DCT: A Deep Collaborative Filtering Approach Based on Content-Text Fused for Recommender Systems

Zhiqiao Zhang(✉), Junhao Wen, and Jianing Zhou

School of Big Data and Software Engineering, Chongqing University, Chongqing, China
{zhangzhiqiao,jhwen,jnzhou}@cqu.edu.com

Abstract. Recommender systems commonly make recommendations by means of user-item interaction ratings. One of the basic methodologies of recommendation is collaborative filtering, which exploits users' and items' latent space features to make predictions of personalized ranking list for an individual user. However, most of existing collaborative filtering approaches only employ explicit interaction ratings data to predict user preferences, and neglect the necessary of exploiting implicit feedback data and auxiliary information in promoting the performance recommendation. In this paper, we raise a novel model of recommendation on the basis of neural networks architecture. Concretely, the model exploits both of interaction data and content text information as input and adopts two parallel neural networks to learn the latent feature representations of users and items for a better performance. We utilize three kinds of real-world data to make extensive evaluations on the model. The experimental results reveal that the method we proposed dramatically outperforms the state-of-the-art methodologies and achieves expressively improvement in performance.

Keywords: Recommender systems · Deep learning · Textual information

1 Introduction

With the development of technology, numerous applications emerge in large numbers resulting in the expansion size in data which has led to the problem such as 'information overload'. Recommender systems (RSs) are an effective means of solving this phenomenon that can find the valuable information to meet users' demands quickly and efficiently. It provides unique user with a personalized recommendation service. The fundamental essence of RS is prediction which utilizes the information of users' preferences on items to recommend [1, 2]. In addition, RSs are everywhere which have been adopted into numerous fields, such as information retrieval, e-commerce and so on.

The methodologies of RSs major include collaborative filtering (CF), content-based recommendation, and hybrid recommendation. CF is the most widely applied approach, which leverages the similarities among items to recommend for users. In other words, if users have bought some products in the past, the strategy of collaborative filtering infers

H. Gao et al. (Eds.): CollaborateCom 2020, LNICST 349, pp. 392–405, 2021.
https://doi.org/10.1007/978-3-030-67537-0_24

that they would be interested in similar stuffs in the future [3]. Matrix factorization (MF) is one of the most commonly used methodologies of collaborative filtering due to its simplicity and scalability. MF employs latent feature factors to denote to items and users respectively, which will be adopted to obtain predicted ratings of users on a set of items.

To make predictions of personalized ranked lists on items for recommendation, matric factorization captures users' and items' low-dimensional space through ratings interaction matrix. Whereas, the method only using the explicit ratings data is insufficient to make a good top-n recommendation [4–6]. Many existing methods take the implicit feedback into consideration for a better performance [7–9]. Such as collaborative denoising autoencoder (CADE) [10] and neural collaborative filtering (NCF) [11]. Nevertheless, both of them only employ implicit feedback and neglect the explicit ratings.

In the meanwhile, the growing of the number over of stuffs and users in applications will sharpen sparseness of rating data of users on items, which will impact the performance of recommendation. Consequently, several methods take account of ratings and auxiliary information (e.g. social relationship, item description documents, etc.). Some previous methodologies (e.g. stacked denoising auto-encoder (SDAE) and latent dirichlet allocation (LDA)) adopt item description documents to enhance the performance [12, 13]. Nevertheless, those hybrid recommendation methodologies commonly only use bag-of-words model to capture content information, neglecting the semantic features and deep latent features, which will affect the performance in predictions.

To dispose the above-mentioned phenomenon, we exploit two parallel deep neural networks to deal with interaction data and auxiliary information at the same time for getting a better performance. Moreover, owing to its ability of powerful representation learning, deep learning has obtained significant progress in image identification, speech recognition, nature language processing and so on. Especially, applying deep neural networks into recommender systems facilitates deeper extract the users' and items' latent representations [14–16].

In our work, we address a novel collaborative filtering method on the basis of neural networks. The model leverages both of interaction data and auxiliary information to make recommender prediction through two parallel networks. In details, we capture items' and users' deep latent feature representations from the interaction ratings data by utilizing the multi-layer neural network. In the meanwhile, we convert content text into latent feature vectors through another parallel neural network layers for enhancing the final item representation. Moreover, we take full advantage of explicit data and implicit feedback information to optimize the model, which could be better to learn the latent features of items and users. As for the experiments, we compared performance variation the model proposed with other five methodologies on three kinds of real-world datasets. In addition, we make extensively evaluations on hyper-parameters of the model. Experimental results signify that the approach we raised is effective.

