An Aspect-Based Co-Attention Model for Recommendation Task

Boheng CAO

44181572-1

Master of Engineering

Supervisor: Professor Jinglu HU

Graduate School of Information, Production and Systems

Waseda University

July 2020

Abstract

The recommender system has become an indispensable technology in the internet era and plays a very important role in various content sharing platforms and e-commerce fields. In order to improve the performance of the recommender system, researchers conducted a lot of research on the recommendation algorithm, including the integration of information learned from multimodal data. Most existing algorithms based on multimodal data use multimodal data such as text information and image information. When they process text information, they often divide it into word level and sentence level or just directly model it. In this paper, we propose to use the aspect-level analysis to pre-process the text, and use the extracted aspect information for modeling, thereby directly filtering out the useless information and grasping the key points in the text. Moreover, we propose to use the co-attention distribution to learn the implicit relationship between the two-modal information in dataset. From this we construct a model consisting of two modules(Aspect Module and Cast Module) and two attention distributions(global attention distribution and co-attention distribution). We compare our model with an advanced multimodal data recommendation algorithm on the same dataset. The result shows that our model performs as good as the existing multimodal data recommendation algorithm, outperforms all of the baseline algorithms, when the training speed is greatly improved.

Keywords: Recommender system, multimodal, co-attention

Acknowledgements

At the completion of this thesis, I would like to first thank Professor Jinglu HU. No matter in research or in the previous courses, he always takes the trouble to answer my questions, guides me when I am confused and helps me get through the difficulties. I would also like to thank the other professors in IPS. Their teachings have always benefited me a lot. At the same time, I would also like to thank my colleagues in our laboratory. They have created a good academic environment for me, which has played a very important role in successfully completing this thesis. Of course I would also express my sincere thanks to my parents for their always support.

During the completion of this thesis, I learned a lot. I will keep working hard in the future and never forget the thing I learned in IPS.

Contents

List of figures **vi**

List of tables **vii**

1 Introduction **1**
1.1 Introduction to Recommender Systems 2
1.1.1 Input and Output of Recommender Systems 2
1.1.2 Classification of Recommender Systems 4
1.2 Introduction to Multimodal Learning 6
1.2.1 Definition of Modality and Multimodal 6
1.2.2 Recommendation Algorithms Based on Multimodal Data 7
1.3 Introduction to Aspect 9
1.3.1 Concept of Aspect 9
1.3.2 Aspect Level Recommendation Algorithms 10
1.4 Motivation and Main Contribution 12
1.5 Organization of the thesis 12

2 Related Work **14**
2.1 Multi-Level Attraction Model for Recommedation 14
2.2 Co-Attention Model in Visual Question Answering 16

3 Multimodal Dataset and Data Preprocessing **20**
3.1 Introduction of MovieLens Dataset 21
3.2 Aspect Extraction Tool Sentires 22
3.2.1 Principle of Sentires 22
3.2.2 Using of Sentires 24
3.3 Data Preprocessing 25

4 Aspect-Based Co-Attention Model **27**
4.1 Problem Definition . . . 27
4.2 Overview of Our Model . . . 28
4.3 Aspect Module . . . 30
4.3.1 Global Attention Distribution . . . 30
4.3.2 Co-Attention Distribution . . . 32
4.3.3 Output of Module . . . 36
4.4 Cast Module . . . 37
4.5 Output and Training of the Model . . . 38
4.5.1 Output of the Model . . . 38
4.5.2 Training of the Model . . . 38
4.6 Summary . . . 39

5 Experiments **41**
5.1 Baselines . . . 41
5.1.1 CENTROID . . . 41
5.1.2 CTR . . . 42
5.1.3 CWER . . . 42
5.1.4 MLAM . . . 42
5.1.5 MLAM-S . . . 42
5.1.6 MLAM-C . . . 43
5.2 Evaluation Criteria . . . 43
5.2.1 P@K . . . 43
5.2.2 R@K . . . 43
5.2.3 MAP@K . . . 44
5.2.4 MRR@K . . . 44
5.2.5 NDCG@K . . . 45
5.3 Results . . . 46
5.4 Model Variant Experiments . . . 48
5.5 Visualization of Co-Attention . . . 50
5.6 Summary . . . 53

6 Conclusion and Future Work **55**
6.1 Conclusion . . . 55
6.2 Future Work . . . 56

References **58**

List of figures

1.1 The framework of JRL[1] 8
1.2 Framework of AARM[2] 10

2.1 The framework of MLAM[3] 16
2.2 Flowchart of hierarchical co-attention model[4] 17
2.3 (a) Parallel co-attention mechanism; (b) Alternating co-attention mechanism[4] 18
2.4 Visualization of image and question co-attention maps[4] 19

3.1 A user review from Amazon[5] 23
3.2 Part of the obtained lexicon using Sentires 24

4.1 Overview of our model 29
4.2 Parallel co-attention mechanism 33
4.3 Alternating co-attention mechanism 35

5.1 MLAM[3] Left:MLAM-C Right: MLAM-S 42
5.2 Our Model(G+C+U) 49
5.3 Gray scale image of correlation score 51
5.4 Highest relation score gainer with aspect word "thriller" 52
5.5 Visualization of importance vector in co-attention of movie Metro(1997) . . 53

List of tables

2.1 Statistics of content-enriched MovieLens dataset[3] 15

3.1 Quantified attributes of MovieLens dataset[6] 22
3.2 Statistics of content-enriched MovieLens dataset[3] 26

5.1 Experiment results (compared with MLAM and its baselines) 47
5.2 Experiment results (compared with variants) 49

Chapter 1

Introduction

In today's era, the development of the Internet has promoted the prevalence of new media. A large amount of new content appears on the Internet at any time, including news, blogs, and videos. Thanks to this, a large number of content sharing platforms have emerged in recent years, such as Twitter and Netflix. After logging in to these platforms, users often do not need to take the initiative to search and browse, the platform will recommend the latest content to the user or content that the user may be interested in. According to the report, "75% to 80% of the content people watch on Netflix comes from what Netflix recommends to them, not from their own search"[3]. At the same time, in the field of e-commerce, the recommendation system also plays a very important role. In fact, most current e-commerce platforms use recommendation algorithms to varying degrees, especially in online shopping platforms such as Amazon. According to data from Amazon, every year Amazon's online bookstore can rely on the recommendation algorithm to contribute nearly 30% of its revenue.

In order to improve the effectiveness of the recommendation algorithm, researchers have conducted many researches on recommendation systems. One of the directions is to use the multimodal data of the recommended objects (for example, when recommending movies, consider the actors and movies story information), and integrate information learned from multiple modal data. Some existing algorithms for multi-modal data such as MLAM[3], VideoReach[7], JRL[1], etc. have achieved good results in the experiments on the corresponding data sets. On the basis of using multimodal data for recommendation, we consider

mining the aspect information hidden in the text information as a new modality. Based on this, a new model is established and the experiment is performed on the corresponding data set.

1.1 Introduction to Recommender Systems

The recommendation system has gone through more than 20 years of development, but so far there is no precise definition. Broadly speaking, a recommendation system is a system that actively recommends items to users. The recommended items may be news, music, movies, commodities, etc., which vary with different application fields. The items recommended by the recommendation system need to meet the conditions that the user may be interested in or helpful to the user. With the development of the Internet, the types and number of items have greatly increased, and users have higher requirements for recommendations. At present, the recommendation system we refer to generally refers to a personalized recommendation system. It is a recommendation system that considers the differences of different users in hobbies, personal experiences, etc. and gives different recommendations to different people in order to meet the different recommendation needs of different users . In this section, we will give an overview of the current recommendation systems and analyze the research status of recommendation algorithms.

1.1.1 Input and Output of Recommender Systems

The output data of the recommender system is very simple and clear. For a given user, the recommender system will output a "recommendation list", in which the items that the user may be interested in are given in order according to priority. The input data of the recommendation system has many forms, but it can be attributed to three levels: Item, User, and Review.

(1) Item

The input item information is usually used to describe some properties of an item, and is often referred to as Item Profile. The Item Profile varies according to the item.

For product recommendation, Item Profile may include product price, manufacturer, product description, etc .; for movie recommendation, Item Profile may be the movie title, starring, plot description, etc.

(2) User

The input user information is usually used to describe a user's characteristics, namely User Profile. The application scenarios of the recommendation system are ever-changing, and the specific algorithms are different. According to these, the representation of the User Profile is also different. It is an intuitive way to enter the user's gender, age, geographic location and other attributes into the system like items. However, such a representation is generally difficult to integrate into the recommendation algorithm, and in reality, even two users with the same attributes such as gender, age, location, etc. are likely to have very different interests and hobbies, so this representation is generally not used. Another more commonly used representation is to use the relationship between user information and item information for representation. For example, we can synthesize the Item Profiles of all products purchased by a user (such as weighted average operation) to get a new Profile, and use it as the User Profile of this user. This User Profile is more practical and more widely used. It has two advantages: first, we can easily calculate the similarity between this User Profile and a new item for recommendation; second, this User Profile can describe a user 's item more accurately At the same time, there is no need to collect user's private information and other private data.

(3) Review

Looking at the current recommendation algorithm, most of them will mention a concept called the user rating matrix. If users and items correspond to the rows and columns in the matrix, respectively, the evaluation is the value in the matrix. It can be said that the evaluation is a bridge between the user and the item. The recommendation system often needs to use the evaluation information to judge the preference relationship between the user and the item. Evaluation information is also diverse, generally we divide it into two categories: one is explicit, such as the evaluation text left by the user after purchasing

the product, the user's rating of an item, etc., all belong to this category; The second is implicit, such as the user's browsing history, the time the user stays when viewing the information of a certain item, and other data that the user naturally generates when using the website. Although the use of explicit information, the first type of evaluation information, currently occupies the mainstream of current research, researchers are increasingly paying attention to the use of implicit data for recommendation. Research in this direction will help to solve the recommendation system 's cold start problem and interpretability.

1.1.2 Classification of Recommender Systems

Through the introduction of the input and output of the recommendation system, we have a general understanding of the recommendation system. In this section, we will classify the existing recommendation systems to show the research status of recommendation algorithms.

(1) Demographic-based Recommendation

The demographic-based recommendations are for the first type of user information we mentioned earlier, namely the user's age, sex, location and other attribute information. This demographic information can be used to determine the type of user—just like determining a particular item. For example, when recommending a restaurant, [8] uses the two types of information to model the user's type by crawling the Internet homepage of users who like a particular restaurant and the homepage of users who do not like the restaurant. This method is based on an important assumption "users with similar attributes also have similar preferences". If this assumption holds, then we can easily use the similarity between users to make recommendations, that is, by quantitatively analyzing the attributes of users The purchase or viewing history of similar users is recommended to specific users. Moreover, due to the certainty of the user information, the similarity calculation of the user information can be completed offline, eliminating the calculation amount for each recommendation. But this assumption is often not true in reality. On the one hand, it is difficult to obtain all the information of the user (privacy issues), on the other hand, even users with the same information such as age and gender

are likely to have completely different interests. This recommendation method is mainly seen in the early recommendation algorithm research, and now it has rarely been used alone.

(2) Collaborative Filtering-Based Recommendation

Collaborative filtering is a method for automatically predicting (filtering) the interests of a particular user by collecting their preferences or tastes (collaboration) from many users. This method is based on a basic assumption: "If user A has the same opinion as user B in one thing, then A will also tend to have the same opinion as B in another thing." The main means of collaborative filtering is to find some relevance through the user's preference for the item, including the relevance of the item itself or the user. Collaborative filtering can also be divided into many categories, such as user-based, item-based, and model-based collaborative filtering. Collaborative filtering can recommend items that are difficult for content analysis and can discover some potential user interests, but collaborative filtering faces a serious problem, that is, when an item has just entered the market and there is no record of use, this item cannot be recommended through a recommendation method based on collaborative filtering. This is the problem of cold start.

(3) Content-based Recommendation

The content-based recommendation system relies on the description information (content) of the item and the user to construct the representation of the item and the User Profile, so that they can be efficiently used to recommend items that are similar to the items he previously liked to the target user. Note that the User Profile here refers to the second User Profile we mentioned earlier. It is not explicitly provided by the user, but constructed from historical records. Content-based recommendations are based on a basic assumption: "The user will tend to like items that he once liked." The typical content-based recommendation method is to first build a User Profile through the user's historical records, that is, "items that I have liked". This User Profile contains the user's preference and interest information, and then these information and item attributes are matched to obtain a relevance score that predicts the target user's level of interest in

the item. Through this score, we can recommend items to the users. The content-based recommendation method solves the problem of cold start, because even if it is a newly added item, as long as we can get its content, we can calculate its Item Profile. However, many current content-based recommendation algorithms cannot capture the user's attention, that is, they do not consider modeling the user's attention, and often require complex pre-processing of item information.

(4) Hybrid Recommendation

A hybrid recommendation system refers to a system that uses multiple recommendation methods in combination to achieve better recommendation results. In many cases, the recommendation based on collaborative filtering is combined with other recommendation methods to solve the cold start problem of collaborative filtering. Common hybrid methods include Weighted Fusion, Switch, Mix, Feature Combination, Cascade, Feature Augmentation, and Meta-level hybrid.

1.2 Introduction to Multimodal Learning

In the content-based recommendation system, content is the most important part of the system, and it is also the same in various hybrid recommendation systems containing content-based recommendation methods. In reality, when we recommend an item, the content information that we can obtain is often a combination of various kinds of data. In our research, we call it multimodal data. In this section, we will define multi-modal data, and briefly introduce several existing recommended algorithms based on multimodal data.

