Noname manuscript No.
(will be inserted by the editor)

DFIAM:Deep Factorization Integrated Attention Mechanism for Smart TV Recommendation

Yijie Zhou[1,2] · Xuewen Shen[3] · Suiyu Zhang[3] · Dingguo Yu[1,2,3] · Guandong Xu[4]

Received: date / Accepted: date

Abstract Users are frequently overwhelmed by their uninterested programs due to the development of smart TV and the excessive number of programs. For addressing this issue, various recommendation methods have been introduced to TV fields. In TV content recommendation, auxiliary information, such as users' personality traits and program features, greatly influences their program preferences. However, existing methods always fail to take auxiliary information into account. In this paper, aiming at personality program recommendation on smart TV platforms, we propose a novel Deep Factorization Integrated Attention Mechanism (DFIAM) model, which fully takes advantage of users' personality traits, program and interaction features to construct users' preference representations. DFIAM consists of two components, FNN component and DMF component. By suitably exploiting auxiliary information, FNN component devises a feature-interaction layer to capture the low- and higher-order feature interactions, while DMF component has a field-interaction layer to acquire higher-order field interactions. The embedding layer is divided into two layers , including feature embedding layer and field embedding layer. The two components share the feature embedding layer to profile latent representations of user and program features to reduce learning pa-

✉Dingguo Yu
E-mail: yudg@cuz.edu.cn

Yijie Zhou
E-mail: yjzhou@cuz.edu.cn

Xuewen Shen
E-mail: shenxuewen@cuz.edu.cn

Suiyu Zhang
E-mail: zhangsuiyu@cuz.edu.cn

Guandong Xu
E-mail: guandong.xu@uts.edu.au

[1]Institute of Intelligent Media Technology, Communication University of Zhejiang, Hangzhou, China
[2]Key lab of Film and TV Media Technology of Zhejiang Province, Hangzhou, China
[3]College of Media Engineering, Communication University of Zhejiang, Hangzhou, China
[4]School of Computer Science, University of Technology Sydney, Sydney, Australia

rameters and computational complexity. And the field embedding layer calculated by feature embedding layer is the input of DMF component. Besides, hierarchical attention networks are applied to self-adapt the influence of each feature and effectively capture more important feature interactions. To evaluate the performance of the DFIAM model, extensive experiments are conducted on two real-world datasets from different scenarios. The results of our proposed model have outperformed the mainstream neural network-based recommendation models in terms of RMSE, MAE and R-square.

Keywords Big data · Recommendation system · Neural network · Attention mechanism · Smart TV recommendation

1 Introduction

Recent decades, with network convergence, the internet and TV broadcasting have paved the way for video service technologies such as mobile TV and website TV, which leads to the diversity of TV broadcasting services[1]. Meanwhile, for the explosive growth of programs with different ways of services, TV viewers face excessive amounts of contents and change their watching habits in accordance with the diversification of them. Therefore, it becomes more and more difficult for TV platforms to satisfy users' preferences under such situations.

At present, recommender system is developed to solve this problem. It is widely used in video [14][30], social media [17][31], music [24], and e-commerce [16][25][13], etc. And it utilizes users' history behavior to improve the effectiveness of searching and to retrieve interesting items. Similar to recommendation applied in various areas mentioned above, the algorithmic advances of recommender system are also suitable for smart TV recommendation, which helps people identify TV contents of users' interests among a large set of options available [18][3]. The approaches of recommender systems can be divided into several categories [23], including collaborative filtering (CF), content-based filtering (CBF), hybrid filtering (HF), etc. So far, many traditional recommender algorithms have made a huge success on video recommendation tasks on website TV platforms [30], such as YouTube and Netflix. However, like collaborative filtering [28][7][11], traditional recommendation algorithms still suffer from some limitations, such as data sparsity. And most of these existing approaches cannot fully exploit auxiliary information such as personality trait data (age, gender), program data (category, actors) and interaction information (location, time) to predict users' TV content preferences.