The rest part organization of this paper is as bellow. First, Sect. 2 has a brief review of related work. Next, Sect. 3 states problems and structure of the model concretely. Then, Sect. 4 analyzes the experiments over three kinds of data. Finally, Sect. 5 outlines the contributions and future work.

2 Related Work

2.1 Collaborative Filtering

It can be perceived as a process of word-of-mouth for the strategy of collaborative filtering (CF). It makes recommendations for users by making use of the rating data and other auxiliary information. Some models of collaborative filtering are quite remarkable.

Wang [17] combined LDA with probabilistic matrix factorization (PMF) in order to optimize the method performance of CF. Hu [18] made full use of implicit feedback data to identify the unique properties. They proposed that the interaction information of users on items can be regarded as indicant of positive and negative preference in reference to disparate confidence levels. Lee [19] adopted variational autoencoders (VAE) to extent the model of CF, in the meanwhile, they leveraged auxiliary information to intensify items' and users' representations. Liang [20] also applied VAE to the model of collaborative filtering, which utilized implicit feedback. Based on that, then they devised a generative model with multinomial likelihood.

The methods of CF try to take advantage of the auxiliary information or learn unsupervised topic models for promoting the performance. However, by reason of data sparse, the low-dimensional representations learned from the model are not perfectly efficient. Meanwhile, these methods only exploit bag-of-words models and neglect the semantic features of content text, which will lead to some limitations. Whereas, deep learning can be utilized to address this issue.

2.2 Deep Learning

Deep Learning lets computational models with multi-processing-layers extract representations of data with abstraction multi-levels [21]. It has a significant attention in object detection, natural language processing (NLP), information retrieval and so on. Furthermore, deep learning has made great progress in recommender systems. And the recommendation models using the strategy of deep learning adopt deep neural networks, then employ user-item interaction information as input for obtaining items' and users' latent representations respectively for which can be used to make personalized prediction.

Xue [22] presented to combine the explicit ratings with implicit feedback, and capture users' and items' low-dimensional representations by utilizing deep neural networks. He [11] raised a common neural collaborative filtering architecture on the basis of deep learning, modeling with linearity and non-linearity of multi-layers. Zheng [23] exploited comments to obtain latent feature space of items and users respectively through parallel neural networks. Then, they jointed items' features and users' preference of learned from model to predict ratings.

These recommendation methods have proved that it is effective to obtain items' and users' latent representations through the content text and deep neural network respectively. Nonetheless, to our knowledge, there is no previous methods that obtain the latent representations of items, users and content text at same time by adopting the deep neural networks, and take both of explicit and implicit feedback into consideration as well.

3 Proposed Model

First, we state the problem of our model in this section. Then, we analyze the model addressed in details. Finally, we put forward the objective function to optimize the method.

3.1 Problem Statement

Assuming that there are M users $U = \{u_1, \ldots, u_M\}$, N items $V = \{v_1, \ldots, v_N\}$ and each item contains content text. On account of users interacting with items, make $R \in \mathbf{R}^{M \times N}$ denotes to the interaction matrix, where R_{ij} indicates the rating attained from user i interacted with item j. If R_{ij} is unknown, we mark it *unk*.

Commonly, existing two kinds of methodologies to devise interaction matrix $Y \in \mathbf{R}^{M \times N}$ of users on items based on the implicit feedback of R, where Y_{ij} indicates the $(i, j)_{th}$ entry of Y. Several recommendation methods employ Eq. (1) to model matrix Y, which regard the observed ratings as 1 and neglect the significance of the value of ratings. While some others construct user-item matrix Y by adopting Eq. (2) that reserves the value of ratings which can be applied to show the degree of preference of user u_i on item v_j. Based on this, we employ Eq. (2) to model interaction matrix Y. Meanwhile, we mark zero if ratings of users on items are unknown, which can be taken as implicit feedback in our work.

$$Y_{ij} = \begin{cases} 0 & if \ R_{ij} = unk \\ 1 & otherwise \end{cases} \tag{1}$$

$$Y_{ij} = \begin{cases} 0 & if \ R_{ij} = unk \\ R_{ij} & otherwise \end{cases} \tag{2}$$

Recommender systems are typically deemed to calculate the unknown entry of matrix Y, which means ratings prediction. Model-based collaborative filtering methods commonly make use of latent factor model (LFM) that exploits the dot product of user' and items' latent vectors for predicting.

$$\hat{Y}_{ij} = F^{LFM}(u_i, v_j | \Theta) \tag{3}$$

Where u_i and v_j respectively indicate the i_{th} user and j_{th} item. $\hat{Y}_{ij}$ is the predicted value of rating by adopting u_i and v_j. And F denotes to the function of mapping the parameters into predicted values. In our work, we follow the LFM and extent it.

