Attention-driven Factor Model for Explainable Personalized Recommendation

Jingwu Chen[1,2], Fuzhen Zhuang[1,2], Xin Hong[1,2], Xiang Ao[1,2], Xing Xie[3], Qing He[1,2]
[1]Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS),
Institute of Computing Technology, CAS, Beijing 100190, China.
[2]University of Chinese Academy of Sciences, Beijing 100049, China.
[3]Microsoft Research Asia
{chenjingwu,zhuangfuzhen,heqing}@ict.ac.cn

ABSTRACT

Latent Factor Models (LFMs) based on Collaborative Filtering (CF) have been widely applied in many recommendation systems, due to their good performance of prediction accuracy. In addition to users' ratings, auxiliary information such as item features is often used to improve performance, especially when ratings are very sparse. To the best of our knowledge, most existing LFMs integrate different item features in the same way for all users. Nevertheless, the attention on different item attributes varies a lot from user to user. For personalized recommendation, it is valuable to know what feature of an item a user cares most about. Besides, the latent vectors used to represent users or items in LFMs have few explicit meanings, which makes it difficult to explain why an item is recommended to a specific user. In this work, we propose the Attention-driven Factor Model (AFM), which can not only integrate item features driven by users' attention but also help answer this "why". To estimate users' attention distributions on different item features, we propose the Gated Attention Units (GAUs) for AFM. The GAUs make it possible to let the latent factors "talk", by generating user attention distributions from user latent vectors. With users' attention distributions, we can tune the weights of item features for different users. Moreover, users' attention distributions can also serve as explanations for our recommendations. Experiments on several real-world datasets demonstrate the advantages of AFM (using GAUs) over competitive baseline algorithms on rating prediction.

KEYWORDS

Personalized Recommendation, Recommendation Explanation, Attention Distribution

ACM Reference Format:
Jingwu Chen[1,2], Fuzhen Zhuang[1,2], Xin Hong[1,2], Xiang Ao[1,2], Xing Xie[3], Qing He[1,2]. 2018. Attention-driven Factor Model for Explainable Personalized Recommendation. In *SIGIR '18: 41st Int'l ACM SIGIR Conference on Research & Development in Information Retrieval, July 8-12, 2018, Ann Arbor, MI, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3209978.3210083

1 INTRODUCTION

In recent years, personalized recommendation has attracted much attention from both research community and industry. By considering users' preferences, recommendation systems will have more chance to attract users. Many researchers have found that it's very beneficial for personalized recommendation systems to give explanations. Given reasonable explanations, users are more likely to buy or try. Furthermore, explanations will help convincing users that the system knows them very well and makes custom-made recommendations for them.

There have been plenty of techniques such as content based algorithms proposed to address this explainability problem. Besides, some review-aware methods [14] based on sentiment analysis have been newly proposed with good performance on some datasets. However, these methods heavily rely on both quality and quantity of users' reviews. Besides, there are some users unlikely to show personal preferences in their reviews, which makes it tough for review-aware methods to capture users' preferences.

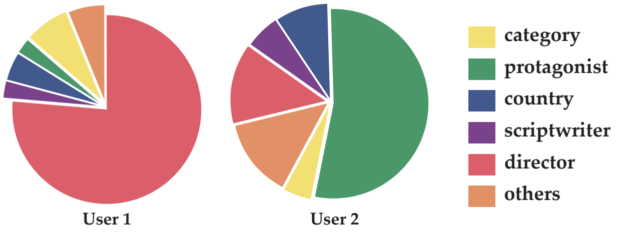

Figure 1: Two users' attention distributions on different item features generated by GAUs

Latent Factor Models such as Matrix Factorization (MF) [8] have become very popular and welcomed by the research community and industry. By representing features with latent vectors, it becomes very convenient to integrate various item features by adding or concatenating. Item features are usually easy to collect and very helpful, because users make decisions generally based on item features. With many advantages, LFMs reach high prediction accuracy on some benchmark datasets. However, LFMs also encounter some problems in personalized recommendation. On one hand, when conducting interactions between latent vectors of users and items, LFMs ignore how users make decisions according to their preference for various item features, which makes LFMs do it the same way for all users. On the other hand, due to the latent representation, the lack of explainability weakens the ability of a personalized recommendation system to gain users' trusts.