1.2.1 Definition of Modality and Multimodal

To define multi-modal, we must first define the modality. Broadly speaking, Modality is the way things happen or are experiencing[9]. In other words, your way of knowing something, including sight, hearing, touch, taste, etc., are different modalities. In our research, the following definition may be more vivid and relevant: Modality refers to a

specific type of information or a representation format for stored information. For example, voice, text, video, image, etc. are all different modalities, and even two different languages can be regarded as different modalities.

Multimodal data refers to data that contains a variety of different modality information. When recommending items, we can often collect data of different modalities, such as pictures and text descriptions of commodities, movie crew lists and plot descriptions. Studying the recommendation algorithm based on multi-modal data is conducive to enhancing the practicality and accuracy of the recommendation system, and has strong practical significance.

From a historical point of view, the study of multimodal has started from a very early stage. As early as the 1970s, researchers have studied multimodal in the field of behavioral; in the late 1980s, researchers began to apply the concept of multimodal to the field of computational; in the 2000s, the concept of multimodal appears in the research of interaction; since 2010, people began to use multimodal learning in the field of deep learning. At present, multimodal machine learning has become a popular research direction, which aims to achieve the ability to understand and process multi-modality information through machine learning methods.

1.2.2 Recommendation Algorithms Based on Multimodal Data

In order to improve the performance of the recommendation system, researchers have conducted a lot of research on recommendation algorithms, including the integration of information learned from multi-modal data. Many existing recommended algorithm models based on multimodal data have achieved better results on corresponding data sets than algorithms that only consider single-modal data. Here we will briefly introduce one of them: JRL. In Chapter 2, we will introduce another model MLAM in more detail.

JRL[1](Joint Representation Learning) is a model based on three modal data (text information, image information and scored numerical information) of user review. It regards the source of information of each modality as a view, and regards each piece of information (for example, a comment) as an entity. In each view, the entity will be represented as embedding based on the Deep Representation Learning architecture. These entity embeddings

will connect with users and items based on observed user-item interactions, which will give a representation of users and items in the corresponding view. Afterwards, the representations from each view are further mapped to a fully connected shared semantic space to obtain integrated user / item representations, and point-wise calculations are performed on these representations to obtain pair-wise learning score for ranking. When the model is learning, it randomly selects items that do not interact with the target user to construct positive and negative item pairs, and uses ranking loss in an end-to-end manner to learn the parameters and representations in the entire framework.

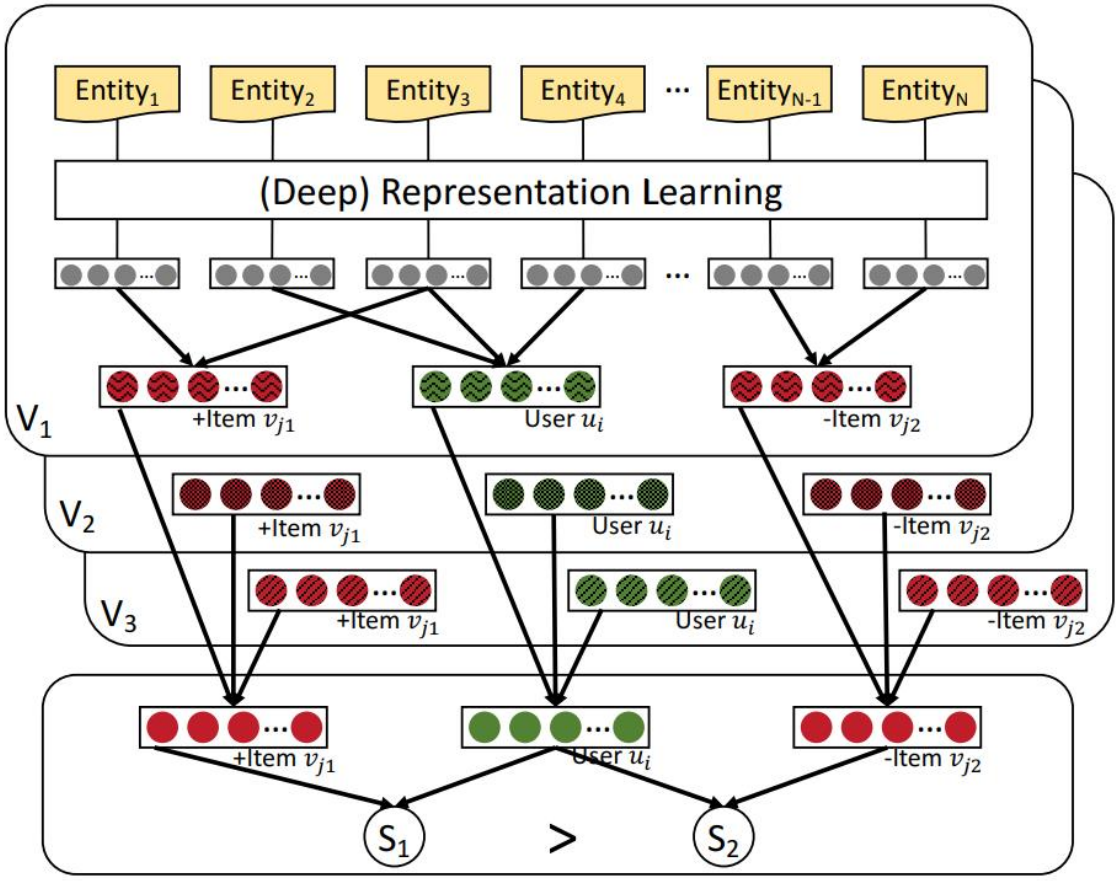

Fig. 1.1 The framework of JRL[1]

When faced with text information, existing recommendation algorithms based on multi-modal data often use it directly to model it, or divide it into word level and sentence level. In fact, we can use aspect-level analysis to preprocess the text and use the extracted aspect

information for modeling. The advantage of this is that it can filter out the useless information in the text and grasp the key points in the text information. Next, we will introduce the concept of the aspect and the current related research.

1.3 Introduction to Aspect

1.3.1 Concept of Aspect

The concept of Aspect is more common in Sentiment Analysis. In sentiment analysis, neither document level nor sentence level analysis can find out what people like or dislike, and Aspect level performs more fine-grained analysis in this regard[10]. The aspect level was also called the feature level in the early days. The aspect level does not focus on the structure of the language (documents, paragraphs, sentences, clauses or phrases), but directly focuses on the viewpoint itself. It is based on the idea that a point of view consists of emotion (positive or negative) and a goal, and a point of view that cannot be determined is often useless. Recognizing the importance of goals can further our understanding of sentiment analysis issues. For example: "Although the service is not so good, I still like this restaurant." Although this sentence is obviously a positive tone, we can't say it is absolutely certain. In fact, this sentence is positive for the goal of "restaurant", but negative for the goal of "service". In many applications, the goal of an opinion is described by entities and their different aspects, so the goal of this level of analysis is to discover entities and their various aspects of emotion. Below we use another example to show. "The call quality of this iPhone is very good, but the battery life is too short." In this sentence, "call quality" and "battery life" are two aspects of the "iPhone" entity. Through aspect-level analysis, we can turn unstructured text into a structured summary of opinions about entities and their aspects, so that various qualitative or quantitative analysis methods can be used.

1.3.2 Aspect Level Recommendation Algorithms

In recent years, many studies have extracted information from user reviews and integrated them to improve the performance of recommendation systems. The common aspects of the evaluation made by a user and the evaluation received by an item indicate some indirect connections between the user and the item. In this section we will introduce a model that uses aspect information for recommendation AARM[2] (Attentive Aspect-based Recommendation Model).

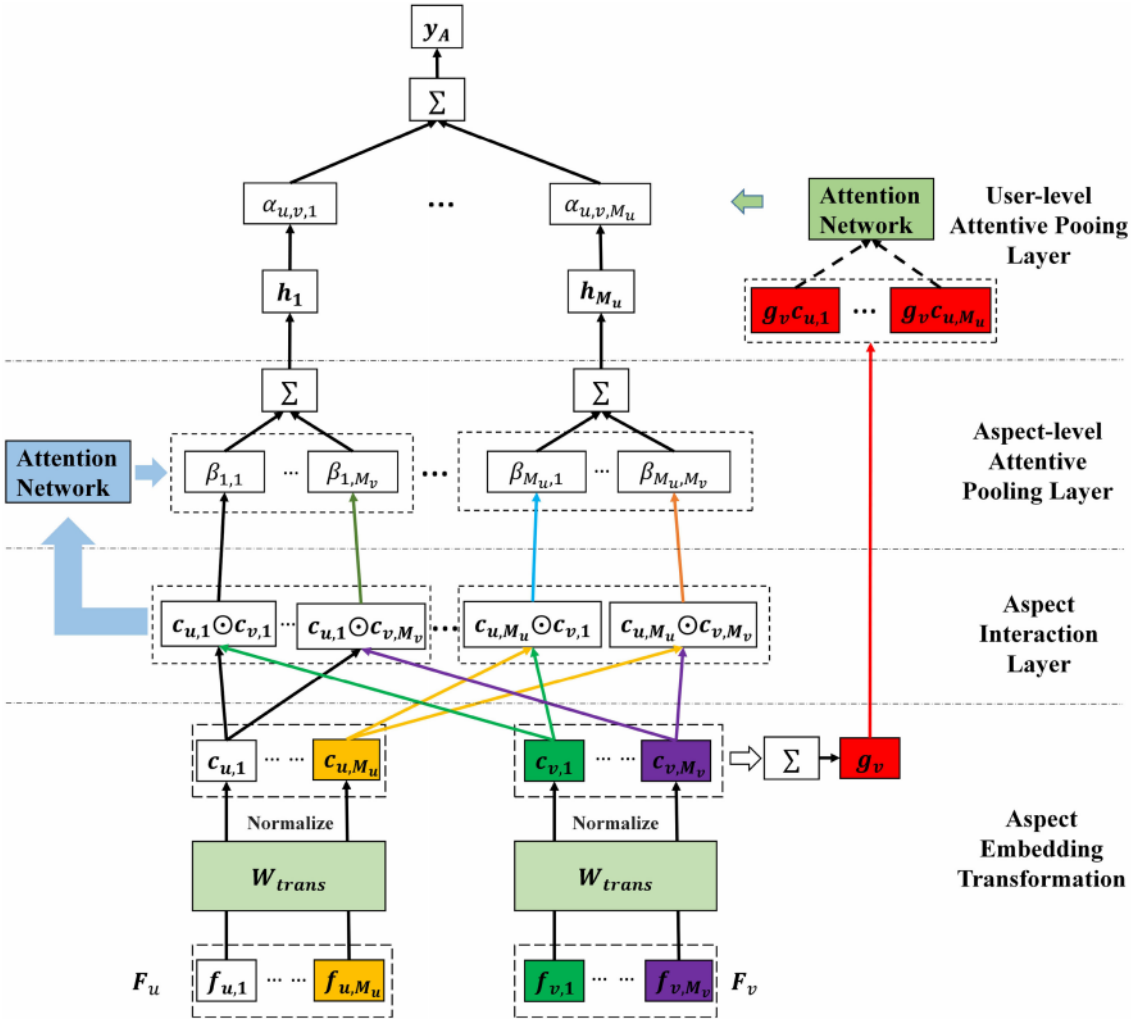

Fig. 1.2 Framework of AARM[2]

AARM defines aspects as words or phrases that users use to describe product features in product reviews. It points out that due to the sparse nature of user-item interactions and

the different evaluation terms of users, the common aspects between the evaluation of target users and items are often very limited. At the same time, when users buy different items (even items of the same category), the information they may focus on will be different. AARM tries to solve these two problems through a unique neural network structure.

At the input layer, users and items are represented as binary sparse vectors using the one-hot encoding method. Above the input layer, the Aspect Interactions part models the interaction between user and item aspect information, while the Global Interactions part is used to model some hidden factors. We mainly Introduce the interactive part. In order to model the similarity between aspects, AARM uses an embedding layer to represent aspects as a continuous vector. The Attentive Aspect-Interaction Pooling Module embeds aspects between users and items. Modeling of the two-way interaction.

The aspect embedding vectors of users and items are normalized by the Aspect Embedding Transformation layer, and the resulting new vectors are calculated by dot multiplication at the aspect interaction layer. Note that in addition to the same aspects, the interactions between different aspects are also calculated, so the model can model synonymous or similar aspects, which solves the problem of sparsity to a certain extent. In the Aspect-level Attentive Pooling Layer, an Attention Network is introduced to model the user's attention to different aspects. This layer will output every aspect of the user for this The total similarity of all aspects of the item. At the user-level attention pooling layer, AARM uses the aspect information of the items to calculate the weights to weight and sum the results of the previous layer. In this way, the modeling problem of the user's different attention to different items is solved. The final score obtained by this module is used to connect with the result of the global interaction part to obtain the total score for recommendation.

AARM and some other existing aspect-level recommendation algorithms use user reviews to perform aspect extraction. We believe that aspect-level concepts can also be applied to other types of text, such as movies based on the plot information, and built a new model based on it, which will be introduced in detail later.

1.4 Motivation and Main Contribution

As we show in 1.2, existing recommendation methods based on multimodal data usually deal with data in different modality separately, without considering the relationship between them. On the other hand, when they deal with texual data, they usually divide it into word level, sentence level and so on, then use relatively complicated neural network to model it. However, recommendation performance may still get hurt because of useless words or sentences in the text.