Besides, on account of the recommending tasks which are related to content text. In our work, we exploit a sequence $X_j = (w_1, \cdots, w_l)$ with l word tokens to denote to the j_{th} item content text, where each word token is in the vocabulary containing W words.

3.2 Proposed Model

Facing the problem of data sparseness, many existing collaborative filtering methodologies try to adopt auxiliary information (e.g. social relationship, item description documents, etc.) for promoting the performance of recommendation. Whereas, most of

previous models do not take the relevance of rating matrix and content text into consideration. They only employ one kind of information data as input of the model to predict. In addition, some traditional methods use topic model through unsupervised learning to pre-train text as features, which is insufficient to obtain the deep latent features of textual.

In our model, we exploit both of interaction ratings data and content text for promoting the performance of recommendation. Concretely, the model adopts two parallel neural networks to deal with interaction dataset and content text information at the same time. Figure 1 demonstrates our model's structure in details. We obtain the deep latent feature representations of items and users from interaction ratings matrix by utilizing the multi-layer neural network. Meanwhile, we convert content text into item latent representations through the embedding-layer and neural network layers. Employing the representation of embedding-based is helpful for learning the latent text features owing to word embedding can be pre-trained on large corpus through the unsupervised way. In addition, neural networks can better understand the deep latent features of content text. We adopt linear strategy to fusion the two kinds of item representations extracted from two parallel neural networks to enhance the final item representation. Moreover, we utilize users' and items' representations to predict the ratings.

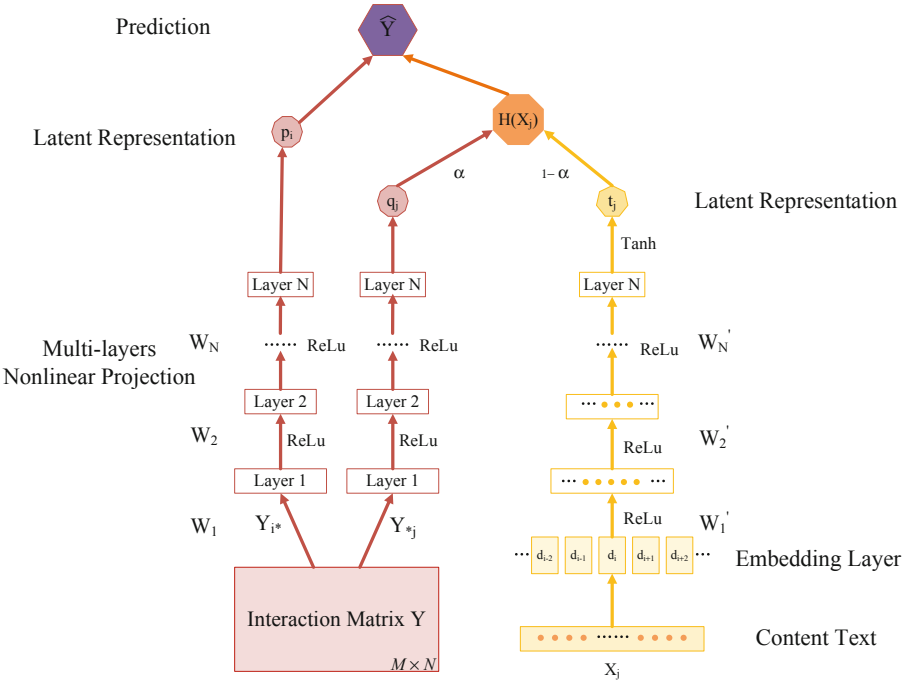

Fig. 1. The architecture of the proposed model.