SIGIR'18, July 8–12, 2018, Ann Arbor, MI, USA
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5657-2/18/07…$15.00
https://doi.org/10.1145/3209978.3210083

To address the challenges above, we develop a novel CF-based model named Attention-driven Factor Model (AFM) as a general framework for personalized recommendation, which makes recommendations according to users' attention on different aspects of the item. For estimating the users' preferences via ratings and item features, we propose the Gated Attention Units (GAUs) for AFM to generate attention distributions for different users, as Figure 1 shows. With users' attention distributions, AFM can reach a high prediction accuracy and give feature-level explanations for users' preferences.

The main contributions of this work are summarized into three folds: 1) By considering users' preferences, we develop a general framework AFM for explainable personalized recommendation, which can give reasonable explanations for users' preferences and keep a high prediction accuracy. 2) We propose the Gated Attention Units(GAUs) to extract explicit users' preferences from latent representations. 3) We perform experiments on several real-world datasets to demonstrate the effectiveness and explainability of AFM.

2 RELATED WORK

Latent Factor Models (LFMs) using Matrix Factorization (MF) have been very popular, as they usually outperform other traditional models on many benchmark datasets. Some typical LFMs have been proposed for different problem settings, such as Singular Value Decomposition (SVD) [8], Non-negative Matrix Factorization (NMF) [9] and Probabilistic Matrix Factorization (PMF) [12]. Since these MF models learn solely from ratings, some other models which can incorporate auxiliary information have been proposed. Factorization Machine (FM) [10] and SVDFeature [3] are two of such models and very famous for their good performance.

As deep learning (DL) techniques have gained immense success on computer vision and natural language programming, lots of efforts have been made to introduce deep learning techniques into recommendation systems. Generally, DL has been applied to feature extraction [13] and prediction [5, 6]. Since attention mechanism has been proved quite effective in machine translation [1], attention-based recommendation models have been developed recently. Chen *et al.* [2] integrated LFM with Neighborhood Model based on item- and component-level attention, which models the implicit feedback in Multimedia Recommendation and performs well.

Apart from making prediction, there has been some researches on recommendation explanation. A popular way to generate explanation is by analyzing users' preferences from their reviews. Zhang *et at.* [14] extracted users' preferences from their reviews based on phrase-level sentiment analysis, which helps generating personalized recommendations together with explanations.

To sum up, how to generate explainable recommendations and keep a high prediction accuracy simultaneously has been one of the major research questions for personalized recommendation.

3 ATTENTION-DRIVEN FACTOR MODEL

In this section, we first present the structure of AFM and explain how AFM serves as a general framework for personalized recommendation. We then introduce the Gated Attention Units (GAUs) and how GAUs generate attention distributions for different users. Lastly, we would like to show how AFM gives reasonable explanations for users' preferences.

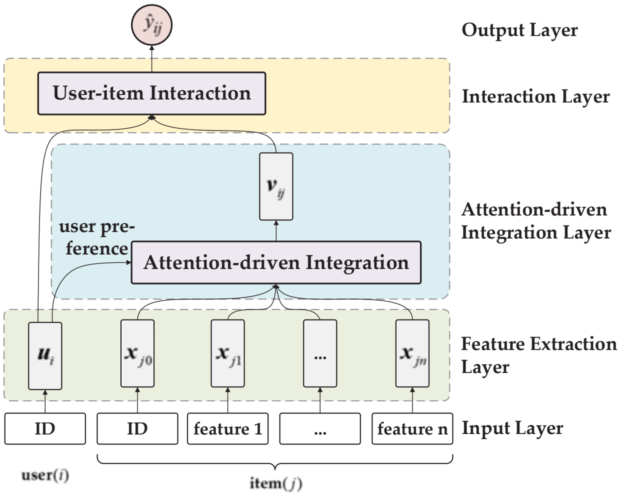

Figure 2: Attention-driven Factor Model

3.1 General Framework

Figure 2 shows the structure of AFM. For training AFM, we need user ratings and item features. Here, let m, n and k denote the number of users, the number of item features and the size of latent vectors respectively.