To deal with the first problem, we decide to use co-attention to discover relationship between data of different modality. Co-attention has been widely used in Question-Answering area, which will be introduced in Chapter 2 in detail. For the second problem, we have tried some data prepocessing of texual data to remove useless information, and finally we find that aspect extraction shows great performance on our task.

This paper is mainly about our aspect-based co-attention model for recommendation task. We also introduce our data preprocessing method in detail, because it provides aspect lexicon for us, which is indispensable in our task. After introduce our data preprocessing and model, we will give the experiment results of our model on MovieLens dataset and do some comparisons with our baselines.

1.5 Organization of the thesis

The rest of the paper is arranged as follows: Chapter 2 introduces the related work in the field of recommendation algorithm based on multi-modal data and the using of co-attention in other research field. Chapter 3 introduces the data set we use, and the data preprocessing we do before experiments, it also introduces the data preprocessing method and tool we use. Chapter 4 is the model design part. In this chapter, we will discuss our aspect-based co-attention model and the different co-attention mechanisms we use in detail. Chapter 5 is the experiments part. We will compare our model with MLAM and its baselines, and we also do some more experiments of model variant and visualization. The last part (Chapter 6) is

the conclusion and future work, where we will discuss the summary of previous work and future possibilities.

Chapter 2

Related Work

Content-based recommender systems use content to find similarities between items. Moreover, researchers have tried to use multimodal data as content to do recommendation. In this chapter, we firstly introduce an existing recommendation algorithm based on multimodal data named MLAM[3]. Then, to fix the problems we find in MLAM, we will introduce an attention mechanism called co-attention, which has been widely used in Question Answering area.

2.1 Multi-Level Attraction Model for Recommedation

As we introduced in 1.1.2, collaborative filtering-based recommendation algorithms use similar users' watch record to do recommendation. However, they all suffer from the cold start problem which means that they can not recommend an item which has not been rated by anyone. Content-based filtering methods[11] discover similarity from contents to avoid the cold start problem. As we all know, when user select items, the selection is always affected by some attractive points, e.g., a singer of a song. However, existing content-based filtering methods usually cannot find these attractive points. Nowadays, some researchers begin to notice that only focusing on improving the accuracy of recommender system may just hurt it[12][13]. In [3], instead of just focusing on higher recommendation accuracy, they try to

find and interpret the attractive points in available contents, although their model can still get comparable accuracy performance.

MLAM (Multi-Level Attraction Model) is an recommendation model based on a movie dataset named MovieLens 1M. MovieLens 1M itself does not contain multi-modal data, but only provides the user's viewing records, but some researchers have provided a mapping from the ID of the data set to DBPedia URI[14], so you can get the corresponding movie story and cast members information from DBPedia. The expanded MovieLens data set attributes are shown in Table 2.1.

Table 2.1 Statistics of content-enriched MovieLens dataset[3]

# movies:	3,900	# users:	6,040
# watch record:	1,000,209	# cast:	9,398
movie story vocabulary	22,582	# sentences per story	10.2
# cast members per movie	6.44	# plays per cast	2.10

The MLAM model contains three main parts: user module, cast attraction module and story attraction module. In the story attraction module, a multi-level attraction model is established to score the user's attention to words and sentences. Similarly, in the cast attraction module, another attention model is established for scoring the attention of the cast members. Finally, the results of these two parts are used together at the top of the model to calculate the user's attention score for a movie.

In each attention module, both the word information and the cast information are first converted into embedding vectors. In this model, a user attraction filter is defined. Using the inner product of this filter vector and embedding vectors, weights are calculated by the ISR function and softmax function. The obtained Weights will be used to do weighted sum of the embedding vectors. The result obtained in this way can reflect the user's attention information. For example, if the user may like a certain actor to watch a movie, this actor will have a higher weight than other actors. This result is combined with another result obtained by using a filter that has nothing to do with the user, and the result of the concerned module is obtained. The story attention module can be said to be an alternative to the sentence level

target module and the phrase level related module. This model also uses ranking loss for learning.

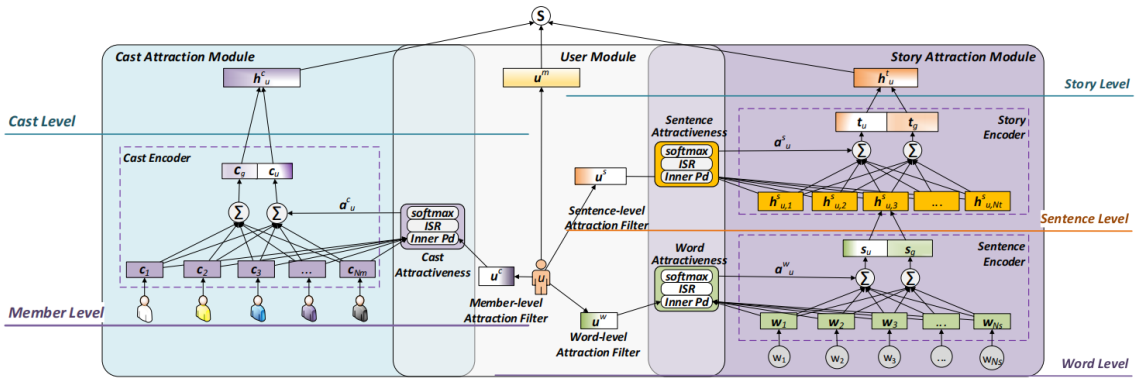

Fig. 2.1 The framework of MLAM[3]

Like most existing recommendation algorithms based on multimodal data, MLAM deal text information by dividing it into word level, sentence level and then using relatively complicated neural network to model it. However, recommendation performance may still get hurt because of useless words or sentences in the text. To deal with this problem, we use some data preprocessing method to avoid it, which will be introduced later. Moreover, most existing recommendation methods based on multimodal data (including MLAM) deal with data in different modality separately, without considering the relationship between them. In our research, we use co-attention to deal with this problem. In next section, we will introduce it in detail.

2.2 Co-Attention Model in Visual Question Answering

Co-attention has been widely used in Question Answering, especially Visual Question Answering area. In both industry and academia, Visual Question Answering (VQA) has become a prominent multidisciplinary research question. To answer questions about an image correctly, the model is supposed to understand information from both question and image. Recently, lots of visual attention based models are implemented for VQA[15][16][17]. In

these models, the attention mechanism typically produces a spatial map highlighting image regions relevant to answering the question.

However, most of these attention models have only focused on the problem of visual attention, which means "where to look". In [4], they argue that it is equally important to identify "which words to listen", or question attention. They notice that there are always some key words in a question, for example "how many apples" in "how many apples can you see in this image" or "how many apples are in this image". It can be said that machines which attends these key words are more robust to word changes that have nothing to do with the meaning and answer of the question.

Motivated by this observation, in addition to reasoning about visual attention, they also address the problem of question attention. Specifically, they present a novel multi-modal attention model for visual question answering with the following two unique features:

- Co-Attention: They propose a novel mechanism that jointly reasons about visual attention and question attention, which we refer to as co-attention.
- Question Hierarchy: They build a hierarchical architecture that co-attends to the image and question at three levels: (a) word level, (b) phrase level and (c) question level.

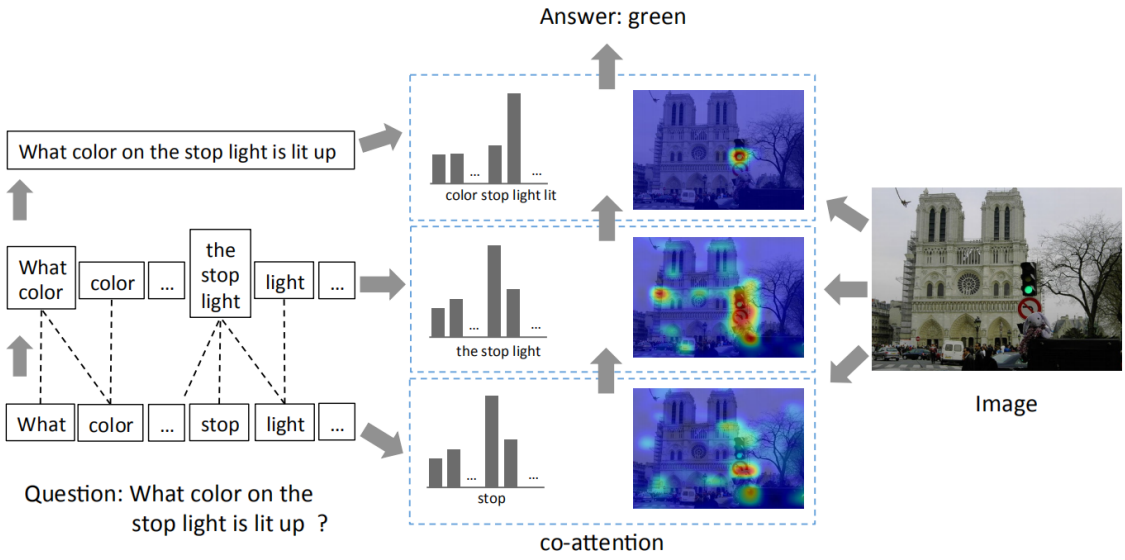

Fig. 2.2 Flowchart of hierarchical co-attention model[4]

Figure 2.2 shows their proposed hierarchical co-attention model. When they face a question, they divide it by question level, phrase level and word level, then extract embeddings of each level. They apply co-attention at each level on both the question and the image. The final answer prediction depends on all of the co-attended question features and image features.

They use two kinds of co-attention mechanism, which are shown in Figure 2.3:

- Parallel co-attention mechanism: Parallel co-attention mechanism generates the question and image attention simultaneously.
- Alternating co-attention mechanism: Alternating co-attention mechanism sequentially alternates between generating image and question attentions.

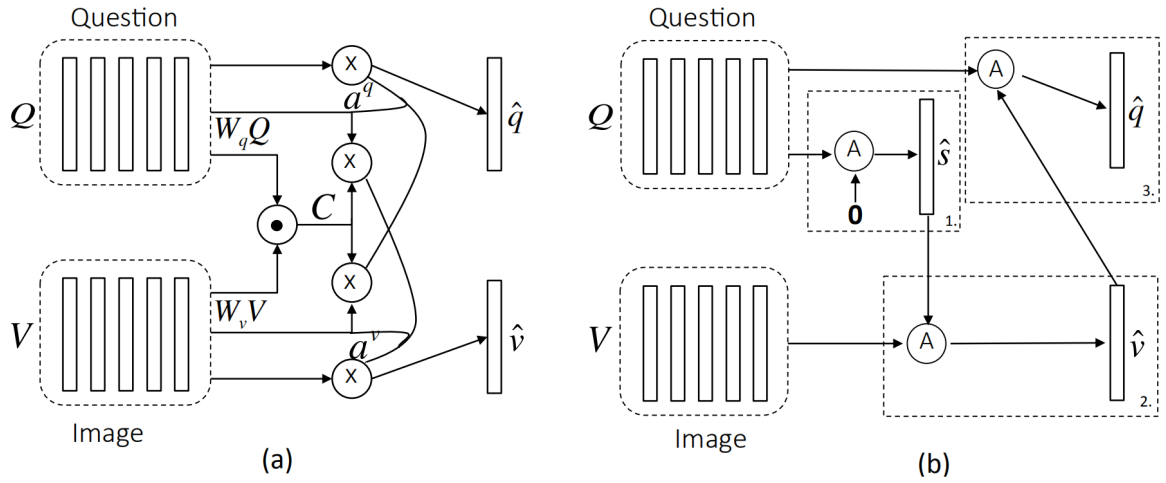

Fig. 2.3 (a) Parallel co-attention mechanism; (b) Alternating co-attention mechanism[4]

Following [18], they treat visual question answering as a classification task. Using co-attended question and image features from three levels, they predict the answer. They use an MLP to recursively encode the attention features. Experiments shows that their model gets better performance, and they also use some visualizations to show the effect of co-attention.

In our research, we also use both two kinds of co-attention mechanism, which will be introduced in detail in Chapter 4.

Fig. 2.4 Visualization of image and question co-attention maps[4]

Chapter 3

Multimodal Dataset and Data Preprocessing

After the introduction in the first two chapters, we have a clear and accurate understanding of the concept and research status of the recommendation system, the concept of multi-modality and aspects. We have proposed a more general aspect-level recommendation algorithm for various multi-modal data sets. In order to facilitate experiments and comparisons, we chose the MovieLens data set mentioned in 2.1 for experiments. The original MovieLens data set only contains the user's viewing record information of the movie, but some researchers have established a good mapping relationship between the data set and DBPedia, so we can easily obtain the movie story information and cast information in the MovieLens data set. On this basis, we preprocessed this data set to meet the requirements of our research on aspect-level recommendation algorithms. In this chapter, we will briefly introduce the MovieLens data set, and introduce the tool Sentires that we use to extract the textual information of the movie story. We will also explain the data preprocessing work we have done.

3.1 Introduction of MovieLens Dataset

The MovieLens dataset was first published in 1998 and described people's preference for movies[6]. These preferences are presented in the form of tuples of <user, item, rating, timestamp>, and each group is a user's preference for a movie at a specific time (a 0-5 star rating). These preferences are entered through the MovieLens website, which is a recommendation system that allows users to give movie ratings to obtain personalized movie recommendations.

