Explainable Recommendation via Interpretable Feature Mapping

Anonymous

ABSTRACT

Latent factor collaborative filtering (CF) has been one of the most widely used techniques for recommender systems by learning the latent representations of users and items. Recently, explainable recommendation has attracted much attention from research community. However, trade-off exists between explainability and performance of the recommendations. To overcome this issue, we present a novel feature mapping approach that maps the uninterpretable latent features onto the interpretable aspect features, achieving both satisfactory performance and explainability in the recommendations by simultaneous minimization of recommendation and user preference losses. Experimental results demonstrate the state-of-the-art recommendation performance as well as the capability of explaining recommendations.

KEYWORDS

Preference Learning, Explainable AI, Matrix Factorization, Attention, Multi-task Learning

ACM Reference Format:
Anonymous. 2019. Explainable Recommendation via Interpretable Feature Mapping. In *RecSys '19: 13th ACM Conference on Recommender Systems, 16th-20th September 2019, Copenhagen, Denmark.* ACM, New York, NY, USA, 5 pages. https://doi.org/xxx

1 INTRODUCTION

Since the inception of Netflix Prize competition, latent factor collaborative filtering (CF) has been continuously adopted by various recommendation tasks due to its strong performance over other methods [12]. Essentially it employs a latent factor model, such as, matrix factorization[12] and/or neural networks[8], to learn user or item feature representations for rendering recommendations. Despite the great success, latent factor CF approaches often suffer from lack of interpretability[23] as other latent factor models. In a contemporary recommender system, explaining why a user likes an item can be as important as the recommendation itself. Research has shown that by providing proper explanations, users are more likely to accept the recommendations [10][20].

To make intuitive explanations for recommendations, recent efforts have been focused on using auxiliary information such as user defined tags[21] and topics from user's review texts[24][15][14][3] to illuminate users preferences. Other works such as [7][24][16][11] use aspect to explain recommendations, where the *aspect* is defined as a set of words that describes the characteristics of an item [11]. However, the interpretable features underlying the explainable recommendations have been sparsely studied thus are poorly understood.

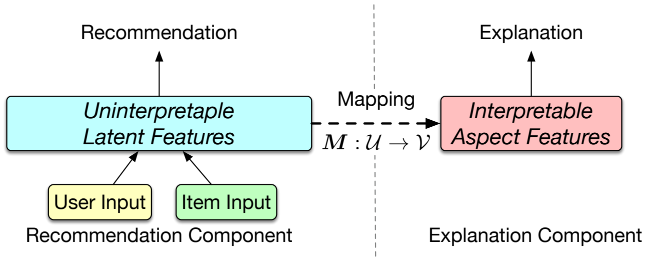
Recommendation
Explanation
Mapping
Unintepretaple Latent Features
$M : \mathcal{U} \rightarrow \mathcal{V}$
Interpretable Aspect Features
User Input
Item Input
Recommendation Component
Explanation Component

Figure 1: An illustration of interpretable feature mapping

Here we propose a novel feature mapping strategy that enjoys advantages of performance in latent factor models as well as explainability via interpretable features. As shown in Figure 1, we use a latent factor CF model (left) for good recommendation performance whereas recommendation explainability is achieved by mapping the uninterpretable latent features to interpretable aspect features (right). We implement this mapping strategy by designing an attentive multi-task learning approach.

Specifically, we first find the basis vectors of interpretable aspect features to span an interpretable aspect space, then we project the uninterpretable latent features from the source model, e.g., a latent factor model, to such an interpretable aspect space. The resulting new features in the interpretable space are then used to render accurate yet explainable recommendations. We formulate the problem as: 1) how to find the interpretable aspect basis; 2) how to perform the interpretable feature mapping, and we describe our approach in detail in Section 3.

The main contributions of our work are: 1) We propose a novel feature mapping approach to map the complex uninterpretable features to interpretable aspect features, enabling explainable recommendations; 2) Borrowing strength across aspects, our model is capable of breaking the trade-off between recommendation performance and explainability with minimum auxiliary information; 3) Real-time explainable recommendation achieved by real-time feature mapping.