Auxiliary information plays an essential role in TV recommendation. Users' demographic data, program data and interaction data contain various helpful information, which can bridge users and related TV contents and help predict users' interests more precisely. For instance, feature Time can impact what to view. Children prefer to watch cartoons after school, adults usually watch TV in the evening, and the aged like to watch TV in the daytime. Furthermore, feature conjunctions are essential for accurate rating prediction. For example, when watching TV, the female may prefer movies and entertainment on weekends, while on weekdays is apt for short shows. If a user has the feature conjunction {female, weekend}, we would like to recommend a movie to her. What's more, when users click TV programs, it does not mean that they want the contents. In addition to predicting

the user' preference, indicators of users' satisfaction, such as the duration of a user staying, need to be taken into consideration [30].

Based on the analysis above, we propose a novel model Deep Factorization Integrated Attention Mechanism for smart TV Recommendation (DFIAM), which can take full advantage of auxiliary information and acquire higher-order interactions to get better performance of TV recommendation. The overview of our method is shown in Figure 1.

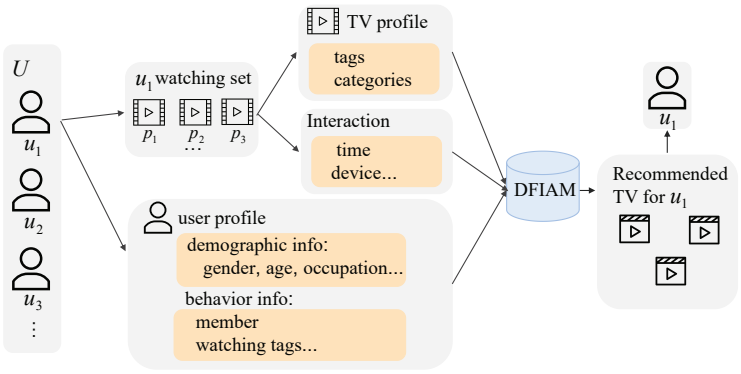

Fig. 1 Architecture overview of our system. The auxiliary information contains user profile, program profile and interaction data. Our proposed model DFIAM incorporates auxiliary information into system.

One purpose of our approach is to explore and exploit auxiliary information into the system. Based on matrix factorization, a method of collaborative filtering, our proposed model can learn the latent representations of users and programs from user-TV ratings or clicks. However, complex interactions and feature conjunctions cannot be captured by matrix factorization (MF) [12] and its variants, such as neural collaborative filtering [6] and deep matrix factorization [27]. MF series methods only estimate the ratings or clicks by the inner product of the learned latent vectors between users and TVs, which will lead to the limited recommendation. Therefore, beyond simply users' and programs' IDs feature vectors learned by MF, we introduce factorization machine (FM) [19] method to analyze users and TV contents by extracting multi-faceted features, including gender, age, occupation, summary, category and interaction attribution. Besides, FM can capture second-order feature interactions as well. Then, we adopt a deep learning framework for incorporating the features and acquire higher-order feature interactions. Furthermore, DFIAM can be divided into two parts, in which users' preference and clicking representations can be learned respectively. What's more, we employ attention mechanism to estimate the importance of different feature interactions and field interactions for the two components.

The main contributions of this work are as follows:

(1) We propose a deep neural network architecture model Deep Factorization Integrated Attention Mechanism for smart TV Recommendation, which devises a new operation in neural network by introducing attention network unit into the

union model of MF and FM, in order to incorporate the auxiliary information into the system and emphasize the weights of relative features and interactions.

(2) DFIAM is a network model with a parallel structure, and feature and field embedding layers are used during training, reducing learnable parameters and computational complexity. Meanwhile, the feature embedding vectors and field embedding vectors from the embedding layers enable feature-interaction and field-interaction layer to learn feature and field interactions, respectively.

(3) We perform extensive experiments on two real-world datasets, an open dataset of Movielens and a private dataset of television Company *. The results show the better effectiveness of the proposed method DFIAM by comparing to mainstream methods.

The remainder of this paper is organized as follows. Section 2 introduces the related works to our work. In section 3, we propose model DFIAM in detail, and we present the experimental results and discussion in section 4. And then Section 5 is the conclusion.