Furthermore, we adopt Eq. (2) to model the interaction matrix Y, where Y_{i*} denotes to the high dimension vector of user u_i which demonstrates the ratings of i_{th} user over all

of items and Y_{*j} is the high dimension vector of item v_j which means ratings of j_{th} item graded by all users. The q_j and p_i respectively indicate the latent representations of item v_j and user u_i. Whereas, we do not directly take the q_j as the item's final representation. The model fuses the latent vector q_j and text latent feature vector t_j which extracted from interaction matrix and item content text respectively. Therefore, the item's final latent representation becomes as Eq. (4).

$$H\left(X_j\right) = \alpha q_j + (1 - \alpha)t_j \tag{4}$$

Where α denotes to be a hyper-parameter. And t_j is the content text latent feature vector. Concretely, at first, the content text of item v_j is deemed to be a sequence $X_j = (w_1, \cdots, w_l)$ with l words. In addition, we leverage the word embedding model to process the item content text and adopt a lookup table where each word can be expressed by a single vector in the vocabulary. Then, we joint vectors of words in text sequence X_j to get a numeric text matrix $D_j^{\rho \times l}$ through the embedding layer.

$$D_j = \begin{bmatrix} | & | & | & \\ \cdots & d_{i-1} & d_i & d_{i+1} & \cdots \\ | & | & | & \end{bmatrix}$$

Where ρ indicates embedding dimension size of each word vector d_i and l signifies the length of content text. At last, we obtain text feature vector t_j through the deep neural network. In our model, we adopt tanh as activation function on the output-layer in text neural network for getting better performances.

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{5}$$

Besides, in the neural network layers, we employ b_i and W_i respectively indicate the i_{th} bias and weight matrix of the middle-hidden-layers l_i, i $\in [1, N]$. In the meanwhile, in order to prevent the vanishing gradient, we adopt ReLU as activation function.

$$\text{ReLU}(x) = max(0, x) \tag{6}$$

Moreover, in our work, we follow and extent LFM to predict ratings which is formulated as Eq. (7). And the model employs H(·) instead of q_i as the final item latent representation on account of integrating item content text feature factors into the model. H(·) can be referred as items' universal feature extractor, which can be extended in various way.

$$\hat{Y}_{ij} = p_j^T H\left(X_j\right) \tag{7}$$

3.3 Loss Function

It is another key component to devise an advisable objective function for optimizing the method in our work, which is on the basis of unobserved implicit feedback information and observed ratings. The most common of objective function is indicated as Eq. (8).

$$L = \sum_{y \in Y} l\left(y, \hat{y}\right) + \Omega(\theta) \tag{8}$$

Where $\Omega(\theta)$ denotes to the regularizer and $l(\cdot)$ is the loss function. Most of existing recommendation methodologies common use squared loss as loss function. However, the squared loss could not do well with implicit feedback data on account of the value of implicit feedback in Y_{ij} is a binarized 1 or 0 indicating that if users interact with items or not. He [11] proposed a binary cross entropy loss function with implicit feedback as follows.

$$L = \sum_{(i,j)\in Y} Y_{ij} \log \hat{Y}_{ij} + \left(1 - Y_{ij}\right) log\left(1 - \hat{Y}_{ij}\right) \tag{9}$$

To integrate the explicit ratings into loss function, extending Eq. (9) to get a new normalized cross entropy loss as below.

$$L = -\sum_{(i,j)\in Y} \frac{Y_{ij}}{max(R)} log\, \hat{Y}_{ij} + \left(1 - \frac{\hat{Y}_{ij}}{max(R)}\right) log\left(1 - \hat{Y}_{ij}\right) \tag{10}$$

Where $\max(R)$ is the maximum in all ratings, which is utilized for normalization. Based on this, diverse value of Y_{ij} has different impacts on loss. In our work, we take the normalized cross entropy loss with implicit and explicit feedback as an objective function. However, for the predicted value of $\hat{Y}_{ij}$ can be negative, we exploit Eq. (11) to convert the value of predictions, where σ is an extremely low number and the value is $1.0e^{-8}$.

$$\hat{Y}'_{ij} = max\left(\sigma, \hat{Y}_{ij}\right) \tag{11}$$

4 Experiments

We make extensive evaluations of performances on the model proposed in this section. Firstly, we briefly review the settings of experiments in details, covering implementation details, evaluation criterion and datasets. Secondly, we compare performances of the model with various settings according to the extensive experiments, which contains the number of neural network layers, integrating ratio, negative sampling ratio and so on.