Feature Extraction Layer. The feature extraction layer is to extract latent representations in dense vector format from original inputs. According to different types of inputs, we can apply suitable extraction methods, which can be very flexible under many situations. For those categorical inputs (*i.e.*, user IDs and item IDs), we can use the embedding method. Besides, we can rescaled embedding vectors by their input values, in order to account for continuous valued inputs [10]. For those textual or pictorial inputs, we can employee deep learning techniques such as recurrent neural network (RNN) and convolution neural network (CNN) to generate latent representations. After feature extraction, we get $\boldsymbol{u}_i \in \mathbb{R}^k$ for the i-th user, and a set of feature vectors $\boldsymbol{\mathcal{X}}_j = \{\boldsymbol{x}_{j0}, \boldsymbol{x}_{j1}, \boldsymbol{x}_{j2}, \ldots, \boldsymbol{x}_{jn}\}$ for the j-th item, where $\boldsymbol{x}_{j0} \in \mathbb{R}^k$ denotes the embedding vector for the ID of the j-th item and $\boldsymbol{x}_{jl} \in \mathbb{R}^k$ ($l \in [1, n]$) denotes the latent vector for the l-th feature of the j-th item.

Attention-driven Integration Layer. The attention-driven integration layer is the core component of AFM. We formulate this layer as:

$$\boldsymbol{v}_{ij} = f_{AI}(\boldsymbol{u}_i, \boldsymbol{\mathcal{X}}_j), \tag{1}$$

where f_{AI} indicates the attention-driven integration transformation and $\boldsymbol{v}_{ij}$ denotes the integration vector of the j-th item driven by the i-th user's preference $\boldsymbol{u}_i$. Different from many other models which simply integrates $\boldsymbol{\mathcal{X}}_j$ by adding or concatenating, we introduce $\boldsymbol{u}_i$ implying the i-th user's preference to conduct the integration.

Inspired by LSTM units [7] and Gated Recurrent Units [4], we introduce the Gated Attention Units (GAUs) as a novel attention-driven integration method, as Figure 3 shows.

We calculate $\boldsymbol{v}_{ij}$ in two steps. First, we use GAUs to generate users' attention distribution as follow:

$$\boldsymbol{\alpha}_i = \text{softmax}(\boldsymbol{W}_a \boldsymbol{u}_i), \tag{2}$$

where $\boldsymbol{W}_a \in \mathbb{R}^{(n+1) \times k}$ denotes the attention mapping matrix and $\boldsymbol{\alpha}_i = \{\alpha_{i0}, \alpha_{i1}, \alpha_{i2}, \ldots, \alpha_{in}\}$ indicates the attention distribution of the i-th user. The attention mapping matrix is the key of GAUs and shared by all users, which can map the latent preference space to the feature-level attention space. If we directly use different $\boldsymbol{\alpha}$ for different users, the number of parameters will increase from $k(n+1)$ to $m(n+1)$, which may result in serious overfitting. The shared $\boldsymbol{W}_a$ can also help learning the latent user vector $\boldsymbol{u}$ during the training process. With the attention distribution $\boldsymbol{\alpha}$, we apply a weighted sum for $\boldsymbol{\mathcal{X}}$ to get $\boldsymbol{v}$:

$$\boldsymbol{v}_{ij} = \sum_{l=0}^{n} \alpha_{il} \boldsymbol{x}_{jl}. \tag{3}$$

From Equation (3), we can get to know some advantages of GAUs. Supposing we are making movie recommendation for a user, who cares most about the protagonist of a movie and pays little attention to other features. In this case, the GAUs can learn a high attention value to emphasize the protagonist feature, meanwhile weaken the influence of other features. It's worth to point out that the embedding vector $\boldsymbol{x}_{j0}$ of the j-th item is quite necessary. Since it is unlikely to collect all the features of the item in practical, the $\boldsymbol{x}_{j0}$ can serve as "others" for features which are not included in inputs.

Interaction Layer. The interaction layer performs the interaction between the latent user vector $\boldsymbol{u}$ and the latent item vector $\boldsymbol{v}$ to get the final prediction $\hat{y}$. There are plenty of interaction methods. As usually used in Matrix Factorization (MF), we can apply an inner product:

$$\hat{y}_{ij} = \boldsymbol{u_i}^T \boldsymbol{v}_{ij}. \tag{4}$$

Since the inner product simply combines the multiplication of latent vectors linearly, He *et al.* [6] proposed to leverage a multi-layer perceptron (MLP) to learn the user-item interaction function:

$$\hat{y}_{ij} = f_{MLP}(\boldsymbol{u}_i, \boldsymbol{v}_{ij}). \tag{5}$$

With non-linear activation function, we can enhance the ability of AFM to learn the non-linear interaction. For training, we can employ appropriate loss function for different tasks, which has nothing special compared to many other CF-based algorithms. Since AFM is a framework, the overall model size and complexity are decided by the methods adopted for each layer.