The MovieLens dataset has harvested a large number of downloads (over 140,000 downloads in 2014) and is frequently cited in research literature (more than 7,500 cited records in Google Scholar). This popularity reflects to some extent the astonishing growth rate of personalized and recommended scientific research, and the important value of such data sets in exploring and verifying ideas in related research. At the same time, this popularity can also be attributed to the flexibility of rating data. Rating data is not only suitable for the research of recommendation algorithms, but also for more general such as summarization, pattern identification, visualization, etc. Data science and technology research. In addition, since the preference of movies depends greatly on personal taste, movie datasets are very suitable for testing personalization techniques. Finally, popularity may reflect the perceptibility and accessibility of movies in the content domain: movies are a common interest, which makes the output of the algorithm easy to discuss.

The MovieLens data set has gone through more than 20 years and has also been released in multiple versions. The size of the data set is also increasing in chronological order: MovieLens 100K, MovieLens 1M, MovieLens 10M, MovieLens 20M. Their names reflect the approximate number of ratings in each version of the dataset. Some quantified attributes of these datasets are shown in Table 3.1:

Note that the "density" value in the table represents the proportion of cells with rating values in the entire user-item matrix. We can see that no matter in which version of MovieLens dataset, the density value is very small. This shows the sparsity of the user-item matrix of the data set. In our research, we selected MovieLens 1M version dataset for experiments. It should be noted that MovieLens itself is not a multimodal data set, but some researchers

Table 3.1 Quantified attributes of MovieLens dataset[6]

Name	Date range	Users	Movies	Ratings	Density
ML 100K	1997.9-1998.4	943	1,682	100,000	6.30%
ML 1M	2000.4-2003.2	6,040	3,706	1,000,209	4.47%
ML 10M	1995.1-2009.1	69,878	10,681	10,000,054	1.34%
ML 20M	1995.1-2015.3	138,493	27,278	20,000,263	0.54%

have established a good mapping relationship between MovieLens and DBPedia. We directly use the expanded MovieLens 1M data set provided by [3] mentioned in 2.1. The properties of this data set are shown in Table 2.1. On the basis of this extended data set, we carried out a series of data preprocessing work, one of which is the aspect extraction.

3.2 Aspect Extraction Tool Sentires

Aspect extraction is an important part of our data preprocessing. In our research, aspects are defined as "words or phrases that can indicate the attributes of the movie in the text information of the movie story." The aspect extraction is to extract these words or phrases from the plot text information. Since the focus of our research is on the design of recommendation algorithms, in this experiment we use external tools to extract aspects of the story text information. The tool we use is Sentires (from yongfeng.me/code), which has been successfully applied in studies such as [2][5][19]. Sentires uses rule-based machine learning hybrid algorithms to extract aspects. Given a data set, it will generate an aspect lexicon, which will be used to construct our aspect data set in our research. In this section, we will briefly introduce the principle and using of Sentires.

3.2.1 Principle of Sentires

The construction of sentiment lexicon is a key part of Sentires' principle. Each entry in the lexicon is a feature-opinion (F-O) word pair with corresponding emotional polarity S, expressed as (F, O, S). For example, such terms can be extracted from the comment text in Figure 3.1: (service, excellent, positive) and (phone quality, perfect, positive).

Apple iPhone 5 16GB (White) - Unlocked by Apple

By [redacted] (Chennai) - See all my reviews

Exceeds the expectation, June 18, 2013

I am very happy to have bought this phone from Amazon and the service rendered from the seller is excellent. Phone quality is perfect as new though I bought an used one. Care to their customers is something a key strategy the seller has followed. I would like to deal again with the same group in near future and recommend to others highly. Thank you.

Fig. 3.1 A user review from Amazon[5]

The previous phrase-level sentiment lexicon building method can only give 70% 80% accuracy of sentiment polarity labels (assignment of polarity S for an F-O pair). The authors of Sentires point out that the reason is that a basic assumption of these methods that “the overall numerical rating of the review represents the overall sentiment of the review text” may not be correct. To avoid this erroneous assumption, they plan to improve the performance of phrase-level sentiment polarity labels in an opposite way, using unsupervised comment-level sentiment classification results to replace digital ratings as an inspiration for phrase-level sentiment labels.

The framework is generally composed of two stages: in the first stage, a comment-level sentiment classifier is used to mark the overall sentiment direction of product reviews; in the second stage, feature-opinion pairs are extracted from the corpus, and then reviews are used As a constraint on the overall emotional direction, a new optimization framework is used to automatically learn the emotional polarity of these pairs. When setting constraints in the second stage, a total of four constraints are considered: Review-level sentiment orientation captures the overall sentiment of a review given by the review-level sentiment classification algorithm in the previous stage; General sentiment lexicon captures some emotions that are not related to context, such as excellent, good, and bad; Linguistic heuristic captures the relationship between "and" and "but" in linguistics. Intuitively, those F-O pairs that are often connected with "and" may have similar emotions, and those that are often connected with "but" often have opposite emotions; Sentential sentiment consistency captures the emotional consistency in the sentence, that is, similar directions of opinions are usually expressed in

consecutive sentences. The framework then builds and learns the objective function based on the above four constraints.

3.2.2 Using of Sentires

The design goal of Sentires is aimed at the user's evaluation text of the product. We applied it to the film plot text and achieved good results. Sentires is provided to us as a java package, we need to change the two parameters [task] and [conf] in the command "java -jar thuir-sentires.jar -t [task] -c [conf]" Come and use it to complete different tasks. The [task] parameter indicates the type of task, and the [conf] parameter indicates the location of the configuration file. In the configuration file, we specify the input and output paths of the task and other configuration information. The several [task] parameters we need to use are: pre, data preprocessing; pos, part of speech tagging; validate, verifying the results of word tagging; lexicon, building an emotional lexicon. These tasks need to be performed in order.

Among them, in the data preprocessing task, there are requirements on the format of the input file: when performing sentiment analysis on the evaluation text, the input of the task needs to be a strip of evaluation text separated by <DOC> </ DOC> Our input file is converted into a series of movie story text separated by <DOC> </ DOC> and then data preprocessing tasks are performed. Each line of the resulting lexicon file represents a feature-opinion pair in the following format: [1 or -1] feature | opinion x y

```
[1] academy awards|best 4   0
[1] action films|heroic 1   0
[1] actor|best  4   100
[1] actor|frank 1   0
[1] actor|fredric   1   0
[1] actress|best    5   105
[1] actress|golden  1   0
[1] actress|hungry  1   0
[1] actress|original    1   0
[-1]    actress|worst   1   10
```

Fig. 3.2 Part of the obtained lexicon using Sentires

Taking the 6th line as example, actress is the feature word and best is the opinion word. [1] indicates that the emotion of the feature-opinion pair is positive. The first number 5 in the back indicates the number of times the feature-opinion pair was used positively (that is, the feature word appears before the opinion word), and the second number 105 indicates that the feature-opinion pair was used reversely (that is, the feature word Appears after the opinion word). We extracted a total of 555 feature-opinion pairs from the MovieLens 1M extended dataset.

3.3 Data Preprocessing

Our basic data set comes from the expanded MovieLens 1M data set provided by [3]. On the basis of this data set, we carried out some data preprocessing to make the data set conform to our research needs.

In the original data set, the story text information of the movie is stored in the form of a json file. We extract this information and convert it into the story text of different movies in a format separated by <DOC> </ DOC>. After extraction, the lexicon file is finally obtained, and its format is shown in Figure 3.2. In order to ensure the integrity of the aspect information, we keep all the feature words and opinion words of the feature-opinion pairs in the lexicon file into the aspect lexicon, because we will later convert the aspect information into the form of word embedding, we split the feature word and opinion word phrases, as shown in Figure 3.2, the action films phrase is split into two words, action and films, and stored in the aspect lexicon. After removing duplicate words, we extracted 477 aspect words from 555 feature-opinion pairs.

In the original data set, a word dictionary has been established. We compared the extracted 477 aspect words with the word dictionary and removed 32 aspect words that could not be found in the word dictionary. Finally, our aspect lexicon has 445 words. Compared with the 22582 vocabulary of the movie plot in Table 2.1, it can be said that it is greatly compressed.

Then we filter the story text information of the movie, first split the story text information into words, and then judge whether these words are in our aspect lexicon one by one, and keep the words which can be found in the aspect lexicon. We use these words to build the aspect information part of the new dataset. Finally, we use the constructed aspect information to directly replace the story text information in the original data set. The extended MovieLens 1M attributes after our preprocessing are shown in Table 3.2:

Table 3.2 Statistics of content-enriched MovieLens dataset[3]

# movies:	3,900	# users:	6,040
# watch record:	1,000,209	# cast:	9,398
movie aspect vocabulary	445	# aspect words per movie	12.6
max movie aspect words	51	# max movie cast members	33
# cast members per movie	6.44	# plays per cast	2.10

Chapter 4

Aspect-Based Co-Attention Model

In the third chapter, we introduced the MovieLens, the dataset to be used in the experiment, and explained the preprocessing we made on the dataset, and explained the aspect extraction tool we used—Sentires. Our dataset is the same as the dataset used by MLAM[3] in 2.1, but we discarded the lengthy text data used by MLAM and replaced it with the key aspect information we extracted. In terms of models, we have also made two improvements based on MLAM. First, we use the extracted aspect information instead of text information, thereby avoiding the use of multi-layer complex neural network structures at the word level, sentence level, and document level; second, MLAM considers the information of the cast and the text information into two parts independently, and only integrates the scores of the two parts at the end, and we have considered extracting the connection between the cast information and the aspect information, using co- attention to model the relationship between the two pieces of information. In this chapter, we will introduce our improved model in detail.

4.1 Problem Definition

We define the movie collection as:

$$M = \{m_1, \cdots, m_{N_M}\} \tag{4.1}$$

For each movie m in the movie collection, it contains an aspect collection:

$$A_m = \{a_1, \cdots, a_N\} \tag{4.2}$$

and a cast collection:

$$C_m = \{c_1, \cdots, c_M\} \tag{4.3}$$

We define the user set U to model the user's interest to the movie. Given a user $u \in U$, the set of movies he has watched is defined as:

$$M_u = \left\{m_{u,1}, \cdots, m_{u,N_u}\right\} \tag{4.4}$$

Given a movie $m \in M_u$, one of our tasks is to learn the attractive information on the aspect set of the movie for user u and generate an aspect level representation h^a. Another task is to learn the weight of attractiveness in the list of cast members C_m and generate a cast level representation h^c based on attractiveness. After that, we can use h^a and h^c to rate the attractiveness of the movie m. After learning the parameters of the model, we can calculate each user's personal attention score on a series of candidate movies M_C to make recommendations. Moreover, we can use the high-weighted aspect information or cast member information to explain our recommendation results, that is, it may be that some aspects or certain cast members have attracted target users.

4.2 Overview of Our Model

The structure overview of our aspect-based co-attention recommendation model based on multimodal data is shown in Figure 4.1. Our model is mainly divided into two modules, Aspect Module and Cast Module. The input of each module is a list. At the bottom of the module, the cast list and aspect list are first converted into embedding vectors for the later calculation.

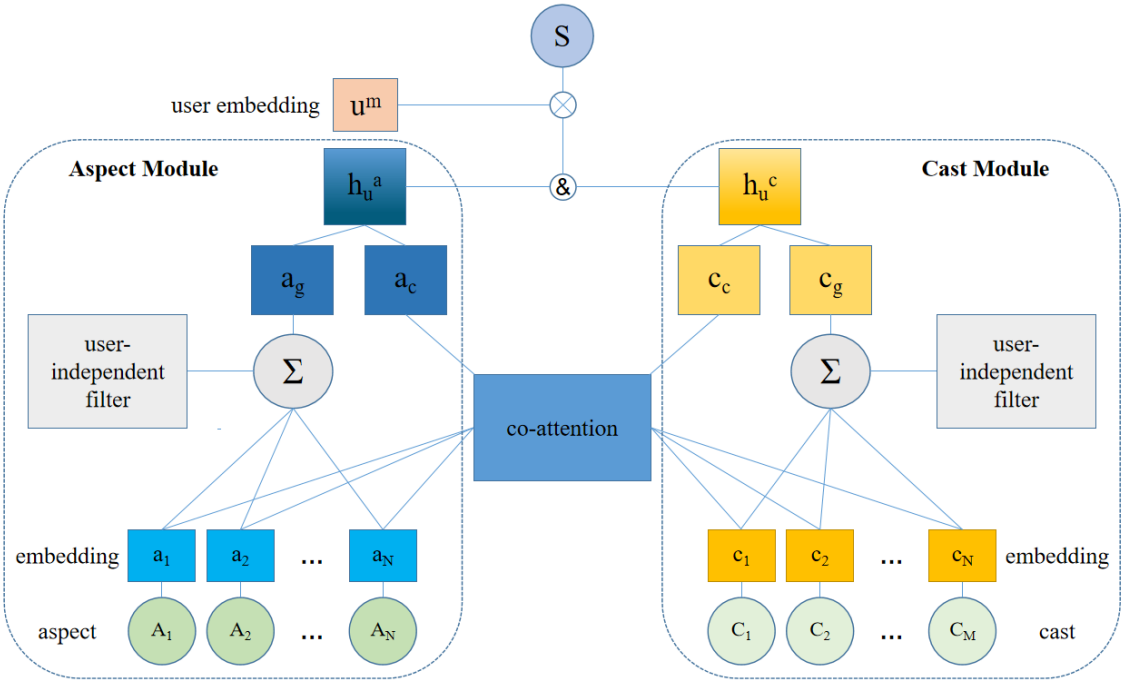

Fig. 4.1 Overview of our model

Our aspect-based recommendation model based on multimodal data mainly contains two modules, aspect module and cast module. The input to each module is a list. At the bottom of each module, the list of aspects or casts is first converted into embedding vectors for subsequent calculations. In each module, we need to calculate two attention distributions, which are global attention distribution and co-attention distribution. Global attention distribution is calculated by weighted summation of the embedding vectors through a global user-independent filter, reflecting the importance of different aspects or different casts in the global view, which is independent of a particular user; The global attention distribution is calculated relatively independently in two modules, while the co-attention distribution requires two modules for cooperative calculation, which reflects the relationship between the aspect information and the cast information. If some directors or actors are good at movies of certain subjects, such information can be learned through the co-attention distribution. At the top of each module, the embedding vectors representing two distributions are concatenated to obtain a comprehensive embedding vector, Finally, we use two comprehensive embedding vectors to calculate the user's individual attraction score towards a movie at the top of the two

modules. Next, we will use the aspect module as an example to explain these two attention distributions one by one.