2 Related Work

The matrix factorization [21][12], widely used in recommendation system, is a typical and effective model of collaborative filtering, which considers users' behavior by factorizing the implicit data matrix into low-dimension latent vectors. The core idea is to map the users' and programs' vectors into a latent feature space, whose element-wise product is on behalf of users' interest. Nevertheless, MF method cannot capture complex feature representations. Recently, due to powerful representation learning abilities, deep learning methods have been successfully applied in various areas, such as natural language processing and audio recognition as well as recommender system. Neural collaborative filtering [6] is proposed to leverage a multi-layer perceptron to learn user-program implicit feedback. Simultaneously, Deep matrix factorization [27] model constructs user-program interaction matrix with explicit ratings. With user vectors and program vectors as input, a deep neural architecture is used to map the two kinds of vectors into a latent and non-linear space. However, apparent drawbacks of the above model are that they cannot make full use of auxiliary information and capture complex feature interactions [10].

To solve the problem, Factorization machine [19] is proposed by Rendle et al. to tackle more generic information. Besides, the second-order interactions can be learned between features by their inner product, which factorizes the cross weights into two latent vectors. Although FM model works well, it still suffers some limitations. One is that FM fails to consider multiple features in the same field. Field-aware FM [9], which originates from FM and PITF [20], groups a class of features into one field. Namely, each feature in FFM has different latent vectors when interacting with different features of different fields. However, like FM, FFM model is also limited to linear functions and weak in capturing potential higher-order feature interactions from raw information. To overcome the problem mentioned above, deep neural networks are introduced into FM model in recent years. It is found that FMs incorporating deep neural network can acquire better performance. NFM [5] and FNN [29] regard the second-order interaction as a bi-interaction vector, which pass through a multi-layer perceptron to extract higher-order interaction information. Wide&Deep [2], which was proposed by Google, is

a parallel model fused with deep neural network to get higher-order feature interactions. And DeepFM [4] borrow the idea of Wide&Deep by replacing LR module with FM module.xDeepFM [12] combines the explicit, implicit higher-order interaction module and traditional FM module. Besides, attention mechanism has also been introduced in recommendation model like Attention-FM [26], IFM [8] and AutoInt [22] to discriminate the importance of different feature interactions.

3 DFIAM algorithm

When users come into the system without any expressed intentions, recommender system will extract users' interests from their historical behavior and then demonstrate the most preferred programs to them. In this session, We propose an efficient neural network architecture for smart TV recommender system. The method comprises two components, Attention FNN for incorporating auxiliary information and features interactions to learn users' click representation, and Attention DMF for obtaining higher-order field interactions and learning users' preference representation. An overview structure of the model is shown in Figure 2.