3.2 Personalized Explanation

Most of the existing factorization models succeed in prediction accuracy but lack of reasonable explanations. We argue that a recommendation system without a clear description of the user's preference is not personalized enough. With GAUs, AFM can learn the feature-level attention distributions of different users. On one hand, we can directly present the distribution diagrams to users, which helps increasing users' interests in our system and shows our great concern for users. On the other hand, we send them interesting messages while making recommendations, like "Dear *XXX*, we have noticed your great interest in [*feature*], so we strongly recommend these for you". Once obtaining the attention distributions of users, there are plenty of ways to give explanations for personalized recommendations and persuade users.

4 EXPERIMENTS

Since the key contributions of this work are on the general framework AFM and the GAUs, we conduct convincing experiments to show the performance on real-world datasets compared to some state-of-the-art models and demonstrate the explainability of the GAUs via case studies.

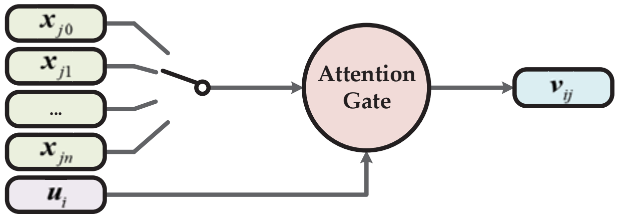

Figure 3: An illustration of the Gated Attention Unit. The attention gate decides whether a feature is masked according to the user's preference.

4.1 Experimental Settings

4.1.1 ***Datasets.*** We experiment with three datasets covering two domains, movie and music recommendations. The characteristics of the three datasets are summarized in Table 1. Ratings in all the three datasets are from 1 to 5.

Table 1: Statistics of the datasets

Dataset	User#	Item#	Rating#	Sparsity
MovieLens	6,040	3,952	1,000,209	95.81%
Douban Music	27,395	31,744	884,224	99.90%
Douban Movie	15,989	10,604	1,061,850	99.37%

MovieLens[1]**.** This is actually one of the most popular benchmarks for recommendation tasks. We use the version which contains about 1 million ratings. Besides user IDs and item IDs, we take the genres of movies as an item feature, which contains 18 different genres (*e.g.*, comedy and adventure, *etc*).

Douban[2]**.** This is a well-known social media website in China, where many people rate and comment on movies, music, books, *etc*. We use two datasets, *Douban Movie* and *Douban Music*, crawled from Douban. In *Douban Movie*, we take five features including director, scriptwriter, protagonist, genre and product country. In *Douban Music*, we only use the music genre as item feature. As you know, people are more likely to care about the musician or album information. Our purpose for using only music genre is to test the ability of models to deal with features which are probably not very effective.

4.1.2 ***Baselines.*** Since AFM uses auxiliary information of the item, we compare our proposed AFM (using GAUs as the attention-driven integration method) with following methods that can also take extra item features as inputs:

- LibFM [11]. This is the official implementation of Factorization Machine (FM) [10] released by Rendle.

- SVDFeature [3]. SVDFeature is a model for feature-based collaborative filtering. We use the official implementation in our experiments.

- CDL [13]. Collaborative Deep Learning (CDL) is a hierarchical Bayesian model using deep learning methods to extract latent

[1] http://grouplens.org/datasets/movielens/1m

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

Table 2: Performance of AFM compared to other baselines

	MovieLens		Douban Music		Douban Movie	
Model	MAE	RMSE	MAE	RMSE	MAE	RMSE
LibFM	0.698	0.880	0.533	0.665	0.576	0.735
SVDFeature	0.693	0.879	0.528	0.665	0.576	0.735
CDL	0.686	0.874	0.546	0.703	0.579	0.740
NFM	0.687	0.878	0.524	0.661	0.574	0.729
AFM	**0.676**	**0.858**	**0.511**	**0.649**	**0.549**	**0.697**

features, which can be easily extended to incorporate auxiliary information.

- **NFM [5].** Neural Factorization Machine (NFM) can not only model the second-order feature interactions as FM does but also model higher-order feature interactions by using the non-linearity of neural network.