4.3 Aspect Module

In the aspect module, the input we get is a list containing all the aspects extracted from a movie story text, see (4.2). Firstly, we transform each aspect a_i to an embedding vector $\mathbf{a_i} \in R^L$, L denotes the length of the embedding vector. We use the pretrained GloVe[20] vectors to initialize our embedding vectors. Later, we will use these embedding vectors to calculate global attention distribution and co-attention distribution of the aspect module.

4.3.1 Global Attention Distribution

In MLAM[3], they use user attention filter and discard most of the information that does not appeal to specific users. However, due to the sparse nature of the user-item matrix of the data set, they cannot guarantee that the learned information that does not attract specific users is completely useless. In order to protect the potentially valuable information filtered by the user attention filter, they use a global user-independent filter to filter the aspect information to obtain a global attention distribution. In our model, we also use global attention distribution to protect the potentially valuable information filtered by co-attention distribution.

We define the aspect-level global attention filter as $g^a \in R^L$, and we use inner product of g^a and each aspect embedding to calculate attention score s_i^a:

$$s_i^a = g^a \mathbf{a_i} + b^a, i \in \{1, \cdots, N\} \tag{4.5}$$

Note that g^a and b^a are global variables which are independent of user. Then normalized weight can be calculated by softmax function:

$$\text{softmax}(s_i) = \frac{e^{s_i}}{\sum_j e^{s_j}} \tag{4.6}$$

However, s_i^a in (4.5) might be very large, which will cause softmax function to output a weight value close to 1. And in actual implementation, a large s_i will easily make exponential function overflow. To solve this problem, we introduce an inverse square root function isr^α to limit the value of s_i:

$$\text{isr}^\alpha(x) = \frac{x}{\sqrt{1+\alpha x^2}}, \text{isr}^\alpha \in (-\alpha^{-\frac{1}{2}}, \alpha^{-\frac{1}{2}}) \tag{4.7}$$

We can use the parameter α to control the upper and lower limits of the function output. A larger α makes the upper and lower limits of the function close to 0. In this case, the softmax function tends to output uniform weights. A smaller α can make the exponential function not overflow, and the softmax function can also output differentiated weight values. For example, when $\alpha = 0.001$, the upper and lower limits of the isr^α function are (-31.6, 31.6). Since the purpose of calculating the global attention distribution is to protect some potentially valuable information, here we set the value of α to be larger to output more uniform weights. In our experiment, after adjusting the parameters for comparison, we set the value of α to 64. In this way, the standardized attention score corresponding to aspect a_i can be expressed as:

$$\bar{s}_i^a = \text{softmax}(\text{isr}^{\alpha=64}(s_i^a)) \tag{4.8}$$

Since $\bar{s}_i^a$ scores the global attention in each aspect, we can use it to calculate the embedding vector representing the global attention distribution. The method is to use it as the weight of the embedding vector of each aspect and then perform weighted summation:

$$\mathbf{a_g} = \sum_i \bar{s}_i^a \mathbf{a_i} \tag{4.9}$$

In this way, we finally get an embedding vector $\mathbf{a_g}$ that can represent the global attention distribution. $\mathbf{a_g}$ retains all aspects information more evenly, but it also reflects some of the differences between different aspects globally. For example, when an aspect is valued by most users, its weight will be relatively high.

4.3.2 Co-Attention Distribution

Co-attention is often used to learn the relationship between two sets of embeddings. In the research of Question Answering [4], it is used to learn the interrelationship between question text and alternative answer text. In the study of Network Embedding[21], it is used to learn the relationship between two vertices. The author proposes that a specific vertex will play different roles when interacting with other different vertices. In other words, each vertex should have its own unique focus on a specific vertex.

In our research, we use co-attention to model the relationship between the story aspect information and the cast member information in a movie. Our modeling is based on the reality that, some directors or actors are good at certain themes, such as American screenwriter and director Eric Red is good at horror movies, then movies have both cast "Eric Red" and aspect "thriller" should receive higher attention scores. This kind of information can not be learned by calculating the aspect module and the cast module independently. But we can learn this implicit “good at” information by linking the two modules together and using co-attention to learn the relationship between the aspect information and the cast information.

As Figure 2.3 shows, there are mainly two kinds of co-attention mechanism: Parralel co-attention mechanism and Alternating co-attention mechanism. In our research, we implement both of them in our model. Next we will introduce the calculation of them in detail.

Parallel Co-Attention Mechanism

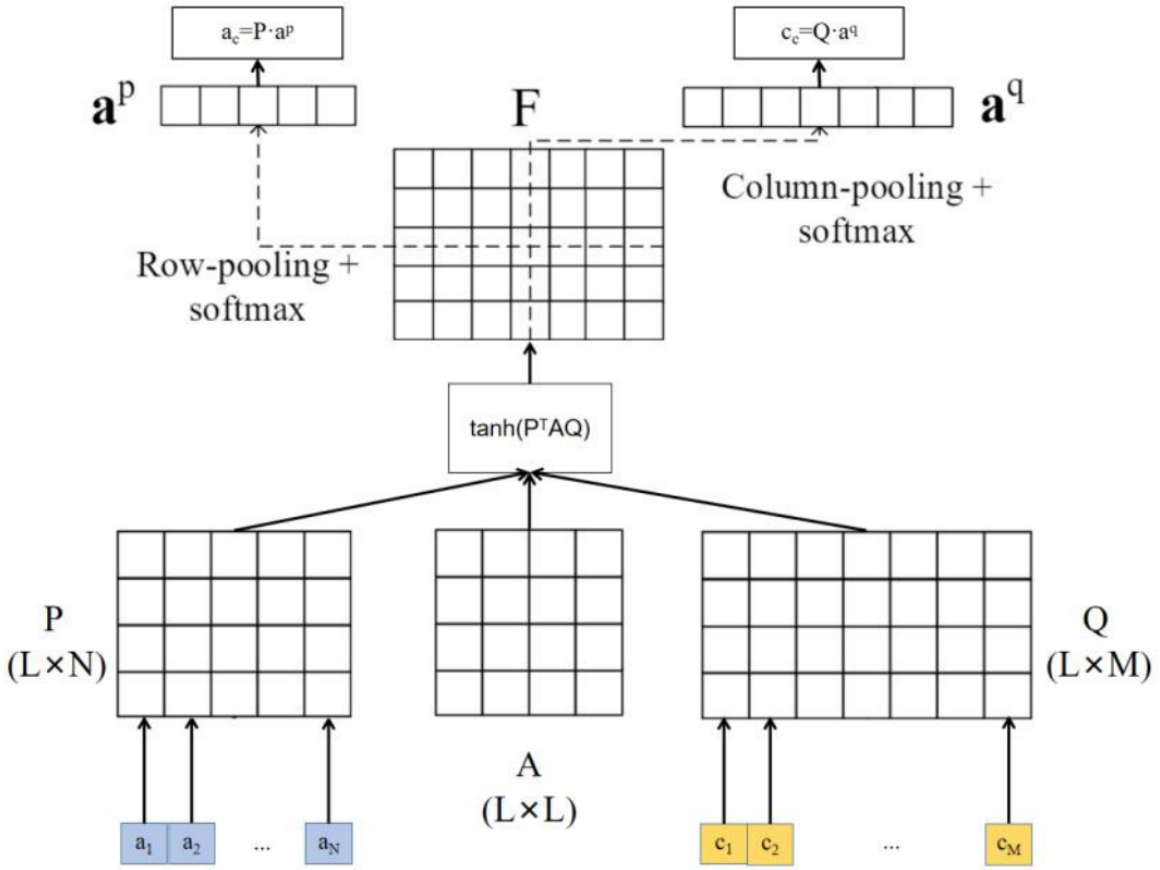

Fig. 4.2 Parallel co-attention mechanism

In figure 4.2, we give the process of obtaining the vector $\mathbf{a_c}$ representing the aspect-level co-attention distribution and the vector $\mathbf{c_c}$ representing the cast-level co-attention distribution, in which we use co-attention learning of aspect embedding sequence $\{\mathbf{a_1}, \mathbf{a_2}, \cdots, \mathbf{a_N}\}$ and the cast embedding sequence $\{\mathbf{c_1}, \mathbf{c_2}, \cdots, \mathbf{c_M}\}$.

Parallel co-attention mechanism attends to the aspect embedding and cast embedding simultaneously. First, we juxtapose the aspect embedding sequences $\{\mathbf{a_1}, \mathbf{a_2}, \cdots, \mathbf{a_N}\}$ to obtain an $L \times N$ matrix P, and juxtapose the sequences $\{\mathbf{c_1}, \mathbf{c_2}, \cdots, \mathbf{c_M}\}$ to obtain an $L \times M$ matrix Q. Here L denotes the length of embedding vectors, N and M are the length of aspect embedding sequence and cast embedding sequence, respectively.

Then, by introducing a learnable attention matrix $A \in R^{L\times L}$, we can calculate the correlation matrix $F \in R^{N\times M}$:

$$F = \tanh(P^T AQ) \tag{4.10}$$

Note that each element $F_{i,j}$ in the correlation matrix F represents a paired relation score between the two embedding vectors $\mathbf{a_i}$ and $\mathbf{c_j}$. Here tanh is an activation function that acts on each element of the matrix:

$$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{4.11}$$

After that, we perform pooling operations on the rows and columns of the correlation matrix F to generate importance vectors, which are called row-pooling and column-pooling, respectively. According to experience, the effect of mean-pooling is stronger than that of max-pooling, so we average-pool the rows and columns of the correlation matrix separately:

$$g_i^p = \text{mean}(F_{i,1}, \cdots, F_{i,M}) \tag{4.12}$$

$$g_i^q = \text{mean}(F_{1,i}, \cdots, F_{N,i}) \tag{4.13}$$

Thus we get the importance vectors of P and Q:

$$g^p = [g_1^p, \cdots, g_N^p]^T \tag{4.14}$$

$$g^q = [g_1^q, \cdots, g_M^q]^T \tag{4.15}$$

Subsequently, we use the softmax function to convert the importance vectors g^p and g^q into attention vectors $\mathbf{a^p}$ and $\mathbf{a^q}$. For example, the i-th element of $\mathbf{a^p}$ will be calculated by the following formula:

$$a_i^p = \frac{\exp(g_i^p)}{\sum_{j\in[1,N]} \exp(g_j^p)} \tag{4.16}$$

Finally, the embedding vectors $\mathbf{a_c}$ and $\mathbf{c_c}$ representing the co-attention distribution of the aspect level and the cast level can be calculated by the following formula:

$$\mathbf{a_c} = P\mathbf{a^p} \tag{4.17}$$

$$\mathbf{c_c} = Q\mathbf{a^q} \tag{4.18}$$

Alternating Co-attention Mechanism

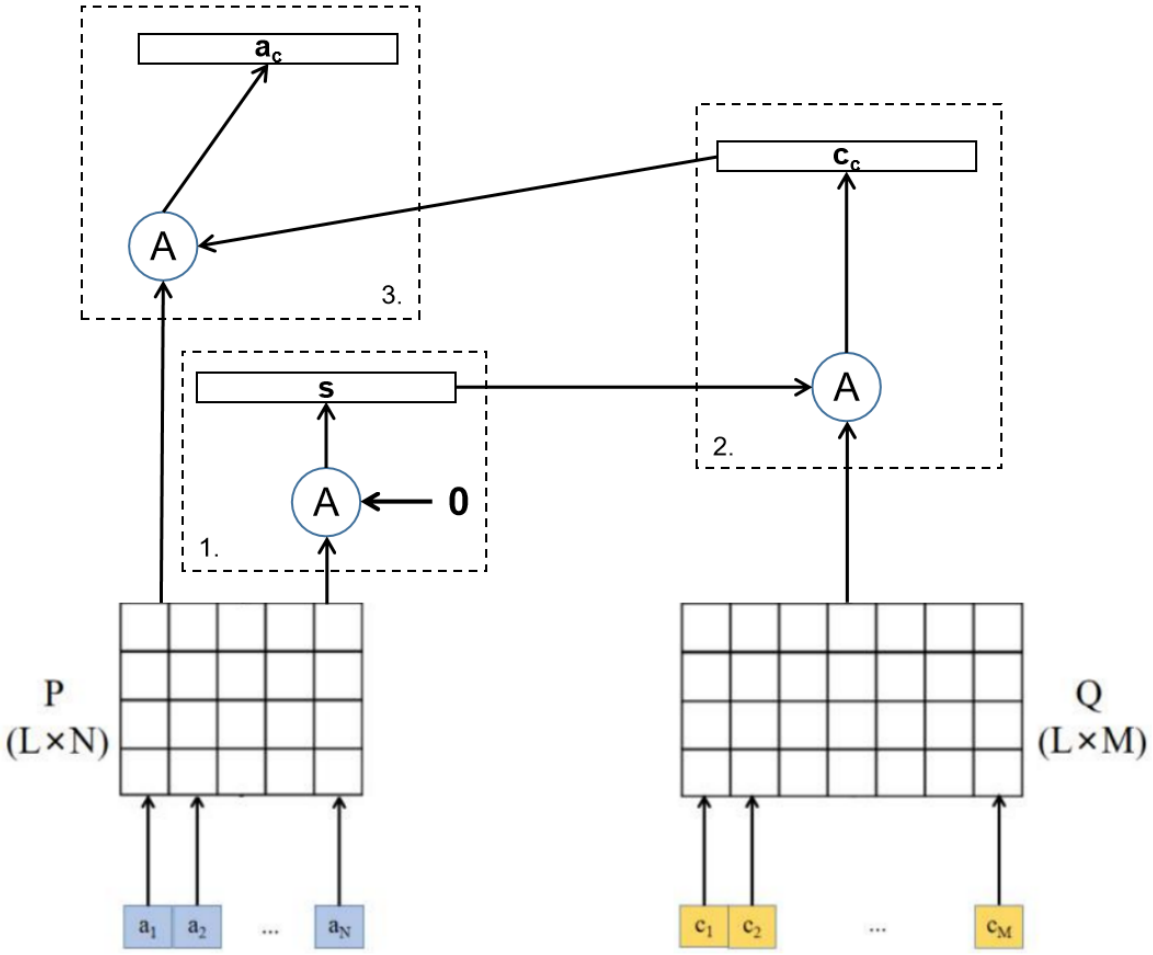

Fig. 4.3 Alternating co-attention mechanism

The same as parallel co-attention mechanism, the inputs of alternating co-attention mechanism are aspect embedding sequence $\{\mathbf{a_1}, \mathbf{a_2}, \cdots, \mathbf{a_N}\}$ and the cast embedding sequence $\{\mathbf{c_1}, \mathbf{c_2}, \cdots, \mathbf{c_M}\}$. And the outputs we expect are vector $\mathbf{a_c}$ representing the aspect-level co-attention distribution and vector $\mathbf{c_c}$ representing the cast-level co-attention distribution.

In alternating co-attention mechanism, we sequentially alternate between generating aspect and cast attention. The process is shown in figure 4.3. First, we also juxtapose the aspect embedding sequences $\{\mathbf{a_1}, \mathbf{a_2}, \cdots, \mathbf{a_N}\}$ to obtain an $L \times N$ matrix P, and juxtapose

the sequences $\{\mathbf{c_1}, \mathbf{c_2}, \cdots, \mathbf{c_M}\}$ to obtain an $L \times M$ matrix Q. Here L denotes the length of embedding vectors, N and M are the length of aspect embedding sequence and cast embedding sequence, respectively. Then there are three steps:1) summarize aspect matrix into a single vector;2) attend to the cast matrix based on the aspect matrix summary; 3) attend to the aspect matrix back based on the attended cast feature.

Concretely, we can define an attention operation $\mathbf{x} = A(\mathbf{X}; \mathbf{g})$, which takes the aspect(or cast) features $\mathbf{X}$ and attention guidance g which is derived from cast(or aspect) features as inputs, and outputs the attended aspect(or cast) vector. The operation can be expressed in the following steps:

$$\mathbf{H} = \tanh(\mathbf{W_x X} + (\mathbf{W_g g} \mathbb{1}^T)) \tag{4.19}$$

$$a^x = \text{softmax}(w_{hx}^T \mathbf{H}) \tag{4.20}$$

$$\mathbf{x} = \sum a_i^x \mathbf{x_i} \tag{4.21}$$

where $\mathbb{1}$ is a vector in which all elements are 1. $\mathbf{W_x}, \mathbf{W_g} \in R^{k \times L}$ and $w_{hx} \in R^k$ are parameters. In our experiments, we set k to 20. a^x is the attention weight of feature $\mathbf{X}$.

The process of alternating co-attention is illustrated in figure 4.3. At the first step, $\mathbf{X} = \mathbf{P}$ and $\mathbf{g} = \mathbf{0}$; Then at the second step, $\mathbf{X} = \mathbf{Q}$ and $\mathbf{g} = \mathbf{s}$ where $\mathbf{s}$ is the intermediate attended aspect feature from the first step; Finally at the third step, $\mathbf{X} = \mathbf{P}$ and $\mathbf{g} = \mathbf{c_c}$ where $\mathbf{c_c}$ is the final cast embedding vector which is from the second step. This means that we use cast embedding vector $\mathbf{c_c}$ as guidance to attend the aspect matrix again.

In this figure, we calculate cast embedding vector $\mathbf{c_c}$ first and then use it as guidance to attend the aspect matrix. Actually it can be reversed, we just need to exchange P and Q and then do the same process. We use both two kinds of order in our experiments.

4.3.3 Output of Module

At the top of the module, we input two embedding vectors representing aspect-level global attention distribution and co-attention distribution into a fully connected layer to

obtain an output vector that comprehensively represents aspect-level attention distribution:

$$h_u^a = \tanh([\mathbf{a_g}, \mathbf{a_c}]\mathbf{W^a} + \mathbf{b^a}) \tag{4.22}$$

Here tanh is the activation function, see equation (4.11). $\mathbf{W^a}$ and $\mathbf{b^a}$ are parameters of this fully connected layer. This vector h_u^a is the final output of the aspect module.

4.4 Cast Module

In the cast module, the input we get is a list containing all the cast numbers of a movie, see equation (4.3). First, we convert each actor number c_i into an embedding vector $\mathbf{c_i} \in R^L$, L represents the length of the embedding vector. In our experiments, we randomly initialize the embedding vectors and set them trainable. Later, we will use these embedding vectors to calculate the global attention distribution and co-attention distribution of the cast module.

Since our model structure is symmetrical, the calculation process of the cast module is similar to the aspect module. So here we only give a brief introduction.

We define the cast-level global attention filter as $g^c \in R^L$, then the weight of global attention distribution can be calculated:

$$\vec{s}_i^c = \text{softmax}[\text{isr}^{\alpha=4}(g^a \mathbf{c_i} + b^c)], i \in \{1, ..., M\} \tag{4.23}$$

Thus, it is possible to calculate the embedding vector representing the global attention distribution of the cast module:

$$\mathbf{c_g} = \sum_i \vec{s}_i^c \mathbf{c_i} \tag{4.24}$$

The embedding vector $\mathbf{c_c}$, which represents the distribution of co-attention at the cast level, is calculated by the formula (4.18) or (4.21).

At the top of the module, we input two embedding vectors representing cast-level global attention distribution and co-attention distribution into a fully connected layer to obtain an

output vector that comprehensively represents cast-level attention distribution:

$$h_u^c = \tanh([\mathbf{c_g}, \mathbf{c_c}]\mathbf{W^c} + \mathbf{b^c}) \tag{4.25}$$

Here tanh is the activation function, see equation (4.11). $\mathbf{W^c}$ and $\mathbf{b^c}$ are parameters of this fully connected layer. This vector h_u^c is the final output of the aspect module.

4.5 Output and Training of the Model

4.5.1 Output of the Model

When we obtain the embedding vector h_u^a which comprehensively represents the aspect-level attention distribution and the embedding vector h_u^c which comprehensively represents the cast-level attention distribution from the aspect module and the cast module, we connect them to form a joint multimodal movie embedding $[h_u^a, h_u^c]$. Subsequently, we define a user's movie-level filter $\mathbf{u}_u^m$ to calculate user u's attention score for movie m:

$$S_{m_u} = \mathbf{u}_u^{m^T} [h_u^a, h_u^c] \tag{4.26}$$

This rating is the output of the model. For a specific user u, movies will be sorted according to this rating, and the top N movies with the highest ratings are the top N recommendation results.

4.5.2 Training of the Model

In our data set, there is no clear data about whether a user likes or dislikes a certain movie, only the user's viewing records, in other words, we only have one-class data. Learning user preferences from such data is often seen as a ranking problem[22]. Given the user u, we can construct a pair of movies for comparison to specify the order of attractiveness, that is, for the movie $m_{u,i} \in M_u$ that the user has watched and the movie $m_{u,j} \notin M_u$ that the user has not watched, we judge that their attractiveness to user u must have this order: $m_{u,i} \geq m_{u,j}$.

Based on this assumption, we can use the following max-margin loss to optimize the ranking order between movie pairs:

$$L_{m_{u,i} \geq m_{u,j}} = \max(0, margin + S_{m_{u,j}} - S_{m_{u,i}}) \tag{4.27}$$

The *margin* here needs to be adjusted according to the data, and we set it to 20 during the training of our model.

Our model uses the user's viewing record as the positive input set during training, constructs a negative input set by randomly selecting movies that the user has not watched, and then uses the max-margin loss (4.27) as the loss function, uses the Adam[23] optimizer to train the model. It should be noted that, because we use the pre-trained GloVe[20] vector to initialize the embedding vector of aspect information, we set the embedding vector of aspect information to be non-trainable.

4.6 Summary

In this chapter, we introduced our aspect-based co-attention recommendation model based on multimodal data in detail. Our model is improved on the basis of MLAM[3]. Firstly, we change the input of the model, replace the original lengthy story text information with the extracted aspect information, and simplified the structure of the model; secondly, we consider the interaction between the two modality information and use co-attention to learn the implicit relationship between the aspect information and the cast information.

Our model is mainly composed of two modules: aspect module and cast module. In each module, we need to learn two different attention distributions: global attention distribution and co-attention distribution. The global attention distribution uses a user-independent global attractiveness filter, which is intended to protect the potentially useful information discarded by the co-attention; and the co-attention distribution is to learn implicit relationship between two modality information, that is, the attention distribution under the joint effect of aspect information and cast information. The two attention distributions are synthesized through

fully connected layers, and the resulting synthesis vector is used to calculate the scoring output of the entire model.

For training, we construct positive-negative movie pairs, then use max-margin loss as the loss function, use Adam[23] optimizer to train the model.

Chapter 5

Experiments

In the last two chapters, we have introduced the data set and model we used in detail. In this chapter, we will conduct a series of experiments to verify the effectiveness of our model. In order to compare with MLAM[3], we divide the data set into the same proportion as in [3]. We randomly select 20% of user viewing records as the test set, and the remaining 80% as the training set. To show the recommendation accuracy, for each watch record, we randomly generate 10 fake records to see whether our model will get higher score for true records.

5.1 Baselines

First, we conduct experiments on the same data set, and compare our model with the effect of MLAM[3]. At the same time, we will also compare with the baseline method of MLAM. Since, as we discussed in 1.1.2, the model of collaborative filtering often cannot solve the situation of rich content or cold start, the models to compare with do not involve the model of collaborative filtering. In this section we will introduce each model we will compare with one by one.

5.1.1 CENTROID

For each user, his user profile is generated from the center of the word embedding vector in the story information of all movies that the user has watched. Subsequently, ranking

recommendation is made based on the similarity between the center of the word embedding vector in each movie story information and the user profile[24].

5.1.2 CTR

Collaborative topic regression[25] performs user regression on the potential topic distribution of movie story learned from the Latent Dirichlet Allocation (LDA).

5.1.3 CWER

Similar to CTR, create collaborative word embedding user regression to regress the center of the word embedding vector of each movie story. The word embedding vectors of these movie stories are initialized by GloVe embedding.

5.1.4 MLAM

The Multi-Level Attraction Model[3], which we introduced in 2.1, includes story attraction module and cast attraction module.

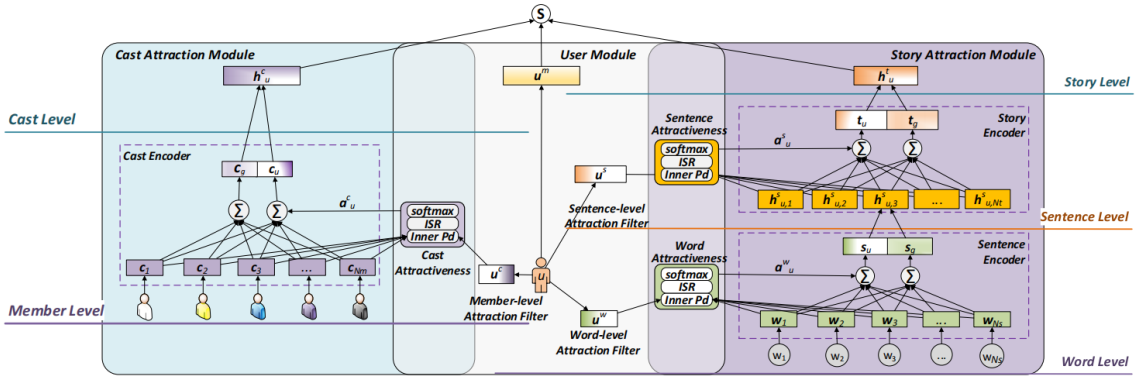

Fig. 5.1 MLAM[3] Left:MLAM-C Right: MLAM-S

5.1.5 MLAM-S

The version of MLAM in which only the story information is considered, that is, it only contains the MLAM story attraction module, and does not include the cast attraction module.

5.1.6 MLAM-C

The version of MLAM in which only the cast information is considered, that is, it only contains the MLAM cast attraction module, and does not include the story attraction module.

We conduct experiments on the same training set and test set, and compare the results of our model with the results of the above six models (given in [3]).

