and Task 4 (Both Cold) as defined in Section 4.1. As for P1, we further hold out 10% of the users ($\approx 8\%$ of $\mathcal{U}$) and 10% of the items ($\approx 8\%$ of $\mathcal{I}$) as the cold-start validation users and cold-start validation items, respectively. The remaining user-item interactions in P1 are randomly divided using a ratio of 80:10:10 into the training set, in-matrix validation set, and in-matrix testing set, i.e. Task 1. As such, both our validation and testing sets include all four scenarios, i.e. in-matrix and the three cold-start scenarios. We repeat this 5 times by holding out a different subset of

users and items each time, whereby every user and item (eventually) gets tested for the various cold-start scenarios. This results in a **5-fold cross-validation** process, which is more robust as compared to most existing work.

Our evaluation procedure resembles [170], whereby the authors held out subsets of users and items for cold-start validation and testing. However, [170] neither evaluated the model performance for in-matrix predictions, i.e. Task 1, nor performed cross-validation. Furthermore, this is a better alternative to the evaluation framework used in [124, 125], whereby a subset (e.g. 30%) of the user and item CF-based representations in the validation and testing sets, are randomly replaced with random vectors. Perhaps unknowingly, [124, 125] are exposing their models to the CB representations of ‘new’ users and ‘new’ items, which may have **different** feature distributions from the existing users and items.

We implemented our proposed model CCF, as well as all *nine* baseline methods, using PyTorch. All models can be updated using stochastic gradient descent, and we do so using Adam [67] with an initial learning rate of 0.002 and a batch size of 128 by minimising the MSE loss. The best hyperparameters for CCF and all baseline methods are determined via a grid search based on the validation RMSE. For all models, d (user and item embeddings) and d_f (feature embeddings) are set as 64 and 16 by default. ACCM, SAIN, and CCF uses cold sampling with the hyperparameter β set between 0.05 to 0.5, and d_a (size of hidden layer in g_{user} and g_{item}) is selected amongst the values $\{8, 16, 32, 64\}$. The pretrained CF representations required for CB2CF and Heater are generated using MF. As for the Multi-head Attention used in CCF, we tried using $H = \{2, 4, 8\}$ attention networks and $K = \{5, 10, 15, 20, 25\}$ as the size of the user and item neighbourhoods, with default values of $H = 4$ and $K = 10$. Following [141], we set the projection size d' as d/H , i.e. the number of latent dimensions divided by the number of attention networks, and the computational cost is kept constant regardless of the number of attention networks used. Finally, most models utilise techniques such as *dropout* [127], *batch normalization* [59], and/or *L_2 regularisation* for better generalisation [41], and the best values (e.g. dropout rate) are chosen based on the validation RMSE.

We report Mean Absolute Error (**MAE**) and Root Mean Squared Error (**RMSE**), which are standard metrics used for datasets with explicit feedback (e.g. ratings) [2, 9, 8, 124, 26, 21, 125, 162].

4.4 Experimental Results

In this section, we start with a discussion about the performance of different models on the four datasets in Section 4.4.1. Furthermore, we examine the effects of 2 key hyperparameters, H and K , which are the number of attention networks and the neighbourhood size, respectively. Finally, we visualise the task-specific attention weights which are learned adaptively.

Table 4.3: MAE & RMSE for *ML-1M*. Best result is made **bold**, and the runner-up is underlined. All statistically significant improvements with $p\text{-value} < 0.05$ and $p\text{-value} < 0.01$ using the paired sample t-test are indicated with * and **, respectively.

Model	Task 1 (In-matrix)		Task 2 (User Cold)		Task 3 (Item Cold)		Task 4 (Both Cold)		Overall	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
MF [69]	0.685	0.867	0.795	<u>0.987</u>	0.904	1.088	0.969	1.144	0.849	1.034
NeuMF [51]	0.688	0.878	0.823	1.003	0.906	1.098	0.983	1.168	0.862	1.049
LR	0.776	0.970	0.792	0.989	0.903	1.105	0.913	<u>1.117</u>	0.850	1.049
FM [111]	0.748	0.940	0.797	0.998	0.892	1.089	0.926	1.130	0.846	1.046
AutoInt [126]	0.729	0.922	0.798	1.006	0.860	1.062	<u>0.912</u>	1.127	0.831	1.036
ACCM [124]	0.682	0.871	0.791	0.990	0.844	1.040	0.953	1.150	0.824	<u>1.020</u>
SAIN [162]	<u>0.677</u>	<u>0.867</u>	<u>0.789</u>	0.990	<u>0.838</u>	1.049	0.929	1.137	<u>0.816</u>	1.021
CB2CF [10]	0.715	0.901	0.815	1.017	0.870	1.067	0.928	1.135	0.841	1.040
Heater [170]	0.713	0.904	0.824	1.033	0.839	<u>1.034</u>	0.918	1.126	0.831	1.033
CCF	0.667	0.857	0.788	0.983	0.804	0.999	0.882	1.076	0.795	0.989
Improvement (%)	1.447 **	1.132 **	0.074	0.330	4.138 **	3.376 **	3.332 **	3.643 **	2.660 **	3.063 **

Table 4.4: MAE & RMSE for *Douban (Movie)*. Best result is made **bold**, and the runner-up is underlined. All statistically significant improvements with $p\text{-value} < 0.05$ and $p\text{-value} < 0.01$ using the paired sample t-test are indicated with * and **, respectively.

Model	Task 1 (In-matrix)		Task 2 (User Cold)		Task 3 (Item Cold)		Task 4 (Both Cold)		Overall	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
MF [69]	0.561	0.707	0.623	0.778	0.720	0.878	0.766	0.928	0.674	0.830
NeuMF [51]	0.562	0.711	0.620	0.775	0.729	0.892	0.793	0.941	0.682	0.837
LR	0.589	0.740	0.678	0.850	0.670	0.833	0.750	0.932	0.677	0.845
FM [111]	0.575	0.727	0.697	0.882	<u>0.669</u>	<u>0.832</u>	0.768	0.956	0.684	0.858
AutoInt [126]	0.585	0.736	0.636	0.798	0.678	0.845	<u>0.722</u>	0.900	0.660	0.825
ACCM [124]	0.557	0.709	0.653	0.820	0.688	0.856	0.779	0.961	0.676	0.844
SAIN [162]	<u>0.554</u>	0.707	0.638	0.801	0.676	0.849	0.758	0.941	0.662	0.831
CB2CF [10]	0.571	0.718	0.628	0.784	0.686	0.850	0.726	<u>0.899</u>	0.658	0.819
Heater [170]	0.557	<u>0.706</u>	0.634	0.794	0.671	0.837	0.725	0.900	<u>0.653</u>	<u>0.816</u>
CCF	0.549	0.702	<u>0.621</u>	<u>0.775</u>	0.652	0.814	0.710	0.881	0.639	0.798
Improvement (%)	0.944 *	0.501	-0.193	-0.010	2.538 **	2.169 **	1.591	2.000 **	2.247 **	2.241 **

4.4.1 Results and Discussion

4.4.1.1 ML-1M and Douban (Movie)

Tables 4.3 and 4.4 present the task-specific results as well as the overall performance⁶ for *ML-1M* and *Douban (Movie)*, respectively.

With the exception of Task 2, CCF achieves the best performance against all baseline methods. Improvements (against the best baseline in each task) are usually statistically significant with $p < 0.01$. Additionally, while CCF has consistently good performance across all tasks, we note that no baseline method constantly outperforms the rest in every task. For example, SAIN may be the best baseline method for some tasks, while Heater or AutoInt performs better in other tasks. We observe that mapping-based approaches may be worse than the end-to-end hybrid methods for Task 1, but their performance in cold-start scenarios are generally much better as they capture the collaborative information (from existing users and items) implicitly via the mapping functions. Overall, the results highlight two salient benefits of our proposed model: **(1)** CCF will perform well even in a dynamic environment with new users and/or new items, and **(2)** By exploiting the collective preferences, CCF is able to outperform existing methods even in Task 4, which is the most challenging and often overlooked.

As for Task 2, the lacklustre performance could be attributed to the poor user content information, as we have previously discussed in Section 4.3.1.1. For the 6,040 users in *ML-1M*, 39.7% of them have ‘*non-unique*’ contents, i.e. for these users, there is at least one other user with *identical* content information. Thus, it is impossible to distinguish two (or more) users who have identical attributes and yet, different preferences. As for *Douban (Movie)*, the available user contents are ‘*social ties*’, which may not be an accurate reflection of the individual user preferences towards these movies.

Based on the experimental results, we can observe that simply including user and item contents does not necessarily lead to large improvements in the model performance. Even for Task 1, whereby the content information complements the historical feedback, the performance of existing hybrid methods are only slightly better than CF-based methods. The quality and hence, usefulness, of the content information is often overlooked by previous work, and we believe that this deserves careful consideration moving forward.

⁶ The overall performance for a given metric is the weighted average of the task-specific results, i.e. Overall MAE (or RMSE) = $\sum_{i=1}^4 |\text{Task } i| * \text{Task } i\text{'s MAE (or RMSE)} / \sum_{i=1}^4 |\text{Task } i|$

Figure 4.4: Overall MAE for ML-100K and Yelp

Figure 4.5: Overall RMSE for ML-100K and Yelp

4.4.1.2 ML-100K and Yelp

In the interest of space, we only show the overall performance for *ML-100K* and *Yelp* in Figures 4.4 and 4.5, respectively. There are no user contents for *Yelp* and the evaluation is only performed for Tasks 1 & 3.

We find that the same observations can be made for these two datasets. For *Yelp*, the overall improvements over the best baseline (Heater) in terms of MAE and RMSE are 1.795% and 0.969%, respectively, and these are statistically significant with $p < 0.01$. As for *ML-100K*, the overall improvements over the best baseline (ACCM) are 3.078% and 2.524% in terms of MAE and RMSE, respectively, and these are statistically significant with $p < 0.05$. Similar to *ML-1M* and *Douban (Movie)*, CCF produces consistently good performance across all tasks (except Task 1 for *Yelp* whereby NeuMF marginally outperforms CCF due to the sparsity of the dataset).

4.4.2 Effects of Hyperparameters H and K

Figures 4.6 (a) and 4.6 (b) show the effects of crucial hyperparameters on the validation RMSE for CCF on *ML-1M* and *Douban (Movie)*, respectively. Although optimal values for the number of attention networks, H , and neighbourhood size, K , most likely depend on the given dataset, we find it to be quite robust across different settings. Generally, we observe that CCF performs well simply by using $H = 4$ attention networks and $K = 10$ (or 15) neighbours.

Space & Time Efficiency. Additionally, while increasing the number of attention networks may improve the performance for certain datasets, both the number of required parameters and computational cost are kept constant as we follow [141] and set the projection size d' as d/H . On the other hand, using a larger neighbourhood size does incur a (moderately) higher computational cost. However, the CCF module involves mainly matrix multiplications and dot products, which are both highly efficient operations that can be performed in parallel. Coupled with the relatively small number of neighbours required for adequate performance, we observe that the time cost per epoch for CCF is only marginally higher than that of other hybrid models such as [124, 162], and CCF can converge in a similar number of epochs.

4.4.3 Task-specific Attention Weights

Figures 4.7 (a), 4.7 (c), 4.7 (b), and 4.7 (d) illustrate the task-specific attention weights for *ML-100K*. For new users (Tasks 2 & 4), we find that $\delta_{u, \text{CCF}}$ is much higher than the remaining two components, as the user contents are not very informative due to its poor quality. The final item representations for new items (Tasks 3 & 4) tend to be

(a): ML-1M

(b): Douban (Movie)

Figure 4.6: Effect of hyperparameters K and H on the validation RMSE

divided between the content-based representations and our CCF-based representations. Since *ML-100K* has additional information from IMDb, the item contents are much more useful. Finally, for Task 1, the CCF-based representations play an important role for some existing items as they do not have many interactions, i.e. less popular movies. These task-specific weights emphasise the importance of our CCF-based representations for both warm- and cold-start recommendation.

Figure 4.7: Task-specific attention weights for ML-100K

4.5 Summary

In this chapter, we presented *Collective Collaborative Filtering (CCF)*, a novel hybrid model which seamlessly handles *both* warm- and cold-start recommendation. By adapting the idea of *Multi-head Attention* to fuse the content space and the CF-based latent space, all users and items, both existing and new, are able to leverage the collaborative information extracted from the historical user-item interactions. Furthermore, we highlighted

challenges which are present in existing work, such as the poor content information in commonly used datasets. Our evaluation framework is robust, and it explicitly shows the performance of different methods for *both* warm-start recommendation as well as the various cold-start scenarios. Experimental results on four publicly available datasets highlight the competitive performance of our proposed method.