2 RELATED WORK

Although latent factor CF demonstrates good prediction performance, the learned latent features are in general uninterpretable. To mitigate this problem, auxiliary information has been incorporated from various sources to increase explainability.

An array of approaches have been developed to render explainable recommendations at aspect level [7][24][16][11][3]. Aspects can be viewed as explicit properties of an item. Formally, we define aspects in the following way[2]:

Definition 1. ***(Aspect)*** *An aspect* $\mathbf{s}$ *is an attribute that characterizes an item. Assuming there are totally* m *aspects, an item can be described by* $\mathcal{I}_i = \{s_{i_1}, s_{i_2}, ..., s_{i_k}\}$, $k \leq m$, *if an item is of aspect* $s_{i_1}, ..., s_{i_k}$ *simultaneously. We say an item* i *is of aspect* s_j *if* $s_j \in \mathcal{I}_i$.

The goal of defining aspects is to describe each item by interpretable aspects and explain user's preferences from these aspects.

In learning user's preferences on different aspects, [7] creates a tripartite graph that constructed by triangles with user-item-aspect triplets, and fits the graph with constraints to find the relationship between users and aspects. [11] imposes a constraint on factorizing user-aspect matrix and item-aspect matrix with the predefined aspects. [24] proposes an explicit factor model that integrates explicit features (i.e. aspects) and implicit features to render explainable recommendations.

Although the above mentioned CF approaches indeed learn user's preference for recommendation, most of them regularize the recommendation with user-aspect and item-aspect relations [24][11][7] extracted from review texts using a third-party tool such as Latent Dirichlet Allocation (LDA). Albeit useful, these explainable approaches can fail when the content-based auxiliary information is not available. And even if the content information is available, the extra processing step to extract the aspects of an item can be a significant bottleneck for rendering real-time explainable recommendation.

Here we define the aspects directly using categorical information of an item and systematically optimize the recommendation performance and explainability using aspects. For example, we use movie genre (e.g., romance, thriller, suspension), or merchandise category (e.g., electronic, furniture, diary) as aspects. By designing an attentive multi-task collaborative filtering (AMCF) approach, we infer a set of aspect features representing the basis of an aspect space, followed by mapping the uninterpretable latent features onto the interpretable aspect space. The advantage of the AMCF approach is to learn user's preferences by a novel interpretable feature mapping without using content-based auxiliary information. Consequently it renders real-time accurate recommendations as other CF approaches yet with clear explanations.

3 THE PROPOSED MODEL

In this section, we first describe motivations and underlying assumptions of our work. Then we propose a general framework for interpretable aspect feature mapping, with a concrete implementation (AMCF) that extracts interpretable aspect features via neural multi-task learning, and projects to the aspect space using an attention mechanism. Finally, we train the feature mapping model by minimizing a new loss function to learn recommendation and user preference simultaneously and systematically.

3.1 Motivations

The trade-off between model interpretability and performance states that we can either achieve high interpretability with simpler models, or achieve high performance with more complex models that are in general hard to interpret[23]. However, recent works [24][23][7] have showed that, with adequate auxiliary information added, it is possible to achieve both goals in the same model. Here we design such a model that renders explainable recommendations using minimum or no auxiliary information. To achieve this, we assume that a simple interpretable representation can be mathematically derived from the corresponding complex representation. More formally:

Assumption 1. *When there are two representations for the same prediction task:* $\boldsymbol{u}$ *in complex feature space* $\mathcal{U}$ *of dimension* p, *and* $\boldsymbol{v}$ *in simpler feature space* $\mathcal{V}$ *of dimension* q, *where* $p > q$, *and* $\mathcal{U} \supset \mathcal{V}$. *We say that* $\boldsymbol{v}$ *is the projection of* $\boldsymbol{u}$ *from space* $\mathcal{U}$ *to space* $\mathcal{V}$, *and there exists a matrix* $\boldsymbol{M}(\boldsymbol{\theta})$ *of* $q \times p$, *with* θ *as hyperparameter, such that* $\boldsymbol{v} = \boldsymbol{M}(\boldsymbol{\theta}) \cdot \boldsymbol{u}$.