4.1 Experimental Settings

Datasets. To indicate efficacy of model presented for personalized ranking recommendation, we exploit three kinds of real-world datasets to conduct experiments. One is MovieLens 100 K (ML-100 K), another is MovieLens 1 M (ML-1 M), and a third is Amazon. Those three kinds of datasets contain the explicit ratings which are on a scale of 1 to 5. The data of Amazon covers the comments of items, which can be deemed as content text. Nonetheless, MovieLens does not have the description of items, we captured the content text related to the items from IMDB. In the meanwhile, since the data of Amazon is highly sparse. We filtrated the Amazon data and reserved the users which have at least 5 interactions with the items. In addition, we removed the items which do not have the relevant content text information on each data. Table 1 illustrates the characters of three kinds of datasets specifically.

Table 1. Statistics of three real-world datasets.

Datasets	Users	Items	Ratings	Rating density
ML-100 K	943	1526	95349	6.626%
ML-1 M	6040	3544	993482	4.641%
Amazon	3374	23963	38648	0.048%

Evaluation. We exploited leave-one-out valuation used in [24–26] to make appraisal of performances on each model. And we split each dataset by holding-out users' recent interaction as test sets and reserving residual dataset as training sets. However, ranking all items for each user would be too time-consuming. In view of this, we adopted the approach in [11] and [22] to sample 100 items that have no interaction with users at random, which we can rank rest items according to the prediction among these items. We utilized the metrics of normalized discounted cumulative gain (NDCG) and hit ratio (HR) to make evaluations on models' performances, where NDCG indicates the ranking ability and quality, HR intuitively reveals that if the item of test is appeared on ranked lists.

Implementation Details. We exploited python and tensorflow to implement the method proposed. In the experiments, mini-batch Adam [27] was used to optimize the model. And each mini-batch's size was set to be 128. Learning rate was set to be 0.0001. In the meanwhile, in our work, to initialize word vectors through the pre-trained word embedding, we used the CBOW [28] which is based on Gensim, and the size of word dimension was set as 200. As for hyper-parameters, we set the value of integrating ratio α as 0.7.

4.2 Performance Comparison

To demonstrate the improvement of our approach on the performance of recommendation, we made an intuitive and full comparison on the performance variance of the model raised and some other the-state-of-art approaches as below.

1. **BPR** [25]. Bayesian personalized ranking is the one of classical methods in ranking for recommendation. The method used pairwise ranking loss to optimize the matrix factorization model. We took the BPR as a baseline method for comparison.
2. **CTR** [18]. Collaborative topic regression combines probabilistic matrix factorization with LDA, and employs both of ratings and documents context to recommend. In the experiments, we took CTR as the baseline methodology for comparing.
3. **NeuMF** [11]. NeuMF employs cross entropy loss to optimize the model. It combines generalized matrix factorization with multi-layer perceptron to get a better performance in item recommendation. However, it only utilizes the ratings data to predict.

4. **ConvMF** [29]. Convolutional matrix factorization employs convolutional neural networks to map content text into probabilistic matrix factorization. Nonetheless, different from our method, it only leverages neural networks to deal with content text. As for the interaction information, it still takes the traditional strategy of PMF. We adjusted the hyper-parameters of this model in accord with it.
5. **DMF** [22]. Deep matrix factorization employs deep neural networks to get users' and items' latent feature factors. However, vary from our model, it only takes interaction ratings information into consideration. This is a state-of-the-art deep neural networks methodology for recommendation, we adjusted the hyper-parameters of this model in accord with it.
6. **DCT**: The model we proposed, which is on the basis of neural networks architecture. We exploit both of interactions matrix of user on item and content text as input for a better performance in rating prediction. Meanwhile, we utilize two parallel neural networks to learn the latent features respectively at the same time.

Table 2 summarizes the comparative results that indicates all models' performances on three kinds of datasets respectively by making use of metrics of NDCG and HR. In details, as is showed in the experiments, contrasted with DMF and NeuMF, which only take use of interaction ratings through neural networks, our method has a significantly improvement on the real-world datasets. It proves that integrate the auxiliary content text into model is helpful in promoting the performance in recommendation and solving the problem of data sparse. Meanwhile, compared with ConvMF, which adopts review document and do not consider the deep latent features of items and users in interaction ratings data, our method also has a breakthrough on three kinds of data, and the results reveal that employ two parallel deep neural networks to train interaction ratings and content text at same time is valid, which can improve the performance.

Table 2. Comparisons over different methodologies.