Chapter 5

The Datasets Dilemma

In order to guarantee the validity of the obtained results, the evaluation procedure needs to be well-defined and robust. As highlighted in Section 2.4.1, several recent studies have expressed concern about the experimental issues present in recommendation systems. In this study, we focus on a fundamental but often neglected aspect of the evaluation procedure, i.e. the datasets themselves.

We perform various analyses and experiments in this study to improve our understanding of the datasets used for evaluating *top-K item recommendation* in a systematic and comprehensive manner. Specifically, we adopt the following three steps: **(1)** First, in Section 5.1, we examined the proceedings of 5 top-tier conferences (*KDD*, *SIGIR*, *TheWebConf*, *WSDM*, and *RecSys*) for the past 5 years (2016 to 2020). The gathered papers are manually filtered, and we narrowed it down to 48 papers which address the problem of top-K item recommendation based on implicit feedback. There are a total of 45 publicly available datasets which have been used in one or more of these papers, and we performed various analyses to illustrate *how* different datasets have been utilised in recent papers. **(2)** After which, in Section 5.2, we introduce a set of well-defined data characteristics. These characteristics can be used as a representation for each of these datasets, and at the same time, are easily obtained by analysing the (binary) user-item interaction matrix. We leveraged these characteristics to clearly understand *what* are the similarities as well as differences between various datasets. **(3)** Finally, in Section 5.3, we performed an empirical study using several representative methods, such as Weighted Matrix Factorization (**WMF**) [58], Item k-Nearest Neighbour (**ItemKNN**) [115], and Mult-VAE [82], to find out *if* the choice of datasets used for evaluating item recommendation could influence the observations and/or conclusions obtained.

Table 5.1: Recent papers for implicit feedback based top-K recommendation

Conference	Year					Total
	2016	2017	2018	2019	2020	
KDD	-	-	[20, 28]	[24]	[130, 60, 98, 62, 94]	8
RecSys	[109, 91, 27]	[102]	[167]	[88, 99, 35]	-	8
SIGIR	[163, 53]	[14]	[50, 34]	[85, 150, 153]	[19, 123, 39, 151, 49, 131]	14
WSDM	[157, 155]	-	-	[100]	[128, 166, 86, 144, 121]	8
TheWebConf	[81]	[52]	[75, 136, 82]	[138]	[61, 87, 64, 79]	10
Total	8	3	8	9	20	48

5.1 Collection and Analysis

5.1.1 Paper and Dataset Collection

We start by examining the proceedings of 5 top-tier conferences (*KDD*, *SIGIR*, *TheWebConf*, *WSDM*, and *RecSys*) for the past 5 years (2016 to 2020), and gathering a list of long papers focusing on implicit feedback based top-K recommendation. Concretely, we performed a keyword search to narrow down to around 400 papers with titles containing either ‘*recommend*’ or ‘*collaborative*’. Next, we manually filtered these papers to include only those which (1) focuses on implicit feedback top-K recommendation, (2) evaluates using classification and/or ranking metrics, such as Precision, Recall, and Normalized Discounted Cumulative Gain (**nDCG**), and most importantly, (3) utilises at least one publicly available dataset from a reliable source¹. In other words, we do not include papers which either (1) formulate it as a rating prediction problem, or (2) tackle other recommendation tasks, e.g. content-aware or session-based recommendation. As summarised in Table 5.1, we managed to obtain a total of 48 long papers which satisfy the aforementioned requirements. Collectively, there are 45 publicly available datasets² which have been used in one or more of these papers. The strict requirements imposed for collecting & filtering the datasets lead to a useful property for analysing the usage patterns: *A dataset used in any single one of these papers can be used in every other paper as well*. Put simply, there is no reason for a dataset to be inapplicable for any of

¹ Examples of reliable sources are grouplens.org/datasets/ and snap.stanford.edu/data/. There are some papers which are evaluated on datasets that are no longer publicly available for various reasons, e.g. some datasets cannot be downloaded after the competition ends.

Figure 5.1: Dataset usage patterns (*Best viewed in colour*)

these papers.

5.1.2 Dataset Usage Analysis

Figure 5.1 illustrates the usage patterns of the 45 datasets across all 48 papers using a scatter plot. The datasets are sorted in descending order (from top to bottom) based on their usage frequencies, i.e. the number of papers using each dataset. For convenience, we have included the usage frequencies corresponding to each dataset on the right side of Figure 5.1. Similarly, the papers are sorted in descending order (from left to right) based on the number of publicly available datasets used in each paper.

There are 11 datasets (24.45%) used in 5 or more papers, 10 datasets (22.22%) utilised in 2 to 4 papers, and 24 datasets (53.33%) which have been used in just 1 paper. Unsurprisingly, a handful of these datasets happen to be really popular, e.g. *Netflix*, *MovieLens-1M* (**ML-1M**), *Yelp*, and *MovieLens-20M* (**ML-20M**). However, we can also observe

² Different versions of a dataset are counted only once. For instance, Yelp (yelp.com/dataset) releases a different version of the dataset for each iteration of its dataset challenge.

Table 5.2: Combinations of datasets used together in 2 or more papers

Datasets	Papers
Epinions, ML-20M, Netflix, Yelp	[88, 87]
ML-20M, Million Song Dataset, Netflix	[35, 128, 121, 82, 64]
Amazon (Books), Gowalla, Yelp	[150, 151, 49]
Amazon (CDs & Vinyl), Amazon (Electronics), Gowalla	[94, 131]
Flixster, ML-10M, Netflix	[28, 27]
ML-100K, ML-1M, Netflix	[75, 138]
ML-10M, Netflix, Yelp	[157, 155]
ML-1M, ML-20M, Meetup (NYC)	[144, 136]

that the majority of these datasets are underutilised.

As for how different datasets have been used in recent papers, we note that there is little to no regularity. On one end, we have papers like [136] and [144] performing their evaluation on 10 datasets and 9 datasets, respectively. On the other end, an overwhelming majority of these papers, i.e. 36 (or 75%) of them, utilised at most 3 publicly available datasets. Furthermore, we use the Apriori algorithm [3] to determine the combinations of datasets which have been used together in 2 or more papers. Table 5.2 presents the dataset quadruplets and triplets which are commonly used together. First, we have [87] and [88] with 4 datasets in common. However, it should be noted that both papers are written by the same author. Next, there are 7 different dataset triplets which appear in 2 or more papers. Notably, the triplet $\{ML-20M, Million\ Song\ Dataset, and\ Netflix\}$ has been used together by 5 different papers. In addition, [136] and [144] ended up having just 3 datasets, i.e. $\{ML-1M, ML-20M, and\ Meetup\ (NYC)\}$, in common despite the substantial number of datasets used for evaluation. Finally, while not shown in Table 5.2, there are also 21 distinct pairs of datasets which have been used in multiple papers. Amongst them, the most frequent pairing would be $\{ML-20M, Netflix\}$, which has been evaluated at the same time in 9 separate papers.

Overall, although all the aforementioned papers address the exact same problem of implicit feedback based top-K recommendation, our analyses have demonstrated that the choice of datasets is often determined arbitrarily. Even when two papers have been evaluated on the same dataset(s), the results are *not* necessarily comparable as they could be using different data pre-processing or splitting strategies [132].

5.2 Datasets and their Characteristics

To facilitate efficient, fair, and reproducible comparison, we could consider having a set of benchmark datasets. For example, [132] has constructed and released an evaluation benchmark with 6 datasets by considering their popularity and domain coverage. However, we argue that the popularity of a dataset is essentially a subjective measure. In practice, many researchers tend to rely on a diverse collection of datasets, e.g. in terms of quantifiable dataset characteristics such as size and density, in order to validate the robustness of their proposed methods. Therefore, as described next in Section 5.2.1, we propose using a more complete set of dataset characteristics as a representation for each dataset. These characteristics are then leveraged in Section 5.2.2 to illustrate the similarities and differences of different datasets.

5.2.1 Dataset Characteristics

Even after decades of research, we still lack a thorough understanding of the characteristics of different recommendation datasets. In [1], the authors used an exploratory modelling (**EM**) approach to investigate the impact of different dataset characteristics on the performance of various recommendation algorithms for the *rating prediction* task. More recently, [31] adopted an identical approach in order to assess the *robustness* of several recommender algorithms against *shilling attacks*, using the same characteristics as [1]. Inspired by [1] and [31], we consider two different types of data characteristics which can be easily derived from the user-item interaction matrix in the *implicit feedback* setting.

5.2.1.1 Structural Characteristics

First, we introduce 3 data characteristics which are based on the structure of the interaction matrix, namely $Space_{log}$, $Shape_{log}$, and $Density_{log}$. For any dataset, we have the set of users $\mathcal{U}$, the set of items $\mathcal{I}$, and the set of interactions $\mathcal{K}$. Structural data characteristics can be easily obtained by building upon basic statistics such as the number of users $|\mathcal{U}|$, the number of items $|\mathcal{I}|$, and the number of interactions $|\mathcal{K}|$.

Definition 5.1 ($Space_{log}$) *For any given dataset, $Space_{log}$ is defined as follows:*

$$Space_{log} = \log_{10}\left(\frac{|\mathcal{U}| \times |\mathcal{I}|}{sc}\right) \quad (5.1)$$

whereby sc is a scaling factor to constrain the possible values of $|\mathcal{U}| \times |\mathcal{I}|$. Following [1], the value of sc is set as 1,000.

Definition 5.2 ($Shape_{log}$) For any given dataset, $Shape_{log}$ can be derived as follows:

$$Shape_{log} = \log_{10}\left(\frac{|\mathcal{U}|}{|\mathcal{I}|}\right) \quad (5.2)$$

Definition 5.3 ($Density_{log}$) For any given dataset, we can calculate its $Density_{log}$ as follows:

$$Density_{log} = \log_{10}\left(\frac{|\mathcal{K}|}{|\mathcal{U}| \times |\mathcal{I}|}\right) \quad (5.3)$$

The number of users $|\mathcal{U}|$ and the number of items $|\mathcal{I}|$ can fall within a wide spread of values, ranging from hundreds to even millions of users/items. Similarly, the number of interactions $|\mathcal{K}|$ can be as little as $\sim 35K$ (FilmTrust), or as much as $\sim 100M$ (Netflix). Therefore, all structural characteristics are log transformed in order to normalise its distributions and enable meaningful comparisons between datasets with significantly different statistics.

Intuitively, $Space_{log}$ reflects the size (and even the scale) of the user-item interaction space, $Shape_{log}$ measures the ratio of users to items, and $Density_{log}$ represents the proportion of observed interactions among all the possible user-item interactions. Structural characteristics are easily interpretable and closely related to one another. For 2 datasets with similar $Space_{log}$ and $Density_{log}$, the dataset with a considerably larger $Shape_{log}$ would have more users and fewer items, and consequently, less observed interactions per user and more observed interactions per item.

5.2.1.2 Distributional Characteristics

Here, we describe 2 data characteristics, i.e. $Gini_{user}$ and $Gini_{item}$, which captures how the interactions are distributed across both the users and the items.

Definition 5.4 ($Gini_{user}$) For any given dataset, $Gini_{user}$ is defined as follows:

$$Gini_{user} = 1 - 2 \sum_{u=1}^{|\mathcal{U}|} \left(\frac{|\mathcal{U}| + 1 - u}{|\mathcal{U}| + 1} \right) \times \left(\frac{|\mathcal{K}_u|}{|\mathcal{K}|} \right) \quad (5.4)$$

where $|\mathcal{K}_u|$ is the number of interactions belonging to the user u .

Definition 5.5 ($Gini_{item}$) *For any given dataset, $Gini_{item}$ can be derived as follows:*

$$Gini_{item} = 1 - 2 \sum_{i=1}^{|\mathcal{I}|} \left(\frac{|\mathcal{I}| + 1 - i}{|\mathcal{I}| + 1} \right) \times \left(\frac{|\mathcal{K}_i|}{|\mathcal{K}|} \right) \quad (5.5)$$

with $|\mathcal{K}_i|$ being the number of interactions belonging to the item i .