This assumption is inspired from [18] in which a simple local approximation can give good interpretation of a complex model in that particular neighborhood. In our assumption, instead of selecting an surrogate interpretable simple model, we map the general complex features to the simpler interpretable aspect features, then render explainable recommendations based on these interpretable simple features, achieving the best of both worlds in keeping the high performance of the complex model as well as gaining the interpretability of the simpler model. The *interpretable simple features* are obtained from *interpretable aspects*, hence the corresponding feature space is called *aspect space*. Here we define aspect projection.

Definition 2. ***(Aspect Projection)*** *We say* $\boldsymbol{v}$ *is an aspect projection of* $\boldsymbol{u}$ *from complex feature space* $\mathcal{U}$ *to aspect feature space* $\mathcal{V}$. *We use* $\tilde{\boldsymbol{v}}$ *to represent the extended representation of* $\boldsymbol{v}$ *in* $\mathcal{U}$.

To achieve the interpretability and performance in the same model, from Definition 2 and Assumption 1, we need to find the matrix $\boldsymbol{M}(\boldsymbol{\theta})$. Here we use generalized matrix factorization (GMF)[8] as the source model which infers complex feature $\boldsymbol{u}$, as shown in Figure 2 (cyan). Then the remaining problem is how to derive the connection between $\boldsymbol{u}$ and $\boldsymbol{v}$. We use the term *source feature* to represent the complex model's feature hereinafter.

3.2 Aspect Feature Extraction

To design a simple and interpretable model, its features should be well aligned to our interest, i.e. the *aspects*. Hence we select q predefined aspects as our interpretable features. From Assumption 1, to make the representation transformation from a source feature space $\mathcal{U}$ to an aspect feature space $\mathcal{V}$, we need to first define the aspect space. We represent the q aspects by q latent vectors in source space $\mathcal{U}$ as the basis that spans aspect space $\mathcal{V}$. These aspect's latent vectors can be inferred from auxiliary information such as user reviews, item profiles and tags. Any aspect-related information can be used to obtain the aspect vectors. Here we build a neural multi-task learning module to extract q aspect latent vectors of p-dimension, supervised by aspects, as shown in Figure 2(red).

The aspects of an item is represented by multi-hot encoding. Taking movie genre as an example: if we use 4 genres (Romance, Comedy, Thriller, Fantasy) as aspects, movie *Titanic*'s aspect should be represented by $(1, 0, 0, 0)$ because it's genre is romance, and the movie *Cinderella*'s aspect is $(1, 0, 0, 1)$ because it's genre falls into both romance and fantasy. In the neural multi-task learning, each

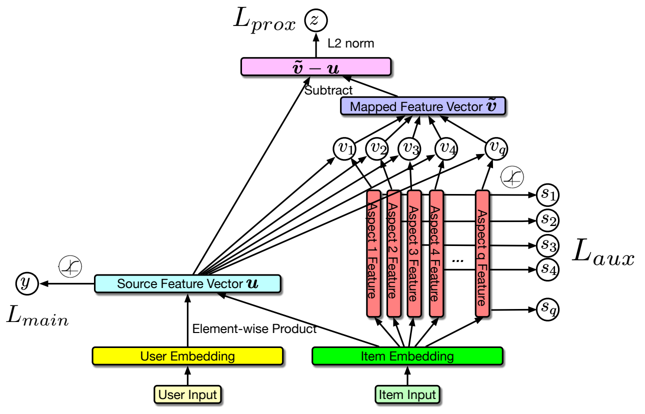

Figure 2: Explainable recommendation via interpretable feature mapping: an AMCF model.

aspect (genre) prediction task of an item (movie) should have a p-dimensional latent vector at its last hidden layer. We take these q aspect latent vectors as basis of the aspect space $\mathcal{V}$, and map the source features to this aspect space. See Section 3.3 below.