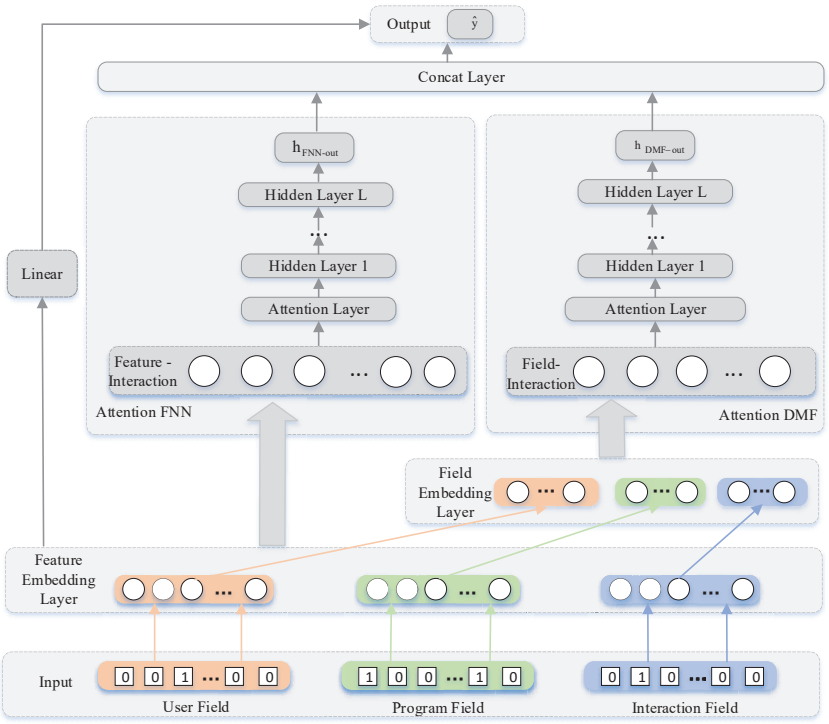

Fig. 2 Deep Factorization Integrated Attention Mechanism (DFIAM) Model. Attention FNN can extract low- and higher-order feature interaction, while Attention DMF obtains field interactions. And the field embedding vectors are the summation of feature embedding vectors of the same field from feature embedding layer.

The model we proposed references the parallel structure design of DeepFM model. Firstly, it is natural to encode categorical features by one-hot or multi-hot method based on different fields. Then all features pass through a shared embedding layer to obtain the low-dimension dense representation of each sample. Then, the learned vectors flow into two sub-modules, attention DMF component and FNN component, respectively. The left sub-module FNN component captures single second-order feature interactions and their higher-order interaction representations, while the right sub-module DMF extracts non-linear and high-order field interactions, which can reduce parameters and computational complexity. And the field interactions are developed from feature interactions. For example, a user's viewing record can contain some features, such as id, gender, age, program name, category, viewing time, etc. In practice, we classify features into three fields, namely user field, program field and interaction field, which are illustrated in Table 1. The field interactions can be acquired by field vectors computed from feature embeddings. Besides, in order to attach different weights to feature and field interactions, we design an attention unit network for both components. Lastly, we concatenate interaction embeddings and a linear embedding to feed the last prediction layer. (Section. 3.4)

Table 1 The classification of fields. According to the properties described, we group all features into three fields, including User, Program and Interaction Field.

Field	Classification	Features
User	demographics	age, gender, occupation, salary, family, character
	custom	pay for TV, activity
Program	TV type	movie, TV series, documentary, variety, cartoon, etc.
	category	history, romance, campus, fantasy, science, war, etc.
	cast	director, actors
Interaction	action	average time spent online, watching time, watching period
	others	device, location

3.1 Embedding Layer

The auxiliary information can be divided into continuous features and categorical features, which are very sparse and high-dimension. And auxiliary information is grouped by different fields. For example, feature {male, age,...} belongs to field User and feature {history, documentary,...} belongs to field Program. Therefore, we should encode the features with one-hot method for each field and then concatenate the feature vectors as the input vectors to acquire field embedding vectors. Suppose there are N features and each feature has M values. A feature is represented as $x_n = [x_1, x_2, ..., x_M]$, and $x_m \in \{0, 1\}$. We group all features into three fields, containing User, Program and Interaction Field. The step for obtaining field embedding vectors as follows:

Firstly, since feature representations are high-dimension and sparse, we employ a feature embedding layer to map them into the same low-dimension space to acquire dense real-value vectors. We transform each feature x_n into v_n. Formally, the dense vector of each feature can be defined as:

$$v_n = W_n x_n \tag{1}$$

where W_n is the weight of feature embedding layer, which can be optimized with other parameters. Next, we concentrate the features in the same field together and apply a field embedding layer to obtain the field embedding vectors. There are three fields {U, P, I}, denoting User, Program and Interaction Field. Suppose $U = [v_1^U, v_2^U, ..., v_u^U]$, $P = [v_1^P, v_2^P, ..., v_p^P]$ and $I = [v_1^I, v_2^I, ..., v_i^I]$.It can be defined as:

$$e^F = \sum_{i=1}^{n} v_i^F x_i^F \tag{2}$$

where F denotes the field index (User, Program, Interaction), and x_i^F denotes the i-th feature value in F field. In this way, we reduce data sparsity and input dimension and pave the way for further data processing. The embedding layers are illustrated in Figure 3.