5.2 Evaluation Criteria

In order to evaluate the quality of the recommendation results, we use the following evaluation criteria:

5.2.1 P@K

Represents the average precision of all test users on the top K recommendation results, that is, the proportion of the correct results in the top K recommendation results to K.

$$P(K) = \frac{\sum_{k=1}^{K} true(k)}{K} \tag{5.1}$$

Here $true(k)$ means whether the k_{th} result is true, which should be 1 or 0.

5.2.2 R@K

Represents the average recall rate of all test users on the top K recommendation results, that is, the ratio of the number of correct results in the top K recommendation results to the total number of correct results(including recommended and non-recommended).

$$R(K) = \frac{\sum_{k=1}^{K} true(k)}{\#true} \tag{5.2}$$

Here $true(k)$ means whether the k_{th} result is true, which should be 1 or 0. $\#true$ means the total number of correct results.

5.2.3 MAP@K

Represents Mean Average Precision of all test users on the top K recommended results.

To explain the Mean Average Precision, we must first explain the Average Precision(AveP). As we all know, the two standards that are commonly used to measure predictive performance: precision and recall can only measure one aspect of performance. The ideal situation is that the values of these two standards are relatively high, but when we increase one of the values, the other value will inevitably be affected. In order to consider these two criteria, we regard the precision rate as a function of the recall rate, and then integrate the function over the recall rate R (from 0 to R(K)) to find the expected mean of P:

$$AveP = \int_0^{R(K)} P(r)dr = \sum_{k=1}^{K} P(k)\Delta(k) = \frac{\sum_{k=1}^{K}(P(k) \times true(k))}{min(\#true, K)} \tag{5.3}$$

Where true (k) indicates whether the k_{th} recommendation is a correct recommendation, or 1 if it is correct, otherwise 0. $P(k)$ represents the accuracy rate of the first k recommendation results, $\#true$ represents the total number of all correct recommendation results.

The *AveP* of each test user's recommendation list is calculated, and then averaged, that is, the Mean Average Precision of all test users on the top K recommendation results, called MAP@K.

5.2.4 MRR@K

Represents the Mean Reciprocal Rank of all test users on the top K recommendation results.

Here, the Reciprocal Rank refers to the value obtained by summing the reciprocal of the ranking of all correct recommendations in the top K recommendations of the recommendation list and dividing by the total number of all correct recommendation results:

$$RR = \frac{\sum_{k=1}^{K}(\frac{1}{k} \times true(k))}{\#true} \tag{5.4}$$

Among them, $true(k)$ indicates whether the k_{th} recommendation list is a correct recommendation, or 1 if it is correct, otherwise 0. $\#true$ indicates the total number of all correct recommendation results.

Then the Mean Reciprocal Rank MRR@K refers to the average value of the reciprocal ranking of each test user recommendation list.

5.2.5 NDCG@K

Represents the Normalized Discounted Cumulative Gain of all test users on the top K recommended results.

First, we introduce the concept of Cumulative Gain (CG). Cumulative gain is usually used in a personalized recommendation system for ranking recommendation based on ratings. Assuming that we recommend the top K items, we can calculate the cumulative gain of the recommendation list CG_K by the following formula:

$$CG_K = \sum_{i=1}^{K} score_i \tag{5.5}$$

$score_i$ represents the score of the i_{th} item. In our experiments, we set all correctly recommended scores to 1 and all incorrectly recommended scores to 0.

On the basis of the cumulative gain, we introduce consideration of the recommendation order, that is, we consider that the same rating should have different effects in different positions of the recommendation list, and the effect of a recommendation result at the end of the recommendation list should be greatly reduced. This resulted in a Discounted Cumulative Gain:

$$DCG_K = \sum_{i=1}^{K} \frac{2^{score_i} - 1}{log_2(i+1)} \tag{5.6}$$

The Discounted Cumulative Gain does not take into account the number of really valid (correct) results in the recommendation list, so we introduce a Normalized Discounted Cumulative Gain to standardize the Discounted Cumulative Gain:

$$NDCG_K = \frac{DCG_K}{IDCG_K} \tag{5.7}$$

Among them, IDCG (Ideal Discounted Cumulative Gain) refers to the Discounted Cumulative Gain under perfect conditions, that is, the Discounted Cumulative Gain calculated by ranking all the correct results before the incorrect ones.

5.3 Results

In this section, we will compare our model with the baselines in 5.1. We use the same data set and divide it into the same proportion as in [3]. So we just take the results given in their paper and do comparison with our experiment result.

As we talked in 4.3.2, there are two kinds of co-attention mechanism. Actually there are also two choices in alternating co-attention mechanism: do attention operation on aspect matrix first, or do attention operation on cast matrix first. So there are three kinds of our model:

Our Model-P

Our model using paralleled co-attention mechanism.

Our Model-Aa

Our model using alternating co-attention mechanism and doing attention operation on aspect matrix first.

Our Model-Ac

Our model using alternating co-attention mechanism and doing attention operation on cast matrix first.

Table 5.1 gives the recommendation results with MAP and MRR as the evaluation criteria. The results of the first six models are derived from the experimental results given in [3].

We can see that our models all show better results than MLAM, while far surpassing other baseline models. We see that the result of CTR is the worst. The reason is that CTR

Table 5.1 Experiment results (compared with MLAM and its baselines)

Model	MAP@5	MAP@20	MRR@5	MRR@20
CENTROID(2016)	0.1738	0.1481	0.0763	0.0958
CTR(2011)	0.1226	0.1069	0.0514	0.0692
CWER(2018)	0.1666	0.1580	0.0798	0.1089
MLAM-C(2018)	0.4243	0.3963	0.2118	0.2398
MLAM-S(2018)	0.3816	0.3451	0.1822	0.2093
MLAM(2018)	0.4252	0.3997	0.2187	0.2464
Our Model-P	**0.4562**	**0.4235**	**0.2349**	**0.2624**
Our Model-Aa	0.4437	0.4148	0.2274	0.2551
Our Model-Ac	0.4465	0.4212	0.2332	0.2605

determines the user profile of the user through the topic distribution of the movie story text information, but in fact there are many worthless and non-informative words in the movie story text. They will affect the topic distribution of the whole story. CENTROID and CWER directly convert the words into embedding vectors and find the user's user profile by taking the center and other methods. There may be some larger elements in the word embedding vector that can reflect the importance difference between different words. Therefore, the results of these two models are better than CTR, but they are still affected by the non-informative words, and they cannot achieve good results. MLAM-S models the user's attention to text information by establishing multiple levels of attention models, and uses the method of attention distribution to solve the effect of non-informative words to a certain extent, and the results of MLAM-C also reflects the value of the cast information. The two multi-modal models MLAM and our model show far better performance than other models, which shows the feasibility of using multi-modal data for recommendation. MLAM deals with story text information and cast information separately while our model takes relationship between different modality information into consideration, so we get better results than MLAM.

Among our three models, Our Model-Aa and Our Model-Ac get close results while Our Model-P performs better than them. We think this is because aspect information and cast information have the same importance in our model. As a result, use which one as guidance first will not make great difference.

Note that our input-the vocabulary of aspect information is only 445, which is two orders of magnitude less than the text information vocabulary of 22582 in MLAM, and the average number of aspect words per movie is only 12.6, which is also much less than the average 10.2 sentences of MLAM. On this basis, our model can still outperform MLAM, which indicates that what we extracted in aspect extraction is indeed the keywords in the text information of the movie story, eliminating a lot of useless information. At the same time, due to the reduced input and simplified network structure of our model, our training speed is much faster than MLAM.

5.4 Model Variant Experiments

In our model, we calculate two attention distributions in each module: global attention distribution and co-attention distribution. But in [3], there is another attention distribution-user attention distribution. They consider that each user has different preferences, not only reflected in the preference of the movie, but also reflected in the attention to the information. They model the user's attention to information by learning the user attraction filter and calculating the user attention distribution. However, we do not use user attention distribution in our model.

Actually, we did lots of experiments to make the decision. Firstly we will introduce some variants of our model.

Our Model(G+C+U)

In addition to calculating global attention distribution and co-attention distribution, this model also calculates user attention distribution and concatenate them in each module. Figure 5.2 shows the structure.

Our Model(G+U)

This model replaces co-attention distribution with user attention distribution in each module.

Our Model(C+U)

This model replaces global attention distribution with user attention distribution in each module.

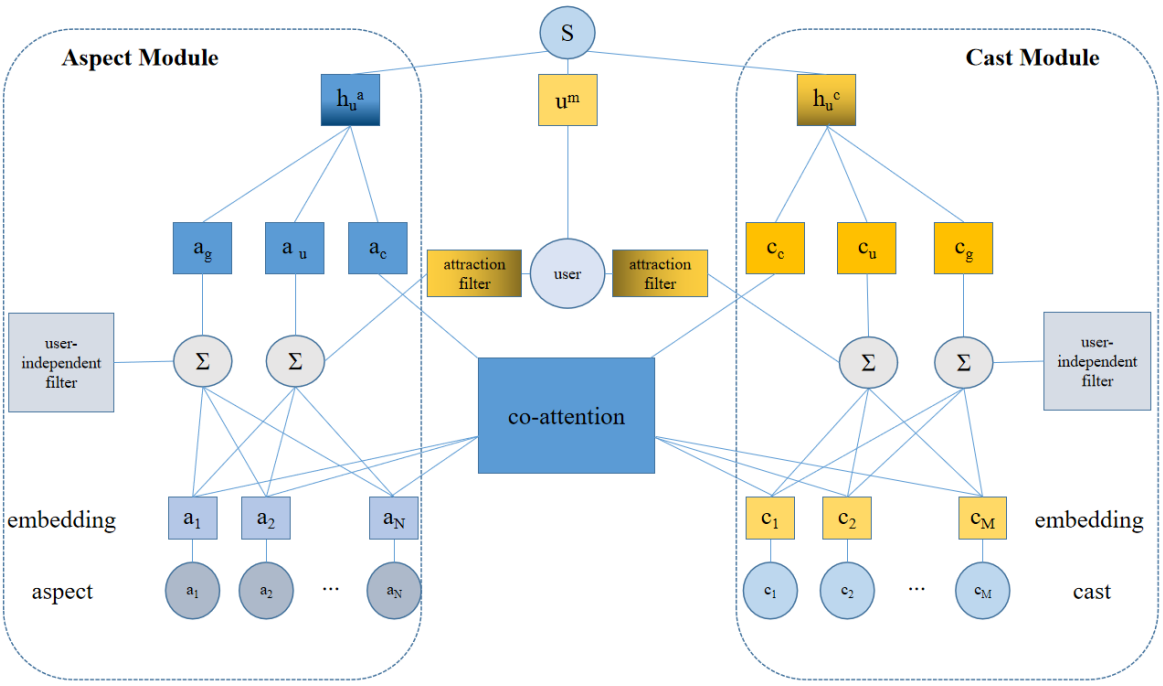

Fig. 5.2 Our Model(G+C+U)

Note that in this section when we talk about co-attention, we use parallel co-attention mechanism.

We use the same data set to do experiments and taking MAP, MRR and NDCG as evaluation criteria, Table 5.2 shows our experiment results.

Table 5.2 Experiment results (compared with variants)

Model	MAP@5	MAP@20	MRR@5	MRR@20	NDCG@5	NDCG@20
Our Model(G+C+U)	0.4278	0.3967	0.2167	0.2437	0.5569	0.5842
Our Model(G+U)	0.4258	0.3970	0.2157	0.2433	0.5551	0.5848
Our Model(C+U)	0.4242	0.3935	0.2144	0.2418	0.5537	0.5819
Our Model	**0.4562**	**0.4235**	**0.2349**	**0.2624**	**0.5804**	**0.6091**

We can see when we add user attention distribution into our model, the recommendation performance gets worse. We think that this result is because our aspect information is key-

words extracted from the story text, and each aspect word has a high amount of information and value. Therefore, the learning of the user attention distribution may cause only a part of the key information to be retained, and more key information is omitted, thereby make the result worse. As a result, we only use global attention distribution and co-attention distribution in our model.

5.5 Visualization of Co-Attention

In the research of recommender systems, interpreting recommendation results is an important and difficult task. In this section, we will visualize co-attention to try to interpret the recommendation results. Here we choose parallel co-attention mechanism to visualize, because it shows better performance in our experiments.

As we introduced in 4.3.2, co-attention utilizes aspect embedding sequence and cast embedding sequence, introduces a learnable attention matrix A to calculate the correlation matrix F, each element in the correlation matrix represents a correlation score between a pair of aspect embedding and cast embedding. The high correlation score indicates that this pair of embedding vectors have a mutually reinforcing effect, which is the embodiment of the "good at" information we mentioned earlier on the co-attention.

We extract the embedding vectors $c_1, ..., c_M$ and attention matrix A after training from the model, and obtain the embedding vectors $a_1, ..., a_N$ from the pre-trained GloVe vector. We can easily calculate the correlation score $R_{i,j}$ between the pair of aspect embedding vector $\mathbf{a_i}$ and the cast embedding vector $\mathbf{c_j}$ without going through the tanh activation function by the following formula:

$$R_{i,j} = \mathbf{a_i} A \mathbf{c_j}^T \tag{5.8}$$

Since we are only concerned about the size of the correlation score here, and tanh is a monotonic function, the correlation scores involved in this section are all correlation scores that have not passed through the tanh activation function.

We calculate the correlation score between all cast members and all words in Glove, and generate the gray scale image, which is shown in Figure 5.3. Vertical axis is words

and horizontal axis is cast members. However we can not find out much useful information except that there is a big difference between correlation scores. So we also do some more visualization of specific aspect words or movies.