We describe the multi-task learning component in our model as shown in Figure 2: an input item is embedded to a latent representation, followed by creating q *normalized* latent vectors of p-dimension (i.e. with unit $l2$ norm) from q parallel fully connected layers (with normalizing activation). For each latent vector, logistic regression is used to predict the specific aspect with non-negative weight constraints. The logistic output is supervised by the aspect's ground truth. The purpose of multi-task learning component is to create a set of reasonable basis for interpretable aspect space $\mathcal{V}$.

3.3 Aspect Projection

For notation conciseness, we drop θ and use $\boldsymbol{M}$ directly instead of $\boldsymbol{M(\theta)}$ to represent the proposed interpretable feature mapping. Let $(\boldsymbol{\psi}_1, ..., \boldsymbol{\psi}_q)$ represent q orthogonal aspect vectors in space $\mathcal{U}$, which form the basis spanning the space $\mathcal{V}$, and $\boldsymbol{\Psi}$ is the corresponding $(p \times q)$ matrix with $(\boldsymbol{\psi}_1, ..., \boldsymbol{\psi}_q)$ as columns. Then $\boldsymbol{v}$ in $\mathcal{V}$ can be extended to space $\mathcal{U}$ by $\tilde{\boldsymbol{v}} = \boldsymbol{\Psi}\boldsymbol{v}$. Based on the Assumption 1 and Definition 2, $\boldsymbol{v}$ is known as the projection of $\boldsymbol{u}$. Since $\boldsymbol{\Psi}$ are orthogonal, we can easily calculate $\boldsymbol{v} = \boldsymbol{\Psi}^T\boldsymbol{u}$, i.e. $\boldsymbol{M} = \boldsymbol{\Psi}^T$, hence $\tilde{\boldsymbol{v}} = \boldsymbol{\Psi}\boldsymbol{\Psi}^T\boldsymbol{u}$. We call $(\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_q)$ as the *basis of aspect features* in the source space $\mathcal{U}$.

However, in most cases, the selection of aspects as basis is not necessarily orthogonal, in the case of non-orthogonal aspect basis, suppose we have q linearly independent *aspect* vectors $(\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_q)$ in space $\mathcal{U}$, which spans space $\mathcal{V}$, and $\boldsymbol{\Psi}$ is the corresponding $(p \times q)$ matrix with $(\boldsymbol{\psi}_1, ..., \boldsymbol{\psi}_q)$ as its columns. Suppose $\boldsymbol{v}$ in space $\mathcal{V}$ (with $\boldsymbol{\Psi}$ as basis) is represented by $\boldsymbol{v} = (v_1, ..., v_q)^T$, and then $\boldsymbol{v}$'s extension in space $\mathcal{U}$ can be represented by $\tilde{\boldsymbol{v}} = \boldsymbol{\Psi}\boldsymbol{v}$. From Assumption 1, by linear algebra we know that $\boldsymbol{v}$ can be calculated by $\boldsymbol{v} = (\boldsymbol{\Psi}^T\boldsymbol{\Psi})^{-1}\boldsymbol{\Psi}^T\boldsymbol{u}$, or $\boldsymbol{M} = (\boldsymbol{\Psi}^T\boldsymbol{\Psi})^{-1}\boldsymbol{\Psi}^T$, hence $\tilde{\boldsymbol{v}} = \boldsymbol{\Psi}(\boldsymbol{\Psi}^T\boldsymbol{\Psi})^{-1}\boldsymbol{\Psi}^T\boldsymbol{u}$.

It is worth noting that if space $\mathcal{U} = \mathcal{V}$, we must have $\tilde{\boldsymbol{v}} = \boldsymbol{u}$, otherwise we have the quantity $||\tilde{\boldsymbol{v}} - \boldsymbol{u}||_2$ represents the distance between $\tilde{\boldsymbol{v}}$ and $\boldsymbol{u}$ in space $\mathcal{U}$.