Datasets	Metrics	Method					
		BPR	CRT	NeuMF	ConvMF	DMF	DCT
ML-100 K	HR	0.534	0.626	0.672	0.701	0.648	**0.896**
	NDCG	0.397	0.401	0.409	0.592	0.415	**0.667**
ML-1 M	HR	0.446	0.598	0.705	0.688	0.712	**0.864**
	NDCG	0.383	0.421	0.457	0.579	0.464	**0.627**
Amazon	HR	0.314	0.487	0.569	0.589	0.575	**0.638**
	NDCG	0.202	0.385	0.408	0.421	0.410	**0.454**

4.3 Sensitivity to Hyper-parameters

Depth of Neural Network Layers. Extracting the latent representations of items and users from the interaction data through the multi-layer neural network plays an important

part in our model. Consequently, an extensive evaluation was conducted on the performance of our model with disparate in numbers of hidden-layers. Figure 2 states the impact on the performance with different in numbers of layers on each iteration. However, we only show the first 15 iterations on the metrics of HR and NDCG on account of the space limitation. As shown in the figure, compared with 1-layer and 3-layers structure, the method with 2 hidden-layers gets best performance on ML-1 M and ML-100 K. Nevertheless, on the dataset of Amazon, our model with 1-hidden-layer performs best on the metric of HR during the iteration, while the model with 2 and 3 hidden-layers have a similar performance on the metric of NDCG instead. Even so, deep layers seem to be a little helpful for prediction accuracy. Maybe due to the reason of our model extracts the auxiliary text latent factor to enhance item representation through a parallel neural network, which does have a good effect on the performance.

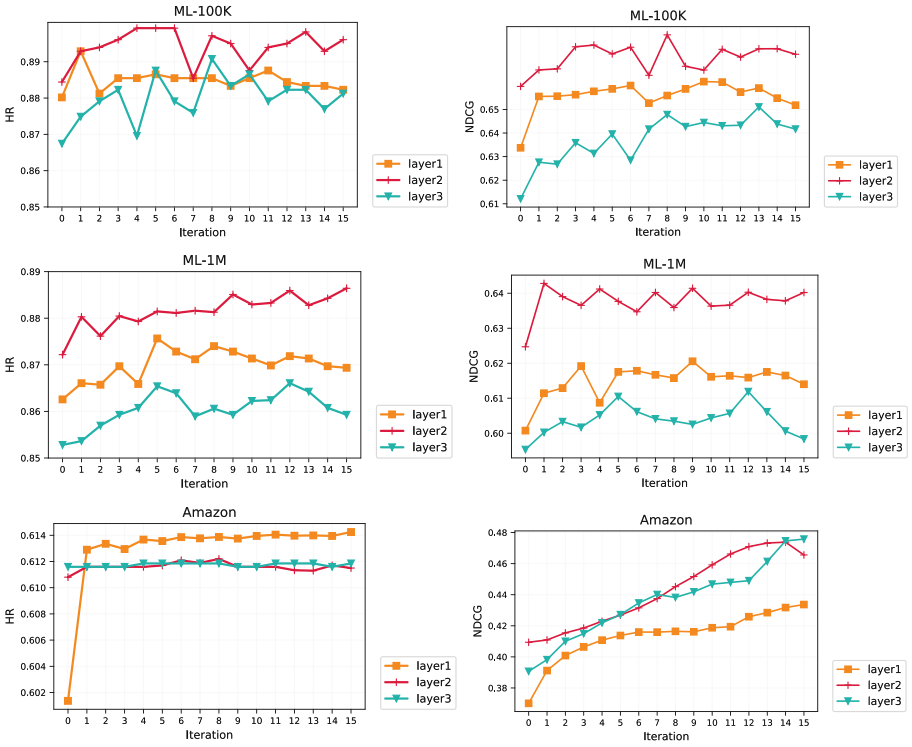

Fig. 2. Experimental results with different layers

Integrating Ratio. In our model, two kinds of item latent vectors are fused into one final item representation through the hyper-parameter α referred to the Sect. 3.2. Table 3 demonstrates the performance variance with disparate values of α on both of three kinds of datasets. As is shown in the table, the value of α around the number of 0.2 and 0.8 achieve the best performance. And it can be proved that integrate the representations

of content text into the model and utilize two parallel neural networks to learn the representations from interaction data and content text at the same time seems useful for improving the performance in recommendation.