$Gini_{user}$ and $Gini_{item}$ utilise the Gini coefficient [40] to measure the distribution of interactions over the set of users $\mathcal{U}$ and the set of items $\mathcal{I}$, respectively. Since these 2 characteristics are similar in nature, we will describe them by using $Gini_{item}$ as an example. $Gini_{item}$ takes on values between 0 and 1 (both inclusive), and a value of 0 implies total equality (i.e. all items are equally popular and have the same number of observed interactions), while a value of 1 indicates maximal inequality whereby all observed interactions belong to a single item. In reality, skewed frequency distributions are rather common across many application scenarios. For example, a blockbuster movie can attract much more viewers as compared to a niche movie with a relatively smaller target audience.

Unlike structural characteristics, distributional characteristics are rarely considered and/or discussed in most recommendation papers. However, we note that there are papers such as [101] which specifically address the long-tail problem whereby a large number of items receive too few interactions to be modelled and recommended satisfactorily. In practice, there are often multiple measures that can be used for characterising the frequency distributions of the user-item interaction data. Having said that, it was found in [1] that most of these metrics are highly correlated with one another. Therefore, following [1] and [31], we only consider $Gini_{user}$ and $Gini_{item}$ in this paper.

5.2.1.3 Influence of Dataset Characteristics

Structural and distributional characteristics are capable of capturing the intricate relationships between users, items, and interactions in a cohesive manner. Notably, their importance should not be underestimated as these data characteristics can affect most recommendation algorithms in one way or another. For example, it is commonly assumed that datasets with higher density would lead to better performance across recommendation algorithms in general as there would be more interactions observed for each user/item under consideration. Similarly, for a dataset with a high $Shape_{log}$ (i.e. more users than

items) and $Gini_{user}$ (i.e. there exists some users with many observed interactions), there is a smaller set of candidate items, and at the same time, there would be a handful of active users who have interacted with many of these items. For such datasets, recommender systems adopting a user-based approach such as UserKNN might perform better as (1) having more candidate users increase the likelihood of potentially finding similar users, and (2) it would be easier to find a similar user among these active users with high item coverage. In some way, it is reasonable to expect these dataset characteristics to influence the performance of commonly used algorithms as most of them are still based on the age-old concept of Collaborative Filtering (**CF**) [58], i.e. identifying similar pairs of items and/or like-minded users. Therefore, we believe that it would be both important and beneficial to have an understanding of recommendation datasets from the perspective of data characteristics.

5.2.2 Similarities and Differences

To the best of our knowledge, there is no existing work which strives to set apart individual recommendation datasets based on their (measurable) similarities or differences. We begin by considering the 45 datasets shown in Figure 5.1 (Section 5.1). 3 of these datasets³ are excluded as they are too small and sparse for any meaningful comparison.

Furthermore, we also include the remaining 9 Amazon datasets⁴ which have not been utilised in any of the recent papers for top-K recommendation. This leaves us with a total of 51 datasets.

In practice, datasets used for evaluating recommender systems are rarely used as they are. Therefore, we preprocess these datasets before deriving their data characteristics. Most, if not all, research papers would remove the cold-start users/items (i.e. users/items

³ The 3 datasets that are excluded are { HetRec2011-Delicious-2K; Twitter (USA); Twitter (WW) }. These datasets have between $\sim 35K$ to $\sim 105K$ interactions and between $\sim 36K$ to $\sim 69K$ items, and are too small and sparse for meaningful analysis after the data preprocessing step.

⁴ The remaining 9 Amazon datasets are { Apps for Android; Baby; Digital Music; Health & Personal Care; Home & Kitchen; Musical Instruments; Office Products; Patio, Lawn & Garden; Tools & Home Improvement }.

⁵ As the final preprocessing step, for datasets with explicit feedback (e.g. ratings), we convert all the observed entries into positive interactions.

⁶ The K -core (e.g. used in [138, 155]) is derived by using a *recursive filter* such that all remaining users and items have $\geq K$ interactions each.

⁷ We use the latest version of the Yelp dataset (Retrieved on 20th January 2021).

⁸ The original dataset consists of more than $1.8M$ users and $717M$ ratings, and it has been partitioned into 10 subsets. In this paper, we only utilise the first subset of $200K$ users.

Table 5.3: Source and processing steps taken for the 51 datasets used in Sections 5.2.2 and 5.3

Dataset(s)	Source	Preprocessing Steps ⁵
24 Amazon Datasets , e.g. Amazon (Books), Amazon (Electronics), etc.	jmcauley.ucsd.edu/data/amazon	We directly use the 5-core ⁶ version whereby all users and items have ≥ 5 interactions each.
Amazon Fine Food	kaggle.com/snap/amazon-fine-food-reviews	We preprocess these datasets to get its 5-core.
BookCrossing	www2.informatik.uni-freiburg.de/~cziegler/BX/	
CiteULike-a	github.com/js05212/citeulike-a	
CiteULike-t	github.com/js05212/citeulike-t	
Epinions	trustlet.org/downloaded.epinions.html	
FilmTrust	guoguibing.github.io/librec/datasets.html	
Flixster	sites.google.com/view/mohsenjamali/flixter-data-set	
GoodReads (Comics)	sites.google.com/eng.ucsd.edu/ucsdbookgraph/home	
HetRec2011-LastFM-2K	grouplens.org/datasets/hetrec-2011/	
Last.fm 1K	ocelma.net/MusicRecommendationDataset/lastfm-1K.html	
Last.fm 360K	ocelma.net/MusicRecommendationDataset/lastfm-360K.html	
Meetup (NYC)	personal.ntu.edu.sg/gaocong/datacode.htm	
Netflix	kaggle.com/netflix-inc/netflix-prize-data	
Yahoo! R1	webscope.sandbox.yahoo.com/catalog.php?datatype=r	
Yahoo! R4	webscope.sandbox.yahoo.com/catalog.php?datatype=r	
Yelp ⁷	yelp.com/dataset	
Gowalla	github.com/dawenl/expo-mf	We use the dataset from [81] where venues have ≥ 20 check-ins, and we remove users with ≤ 5 check-ins.
Million Song Dataset	millionsongdataset.com/	We follow [82] and retain users with ≥ 20 interactions and songs with ≥ 200 interactions.
HetRec2011-ML-2K	grouplens.org/datasets/hetrec-2011/	The users in these datasets have ≥ 20 interactions, and we remove items with ≤ 5 interactions.
ML-100K	grouplens.org/datasets/movielens/100k/	
ML-10M	grouplens.org/datasets/movielens/10m/	
ML-1M	grouplens.org/datasets/movielens/1m/	
ML-20M	grouplens.org/datasets/movielens/20m/	
Pinterest	github.com/hexiangnan/neural_collaborative_filtering	
Goodbooks-10k	github.com/zygmuntz/goodbooks-10k	We directly use these datasets as they are.
Million Song Dataset (Taste Profile Subset)	millionsongdataset.com/tasteprofile/	
Yahoo! R2 ⁸	webscope.sandbox.yahoo.com/catalog.php?datatype=r

Figure 5.2: Pairwise Euclidean distance between different datasets (*Best viewed in colour*)

with too few interactions) as there is insufficient information for recommendation systems to provide inference for these extremely inactive users or highly unpopular items. We would like to point out that some of these datasets are made publicly available in a (partially) preprocessed form. For instance, the classic MovieLens datasets (i.e. ML-100K, ML-1M, ML-10M, and ML-20M) do not include users with less than 20 interactions. As for datasets with explicit feedback (e.g. ratings), we simply follow the commonly used approach (e.g. in [27, 102, 167, 99]) of converting all the observed entries into positive interactions so as to make them suitable for the implicit feedback setting. Table 5.3 provides the source and preprocessing steps taken for the 51 datasets used in the remainder of this paper.

By using the 5 data characteristics described in Section 5.2.1, we can depict the similarities and differences between various datasets based on their pairwise Euclidean distance as shown in Figure 5.2. Datasets with a small Euclidean distance are more similar, and datasets with a large Euclidean distance are significantly different from one

Table 5.4: Dataset clusters obtained using k-means

Cluster	Datasets	Size
1	Amazon (Books; CDs & Vinyl; Electronics; Kindle Store; Movies & TV); GoodReads (Comics); Last.fm 360K; Million Song Dataset; Yahoo! R1; Yahoo! R2; Yelp	11
2	Amazon (Amazon Instant Video; Automotive; Digital Music; Office Products); Amazon Fine Food; CiteULike-a; CiteULike-t; HetRec2011-LastFM-2K; HetRec2011-ML-2K; Last.fm 1K; ML-1M; Yahoo! R4	12
3	Amazon (Apps for Android; Baby; Beauty; Cell Phones & Accessories; Clothing, Shoes & Jewelry; Grocery & Gourmet Food; Health & Personal Care; Home & Kitchen; Pet Supplies; Sports & Outdoors; Tools & Home Improvement; Toys & Games; Video Games); BookCrossing; Epinions; Gowalla; Meetup (NYC)	17
4	Amazon (Musical Instruments; Patio, Lawn & Garden); FilmTrust; ML-100K	4
5	Flixster; Goodbooks-10K; ML-10M; ML-20M; Million Song Dataset (Taste Profile Subset); Netflix; Pinterest	7

another. For example, Amazon (Musical Instruments) and Amazon (Patio, Lawn & Garden) have nearly identical characteristics as they are both small and dense datasets with slightly more users than items. In contrast, a gigantic but sparse dataset like Amazon (Books) (with $\sim 9M$ interactions) is very different from small and dense datasets like FilmTrust and ML-100K (with less than $100K$ interactions).

5.2.2.1 Clustering Datasets based on their Characteristics

In order to obtain a better overview of recommendation datasets, we can cluster these datasets and take a closer look at the characteristics of each cluster. Naturally, datasets in the same cluster are more similar to one another, and at the same time, considerably different from datasets in other clusters. Each dataset is represented by a 5-dimensional vector, i.e. $[Space_{log}, Shape_{log}, Density_{log}, Gini_{user}, Gini_{item}]$, and we use k-means (i.e. based on Euclidean distance) to cluster these 51 datasets. Empirically, we set the number of clusters to 5⁹. Table 5.4 lists the datasets in each cluster while Table 5.5 presents the characteristics of each cluster (i.e. the cluster centroid).

Cluster 1 consists of gigantic but sparse datasets with the most users and items, e.g. Last.fm 360K which has $\sim 359K$ users, $\sim 88K$ items, and $\sim 17M$ interactions. As for **Cluster 2**, it contains datasets with much fewer interactions ($\leq 1M$ interactions) and the number of users and items are often just in the thousands. Unlike other clusters, there

⁹ For the clustering, we considered internal validation measures such as Silhouette Coefficient and Davies-Bouldin Score, and the optimal number of clusters is 4 or 5.

Table 5.5: Characteristics (i.e. cluster centroid) of all dataset clusters. For individual data characteristic, we rank the clusters based on their corresponding values in descending order.

Cluster	$Space_{log}$	$Shape_{log}$	$Density_{log}$	$Gini_{user}$	$Gini_{item}$
1	7.274 (1)	0.497 (2)	-3.412 (5)	0.477 (2)	0.657 (2)
2	4.340 (4)	-0.134 (5)	-2.162 (3)	0.441 (3)	0.517 (4)
3	5.619 (3)	0.272 (3)	-3.106 (4)	0.337 (4)	0.504 (5)
4	3.167 (5)	0.116 (4)	-1.670 (1)	0.289 (5)	0.557 (3)
5	6.307 (2)	0.878 (1)	-2.120 (2)	0.502 (1)	0.767 (1)

are several datasets here with significantly more items than users (hence the negative $shape_{log}$). For instance, CiteULike-t has $\sim 4K$ users, $\sim 7K$ items, and $\sim 82K$ interactions. The datasets in **Cluster 3** have a moderate number of users, items, and interactions, and are relatively sparse as compared to datasets in other clusters. Next, we have **Cluster 4** with just 4 datasets. This is an interesting cluster as the datasets here have the least number of users, items, and interactions, as well as the highest densities. The frequently used ML-100K dataset, which has 943 users, 1,349 items, and 99,287 interactions, turns out to be the dataset with the most interactions within this cluster. Finally, **Cluster 5** contains datasets with a considerable amount of users and items (much lesser than Cluster 1, but slightly more than Cluster 3) and the datasets are quite dense as well. Notably, the datasets in this cluster have a lot more users than items. For example, ML-20M has roughly 7 times more users than items with $\sim 138K$ users and $\sim 18K$ items. In addition, many frequently used datasets such as ML-10M, ML-20M, Netflix, and Pinterest belong to Cluster 5. Last but not least, we can observe that the interactions are usually more evenly distributed across users than items across all clusters. The most popular items can easily amass thousands or even millions of interactions, but it is rather unlikely even for an extremely active user to have that many interactions. By leveraging the various characteristics and focusing solely on the datasets themselves, we are able to put forward a thorough analysis, and consequently, improve our perception of different recommendation datasets substantially.