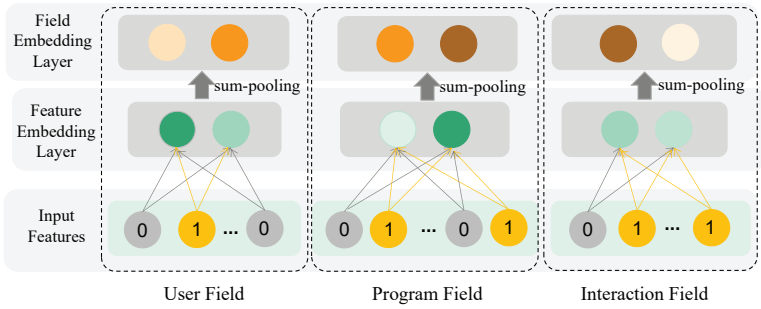

Fig. 3 The structure of embedding layers, which are divided into feature and field embedding layer. Field contains User, Program and Interaction. The feature embedding layer input is one-hot or multi-hot sparse vectors, and the output is low-dimension dense real-value feature embeddings, which are called feature embedding vectors. The feature embedding vectors of the same field are concatenated as the input of field embedding layer.

3.2 Attention FNN Component

In this session, after feature extraction for learning user and program modeling, we present Attention FNN sub-module, which utilizes attention unit under the fused model of FM with deep framework to learn the correlations of user, program features and their interactions. Meanwhile, the kernel FM with a deep network focuses on second- and higher-order interactions between different features. And attention unit focuses on different contributions of feature interactions to prediction.

When feature embedding vectors from feature embedding layer pass through feature-interaction layer, which can estimate feature interaction vectors via the inner product. The vector is expressed as $v_i x_i \odot v_j x_j$, where x_i and v_i denote the i-th feature value and the latent feature vector, respectively. $\odot$ is the element-wise product of two vectors. And then, we employ an attention layer after feature-interaction layer to assign different weights on interaction vectors, which can be

defined as:

$$f_{attention} = \sum_{i=1}^{n} \sum_{j=i+1}^{n} a_{ij}(v_i x_i \bigodot v_j x_j) \tag{3}$$

where a_{ij} is the attention weight for the feature interaction, coming from a feed-forward network. We regard the network as an activation network. The input of the network is the second-order feature interaction vectors we extract in feature-interaction layer. Formally, the attention network can be defined as:

$$a_{ij}^{1} = Relu(W(v_i x_i \bigodot v_j x_j) + b) \tag{4}$$

$$a_{ij} = softmax(a_{ij}^{1}) \tag{5}$$

where W, b denote the weight matrix and bias of attention network. Then, we use the softmax function to normalize the attention weights for limiting the parameters to space of [0,1] and the sum of all attention scores is 1. The output of the attention layer is a vector of K-dimension, which contains all second-order feature interactions by assigning them different weights and compute the sum $f_{attention}$ of these vectors. After the attention layer, for obtaining higher-order feature interactions, we design a stack of fully connected layers, which can be able to learn higher-order feature interactions. Formally, the MLP layer can be defined as:

$$h_{FNN-out} = f(W_L^T(...f(W_1^T f_{attention} + b_1)...) + b_L) \tag{6}$$

where L denotes the number of hidden layers, W_i, b_i are the weights and bias of the i-th layer, respectively, f denotes non-linear activation function, such as Sigmoid, Tanh and ReLU.

3.3 Attention DMF Component

In this part, the sub-module is called Attention DMF module, which focuses on learning field interactions and users' preference representation. Specially, we focus on the field interactions between the field of User, Program and Interaction. The user field embedding vectors represent the users' information, and Program field embedding vectors are on behalf of the program's general information. Iteration field embedding vectors denote users' action representations. Let the conjunction of User field and Program field indicate users' long-term taste factors, and the combination of Program and Interaction field represents the short preferences. Then we present a multi-layer perceptron fused with attention mechanism based on MF to capture the non-linear and high-order field interactions by users' behavior. Compared to feature interactions, learning field interactions can reduce computational complexity and avoid overfitting.