Table 3. Comparisons with different integrating ratio

α	ML-100 K		ML-1 M		Amazon	
	HR	NDCG	HR	NDCG	HR	NDCG
0.1	0.899	0.669	0.876	0.632	0.612	0.432
0.2	**0.905**	**0.672**	**0.885**	**0.640**	**0.623**	**0.451**
0.3	0.886	0.663	0.879	0.637	0.617	0.438
0.4	0.899	0.676	0.866	0.626	0.612	0.436
0.5	0.883	0.669	0.880	0.636	0.613	0.434
0.6	0.893	0.674	0.882	0.634	0.616	0.433
0.7	0.886	0.663	0.874	0.630	0.619	0.438
0.8	**0.901**	**0.675**	**0.883**	**0.644**	**0.621**	**0.446**
0.9	0.894	0.666	0.878	0.636	0.615	0.437

Negative Sampling Ratio. In our model, we need to sample negative instances from unobserved data to train. Based on this, we employed different in numbers of negative instances to observer the impact on the model. Table 4 indicates that the performance variance with different value of sampling ratio. From the experimental, we can see that the negative instance's value around 5 getting the optimal performance on both three datasets, which is agree with results of work in [22]. Besides, the experiment result expresses that the more negative instances, the better performance of the model would be.

Table 4. Comparisons with different negative sampling ratio.

Datasets	Metrics	Negative sampling ratio					
		1	2	5	7	9	10
ML-100 K	HR	0.889	0.890	**0.896**	0.892	0.892	0.882
	NDCG	0.637	0.655	**0.664**	0.666	0.663	0.657
ML-1 M	HR	0.874	0.878	**0.887**	0.885	0.886	0.883
	NDCG	0.629	0.635	**0.639**	0.637	0.634	0.636
Amazon	HR	0.609	0.612	**0.627**	0.623	0.611	0.613
	NDCG	0.426	0.437	**0.452**	0.432	0.423	0.433

Final Latent Space Factors Latent factors are another sensitive parameter in the method. We employ the model of 2 hidden-layers structure to conduct this experiment. Table 5 demonstrates the performance with disparate in numbers of factors on a scale of 8 to 128. As can be clearly seen from results, the model utilizing 128 latent factors gets the best performance over three kinds of datasets. In general, the representations with more factors can be more effective in performance for sparse datasets.

Table 5. Comparisons with different factors

Factors	ML-100 K		ML-1 M		Amazon	
	HR	NDCG	HR	NDCG	HR	NDCG
8	0.871	0.623	0.808	0.542	0.575	0.409
16	0.889	0.643	0.851	0.601	0.583	0.418
32	0.892	0.669	0.871	0.623	0.604	0.426
64	0.904	0.676	0.882	0.638	0.609	0.435
128	0.905	0.682	0.889	0.648	0.617	0.445

5 Conclusion and Future Work

In this paper, we present a new approach of recommendation, which integrates content text as auxiliary information into neural networks architecture to enhance items' representations. Meanwhile, this method exploits both of the explicit ratings and implicit feedback information to better understand the latent features of items and users. Compared with other approaches, the model we addressed utilizing two parallel neural networks to deal with interaction data and text information at the same time is proved to be valid and achieves a dramatically improvement performance in item recommendation.

As for the future work, we attempt to joint other auxiliary data (e.g. user description documents, social network, etc.) into the model for getting a better performance. In addition, we will try to devise some other methodologies to capture text latent factors by adopting some other natural language processing methods which can learn the semantic context better.

Acknowledgement. This research was supported in part by the National Key Research and Development Program of China under grant 2018YFF0214700.

References

1. Lu, J., Wu, D., Mao, M., et al.: Recommender system application developments: a survey. Decis. Support Syst. **74**, 12–32 (2015)