5.3 Experiments and Results

Up to this point, we have seen how the choice of datasets is often determined arbitrarily (Section 5.1), as well as how various datasets can be distinctively different from one

Table 5.6: Representative datasets for each dataset cluster

Cluster	Representative Datasets
1	Amazon (Electronics); Amazon (Movies & TV); Last.fm 360K
2	Amazon (Digital Music); CiteULike-t; Yahoo! R4
3	Amazon (Beauty); Amazon (Toys & Games); Amazon (Video Games)
4	Amazon (Musical Instruments); Amazon (Patio, Lawn & Garden); ML-100K
5	Flixster; ML-10M; ML-20M

another (Section 5.2). Nonetheless, a key question still remains: Could it actually affect the observations and/or conclusions obtained *if* we select the datasets in a different manner? For instance, would we obtain similar results across different recommendation algorithms if we utilise datasets with similar characteristics? In this section, we conduct an empirical study to investigate the performance of several well known recommendation algorithms using a wide variety of datasets with different characteristics.

5.3.1 Experimental Setup

First of all, it is impractical to evaluate and analyse the performance of multiple recommendation algorithms on all 51 datasets in Table 5.3. Therefore, for each cluster, we select the 3 datasets which are closest to the corresponding cluster centroid based on their Euclidean distances. These selected datasets would have data characteristics which are highly representative of their own cluster, and at the same time, be significantly different from datasets in other clusters. Overall, as shown in Table 5.6, we conduct the experiments on a total of 15 datasets (5 clusters x 3 datasets).

Next, for each dataset, we adopt the widely used *leave-one-out* approach [62, 50, 52, 136, 138] to construct the training, validation, and testing sets. For each user, the latest interaction is used for testing, the penultimate interaction is used for validation, and all her remaining interactions are used for training. As for datasets without timestamps, the validation and testing interactions for each user are sampled randomly.

For all 15 datasets, we evaluate the performance of 4 different recommendation algorithms as follows:

- (i) **UserKNN**: UserKNN [54] is a classic *user neighbourhood-based* CF approach using the Cosine Similarity to find like-minded users. Following [30], both the neighbourhood size and shrinkage term are selected from [5, 1000].
- (ii) **ItemKNN**: ItemKNN [115] is an *item neighbourhood-based* variant of UserKNN. Similarly, we use the Cosine Similarity, and both the neighbourhood size and shrinkage term are selected from [5, 1000].
- (iii) **RP3beta**: RP3beta [106] is a simple yet effective *graph-based* method which performs a random walk between users and items based on the observed interaction matrix. The similarity between two items is further divided by each item’s popularity raised to the power of a coefficient β . The number of neighbours is set between [5, 1000], and both the damping factor α and the coefficient β are set to real values between [0, 2].
- (iv) **WMF**: Weighted Matrix Factorization (**WMF**) [58] uses a *latent factor model* approach designed specifically for the implicit feedback setting. Following [82, 78], the weight for unobserved entries is set to 1, while the weight for observed entries (i.e. *confidence*) is chosen from [2, 100]. The number of latent factors is set between [100, 200], and WMF is trained using Alternating Least Squares (**ALS**).

The recommendation algorithms are selected for various reasons: **(1)** They represent different ‘*families*’ of recommendation methods with distinct inductive bias, i.e. *neighbourhood-based*, *graph-based*, and *latent factor models*. **(2)** These straightforward and yet effective algorithms do not require extensive hyperparameter tuning to achieve satisfactory performance. **(3)** Finally, if the hyperparameters are selected adequately [30], conceptually simple methods (e.g. ItemKNN and RP3beta) have been shown to outperform recently proposed methods such as CMN [34], NeuMF [52], and SpectralCF [167].

The evaluation is performed using both classification and ranking based metrics, i.e. Recall@K and nDCG@K, respectively. Recall@K measures whether the ground truth item is present in the top-K recommendation list. Since we are using the *leave-one-out* approach, Recall@K is equivalent to the Hit Ratio (**HR**) [52]. Other than that, we also consider the Normalized Discounted Cumulative Gain @ K (**nDCG@K**) which accounts for the position of the ground truth item within the top-K recommendation list. As suggested by [70], we use the *exact version* of these metrics (i.e. we rank the ground truth item among all candidate items), and the number of recommended items, i.e. K, is set to 10. Additionally, to determine the ideal hyperparameters for all recommendation algorithms on each dataset, we adopt a Bayesian search strategy¹⁰

based on the $nDCG@10$ of the corresponding validation set. For each algorithm, we record its $Recall@10$ and $nDCG@10$ obtained on the testing set when the corresponding validation $nDCG@10$ is the highest.

Figure 5.3: $Recall@10$ of 4 recommendation algorithms for the 15 datasets (*Best viewed in colour*)

Figure 5.4: $nDCG@10$ of 4 recommendation algorithms for the 15 datasets (*Best viewed in colour*)

¹⁰ Specifically, we use a wrapper function from <https://scikit-optimize.github.io>, and for each algorithm, we execute a total of 35 runs whereby the first 5 runs are random initial points. The Bayesian search strategy facilitates efficient hyperparameter tuning for all recommendation algorithms, and reduces the likelihood of subpar performance due to poorly chosen hyperparameters.

5.3.2 Experimental Results

Figures 5.3 and 5.4 present the Recall@10 and the nDCG@10 of different recommendation algorithms for the 15 datasets, respectively. As the results (i.e. relative performance of different algorithms for each dataset) are nearly identical, we will focus on nDCG@10. Unless stated otherwise, the observations do hold in terms of Recall@10 as well.

- For **Cluster 1**, in most cases, RP3beta tends to perform well while WMF is always the worst method¹¹. While the results for Amazon (Electronics) and Amazon (Movies & TV) are rather similar, we can see that UserKNN does surprisingly well for the Last.fm 360K dataset. The Last.fm 360K dataset is structurally similar to those in Cluster 5, i.e. with a large number of interactions as well as much more users than items, and this could have contributed to the stronger performance of UserKNN.
- As for **Cluster 2**, the graph-based RP3beta is always the best performing method. On the *Yahoo! R4* dataset, RP3beta and UserKNN performs similarly (i.e. the difference is not statistically significant).
- When it comes to **Cluster 3**, there are a number of interesting observations. First, across all 3 datasets, RP3beta has the best performance while WMF has the worst performance. Next, both UserKNN and ItemKNN perform significantly worse (with a 3% to 8% relative difference) than RP3beta. At the same time, both UserKNN and ItemKNN perform significantly better (with a 3% to 15% relative difference) than WMF. Note that for all 3 datasets, the difference between UserKNN and ItemKNN is never statistically significant.
- For **Cluster 4**, the relative performance of each algorithm changes drastically across different datasets. In other words, for this particular cluster, we do not have any algorithm which consistently performs better or worse than all the other algorithms.
- Finally, for **Cluster 5**, we can immediately see the results are drastically different from that of Cluster 3. Across all 3 datasets in Cluster 5, UserKNN, and WMF perform incredibly well and hold a 20% relative improvement over ItemKNN and RP3beta. This is in stark contrast with the results of Cluster 3, whereby RP3beta triumphs over remaining algorithms while WMF consistently ranks last.

¹¹ Whenever we state that method A performs better (or worse) than method B, the difference between those two methods is statistically significant with $p\text{-value} < 0.05$ using the paired sample t-test.

Our results highlight a clear and consistent trend: For datasets with relatively similar characteristics, there could be a particular recommendation algorithm which tends to perform significantly better (or worse) than all remaining algorithms. As a matter of fact, it seems entirely plausible that a very different ‘*ordering*’ of recommendation algorithms¹² can be obtained just by selecting datasets from different clusters (e.g. Cluster 3 vs. Cluster 5). Consequently, it could lead to completely different observations and/or conclusions.

5.3.2.1 Relating the Recommendation Performance for each Cluster to its Data Characteristics

In order to have a better understanding of the results, we provide a few examples relating the performance of various recommender algorithms to the data characteristics of a particular dataset cluster.

As mentioned in Section 5.2.2.1, the datasets in **Cluster 3** have very moderate characteristics in terms of the number of users, items, and interactions. At the same time, the relatively low $Gini_{user}$ and $Gini_{item}$ scores imply that the observed interactions are much more equally distributed across both users and items. For these datasets, their corresponding user-item bipartite graphs would have much higher connectivity, and it would be less likely to encounter multiple disconnected vertices (either users or items) when performing the random walks. We hypothesise that it could be the one of the reasons behind the exceptionally strong performance of the *graph-based* algorithm RP3beta for this particular cluster. Additionally, since we have a similar number of users and items in these datasets, both *neighbourhood-based* approaches, i.e. the user-based UserKNN and the item-oriented ItemKNN, have nearly identical performance.

Next, for **Cluster 4**, we have noted that the performance of different algorithms appears to be relatively unstable across the individual datasets. We believe that this could be attributed to the extremely small dataset sizes as the largest dataset in this cluster, i.e. ML-100K, has just $\sim 100K$ interactions as its name implies. As such, it is unlikely for any particular method to learn well enough from the training data, which then leads to unreliable and seemingly random predictions.

Finally, for **Cluster 5**, the *user neighbourhood-based* UserKNN performs incredibly well as it could be easier to find like-minded users among the large number of candidate

¹² The ‘*ordering*’ for **Cluster 3** would be $RP3beta > UserKNN, ItemKNN > WMF$, while the ‘*ordering*’ for **Cluster 5** would be $UserKNN, WMF > ItemKNN, RP3beta$.

users in these datasets with an exceptionally high $Shape_{log}$ (i.e. much more users than items). If we keep the number of interactions (and hence, density) fixed and simply adjust the ratio of users to items, the feasibility of deriving either the user-user similarity matrix or the item-item similarity matrix would naturally be affected [1].

To summarise, as different types of recommendation algorithms, i.e. *neighbourhood-based*, *graph-based*, and *latent factor models*, have distinct inductive bias, the data characteristics would inherently affect them differently. However, we find the effect that each data characteristic has on a recommendation algorithm might not always be immediately apparent. For example, having 10% *more users than items* might not have a significant effect on the recommendation performance. However, if the dataset has *3 times more users than items*, it is likely to provide a user-based approach such as UserKNN with a considerable advantage over its item-oriented counterpart.

5.3.3 Recent Neural Recommendation Approaches

Unsurprisingly, many of the recent papers (as shown in Table 5.1) are proposing new neural recommendation approaches. The authors in [30] have reported that simple, but well-tuned, baselines (e.g. ItemKNN and RP3beta) are able to outperform some of these new recommendation approaches, such as CMN [34] and NeuMF [52]. However, we note these neural recommendation approaches are individually compared against the simpler baselines using **(1) different evaluation procedures**, i.e. datasets, data pre-processing or splitting strategies, evaluation metrics, etc., and **(2) sampled metrics**, i.e. by evaluating the ground-truth item(s) against a randomly sampled subset of candidate items, which have been shown to be inaccurate in [70].

Hence, in this section, we repeat the experiments as described previously in Section 5.3.1, but we consider 4 recently proposed neural recommendation approaches as follows:

- (i) **Mult-VAE**: Mult-VAE [82] is a deep generative model which has been shown to be a strong neural recommendation method [30]. The autoencoder-based architecture for Mult-VAE with 1 hidden layer (of size 600) is $[I \rightarrow 600 \rightarrow 200 \rightarrow 600 \rightarrow I]$, whereby I is the number of items. The hyperparameter β (for KL annealing) is selected from $\{0.1, 0.2, 0.3, 0.5, 1.0\}$, and it is optimised by Adam with a learning rate 0.001 for 200 epochs.