Firstly, we compute field interactions by feature representations obtained from feature embedding layer. There are three fields {U, P, I}, denoting User, Program and Interaction Field, respectively. Then the vectors pass through field-interaction layer, which can estimate field interaction vectors via the inner product. We use equation(2) to compute the field embedding vectors. And then, the field interactions can be formalized as:

$$S_{UP} = e^{U} \bigodot e^{P} \tag{7}$$

$$S_{PI} = e^U \bigodot e^I \tag{8}$$

$$S_{UI} = e^P \bigodot e^I \tag{9}$$

Where S_{UP} denotes interactions between user features and program features, representing users' long-term taste factor. S_{PI} is regarded as users' short-term taste factors, the interactions between program features and interaction features. And S_{UI} is the interaction of user features and interaction features.

Then, because the attention mechanism can discriminate the importance of different field interactions, DMF module employs attention unit over the field interactions. We concatenate the field interactions, which is defined as $Z_1 = concat(S_{UP}, S_{PI}, S_{UI})$. The framework is formalized as:

$$g_A = g(W_A^T Z_1 + b_A) \tag{10}$$

$$h_{DMF-out} = f(W_L^T(...f(W_1^T concat(Z_1, g_A) + b_1)...) + b_L) \tag{11}$$

where $concat(\cdot)$ denotes the concatenation operation, g is a softmax function, W_A,b_A denote the weights and bias of attention unit network, W_i and b_i are the weights and bias of the i-th hidden layer. For activation function f of the hidden layer, we can freely choose ReLU, Tanh and Sigmoid. This paper uses ReLU as the active function for each layer, which is well-suited for sparse data and making model less likely overfitting.

3.4 Prediction Layer

Finally, we combine the two sub-module and a linear module to predict the user's rating score. The output vector $h_{FNN-out}$ from attention FNN module is concatenated with the output vector $h_{DMF-out}$ from attention DMF module, which is regarded as the input vector of the last hidden layer. Formally, with a linear module, the predicted output can be define as :

$$\hat{y} = w_0 + \sum_{i=1}^{n} w_i x_i + \sigma(W_{pre} concat(h_{FNN-out}, h_{DMF-out}) + b_{pre}) \tag{12}$$

where W_{pre}, b_{pre} denote the weights and bias of the last hidden layer, σ denotes the non-linear activation function, And w_0 is the global bias, x_i and w_i denote the i-th feature value and i-th feature weights of the linear model, respectively. Lastly, we apply a sigmoid function to make the last prediction.

Because the prediction result is a rating score, our loss function is a mean absolute error for the smart TV ranking prediction task, which is defined as follows:

$$L = \frac{1}{M} \sum_{i=1}^{M} \sqrt{(y_i - \hat{y}_i)^2} + \epsilon ||\theta||^2 \tag{13}$$

where M is the number of user-program interaction records in the training set, θ is the parameter set of DFIAM, and ϵ is the regularization parameter. We set a threshold based on the ratio of the user's viewing time to the program duration. In terms of the negative sample selection, we choose program which the threshold is less than 0.5.

4 Experiments

In this section, we conduct the experiment on two real-world datasets to evaluate the effectiveness of our proposed method DFIAM and we intend to answer the following research questions:
RQ1 How do the key hyper-parameters of DFIAM (embedding size, attention size, learning rate, dropout) influence its performance?
RQ2 Does the hybrid model of DFIAM outperform mainstream methods for rating prediction?
RQ3 Is it effective to take advantage of multi-faceted features from the auxiliary information?
RQ4 What's the effect of the hierarchical attention layers?
RQ5 How does the sparsity of datasets influence the performance of DFIAM?
RQ6 Is field-level feature interaction beneficial to the performance of DFIAM?

4.1 Experimental Settings

Data description
We conduct our experiment and evaluate the effectiveness of our proposed model with two real-world datasets. One is an open dataset MovieLens, and the other is a private dataset from company *. The statistics of the two datasets are given in Table 2.