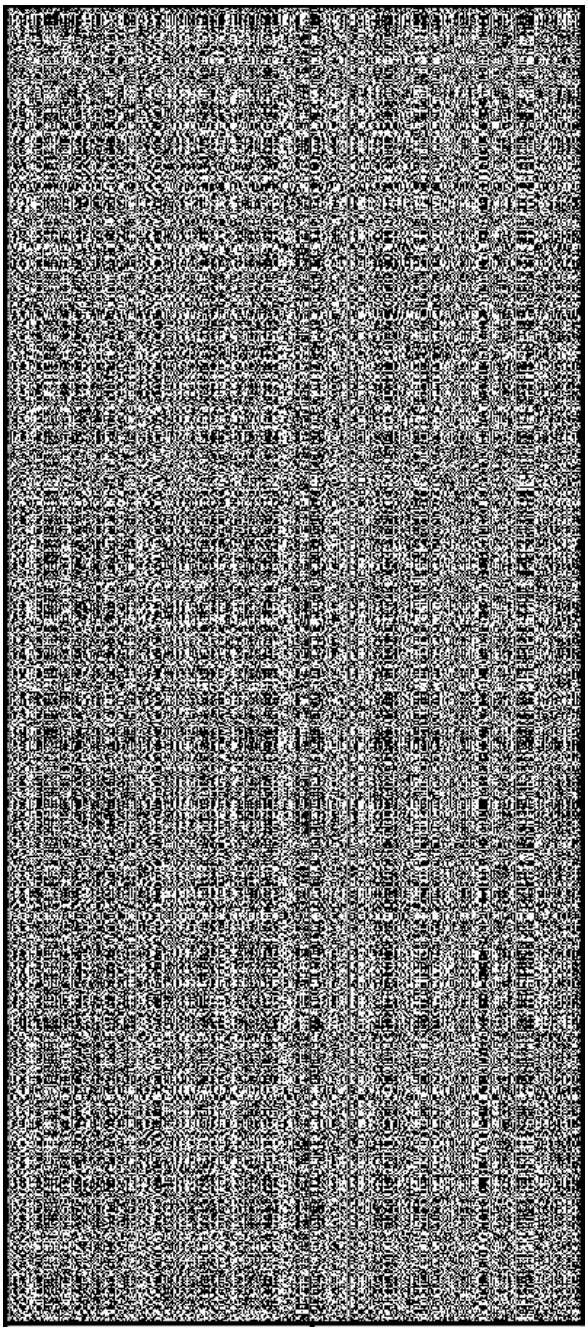

Fig. 5.3 Gray scale image of correlation score

Firstly we choose one typical aspect word "thriller", and calculate the relation score between its embedding and all cast embeddings.

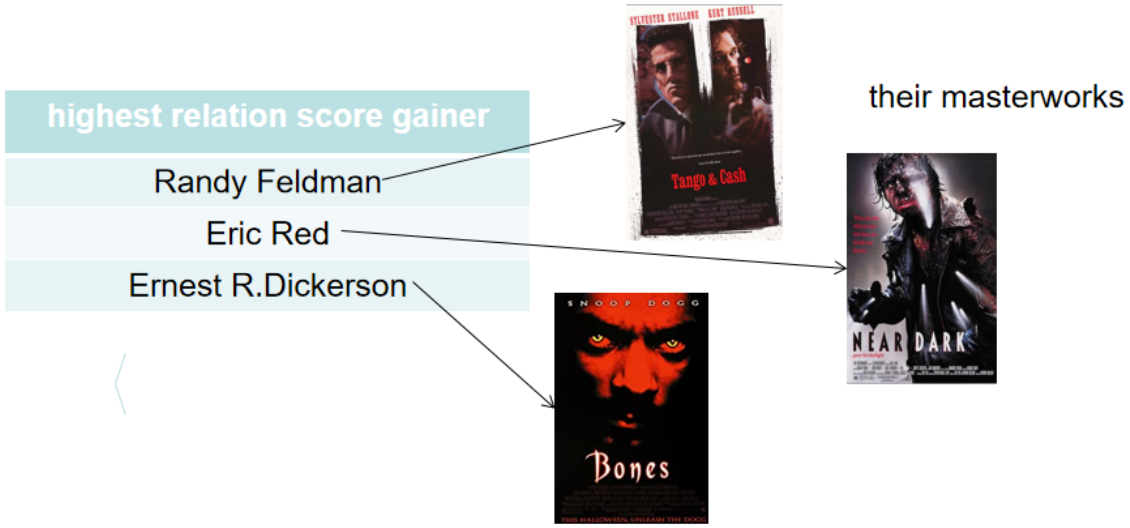

Fig. 5.4 Highest relation score gainer with aspect word "thriller"

Figure 5.4 shows the highest relation score gainer and their masterworks. We may not be familiar with the names of the cast members in the table, so we searched their information on the Internet. Randy Feldman is a producer and writer, known for Tango & Cash (1989), Metro (1997) and The Reading Room (2005). Among them, Tango & Cash (1989) and Metro (1997) are two comedy thriller films. Eric Red is an American screenwriter and director, the most famous works are the horror movies The Hitcher(1986) and Near Dark(1987). Ernest R. Dickerson is an American director, famous for the thrillers Juice(1992), Demon Knight(1995), Bones(2001) and Never Die Alone(2004). Most of the films he directed are also thrillers.

To further prove the value of co-attention, we choose one movie Metro (1997) to calculate importance vector g^p and g^q given in equation (4.14) and (4.15), which shows the average correlation value of aspect words and cast members.

The visualization is shown in Figure 5.5, the deeper color of font denotes the larger value in importance vector, and the larger weight they get in co-attention mechanism.

story	Metro is a 1997 American action comedy thriller film which was directed by Thomas Carter, produced by Roger Birnbaum, and starring Eddie Murphy as Scott Roper, a hostage negotiator and inspector for the San Francisco Police Department who immediately seeks revenge against a psychotic jewel thief, Michael Korda (Michael Wincott), who murdered Roper's best friend, Lt. Sam Baffert (Art Evans). Released on January 17, 1997 in the United States, Metro grossed $32,000,301 in the domestic market, which failed to bring back its $55,000,000 budget.
aspect	**American**, **action**, **comedy**, **thriller**, film, directed, best, release, United States, gross, budget
cast	Denis Arndt, Michael Wincott, Eddie Murphy, Michael Rapaport, **Randy Feldman**, Thomas Carter, Carmen Ejogo

Fig. 5.5 Visualization of importance vector in co-attention of movie Metro(1997)

We can see our co-attention mechanism catches the key information well, such as the movie type "comedy", "thriller" and the nationality "American". It also unearthed the inner connection between cast member "Randy Feldman" and aspect words "comedy" and "thriller".

From the above results, we can see that co-attention can indeed learn the implicit relationship between cast members and aspect information, that is, to learn such information that a certain cast member is "good at" a certain aspect. And we can also use co-attention to explain the recommendation results.

5.6 Summary

In this chapter, we first introduced our training set&test set, some models we compared with and our evaluation criteria, and then we conducted a series of experiments to explore the effectiveness of our model. First we do experiments on the same data set as [3], and compared the recommendation result of our model with baselines. The results show that our model is superior to MLAM and far ahead of other baseline algorithms. At the same time, due to the reduced input and simplified network structure of our model, the training speed is much faster than MLAM.

Then we consider adding user attention distribution to our model and do model variant experiments. The results show that adding user attention distribution makes recommendation performance worse.

Finally, we performed a visual analysis of co-attention to try to explain the recommendation results. The results show that co-attention can indeed learn the implicit relationship between the cast information and aspect information.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this paper, we build a new aspect-level recommendation model based on the multi-modal data of the movie. The innovation of this article lies in the following two points: First, we apply the aspect extraction usually used for user comment text to the story text of the movie, and use the extracted aspect information to establish an aspect level recommendation model based on multimodal data; second we use the co-attention distribution to learn the implicit correlation between the two modality information, which are aspect information and cast information in our data set.

Our model is mainly composed of two modules: aspect module and cast module. In each module, we need to learn two different attention distributions: global attention distribution and co-attention distribution. The global attention distribution uses a user-independent global attractiveness filter, which is intended to protect the potentially useful information discarded by the co-attention; and the co-attention distribution is to learn implicit relationship between two modality information, that is, the attention distribution under the joint effect of aspect information and cast information. The two attention distributions are synthesized through fully connected layers, and the resulting synthesis vector is used to calculate the scoring output of the entire model.

For comparison, we use the same data set as our baseline MLAM[3] and on this basis we do our own data preprocessing-replace the original lengthy story text information with the extracted aspect information. For training, we construct positive-negative movie pairs, then use max-margin loss as the loss function, use Adam[23] optimizer to train the model. For testing, we divide the data set into the same proportion as in [3]. For each watch record, we randomly generate 10 fake records to see whether our model will get higher score for true records.

The results of comparison experiments show that our model is superior to MLAM and far ahead of other baseline algorithms. At the same time, due to the reduced input and simplified network structure of our model, the training speed is much faster than MLAM. We also performed a visual analysis of co-attention to try to explain the recommendation results. The results show that co-attention can indeed learn the implicit relationship between the cast information and aspect information.

6.2 Future Work

Our model still has some shortcomings. First of all, the aspect information we extracted appears in the form of feature opinion pairs, but when we filter the plot text information, we simply split them into words for filtering, and do not consider the difference between feature words and opinion words. As a result, the data set may be mixed with some words that do not have aspect information in the original text. In addition, our aspect embedding vectors are directly initialized with pre-trained GloVe word embedding vectors and then set to be untrainable, which may cause our aspect embedding vectors to not represent aspect information well.

Our model still has some room for improvement and we still have some work to do. First, we can perform more detailed data preprocessing to filter out the aspect information more accurately; second, we can consider using a trainable embedding vector to represent the aspect information to obtain a word embedding vector that more closely matches the characteristics of the aspect information. In addition, we also need to verify the performance

of our model on other data sets to prove the universality of our model on various multi-modal data sets.

References

[1] Yongfeng Zhang, Qingyao Ai, Xu Chen, and W Bruce Croft. Joint representation learning for top-n recommendation with heterogeneous information sources. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1449–1458, 2017.

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

[3] Liang Hu, Songlei Jian, Longbing Cao, and Qingkui Chen. Interpretable recommendation via attraction modeling: Learning multilevel attractiveness over multimodal movie contents. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018*, pages 3400–3406, 2018.

[4] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. Hierarchical question-image co-attention for visual question answering. In *Advances in neural information processing systems*, pages 289–297, 2016.

[5] Yongfeng Zhang, Haochen Zhang, Min Zhang, Yiqun Liu, and Shaoping Ma. Do users rate or review? boost phrase-level sentiment labeling with review-level sentiment classification. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 1027–1030, 2014.

[6] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.

[7] Tao Mei, Bo Yang, Xian-Sheng Hua, and Shipeng Li. Contextual video recommendation by multimodal relevance and user feedback. *ACM Transactions on Information Systems (TOIS)*, 29(2):1–24, 2011.

[8] Michael J Pazzani. A framework for collaborative, content-based and demographic filtering. *Artificial intelligence review*, 13(5-6):393–408, 1999.

[9] Louis-Philippe Morency and Tadas Baltrušaitis. Multimodal machine learning: integrating language, vision and speech. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 3–5, 2017.

[10] Bing Liu and Lei Zhang. A survey of opinion mining and sentiment analysis. In *Mining Text Data*, pages 415–463. 2012.

[11] Marco De Gemmis, Pasquale Lops, Cataldo Musto, Fedelucio Narducci, and Giovanni Semeraro. Semantics-aware content-based recommender systems. In *Recommender Systems Handbook*, pages 119–159. Springer, 2015.

[12] Sean M McNee, John Riedl, and Joseph A Konstan. Being accurate is not enough: how accuracy metrics have hurt recommender systems. In *CHI'06 extended abstracts on Human factors in computing systems*, pages 1097–1101, 2006.

[13] Liang Hu, Longbing Cao, Shoujin Wang, Guandong Xu, Jian Cao, and Zhiping Gu. Diversifying personalized recommendation with user-session context. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017*, pages 1858–1864, 2017.

[14] Tommaso Di Noia, Vito Claudio Ostuni, Paolo Tomeo, and Eugenio Di Sciascio. Sprank: Semantic path-based ranking for top-n recommendations using linked open data. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(1):1–34, 2016.

[15] Kevin J Shih, Saurabh Singh, and Derek Hoiem. Where to look: Focus regions for visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4613–4621, 2016.

[16] Caiming Xiong, Stephen Merity, and Richard Socher. Dynamic memory networks for visual and textual question answering. In *International conference on machine learning*, pages 2397–2406, 2016.

[17] Huijuan Xu and Kate Saenko. Ask, attend and answer: Exploring question-guided spatial attention for visual question answering. In *European Conference on Computer Vision*, pages 451–466. Springer, 2016.

[18] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015.

[19] 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 & development in information retrieval*, pages 83–92, 2014.

[20] 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)*, pages 1532–1543, 2014.

[21] Cunchao Tu, Han Liu, Zhiyuan Liu, and Maosong Sun. Cane: Context-aware network embedding for relation modeling. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1722–1731, 2017.

[22] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: bayesian personalized ranking from implicit feedback. In *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*, pages 452–461. AUAI Press, 2009.

[23] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[24] Cataldo Musto, Giovanni Semeraro, Marco de Gemmis, and Pasquale Lops. Learning word embeddings from wikipedia for content-based recommender systems. In *European Conference on Information Retrieval*, pages 729–734. Springer, 2016.

[25] 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*, pages 448–456, 2011.