- (ii) **NeuMF**: NeuMF [33, 52] is a generalised matrix factorization approach belonging to the same ‘*family*’ of recommendation algorithms as WMF, i.e. latent factor models. However, NeuMF explores the non-linear relationship between users and items by modelling it using a Multilayer Perceptron (**MLP**). We use the publicly available source code from the authors, and adopt a 3-layer MLP with a tower structure based on their suggestion [52]. The number of predictive factors, for both the Generalised Matrix Factorization (**GMF**) and the MLP components, is selected from $\{8, 16, 32, 64\}$. NeuMF is optimised using Adam [67] with a default learning rate of 0.001, and the number of negative samples for each observed entry is chosen from 2 to 8.
- (iii) **CMN**: CMN [34] can be considered as a hybrid approach combining the latent factor model with a (user) neighbourhood-based approach. Furthermore, CMN utilises a multi-hop neural attention mechanism (i.e. memory network [129, 152]) to learn the importance of neighbouring users with respect to the target user and item. We use the publicly available source code from the authors, and following [34], we set the default number of hops to 2 and number of negative samples to 4. Similarly, we adopt the RMSProp optimiser with a learning rate of 0.001 and 0.9 for both decay and momentum, and the number of latent factors is selected from $\{20, 30, \dots, 90, 100\}$.
- (iv) **LRML**: Latent Relational Metric Learning (**LRML**) [136] is a recently proposed metric learning-based approach. The user-item relational vectors are learned adaptively in Euclidean space using a neural attention mechanism. Following [136], the number of latent factors for the user and item embeddings is selected from $\{20, 50, 100\}$, the number of slices used for the memory matrix is chosen from $\{5, 10, 20, 25, 50, 100\}$, and the margin λ used for the hinge loss is selected from $\{0.1, 0.2, 0.5\}$. Similar to Mult-VAE and NeuMF, the optimisation for LRML uses the Adam [67] optimiser with a default learning rate of 0.001.

The aforementioned neural recommendation algorithms are selected for various reasons: **(1)** They are representative neural recommender approaches with different inductive bias, e.g. *latent factor models*, *generative models*, and *metric learning-based approaches*. **(2)** The source code is publicly available for these neural recommendation algorithms. **(3)** As shown in Figure 5.1, these algorithms have been evaluated on rather different datasets in their respective papers. For example, the evaluation for CMN [34] was performed using $\{Epinions, CiteULike-a, \text{ and } Pinterest\}$, while Mult-VAE [82] uses

Figure 5.5: Recall@10 of 8 recommendation algorithms for the 15 datasets (*Best viewed in colour*)

Figure 5.6: $nDCG@10$ of 8 recommendation algorithms for the 15 datasets (*Best viewed in colour*)

$\{ML-20M, Netflix, \text{ and } Million \text{ Song Dataset}\}$. Finally, in order to determine the optimal hyperparameters for these neural recommendation approaches on each dataset, we adopt a grid search strategy¹³ based on the nDCG@10 of the corresponding validation set.

5.3.3.1 Experimental Results for Neural Recommendation Approaches

Figures 5.5 and 5.6 present the Recall@10 and the nDCG@10 of **8** different recommendation algorithms for the 15 datasets, respectively. As the results (i.e. relative performance of different algorithms for each dataset) are nearly identical, we will focus on nDCG@10 once again. Unless stated otherwise, the observations do hold in terms of Recall@10 as well.

First of all, we are unable to get the results¹⁴ of CMN for **Clusters 1** and **5** due to its time complexity. Clusters 1 and 5 consist of large datasets with a considerable amount of users and items. For instance, the Last.fm 360K dataset in Cluster 1 has $\sim 359K$ users and $\sim 88K$ items, while the ML-20M in Cluster 5 has $\sim 138K$ users and $\sim 18K$ items. For a user u and an item i , CMN considers all users who have interacted with the item i as the user neighbourhood. Therefore, for more popular items, the user neighbourhood could easily be as large as thousands of users. Furthermore, the multi-hop attention mechanism incurs additional time complexity as the attention weights over the user neighbourhood needs to be calculated repetitively (i.e. once per hop). As such, we believe that CMN might not be feasible in practice.

Next, for **Cluster 4**, the performance of different algorithms changes drastically across the individual datasets. As mentioned previously, this could be attributed to the extremely small dataset sizes. Since the results for Cluster 4 tend to be inconclusive, we do not make further reference to it in our following discussion.

As we can see from Figures 5.5 and 5.6, there are several interesting observations, and we summarise them as follows:

¹³ When using the grid search strategy, we are able to run the same model with different hyperparameter settings (within the grid) in a parallel fashion by using multiple devices as and when available. This is somewhat useful for some of these neural recommendation methods which come with an extremely high time complexity.

¹⁴ For all 4 neural recommendation approaches, we note that their evaluation were performed using *sampled metrics*. However, as sampled metrics are inaccurate [70], we use exact metrics in our empirical study. As such, for each user, we have to derive the scores for all candidate items in order to generate the ranked list of top-K items. Our evaluation approach is much more robust, but it requires significantly more time to evaluate each model. For example, on larger datasets, it takes a few days just to perform a single validation / testing step for CMN.

- First of all, with the exception of Mult-VAE, the neural recommendation methods tend to be much less competitive as compared to simpler baseline methods. Across all clusters (and datasets), we can see that Mult-VAE performs relatively well, while CMN tends to be slightly worse than Mult-VAE. On the other hand, LRML lags significantly behind, while NeuMF’s performance happens to be the worst across all clusters (and datasets).
- For Mult-VAE, we find that it performs extremely well for **Cluster 5** which contains large, but relatively dense, datasets such as Flixster, ML-10M, and ML-20M. This is once again in stark contrast with its performance in **Cluster 3**, which contains datasets that are considerably different, i.e. datasets which are much smaller and sparser. Furthermore, the datasets in Cluster 5 have an exceptionally high $Shape_{log}$ value, i.e. they have much more users than items. This could be advantageous to a generative model like Mult-VAE which happens to be inductive over users [82, 78]. Finally, we would like to point out that the Mult-VAE was evaluated against other methods using large and highly skewed (i.e. much more users than items) datasets such as ML-20M and Netflix in the original paper [82]. Coincidentally, based on our analysis of the dataset characteristics, both ML-20M and Netflix belong to Cluster 5. As we can see from the results, the characteristics of such datasets might be favourable for Mult-VAE, and the results reported could have been misleading.
- For both LRML and NeuMF, their performance seem to be less than ideal despite extensive hyperparameter tuning. It could be possible that the (range of) hyperparameters suggested in their respective papers [136, 52] do not work well for other datasets with considerably different characteristics. For instance, the evaluation for NeuMF was performed using relatively small datasets, such as ML-1M and Pinterest. In addition, for those two datasets, only users with at least 20 interactions were retained, and the pre-processed datasets would have been a lot denser than most datasets considered in our empirical study. Furthermore, we note that several recent work (e.g. [30, 82, 78]) have encountered difficulty when trying to obtain competitive performance for NeuMF as well.
- Finally, unlike the simpler recommendation algorithms introduced in Sections 5.3.1 and 5.3.2, the influence of these dataset characteristics on neural recommendation methods might be less straightforward. For instance, there are a lot more factors that could significantly affect the overall performance of a neural recommendation algorithm. Some examples are the choice of hyperparameters, the optimisation technique used, or even the types of non-linearities used within the algorithm [41].

5.3.4 Key Findings

In this section, we have evaluated a total of 8 different recommendation algorithms using 15 distinct datasets. We provide an overview of our key findings as follows:

(i) First of all, for most recommendation algorithms, we note that their performance tend to be quite consistent across datasets with similar characteristics. In our experimental study, we have shown that the (relative) performance of a recommendation algorithm does not vary much for datasets within the same cluster.

(ii) Next, we can observe that recommendation algorithms do perform differently on datasets with different characteristics. However, the impact of each data characteristic on a recommendation algorithm might not always be that obvious. In fact, we find that some datasets might have relatively ‘*neutral*’ characteristics, e.g. they are neither too sparse nor too dense, or that the number of users and items are nearly identical. For such scenarios, the difference in the performance of one algorithm over another might be less significant.

(iii) Last but not least, we have clearly shown that the choice of datasets utilised in an empirical evaluation has a substantial influence on the observations and/or conclusions obtained. In general, coupled with our observation that existing work tend to select the datasets used for evaluation in an arbitrary manner, we find that their reported results might not be as comprehensive as it should be. Hence, we believe that this is an important facet of the evaluation process that requires much more deliberation in order to improve its robustness.

5.3.5 Suggestions

Our findings have shown that the approach used for selecting datasets could in fact influence the observations and/or conclusions obtained from the experimental evaluation. As researchers may face different constraints (e.g. limited to datasets from a specific domain) when performing their experiments, we believe that restricting everyone to a common set of benchmark datasets may not be the ideal solution.

Instead, given the large variety of publicly available datasets to choose from, we strongly suggest using datasets with *considerably different characteristics* as part of the evaluation process. For example, as shown in Section 5.1.2, frequent dataset combinations

such as $\{ML-10M, Netflix\}$ and $\{ML-20M, Netflix\}$ are all made up of large and dense datasets from Cluster 5. Therefore, the results in existing work could be biased and the same model might not work well for other scenarios.

Furthermore, we note that the characteristics described in Section 5.2.1 can be easily derived for any dataset with minimal effort. As such, the comparison of multiple datasets can be done in a convenient and efficient manner. In addition, we recommend that empirical evaluation performed using extremely small datasets such as *ML-100K* should be avoided if possible. As shown in Figures 5.3, 5.4, 5.5, and 5.6, the results obtained using datasets in Cluster 4 tend to be inconclusive in general.

Finally, while it is only natural for a recommendation algorithm to perform much better on certain datasets, we are usually unaware of the specific type(s) of datasets which a particular algorithm would perform well on. Therefore, being able to relate the performance of a proposed algorithm to measurable aspects of different datasets could be highly beneficial to the recommendation systems community. For instance, it might even allow us to efficiently narrow down the set of suitable algorithms for a completely new dataset based on its characteristics. Hence, we encourage paying more attention to these data characteristics as they could be used as a beacon to guide future research efforts.

5.4 Summary

In this chapter, we conducted a retrospective survey on datasets used for implicit feedback based top-K recommendation. We demonstrated how various datasets have been utilised rather arbitrarily in recent papers, and illustrated the similarities and differences between many widely known datasets using a set of well-defined data characteristics. Additionally, we investigated whether the datasets used, with respect to their data characteristics, could in fact lead to different observations and/or conclusions drawn from an empirical study. Our findings have shown that data characteristics can have a significant impact on the obtained results, and we strongly suggest using datasets with considerably different characteristics to improve the robustness of the evaluation process.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

In this dissertation, we propose novel hybrid models to overcome a long-standing challenge for personalised recommendation, i.e. the *cold-start problem*, by leveraging different types of content information in conjunction with recent advances in deep learning. Furthermore, we identify and discuss about challenges, as well as experimental issues, that persist in personalised recommendation.

In Chapter 3, we look at how recommender systems can make use of textual reviews to combat the cold-start problem in the “*wide sense*”. To this end, we propose a novel aspect-based neural recommender system, ANR, which performs aspect-based representation learning for users and items, by designing an attention mechanism to focus on the relevant parts of these reviews while learning the representation of aspects on the task. Furthermore, we estimate aspect-level user and item importance in a joint manner using the idea of co-attention, which allows us to model the finer-grained interactions between users and items. To the best of our knowledge, we are the first to propose an end-to-end neural aspect-based recommender system which concurrently addresses the above-mentioned requirements. Extensive experiments have been conducted on 25 benchmark datasets from *Amazon* and *Yelp* to evaluate our proposed model against several state-of-the-art baselines such as DeepCoNN [168], D-Attn [119], and ALFM [25]. Furthermore, we investigate how the different components in our proposed model contribute to its effectiveness. In particular, we include a qualitative analysis of the aspects which are learned automatically by our model without any external supervision.

However, textual reviews are ultimately a form of explicit feedback, which are not available for completely new users and items. Therefore, in Chapter 4, we look at how recommendation systems can utilise other forms of user and item contents to deal with the cold-start problem in the “*narrow sense*”. More specifically, we consider the problem of generating recommendations satisfactorily in *both* warm- and cold-start scenarios via a *single model*. To this end, we present an innovative hybrid model using our proposed neighbourhood-based **Collective Collaborative Filtering (CCF)** component, which aims to maximise the performance of a recommender system in the presence of a dynamic and constantly evolving environment. Evaluation procedures adopted in existing work for cold-start recommendation are examined critically, and a key finding is that the content information in commonly used datasets are often insufficient for differentiating one user (or item) from another. Furthermore, we adopt an evaluation framework which explicitly shows the model performance for both warm-start recommendation and the various cold-start scenarios, i.e. *user cold-start*, *item cold-start*, and *both user & item cold-start*. Experimental results on four datasets, including *ML-100K* & *ML-1M*, show that our proposed model consistently outperforms baseline methods for various cold-start scenarios, and achieves the best overall performance in all four datasets.