Table 2 The description of the two real-world datasets, MoviesLens and Company * Data

Dataset	Users	Program	Interaction	Sparsity
MovieLens	6,040	3,706	1,000,209	95.53%
Company* Data	29,217	3,587	317,671	99.69%

MovieLens.The MovieLens datasets are released by GroupLens[3]. These movie ratings datasets have been widely used in recommendation research. There are several versions of these datasets. In our experiment, we use the version of one million records, which contains 1,000,209 ratings of 6,040 users on 3,706 items, scored in the [1,5] range. And we normalize it to [0,1] range. Each record contains 24 features, including userid, gender, age, programid, category, etc.

Company* Data. Company* Data is a real program dataset with over 10 million samples from a famous television company. The original dataset is huge but high sparse, and the records lack side information. Therefore, we filter the dataset and remain only users with at least five records. This result in the data that only contains 317,671 records of 29,217 users' watching 3,587 programs. Each watching record includes 68 features, such as user ID, program ID, age, occupation, program categories, viewing time, program duration, etc. However, the click of TV program cannot fully represent users' preferences. So we regard the ratio of user's viewing time to program duration as his/her satisfaction score, and the scores are in the [0,1] range.

Evaluation Metrics
For accuracy, we use three metrics for our experiment evaluation: RMSE(Root Mean Square Error), MAE(Mean Absolute Error) and R2(R-square), which are widely used for evaluating regression tasks such as recommendation with ratings. A lower RMSE score, a lower MAE score or a higher R2 score indicate better performance. We take 80% users' historical records for training dataset, and the remaining 20% for testing dataset according to each user of the two datasets.
Baseline
We compare our proposed DFIAM model with the following 6 baseline models.

- MF [12]: This is the original Matrix Factorization which is a common model of collaborative filtering. And in this part, we rewrote the model using a neural network architecture.
- NCF [6]: This is a method that fuses deep neural network into original MF model.
- FM [19]: This is the original FM to learn the second-order interactions automatically. In this part, like MF, we rewrote it with the way of neural network architecture.
- NFM [5]: It extracts the higher-order feature interactions on a bi- interaction pooling layer by deep neural network.
- AFM [26]: It learns the importance of feature interactions with an attention mechanism, which can assign different weights to second-order interactions.
- DeepFM [2]: It extends FM model with a neural framework. The model consists of a shallow component to extract low-order feature interactions and a deep component to acquire higher-order interactions.
- xDeepFM [15]: It is a deep model which combines the explicit, implicit higher-order interaction module with a compressed interaction network.
- AutoInt [22]: It learns higher-order feature combinations with multi-head self-attentive neural networks.

Parameter Settings
To evaluate our model on Movielens, we reference the parameter settings in AutoInt. The embedding size and attention size are set to 16 and the hidden unit is 32. The deep part consists of four feed-forward layers, and each layer with 100 hidden units. ReLU activation is adopted for each deep layer and the attention unit network. To prevent overfitting, the dropout is 0.5 for each layer and Adam for the optimizer.

4.2 Hyper-parameter Investigation (RQ1)

We study the impact of different key hyper-parameters on Company* Data of DFIAM in this section, including (1) embedding size, attention size and learning rate; (2) dropout ratio; (3)number of hidden layer.
Embedding size, Attention size and Learning rate
We explore the effect of embedding size, attention size and learning rate. The embedding size is equal to attention size, which is searched between [16,32,64] and the learning rate is in [0.01,0.02,0.03]. And then, we have manually tested them on our implementation of our model to select the best combination. As shown in Table 3 ,for the Company* Data, the performance is better when embedding

size, the attention size and leaning rate are [32,32,0.02],respectively. Besides, by grouped embedding size and attention size, the model has the best performance when learning rate is at 0.02.