As shown above, the aspect projection matrix $\boldsymbol{M}$ can be obtained easily once we already obtained these aspect feature vectors in aspect feature extraction (Section 3.2). However the calculations are not trivial, especially in the case of non-orthogonal basis, the involvement of matrix inversion makes the calculation more challenging. However, one can train a neural network and use the latent layers as a surrogate for $\boldsymbol{M}$. We employ attention mechanism [1] as a natural way to approximate feature mapping matrix $\boldsymbol{M}$ in neural networks, the attention weights are an approximation of $\boldsymbol{v}$ in space $\mathcal{V}$ as shown in Figure 2 ($v_1, v_2, ...v_q$). If we choose GMF with non-negative constraint on the last hidden layer as the source model, then the GMF layer in Figure 2(cyan) represents the uninterpretable source feature $\boldsymbol{u}$, and the multi-task feature vectors represent the aspect feature basis $\boldsymbol{\Psi} = (\boldsymbol{\psi}_1, ..., \boldsymbol{\psi}_q)$, and hence we can calculate the attention weights by $\boldsymbol{v} = \boldsymbol{\Psi}\boldsymbol{u}$ for orthogonal basis. The use of attention mechanism is to create a feature mapping, i.e., $\boldsymbol{M}$, from source space $\mathcal{U}$ to aspect space $\mathcal{V}$.

Other aspect-level explaining methods [24][11][7] typically need a complete triangle (or tripartite[7]) for explainable recommendations, i.e. user-item, user-aspect and item-aspect. Here, our feature mapping approach doesn't introduce user-aspect relations and we show that with any two of the relations, we can derive the third. The proof is straightforward: since the user-item relation is the $(N \times M)$ rating matrix R, we denote $(N \times S)$ user-aspect matrix as P, and $(M \times S)$ item-aspect matrix as Q. *Assume the aspects are complete* (i.e., $\mathcal{U} = \mathcal{V}$), we must have $R \approx PQ^T$, $P \approx RQ(Q^TQ)^{-1}Q^T$, and $Q \approx P(P^TP)^{-1}P^TR$. Even in the cases where $\mathcal{U} \supset \mathcal{V}$, these are still good approximations. This explains why it is sufficient that our approach only requires item-aspect information, i.e., categorical information of the item.

3.4 The Loss Function

The loss function for finding the feature mapping $\boldsymbol{M}$ to achieve both interpretability and performance of the recommender model is composed of 3 components:

- L_{aux} auxiliary loss to the selection of aspects.
- L_{prox} proximity loss to the source feature $\boldsymbol{u}$. This loss item is to quantify the proximity of the reconstructed $\tilde{\boldsymbol{v}}$ to $\boldsymbol{u}$ in space $\mathcal{U}$.
- L_{main} main loss to item ratings, which is the loss function for the original CF model.

Given the attentive multi-task implementation of our feature mapping framework , the auxiliary loss component can then be defined by

$$L_{aux} = \sum_{(u,i)} \sum_{t=1}^{q} \left[s_i^t \log \hat{s}_{ui}^t + (1 - s_i^t) \log(1 - \hat{s}_{ui}^t) \right], \qquad (1)$$

where $\hat{s}_{ui}^t$ represents the logistic output for aspect t given inputs user u and item i, and s_i^t is the ground truth, where $s_i^t = 1$ if item i is of aspect t, $s_i^t = 0$ otherwise. By the loss component L_{aux}, we know that each aspect feature $\boldsymbol{\psi}_i$ (from last hidden layer) can be used to predict aspect s_i, i.e., there exists some mapping g such that $g(\boldsymbol{\psi}_i) = s_i$. We hence can view $\boldsymbol{\psi_i}$ as a representation of $\boldsymbol{s}_i$ in space $\mathcal{U}$. Since $\boldsymbol{v}$ is the coordinate representation with $\boldsymbol{\Psi}$ as the

Dateset	# of ratings	# of items	# of users	Sparsity
MovieLens[6]	1,000,209	3,706	6,040	95.53%
CiaoDVD[5]	72,665	16,121	17,615	99.96%

Table 1: Summary statistics of the datasets.