Finally, while it is paramount to constantly develop new algorithms to advance the state-of-the-art, we believe that the evaluation procedure needs to be well-defined and robust so as to guarantee the validity of the obtained results. In Chapter 5, we conduct a retrospective survey to provide an in-depth look at a fundamental but often neglected aspect of the evaluation procedure, i.e. the datasets themselves. To do so, we adopt a systematic and comprehensive approach to understand the datasets used for implicit feedback based top-K recommendation. We start by examining recent papers from several top-tier conferences to find out how different datasets have been utilised thus far. Next, we study the characteristics of these datasets to understand their similarities and differences. Finally, we conduct an empirical study to determine whether the choice of datasets used for evaluation can influence the observations and/or conclusions obtained. Our findings suggest that greater attention needs to be paid to the selection process of datasets used for evaluating recommender systems in order to improve the robustness of the obtained results.

6.2 Future Work

6.2.1 Explainable Recommendation

For each user-item interaction, our aspect-based recommender system ANR (Chapter 3) is capable of estimating the importance of each aspect. Additionally, for each aspect, we can identify the relevant document words which contribute to its representation.

This information can be easily post-processed to provide a simple and intuitive explanation for our recommendations. For example, for any potential user-item interaction, we can identify the 3 most important aspects. Furthermore, for each aspect, we display the most relevant segments from the reviews to support our recommendation. By showing how the system works (i.e. at an aspect-level), and helping the users make an informed choice through the identification of relevant review contents, we can improve the transparency and effectiveness of the proposed recommender system. In fact, the necessity and intrinsic value of explainable recommender systems have been presented and explored in various recent work [18, 148, 149, 21].

6.2.2 Domain-independent Aspect-based Recommendation

Currently, similar to most existing aspect-based recommender systems, ANR (Chapter 3) can be considered as a domain-dependent framework. In other words, it is necessary for us to train a separate model for each category (or domain).

We believe that an interesting future direction would be to extend ANR into a domain-independent framework, which will be able to handle multiple categories simultaneously, by incorporating either transfer learning [103] or multi-task learning [17]. For different domains, there are certain aspects that are likely to be shared, e.g. ‘price’. On the other hand, aspects such as ‘quality’ can refer to rather different concepts for separate domains. This would be one of the key challenges to be addressed when extending ANR to two or more domains.

6.2.3 Datasets and their Characteristics

In Chapter 5, we focus on datasets used for implicit feedback-based general recommendation. However, there are other recommendation tasks, such as content-aware or session-based recommendation. As these tasks tend to utilise datasets which are much more diverse and informative, it would be an interesting research direction to characterise these datasets for analysis and comparison as well.

Furthermore, as we have seen in Section 5.3.3, neural recommendation methods tend to perform much less competitively in certain scenarios. Generally, neural network-based methods might require careful, and often extensive, hyperparameter tuning in order to perform well for a given dataset. For example, the optimal hyperparameters (and even model structure) might differ drastically from dataset to dataset. An interesting research direction would be to identify the relationship(s) between the hyperparameter search space and the characteristics of a dataset. For instance, a key hyperparameter used in many neural recommendation algorithms would be the user and item embedding size. If we are able to associate key hyperparameters with the data characteristics, we might be able to save a lot of time and effort when tuning a given recommendation algorithm for a specific dataset.

6.2.4 Summary of Future Work

By solving the above-mentioned problems in our future work, recommendation systems will be more transparent and trustworthy as users can better understand the rationale behind certain recommendations. Furthermore, by extending ANR into a domain-independent framework, less popular domains with fewer historical user-item interactions can greatly benefit from more popular domains with richer information. Last but not least, a better understanding of datasets based on their characteristics will enable practitioners to spend less time selecting and tuning the best recommendation algorithm for a given problem.

List of Publications

- Xiucheng Li, **Jin Yao Chin**, Yile Chen, Gao Cong. Sinkhorn Collaborative Filtering. In Proceedings of the Web Conference 2021 (**WWW 2021**), Pages 582-592, Ljubljana, Slovenia, April 19–23, 2021.
- Kaiqi Zhao, Gao Cong, **Jin Yao Chin**, Rong Wen. Exploring Market Competition over Topics in Spatio-temporal Document Collections. In The VLDB Journal 28(1): 123-145 (2019).
- **Jin Yao Chin**, Kaiqi Zhao, Shafiq Joty, and Gao Cong. ANR: Aspect-based Neural Recommender. In The 27th ACM International Conference on Information and Knowledge Management (**CIKM 2018**), Pages 147-156, Torino, Italy, October 22–26, 2018.
- **Jin Yao Chin**, Gao Cong, Kaiqi Zhao, Miao Xie. CCF: Collective Collaborative Filtering for Seamless Warm- and Cold-start Recommendation. *Under Review*.
- **Jin Yao Chin**, Yile Chen, Gao Cong. The Datasets Dilemma: How Much Do We Really Know About Recommendation Datasets? *Under Review*.

References

- [1] Gediminas Adomavicius and Jingjing Zhang. Impact of data characteristics on recommender systems performance. *ACM Transactions on Management Information Systems (TMIS)*, 3(1), April 2012.
- [2] Deepak Agarwal and Bee-Chung Chen. Regression-based latent factor models. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, page 19–28, 2009.
- [3] Rakesh Agrawal, Heikki Mannila, Ramakrishnan Srikant, Hannu Toivonen, and A. Inkeri Verkamo. *Fast Discovery of Association Rules*, page 307–328. American Association for Artificial Intelligence, USA, 1996.
- [4] Amjad Almahairi, Kyle Kastner, Kyunghyun Cho, and Aaron Courville. Learning distributed representations from reviews for collaborative filtering. In *Proceedings of the 9th ACM Conference on Recommender Systems*, RecSys '15, pages 147–154, 2015.
- [5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.
- [6] Trapit Bansal, David Belanger, and Andrew McCallum. Ask the gru: Multi-task learning for deep text recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, pages 107–114, 2016.
- [7] Yang Bao, Hui Fang, and Jie Zhang. TopicMF: Simultaneously exploiting ratings and reviews for recommendation. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, AAAI '14, pages 2–8, 2014.

- [8] I. Barjasteh, R. Forsati, D. Ross, A. Esfahanian, and H. Radha. Cold-start recommendation with provable guarantees: A decoupled approach. *IEEE Transactions on Knowledge and Data Engineering*, 28(6):1462–1474, June 2016.
- [9] Iman Barjasteh, Rana Forsati, Farzan Masrour, Abdol-Hossein Esfahanian, and Hayder Radha. Cold-start item and user recommendation with decoupled completion and transduction. In *Proceedings of the 9th ACM Conference on Recommender Systems*, RecSys ’15, page 91–98, 2015.
- [10] Oren Barkan, Noam Koenigstein, Eylon Yogev, and Ori Katz. Cb2cf: A neural multiview content-to-collaborative filtering model for completely cold item recommendations. In *Proceedings of the 13th ACM Conference on Recommender Systems*, RecSys ’19, page 228–236, 2019.
- [11] Konstantin Bauman, Bing Liu, and Alexander Tuzhilin. Aspect based recommendations: Recommending items with the most valuable aspects based on user reviews. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD ’17, pages 717–725, 2017.
- [12] Daniel Billsus and Michael J. Pazzani. Learning collaborative information filters. In *Proceedings of the 15th International Conference on Machine Learning*, ICML’15, page 46–54, 1998.
- [13] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research (JMLR)*, 3:993–1022, March 2003.
- [14] Rocío Cañamares and Pablo Castells. A probabilistic reformulation of memory-based collaborative filtering: Implications on popularity biases. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’17, page 215–224, 2017.
- [15] Da Cao, Xiangnan He, Lianhai Miao, Yahui An, Chao Yang, and Richang Hong. Attentive group recommendation. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’18, page 645–654, 2018.

REFERENCES

- [16] Da Cao, Xiangnan He, Lianhai Miao, Yahui An, Chao Yang, and Richang Hong. Attentive group recommendation. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '18, pages 645–654, 2018.
- [17] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, July 1997.
- [18] Rose Catherine and William Cohen. TransNets: Learning to transform for recommendation. In *Proceedings of the 11th ACM Conference on Recommender Systems*, RecSys '17, pages 288–296, 2017.
- [19] Dong-Kyu Chae, Jihoo Kim, Duen Horng Chau, and Sang-Wook Kim. Ar-cf: Augmenting virtual users and items in collaborative filtering for addressing cold-start problems. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 1251–1260, 2020.
- [20] Chaochao Chen, Ziqi Liu, Peilin Zhao, Longfei Li, Jun Zhou, and Xiaolong Li. Distributed collaborative hashing and its applications in ant financial. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '18, page 100–109, 2018.
- [21] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. Neural attentional rating regression with review-level explanations. In *Proceedings of the Web Conference 2018*, WWW '18, pages 1583–1592, 2018.
- [22] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. Attentive collaborative filtering: Multimedia recommendation with item- and component-level attention. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, pages 335–344, 2017.
- [23] Xu Chen, Zheng Qin, Yongfeng Zhang, and Tao Xu. Learning to rank features for recommendation over multiple categories. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, pages 305–314, 2016.

- [24] Yihong Chen, Bei Chen, Xiangnan He, Chen Gao, Yong Li, Jian-Guang Lou, and Yue Wang. λ opt: Learn to regularize recommender models in finer levels. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '19, page 978–986, 2019.
- [25] Zhiyong Cheng, Ying Ding, Lei Zhu, and Mohan Kankanhalli. Aspect-aware latent factor model: Rating prediction with ratings and reviews. In *Proceedings of the Web Conference 2018*, WWW '18, pages 639–648, 2018.
- [26] Jin Yao Chin, Kaiqi Zhao, Shafiq Joty, and Gao Cong. ANR: Aspect-based neural recommender. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, page 147–156, 2018.
- [27] Evangelia Christakopoulou and George Karypis. Local item-item models for top-n recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, page 67–74, 2016.
- [28] Evangelia Christakopoulou and George Karypis. Local latent space models for top-n recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '18, page 1235–1243, 2018.
- [29] Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. Very deep convolutional networks for text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '17, pages 1107–1116, 2017.
- [30] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. Are we really making much progress? a worrying analysis of recent neural recommendation approaches. In *Proceedings of the 13th ACM Conference on Recommender Systems*, RecSys '19, page 101–109, 2019.
- [31] Yashar Deldjoo, Tommaso Di Noia, Eugenio Di Sciascio, and Felice Antonio Merra. How dataset characteristics affect the robustness of collaborative recommendation models. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 951–960, 2020.

- [32] Qiming Diao, Minghui Qiu, Chao-Yuan Wu, Alexander J. Smola, Jing Jiang, and Chong Wang. Jointly modeling aspects, ratings and sentiments for movie recommendation (JMARS). In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 193–202, 2014.
- [33] Gintare Karolina Dziugaite and Daniel M. Roy. Neural network matrix factorization. *CoRR*, abs/1511.06443, 2015.
- [34] Travis Ebesu, Bin Shen, and Yi Fang. Collaborative memory network for recommendation systems. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '18, page 515–524, 2018.
- [35] Ehtsham Elahi, Wei Wang, Dave Ray, Aish Fenton, and Tony Jebara. Variational low rank multinomials for collaborative filtering with side-information. In *Proceedings of the 13th ACM Conference on Recommender Systems*, RecSys '19, page 340–347, 2019.
- [36] Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research (JMLR)*, 11:625–660, March 2010.
- [37] Shanshan Feng, Lucas Vinh Tran, Gao Cong, Lisi Chen, Jing Li, and Fan Li. HME: A hyperbolic metric embedding approach for next-poi recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 1429–1438, 2020.
- [38] Z. Gantner, L. Drumond, C. Freudenthaler, S. Rendle, and L. Schmidt-Thieme. Learning attribute-to-feature mappings for cold-start recommendations. In *Proceedings of the 10th IEEE International Conference on Data Mining*, ICDM '10, pages 176–185, 2010.
- [39] Chen Gao, Chao Huang, Dongsheng Lin, Depeng Jin, and Yong Li. DPLCF: Differentially private local collaborative filtering. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 961–970, 2020.