2. Bobadilla, J., Ortega, F., Hernando, A., et al.: Recommender systems survey. Knowl.-Based Syst. **46**, 109–132 (2013)
3. Linden, G., Smith, B., York, J.: Amazon.com recommendations: item-to-item collaborative filtering. IEEE Internet Comput. **7**(1), 76–80 (2003)
4. Ashley-Dejo, E., Ngwira, S., Zuva, T.: A survey of context-aware recommender system and services. In: International Conference on Computing, Communication and Security (ICCCS), pp. 1–6, IEEE (2015)
5. Bi, Z., Zhou, S., Yang, X., Zhou, P., Wu, J.: An approach for item recommendation using deep neural network combined with the Bayesian personalized ranking. In: Wang, X., Gao, H., Iqbal, M., Min, G. (eds.) CollaborateCom 2019. LNICST, vol. 292, pp. 151–165. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-30146-0_11
6. Abdulkarem, H.F., Abozaid, G.Y., Soliman, M.I.: Context-aware recommender system frameworks, techniques, and applications: a survey. In: International Conference on Innovative Trends in Computer Engineering (ITCE), pp. 180–185. IEEE (2019)
7. He, R., Julian, M.: VBPR: visual Bayesian personalized ranking from implicit feedback. In: 30th AAAI Conference on Artificial Intelligence, pp. 144–150 (2016)
8. Purushotham, S., Liu, Y., Kuo, C.C.J.: Collaborative topic regression with social matrix factorization for recommendation systems. In: 29th International Conference on International Conference on Machine Learning (ICML), pp. 691–698 (2012)
9. Li, H., Diao, X., Cao, X., et al.: Collaborative filtering recommendation based on all-weighted matrix factorization and fast optimization. IEEE Access **6**, 25248–25260 (2018)
10. Wu, Y., DuBois, C., Zheng, A.X., et al.: Collaborative denoising auto-encoders for top-N recommender systems. In: 9th ACM International Conference on Web search and Data Mining (WSDM), pp. 153–162 (2016)
11. He, X., Liao, L., Zhang, H., et al.: Neural collaborative filtering. In: 26th International Conference on World Wide Web (WWW), pp. 173–182 (2017)
12. Ling, G., Lyu, M.R., King, L.: Ratings meet reviews, a combined approach to recommend. In: 8th ACM conference on Recommender systems (RecSys), pp. 105–112 (2014)
13. McAuley, J., Leskovec, J.: Hidden factors and hidden topics: understanding rating dimensions with review text. In: 7th ACM Conference on Recommender Systems, pp. 165–172 (2013)
14. He, X., Du, X., Wang, X., et al.: Outer product based neural collaborative filtering. In: IJCAI Conference, pp. 2227–2233 (2018)
15. Cheng, H., Koc, L., Harmsen, J, et al.: Wide & deep learning for recommender systems. In: 1st Workshop on Deep Learning for Recommender Systems, pp. 7–10 (2016)
16. de Souza Pereira Moreira, G., Ferreira, F., da Cunha, A.M.: News session-based recommendations using deep neural networks. In: 3rd Workshop on Deep Learning for Recommender Systems, pp. 15–23 (2018)
17. Wang, C., Blei, D.M.: Collaborative topic modeling for recommending scientific articles. In: 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 448–456 (2011)
18. Hu, Y., Koren, Y., Volinsky, C.: Collaborative filtering for implicit feedback datasets. In: 8th IEEE International Conference on Data Mining, pp. 263–272 (2008)
19. Lee, W., Song, K., Moon, I.C.: Augmented variational autoencoders for collaborative filtering with auxiliary information. In: ACM on Conference on Information and Knowledge Management, pp. 1139–1148 (2017)
20. Liang, D., Krishnan, R.G., Hoffman, M.D., et al.: Variational autoencoders for collaborative filtering. In: Word Wide Web Conference, pp. 689–698 (2018)
21. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**, 436 (2015)
22. Xue, H.J., Dai, X., Zhang, J., et al.: Deep matrix factorization models for recommender systems. In: IJCAI Conference, pp. 3203–3209 (2017)

23. Zheng, L., Noroozi, V., Yu, P.S.: Joint deep modeling of users and items using reviews for recommendation. In: 10th ACM International Conference on Web Search and Data Mining, pp. 425–434 (2017)
24. Bayer, I., He, X., Kanagal, B., et al.: A generic coordinate descent framework for learning from implicit feedback. In: 26th International Conference on World Wide Web, pp. 1342–1350 (2017)
25. He, X., Zhang, H., Kan, M.Y., et al.: Fast matrix factorization for online recommendation with implicit feedback. In: 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 549–558 (2016)
26. Rendle, S., Freudenthaler, C., Gantner, Z., et al.: BPR: Bayesian personalized ranking from implicit feedback. In: Conference on Uncertainty in Artificial Intelligence, pp. 452–461 (2012)
27. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. unpublished (2014)
28. Mikolov, T., Sutskever, I., Chen, K., et al.: Distributed representations of words and phrases and their compositionality. In: Neural Information Processing Systems, pp. 3111–3119 (2013)
29. Kim, D., Park, C., Oh, J., et al.: Convolutional matrix factorization for document context-aware recommendation. In: 10th ACM Conference on Recommender Systems, pp. 233–240 (2016)