4.1.3 ***Implementation Details.*** As all the features in our experiments are categorical, we transform them into one-hot inputs for all models. For NFM and AFM, we both use one interaction layer with sigmoid as activity function.

For each dataset, we conduct the five-fold cross-validation for training and testing. We adopt mean absolute error (MAE) and root-mean-square error (RMSE) as evaluation metrics. For a fair comparison, we learn all models by optimizing the square loss using SGD as optimizer and set 64 as the embedding size for all models.

4.2 Performance Comparison

Table 2 shows the performance of all models on the three datasets. First, we can see that our proposed AFM is quite comparable to other state-of-the-art models. On the first two datasets with only genre as the item feature, AFM slightly outperforms other baselines. It beats others by a large margin (about 4.4% over NFM) on *Douban Movie*. We conjecture with more item features as inputs AFM may be more effective. It is also worth to notice CDL, which uses item feature vector plus a bias as latent item vector, performs not that well on *Douban Music*. By checking the users' attention distribution generated by the GAUs, we find that there are nearly 80% users have less than 20% attention on the music genre, which means the music genre may be not a highly concerned feature for most users. This actually explains why performance of CDL on *Douban Music* is not as good as in other datasets. It also shows AFM can deal with features which are not very effective by masking them with low attention values.

4.3 Case Study

Here, we present some cases in datasets to highlight the performance of the GAUs. First, let's talk about two interesting users from *Douban Movie*. One is probably a big fan of Christopher Nolan (a world-famous director). About 40% movies he has watched are directed by Nolan, and for nearly 80% of them he rates 5. Another likes Donnie Yen and Jet Li (both are well-known Chinese KongFu stars) very much and rates high for their movies. We present their attention distributions generated by GAUs, just as Figure 1 shows ("User 1" for the fan of Nolan and "User 2" for the fan of Chinese Kongfu stars), which quite accords with their ratings.

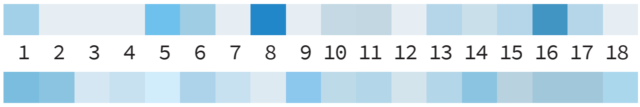

Figure 4: Distributions of two users' favourite movies (rated 5) on 18 different genres, where each number denotes a certain genre. The deeper the color, the greater the proportion.

We also pick two users from *MovieLens* and visualize the distributions of their favorite movies on the 18 genres, shown as Figure 4. The user above seems to prefer the 8-th and the 16-th genres (drama and thriller), while the user below seems to care much less about the genre compared to the one above. Calculated by GAUs, the user above pays 66.5% of attention on the genre, while the user below pays only 4.6%. This also conforms the fact very well.

5 CONCLUSION

In this work, we propose AFM as a general framework for personalized recommendation, which addresses concerns on both prediction accuracy and explainability. At its heart, we propose the Gated Attention Units (GAUs). The GAUs actually generate explicit attention distributions from latent user vectors. Experiments and case studies demonstrate the effectiveness of AFM and GAUs.

6 ACKNOWLEDGEMENTS

This research work is supported by the National Key Research and Development Program of China under Grant No. 2017YFB1002104, the National Natural Science Foundation of China under Grant No. 61773361, 61473273, 91546122, 61573335, 61602438, Guangdong provincial science and technology plan projects under Grant No. 2015 B010109005, and the Youth Innovation Promotion Association CAS 2017146.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. In *ICLR*.

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

[3] Tianqi Chen, Weinan Zhang, Qiuxia Lu, Kailong Chen, Zhao Zheng, and Yong Yu. 2012. SVDFeature: a toolkit for feature-based collaborative filtering. *JMLR* (2012).

[4] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*.

[5] Xiangnan He and Tat-Seng Chua. 2017. Neural factorization machines for sparse predictive analytics. In *SIGIR*.

[6] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW*.

[7] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* (1997).

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

[9] Daniel D Lee and H Sebastian Seung. 2001. Algorithms for non-negative matrix factorization. In *NIPS*.

[10] Steffen Rendle. 2010. Factorization machines. In *ICDM*.

[11] Steffen Rendle. 2012. Factorization Machines with libFM. *ACM TIST* (2012).

[12] Ruslan Salakhutdinov and Andriy Mnih. 2008. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *ICML*.

[13] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative deep learning for recommender systems. In *KDD*.

[14] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *SIGIR*.