REFERENCES

- [40] Corrado Gini. Measurement of inequality of incomes. *The Economic Journal*, 31(121):124–126, 1921.
- [41] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [42] Ihsan Gunes, Cihan Kaleli, Alper Bilge, and Huseyin Polat. Shilling attacks against recommender systems: A comprehensive survey. *Artificial Intelligence Review*, 42, 01 2013.
- [43] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. DeepFM: A factorization-machine based neural network for ctr prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI’17*, page 1725–1731, 2017.
- [44] Lei Guo, Hongzhi Yin, Qinyong Wang, Bin Cui, Zi Huang, and Lizhen Cui. Group recommendation with latent voting mechanism. In *Proceedings of the 2020 IEEE 36th International Conference on Data Engineering, ICDE ’20*, pages 121–132, 2020.
- [45] Qing Guo, Zhu Sun, Jie Zhang, Qi Chen, and Yin-Leng Theng. Aspect-aware point-of-interest recommendation with geo-social influence. In *Adjunct Publication of the 25th Conference on User Modeling, Adaptation and Personalization, UMAP ’17*, page 17–22, 2017.
- [46] Ruining He, Wang-Cheng Kang, and Julian McAuley. Translation-based recommendation. In *Proceedings of the 11th ACM Conference on Recommender Systems, RecSys ’17*, page 161–169, 2017.
- [47] Ruining He and Julian McAuley. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the Web Conference 2016, WWW ’16*, pages 507–517, 2016.
- [48] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. TriRank: Review-aware explainable recommendation by modeling aspects. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, CIKM ’15*, pages 1661–1670, 2015.

- [49] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 639–648, 2020.
- [50] Xiangnan He, Zhankui He, Xiaoyu Du, and Tat-Seng Chua. Adversarial personalized ranking for recommendation. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '18, page 355–364, 2018.
- [51] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the Web Conference 2017*, WWW '17, page 173–182, 2017.
- [52] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the Web Conference 2017*, WWW '17, page 173–182, 2017.
- [53] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, page 549–558, 2016.
- [54] Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers, and John Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, page 230–237, 1999.
- [55] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S. Yu. Leveraging meta-path based context for top- n recommendation with a neural co-attention model. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '18, pages 1531–1540, New York, NY, USA, 2018. ACM.
- [56] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 168–177, 2004.

- [57] Peng Hu, Rong Du, Yao Hu, and Nan Li. Hybrid item-item recommendation via semi-parametric embedding. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence, IJCAI'19*, pages 2521–2527, 7 2019.
- [58] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining, ICDM '08*, page 263–272, 2008.
- [59] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning, ICML '15*, page 448–456, 2015.
- [60] Shuyi Ji, Yifan Feng, Rongrong Ji, Xibin Zhao, Wanwan Tang, and Yue Gao. Dual channel hypergraph collaborative filtering. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '20*, page 2020–2029, 2020.
- [61] Jyun-Yu Jiang, Patrick H. Chen, Cho-Jui Hsieh, and Wei Wang. Clustering and constructing user coresets to accelerate large-scale top-k recommender systems. In *Proceedings of the Web Conference 2020, WWW '20*, page 2177–2187, 2020.
- [62] Manas R. Joglekar, Cong Li, Mei Chen, Taibai Xu, Xiaoming Wang, Jay K. Adams, Pranav Khaitan, Jiahui Liu, and Quoc V. Le. Neural input search for large scale recommendation models. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '20*, page 2387–2397, 2020.
- [63] Rie Johnson and Tong Zhang. Deep pyramid convolutional neural networks for text categorization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL '17*, pages 562–570, 2017.
- [64] Farhan Khawar, Leonard Poon, and Nevin L. Zhang. Learning the structure of auto-encoding recommenders. In *Proceedings of the Web Conference 2020, WWW '20*, page 519–529, 2020.
- [65] Donghyun Kim, Chanyoung Park, Jinoh Oh, Sungyoung Lee, and Hwanjo Yu. Convolutional matrix factorization for document context-aware recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems, RecSys '16*, pages 233–240, 2016.

REFERENCES

- [66] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP '14*, pages 1746–1751, 2014.
- [67] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.
- [68] Yehuda Koren. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '08*, page 426–434, 2008.
- [69] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, August 2009.
- [70] Walid Krichene and Steffen Rendle. On sampled metrics for item recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '20*, page 1748–1757, 2020.
- [71] Shyong K. Lam and John Riedl. Shilling recommender systems for fun and profit. In *Proceedings of the 13th International Conference on World Wide Web, WWW '04*, page 393–402, 2004.
- [72] Daniel Lee and H. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 11 1999.
- [73] Hoyeop Lee, Jinbae Im, Seongwon Jang, Hyunsouk Cho, and Sehee Chung. MeLU: Meta-learned user preference estimator for cold-start recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '19*, page 1073–1082, 2019.
- [74] Christiane Lemke, Marcin Budka, and Bogdan Gabrys. Metalearning: A survey of trends and technologies. *Artificial Intelligence Review*, 44(1):117–130, June 2015.
- [75] Dongsheng Li, Chao Chen, Qin Lv, Hansu Gu, Tun Lu, Li Shang, Ning Gu, and Stephen M. Chu. AdaError: An adaptive learning rate method for matrix approximation-based collaborative filtering. In *Proceedings of the Web Conference 2018, WWW '18*, page 741–751, 2018.

- [76] Jingjing Li, Mengmeng Jing, Ke Lu, Lei Zhu, Yang Yang, and Zi Huang. From zero-shot learning to cold-start recommendation. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, AAAI '19, pages 4189–4196, 2019.
- [77] Sheng Li, Jaya Kawale, and Yun Fu. Deep collaborative filtering via marginalized denoising auto-encoder. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM '15, pages 811–820, 2015.
- [78] Xiucheng Li, Jin Yao Chin, Yile Chen, and Gao Cong. Sinkhorn collaborative filtering. In *Proceedings of the Web Conference 2021*, WWW '21, page 582–592, 2021.
- [79] Defu Lian, Haoyu Wang, Zheng Liu, Jianxun Lian, Enhong Chen, and Xing Xie. Lightrec: A memory and search-efficient recommender system. In *Proceedings of the Web Conference 2020*, WWW '20, page 695–705, 2020.
- [80] Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. xDeepFM: Combining explicit and implicit feature interactions for recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, page 1754–1763, 2018.
- [81] Dawen Liang, Laurent Charlin, James McInerney, and David M. Blei. Modeling user exposure in recommendation. In *Proceedings of the Web Conference 2016*, WWW '16, page 951–961, 2016.
- [82] Dawen Liang, Rahul G. Krishnan, Matthew D. Hoffman, and Tony Jebara. Variational autoencoders for collaborative filtering. In *Proceedings of the Web Conference 2018*, WWW '18, page 689–698, 2018.
- [83] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.
- [84] Guang Ling, Michael R. Lyu, and Irwin King. Ratings meet reviews, a combined approach to recommend. In *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys '14, pages 105–112, 2014.

REFERENCES

- [85] Chenghao Liu, Tao Lu, Xin Wang, Zhiyong Cheng, Jianling Sun, and Steven C.H. Hoi. Compositional coding for collaborative filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '19, page 145–154, 2019.
- [86] Feng Liu, Huifeng Guo, Xutao Li, Ruiming Tang, Yunming Ye, and Xiuqiang He. End-to-end deep reinforcement learning based recommendation with supervised embedding. In *Proceedings of the 13th ACM International Conference on Web Search and Data Mining*, WSDM '20, page 384–392, 2020.
- [87] Huafeng Liu, Liping Jing, Jingxuan Wen, Zhicheng Wu, Xiaoyi Sun, Jiaqi Wang, Lin Xiao, and Jian Yu. Deep global and local generative model for recommendation. In *Proceedings of the Web Conference 2020*, WWW '20, page 551–561, 2020.
- [88] Huafeng Liu, Jingxuan Wen, Liping Jing, and Jian Yu. Deep generative ranking for personalized recommendation. In *Proceedings of the 13th ACM Conference on Recommender Systems*, RecSys '19, page 34–42, 2019.
- [89] Nathan N. Liu, Xiangrui Meng, Chao Liu, and Qiang Yang. Wisdom of the better few: Cold start recommendation via representative based rating elicitation. In *Proceedings of the 5th ACM Conference on Recommender Systems*, RecSys '11, page 37–44, 2011.
- [90] Yiding Liu, Tuan-Anh Nguyen Pham, Gao Cong, and Quan Yuan. An experimental evaluation of point-of-interest recommendation in location-based social networks. *Proceedings of the VLDB Endowment*, 10(10):1010–1021, June 2017.
- [91] Ramon Lopes, Renato Assunção, and Rodrygo L.T. Santos. Efficient bayesian methods for graph-based recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, page 333–340, 2016.
- [92] Pasquale Lops, Marco de Gemmis, and Giovanni Semeraro. *Content-based Recommender Systems: State of the Art and Trends*, pages 73–105. 01 2011.
- [93] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. Hierarchical question-image co-attention for visual question answering. In *Proceedings of 29th International Conference on Neural Information Processing Systems*, NIPS '16, pages 289–297, 2016.

- [94] Chen Ma, Liheng Ma, Yingxue Zhang, Ruiming Tang, Xue Liu, and Mark Coates. Probabilistic metric learning with adaptive margin for top-k recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '20, page 1036–1044, 2020.
- [95] Julian McAuley and Jure Leskovec. Hidden factors and hidden topics: Understanding rating dimensions with review text. In *Proceedings of the 7th ACM Conference on Recommender Systems*, RecSys '13, pages 165–172, 2013.
- [96] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, pages 43–52, 2015.
- [97] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of 26th International Conference on Neural Information Processing Systems*, NIPS '13, pages 3111–3119, 2013.
- [98] Khalil Muhammad, Qinqin Wang, Diarmuid O'Reilly-Morgan, Elias Tragou, Barry Smyth, Neil Hurley, James Geraci, and Aonghus Lawlor. FedFast: Going beyond average for faster training of federated recommender systems. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '20, page 1234–1242, 2020.
- [99] Athanasios N. Nikolakopoulos, Dimitris Berberidis, George Karypis, and Georgios B. Giannakis. Personalized diffusions for top-n recommendation. In *Proceedings of the 13th ACM Conference on Recommender Systems*, RecSys '19, page 260–268, 2019.
- [100] Athanasios N. Nikolakopoulos and George Karypis. Recwalk: Nearly uncoupled random walks for top-n recommendation. In *Proceedings of the 12th ACM International Conference on Web Search and Data Mining*, WSDM '19, page 150–158, 2019.
- [101] Xichuan Niu, Bofang Li, Chenliang Li, Rong Xiao, Haochuan Sun, Hongbo Deng, and Zhenzhong Chen. A dual heterogeneous graph attention network to improve

REFERENCES

- long-tail performance for shop search in e-commerce. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '20, page 3405–3415, 2020.
- [102] Rasmaq Otunba, Raimi A. Rufai, and Jessica Lin. Mpr: Multi-objective pairwise ranking. In *Proceedings of the 11th ACM Conference on Recommender Systems*, RecSys '17, page 170–178, 2017.
- [103] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transaction on Knowledge and Data Engineering*, 22(10):1345–1359, October 2010.
- [104] Seung-Taek Park and Wei Chu. Pairwise preference regression for cold-start recommendation. In *Proceedings of the 3rd ACM Conference on Recommender Systems*, RecSys '09, page 21–28, 2009.
- [105] Yoon-Joo Park and Alexander Tuzhilin. The long tail of recommender systems and how to leverage it. In *Proceedings of the 2nd ACM Conference on Recommender Systems*, RecSys '08, page 11–18, 2008.
- [106] Bibek Paudel, Fabian Christoffel, Chris Newell, and Abraham Bernstein. Updatable, accurate, diverse, and scalable recommendations for interactive applications. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 7(1), December 2016.
- [107] Michael J. Pazzani and Daniel Billsus. *Content-Based Recommendation Systems*, page 325–341. 2007.
- [108] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, EMNLP '14, pages 1532–1543, 2014.
- [109] Shameem A. Puthiya Parambath, Nicolas Usunier, and Yves Grandvalet. A coverage-based approach to recommendation diversity on similarity graph. In *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, page 15–22, 2016.
- [110] Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. Opinion word expansion and target extraction through double propagation. *COLING*, 37(1):9–27, March 2011.