Metrics	ItemPop	ItemKNN	eALS	NeuMF	AMCF
HR@10	0.406	0.600	0.621	0.688	**0.692**
HR@20	0.455	0.673	0.701	0.841	**0.843**
NDCG@10	0.231	0.334	0.356	0.411	**0.419**
NDCG@20	0.251	0.410	0.424	0.450	**0.452**
HR@10	0.349	0.553	0.581	0.631	**0.646**
HR@20	0.440	0.659	0.701	0.750	**0.753**
NDCG@10	0.221	0.304	0.344	0.408	**0.418**
NDCG@20	0.265	0.330	0.386	0.438	**0.441**

Table 2: Performances comparison using MovieLens 1M Dataset (upper) and CiaoDVD Dataset (lower).

basis, we can say the corresponding attention weight $\boldsymbol{v}_i$ indicates the importance of such aspect $\boldsymbol{s}_i$ (via $\boldsymbol{\psi}_i$).

We define the proximity loss component as follows:

$$L_{prox} = ||\boldsymbol{u} - \tilde{\boldsymbol{v}}||_2. \tag{2}$$

The loss component L_{prox} encourages the reconstructed representation obtained from the attentive multi-task framework to be a good approximation of the source representation.

We define the item rating loss component for CF as:

$$L_{main} = \sum_{(u,i)} \left[y_{ui} \log \hat{y}_{ui} + (1 - y_{ui}) \log(1 - \hat{y}_{ui}) \right], \tag{3}$$

where y_{ui} represents the main output (prediction of item ratings).

Hence the whole loss function is $L = L_{main} + \lambda_1 L_{aux} + \lambda_2 L_{prox}$, where λ_1 and λ_2 are tuning parameters to leverage importance of different loss components. λ_1 regularizes how good the basis represents aspects (explainability), λ_2 regularizes the proximity between the projected interpretable features and the uninterpretable source features.

4 EXPERIMENTS AND DISCUSSION

Dataset and Pre-processing: We perform experiments on MovieLens 1M and CiaoDVD datasets to demonstrate both recommendation and explanation performance of our approach where the content-based information are not available. In MovieLens dataset, we transform the explicit ratings to implicit ratings by setting 1 when a user rates a movie and 0 otherwise. We use movie genre as the aspect for interpretable feature mapping. CiaoDVD is a crawled dataset [5] and we pre-process CiaoDVD in the same way as the MovieLens dataset. The summary statistics of the both datasets are displayed in Table 1.

We use the negative sampling strategy as in [8][22][13][17] to sample a small portion of negative samples from the pool of entries with missing ratings to avoid massive computation from all missing/negative ratings and possible overfitting.

We set the number of factors in GMF to 32 as further increase makes marginal improvement in recommendation performance at the expense of overfitting. The negative sampling size is set to be 4 ([8]) and the values of λ_1 and λ_2 are set to 0.5 in performance comparison. Other values of λ's exhibit negligible impact on the recommendation performance (data not shown). We compare recommendation performance of our proposed AMCF method with the following baseline methods: ItemPop, ItemKNN[19], eALS[9] and NeuMF[8], using the widely adopted leave-one-out evaluation technique [8][4][22], in terms of Hit Ratio (HR) and Normalized Discounted Cumulative Gain (nDCG).

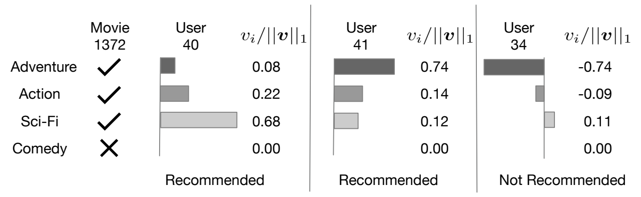

Figure 3: Examples of explainable recommendations.