REFERENCES

- [111] Steffen Rendle. Factorization machines. In *Proceedings of the 10th IEEE International Conference on Data Mining, ICDM '10*, pages 995–1000, 2010.
- [112] Alexander M. Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP '15*, pages 379–389, 2015.
- [113] Ruslan Salakhutdinov and Andriy Mnih. Probabilistic matrix factorization. In *Proceedings of the 20th International Conference on Neural Information Processing Systems, NIPS '07*, page 1257–1264, 2007.
- [114] Aravind Sankar, Yanhong Wu, Yuhang Wu, Wei Zhang, Hao Yang, and Hari Sundaram. GroupIM: A mutual information maximization framework for neural group recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '20*, page 1279–1288, 2020.
- [115] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web, WWW '01*, page 285–295, 2001.
- [116] Martin Saveski and Amin Mantrach. Item cold-start recommendations: Learning local collective embeddings. In *Proceedings of the 8th ACM Conference on Recommender Systems, RecSys '14*, page 89–96, 2014.
- [117] Andrew I. Schein, Alexandrin Popescul, Lyle H. Ungar, and David M. Pennock. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '02*, page 253–260, 2002.
- [118] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, Lexing Xie, and Darius Braziunas. Low-rank linear cold-start recommendation from social data. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence, AAAI '17*, page 1502–1508, 2017.

- [119] Sungyong Seo, Jing Huang, Hao Yang, and Yan Liu. Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In *Proceedings of the 11th ACM Conference on Recommender Systems, RecSys '17*, pages 297–305, 2017.
- [120] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarczyk, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *Proceedings of the 5th International Conference on Learning Representations, ICLR '17*, 2017.
- [121] Ilya Shenbin, Anton Alekseev, Elena Tutubalina, Valentin Malykh, and Sergey I. Nikolenko. RecVAE: A new variational autoencoder for top-n recommendations with implicit feedback. In *Proceedings of the 13th ACM International Conference on Web Search and Data Mining, WSDM '20*, page 528–536, 2020.
- [122] Chuan Shi, Binbin Hu, Wayne Xin Zhao, and Philip S. Yu. Heterogeneous information network embedding for recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 31(2):357–370, 2019.
- [123] Shaoyun Shi, Weizhi Ma, Min Zhang, Yongfeng Zhang, Xinxing Yu, Houzhi Shan, Yiqun Liu, and Shaoping Ma. Beyond user embedding matrix: Learning to hash for modeling large-scale users in recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '20*, page 319–328, 2020.
- [124] Shaoyun Shi, Min Zhang, Yiqun Liu, and Shaoping Ma. Attention-based adaptive model to unify warm and cold starts recommendation. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM '18*, page 127–136, 2018.
- [125] Shaoyun Shi, Min Zhang, Xinxing Yu, Yongfeng Zhang, Bin Hao, Yiqun Liu, and Shaoping Ma. Adaptive feature sampling for recommendation with missing content feature values. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM '19*, page 1451–1460, 2019.
- [126] Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. AutoInt: Automatic feature interaction learning via self-attentive neural networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM '19*, page 1161–1170, 2019.

REFERENCES

- [127] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research (JMLR)*, 15:1929–1958, 2014.
- [128] Harald Steck, Maria Dimakopoulou, Nickolai Riabov, and Tony Jebara. Admm slim: Sparse recommendations for many users. In *Proceedings of the 13th ACM International Conference on Web Search and Data Mining, WSDM '20*, page 555–563, 2020.
- [129] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In *Proceedings of 26th International Conference on Neural Information Processing Systems, NIPS '15*, page 2440–2448, 2015.
- [130] Jianing Sun, Wei Guo, Dengcheng Zhang, Yingxue Zhang, Florence Regol, Yaochen Hu, Huifeng Guo, Ruiming Tang, Han Yuan, Xiuqiang He, and Mark Coates. A framework for recommending accurate and diverse items using bayesian graph convolutional neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '20*, page 2030–2039, 2020.
- [131] Jianing Sun, Yingxue Zhang, Wei Guo, Huifeng Guo, Ruiming Tang, Xiuqiang He, Chen Ma, and Mark Coates. Neighbor interaction aware graph convolution networks for recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '20*, page 1289–1298, 2020.
- [132] Zhu Sun, Di Yu, Hui Fang, Jie Yang, Xinghua Qu, Jie Zhang, and Cong Geng. Are we evaluating rigorously? benchmarking recommendation for reproducible evaluation and fair comparison. In *Proceedings of the 14th ACM Conference on Recommender Systems, RecSys '20*, page 23–32, 2020.
- [133] Seung taek Park, David Pennock, Omid Madani, Nathan Good, and Dennis De-coste. Naïve filterbots for robust cold-start recommendations. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '06*, pages 699–705, 2006.

REFERENCES

- [134] Yunzhi Tan, Min Zhang, Yiqun Liu, and Shaoping Ma. Rating-boosted latent topics: Understanding users and items with ratings and reviews. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence, IJCAI'16*, pages 2640–2646, 2016.
- [135] Jiayi Tang and Ke Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining, WSDM '18*, page 565–573, 2018.
- [136] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. Latent relational metric learning via memory-based attention for collaborative ranking. In *Proceedings of the Web Conference 2018, WWW '18*, page 729–739, 2018.
- [137] Yi Tay, Anh Tuan Luu, and Siu Cheung Hui. Multi-pointer co-attention networks for recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '18*, pages 2309–2318, 2018.
- [138] Thanh Tran, Xinyue Liu, Kyumin Lee, and Xiangnan Kong. Signed distance-based deep memory recommender. In *Proceedings of the Web Conference 2019, WWW '19*, page 1841–1852, 2019.
- [139] Aaron van den Oord, Sander Dieleman, and Benjamin Schrauwen. Deep content-based music recommendation. In *Proceedings of 26th International Conference on Neural Information Processing Systems, NIPS '13*, pages 2643–2651. 2013.
- [140] Manasi Vartak, Arvind Thiagarajan, Conrado Miranda, Jeshua Bratman, and Hugo Larochelle. A meta-learning perspective on cold-start recommendations for items. In *Proceedings of 30th International Conference on Neural Information Processing Systems, NIPS '17*, page 6907–6917, 2017.
- [141] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of 30th International Conference on Neural Information Processing Systems, NIPS '17*, pages 5998–6008. 2017.
- [142] Ricardo Vilalta and Youssef Drissi. A perspective view and survey of meta-learning. *Artificial Intelligence Review*, 18(2):77–95, October 2002.

REFERENCES

- [143] Lucas Vinh Tran, Tuan-Anh Nguyen Pham, Yi Tay, Yiding Liu, Gao Cong, and Xiaoli Li. Interact and decide: Medley of sub-attention networks for effective group recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '19, page 255–264, 2019.
- [144] Lucas Vinh Tran, Yi Tay, Shuai Zhang, Gao Cong, and Xiaoli Li. HyperML: A boosting metric learning approach in hyperbolic space for recommender systems. In *Proceedings of the 13th ACM International Conference on Web Search and Data Mining*, WSDM '20, page 609–617, 2020.
- [145] Maksims Volkovs, Guangwei Yu, and Tomi Poutanen. DropoutNet: Addressing cold start in recommender systems. In *Proceedings of 30th International Conference on Neural Information Processing Systems*, NIPS '17, pages 4957–4966. 2017.
- [146] Chong Wang and David M. Blei. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 448–456, 2011.
- [147] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. Collaborative deep learning for recommender systems. In *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 1235–1244, 2015.
- [148] Nan Wang, Hongning Wang, Yiling Jia, and Yue Yin. Explainable recommendation via multi-task learning in opinionated text data. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '18, pages 165–174, 2018.
- [149] Xiang Wang, Xiangnan He, Fuli Feng, Liqiang Nie, and Tat-Seng Chua. TEM: Tree-enhanced embedding model for explainable recommendation. In *Proceedings of the Web Conference 2018*, WWW '18, pages 1543–1552, 2018.
- [150] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. Neural graph collaborative filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '19, page 165–174, 2019.

REFERENCES

- [151] Xiang Wang, Hongye Jin, An Zhang, Xiangnan He, Tong Xu, and Tat-Seng Chua. Disentangled graph collaborative filtering. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 1001–1010, 2020.
- [152] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. In *Proceedings of the 3rd International Conference on Learning Representations*, ICLR '15, 2015.
- [153] Ga Wu, Maksims Volkovs, Chee Loong Soon, Scott Sanner, and Himanshu Rai. Noise contrastive estimation for one-class collaborative filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '19, page 135–144, 2019.
- [154] Jibang Wu, Renqin Cai, and Hongning Wang. Déjà vu: A contextualized temporal attention mechanism for sequential recommendation. In *Proceedings of the Web Conference 2020*, WWW '20, page 2199–2209, 2020.
- [155] Yao Wu, Christopher DuBois, Alice X. Zheng, and Martin Ester. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the 9th ACM International Conference on Web Search and Data Mining*, WSDM '16, page 153–162, 2016.
- [156] Yao Wu and Martin Ester. FLAME: A probabilistic model combining aspect based opinion mining and collaborative filtering. In *Proceedings of the 8th ACM International Conference on Web Search and Data Mining*, WSDM '15, pages 199–208, 2015.
- [157] Yao Wu, Xudong Liu, Min Xie, Martin Ester, and Qing Yang. Cccf: Improving collaborative filtering via scalable user-item co-clustering. In *Proceedings of the 9th ACM International Conference on Web Search and Data Mining*, WSDM '16, page 73–82, 2016.
- [158] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. Attentional factorization machines: Learning the weight of feature interactions via attention networks. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI '17, pages 3119–3125, 2017.

REFERENCES

- [159] Yang Xiao, Ding Xiao, Binbin Hu, and Chuan Shi. ANE: Network embedding via adversarial autoencoders. In *Proceedings of the 2018 IEEE International Conference on Big Data and Smart Computing*, BigComp 2018, pages 66–73, 2018.
- [160] Caiming Xiong, Victor Zhong, and Richard Socher. Dynamic coattention networks for question answering. *CoRR*, abs/1611.01604, 2016.
- [161] Fuqiang Yu, Lizhen Cui, Wei Guo, Xudong Lu, Qingzhong Li, and Hua Lu. A category-aware deep model for successive poi recommendation on sparse check-in data. In *Proceedings of the Web Conference 2020*, WWW '20, page 1264–1274, 2020.
- [162] Seongjun Yun, Raehyun Kim, Miyoung Ko, and Jaewoo Kang. SAIN: Self-attentive integration network for recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, page 1205–1208, 2019.
- [163] Hanwang Zhang, Fumin Shen, Wei Liu, Xiangnan He, Huanbo Luan, and Tat-Seng Chua. Discrete collaborative filtering. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, page 325–334, 2016.
- [164] Yongfeng Zhang. Incorporating phrase-level sentiment analysis on textual reviews for personalized recommendation. In *Proceedings of the 8th ACM International Conference on Web Search and Data Mining*, WSDM '15, pages 435–440, 2015.
- [165] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '14, pages 83–92, 2014.
- [166] Yuan Zhang, Xiaoran Xu, Hanning Zhou, and Yan Zhang. Distilling structured knowledge into embeddings for explainable and accurate recommendation. In *Proceedings of the 13th ACM International Conference on Web Search and Data Mining*, WSDM '20, page 735–743, 2020.

REFERENCES

- [167] Lei Zheng, Chun-Ta Lu, Fei Jiang, Jiawei Zhang, and Philip S. Yu. Spectral collaborative filtering. In *Proceedings of the 12th ACM Conference on Recommender Systems*, RecSys '18, page 311–319, 2018.
- [168] Lei Zheng, Vahid Noroozi, and Philip S. Yu. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the 10th ACM International Conference on Web Search and Data Mining*, WSDM '17, pages 425–434, 2017.
- [169] Ziwei Zhu, Yun He, Xing Zhao, Yin Zhang, Jianling Wang, and James Caverlee. Popularity-opportunity bias in collaborative filtering. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, WSDM '21, page 85–93, 2021.
- [170] Ziwei Zhu, Shahin Sefati, Parsa Saadatpanah, and James Caverlee. Recommendation for new users and new items via randomized training and mixture-of-experts transformation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 1121–1130, 2020.
- [171] Ziwei Zhu, Jianling Wang, and James Caverlee. Measuring and mitigating item under-recommendation bias in personalized ranking systems. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 449–458, 2020.