Results: Table 2 shows the recommendation performance comparisons between our AMCF method and the baselines on MovieLens 1M and CiaoDVD datasets. AMCF method consistently outperforms the competing Neural MF method [8] and other baselines. The superior recommendation performance may due to exploiting genre information and borrowing strength among movie ratings across genres. Through mapping latent uninterpretable features to interpretable aspect features, our AMCF method not only demonstrates remarkable recommendation performance but also gives compelling explanation for its recommendation. Further, AMCF is readily to be deployed to render accurate yet explainable recommendation in real time.

For recommendation explanation, given a user u, and an item i, our AMCF model predicts a vector $\boldsymbol{v}$ in real time, representing a user's preference on the item with regard to all predefined aspects. Specifically, the normalized signed magnitude of each entry of $\boldsymbol{v}$ (i.e. $v_i/||\boldsymbol{v}||_1$) represents the impact of a specific aspect on whether an item is recommended to a user or not. For example, in Figure 3, the movie 1372 is recommended to both users 40 and 41 in terms of the 3 pre-defined genres, however, with differential explanations: the former user's preference is more on the *Sci-Fi* genre whereas the latter user's preference is more on *Adventure*. On the other hand, the same movie is not recommended to user 34 mainly due to the dislike of *Adventure* genre.

5 CONCLUSION

Modelers tend to better appreciate the interpretable recommender systems whereas users more likely to accept the explainable recommendations. In this paper, we proposed a novel interpretable feature mapping strategy attempting to achieve the two-folded goal: systems interpretability and recommendation explanation, without the need for content-based auxiliary information. Our implementation of AMCF method demonstrates strong performance in both real-time recommendation and explanation.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[2] Konstantin Bauman, Bing Liu, and Alexander Tuzhilin. 2017. Aspect based recommendations: Recommending items with the most valuable aspects based on user reviews. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 717–725.

[3] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Neural attentional rating regression with review-level explanations. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1583–1592.

[4] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. ACM, 7–10.

[5] G. Guo, J. Zhang, D. Thalmann, and N. Yorke-Smith. 2014. ETAF: An Extended Trust Antecedents Framework for Trust Prediction. In *Proceedings of the 2014 International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*.

[6] F Maxwell Harper and Joseph A Konstan. 2016. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)* 5, 4 (2016), 19.

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

[8] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.

[9] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 549–558.

[10] Jonathan L Herlocker, Joseph A Konstan, and John Riedl. 2000. Explaining collaborative filtering recommendations. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*. ACM, 241–250.

[11] Yunfeng Hou, Ning Yang, Yi Wu, and S Yu Philip. 2019. Explainable recommendation with fusion of aspect information. *World Wide Web* 22, 1 (2019), 221–240.

[12] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 8 (2009), 30–37.

[13] Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. 2018. Variational Autoencoders for Collaborative Filtering. *arXiv preprint arXiv:1802.05814* (2018).

[14] Yichao Lu, Ruihai Dong, and Barry Smyth. 2018. Why I like it: multi-task learning for recommendation and explanation. In *Proceedings of the 12th ACM Conference on Recommender Systems*. ACM, 4–12.

[15] Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*. ACM, 165–172.

[16] Claudiu Cristian Musat, Yizhong Liang, and Boi Faltings. 2013. Recommendation using textual opinions. In *Twenty-Third International Joint Conference on Artificial Intelligence*.

[17] Steffen Rendle and Christoph Freudenthaler. 2014. Improving pairwise learning for item recommendation from implicit feedback. In *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 273–282.

[18] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 1135–1144.

[19] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. ACM, 285–295.

[20] Nava Tintarev and Judith Masthoff. 2007. A survey of explanations in recommender systems. In *2007 IEEE 23rd international conference on data engineering workshop*. IEEE, 801–810.

[21] Jesse Vig, Shilad Sen, and John Riedl. 2009. Tagsplanations: explaining recommendations using tags. In *Proceedings of the 14th international conference on Intelligent user interfaces*. ACM, 47–56.

[22] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. 2016. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. ACM, 153–162.

[23] Yongfeng Zhang and Xu Chen. 2018. Explainable recommendation: A survey and new perspectives. *arXiv preprint arXiv:1804.11192* (2018).

[24] 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 *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 83–92.