Table 3 Performance comparison with the combination of embedding size, attention size and learning rate

Parameters	Company* Data		
	RMSE	MAE	R2
[16,16,0.01]	0.0934	0.0596	0.8754
[16,16,0.02]	0.0899	0.0577	0.8846
[16,16,0.03]	0.0905	0.0577	0.8832
[32,32,0.01]	0.0864	0.0554	0.8862
[32,32,0.02]	**0.0848**	0.0542	**0.8972**
[32,32,0.03]	0.0869	0.0557	0.8923
[64,64,0.01]	0.0866	0.0551	0.8733
[64,64,0.02]	0.0849	**0.0534**	0.8781
[64,64,0.03]	0.0855	0.0548	0.8764

Dropout Ratio

We then explore whether the dropout is beneficial to our proposed model DFIAM. We tune the dropout ratio from between {0.1,0.2,...,0.9}. Figure 4 shows the performance of different dropout ratios, which illustrates that the dropout slightly affects our proposed model's performance. And we choose the relatively optimal dropout ratio of 0.5 on Company* Data.

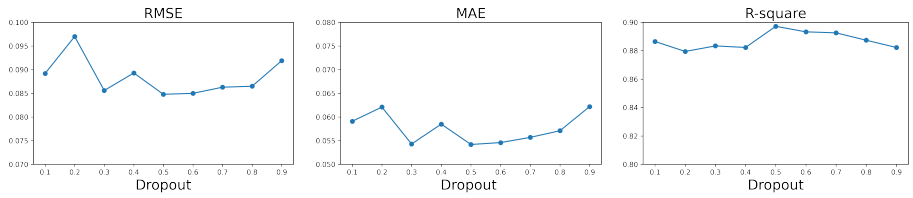

Fig. 4 RMSE, MAE and R-square comparison of dropout ratio.

Number of layers

Table 4 shows that the increasing number of hidden layers improve the model performance at the very beginning. However, when the number of depth layers is more than three layers, the performance is a little flat and even degraded, caused by overfitting.

All in all, We design the deep part with the following parameter settings. The deep part consists of three hidden layers, and the numbers of neurons for each layer are 64,32 and 16, respectively. ReLU activation is adopted for each deep layer and the attention unit network. The dropout is 0.5 for each layer and Adam for optimizer. What's more, the embedding size is equal to attention size at 32, and the learning rate is at 0.02.

Table 4 Performance comparison with number of hidden layers

Number of Layers	Company* Data		
	RMSE	MAE	R2
2	0.0961	0.0613	0.8788
3	**0.0848**	**0.0542**	**0.8972**
4	0.0868	0.0551	0.8965
5	0.0988	0.0612	0.8765

4.3 Performance Comparison(RQ2)

In this section, we compare the performance of DFIAM with the baseline models on two datasets. Table 5 shows the performance of the compared models and our proposed model DFIAM. It is obviously seen that our proposed model has the best performance, achieving 0.1359, 0.1062 and 0.6287 in RMSE, MAE and R2 on MovieLens(0.0848, 0.0542 and 0.8972 on Company * Data). We also have the following observations:

Table 5 Comparison between the proposed method DFIAM and baselines

Model	MovieLens			Company* Data		
	RMSE	**MAE**	**R2**	**RMSE**	**MAE**	**R2**
MF	0.1905	0.1546	0.2514	0.2021	0.1454	0.4299
NCF	0.1785	0.1410	0.3607	0.1812	0.1235	0.5682
FM	0.1805	0.1437	0.4059	0.1818	0.1260	0.6425
NFM	0.1785	0.1385	0.3616	0.1790	0.1260	0.5787
AFM	0.1452	0.1133	0.5906	0.1090	0.0768	0.8644
DeepFM	0.1551	0.1218	0.5098	0.1191	0.0820	0.8286
xDeepFM	0.1401	0.1094	0.6070	0.1109	0.0724	0.8227
AutoInt	0.1367	0.1071	0.6191	0.0866	0.0581	0.8915
DFIAM	**0.1359**	**0.1062**	**0.6287**	**0.0848**	**0.0542**	**0.8972**

Firstly, due to the deep neural network, NCF outperforms MF by 6.3% and 10% in terms of RMSE (8.7% and 15% in terms of MAE) on MovieLens and Company * Data. Similarly, NFM is also better than FM on two datasets. Therefore, deep network model improves the performance of the predictive result significantly. Secondly, FM performs better than MF with 5.2% and 7% in terms of RMSE(10.4% and 13.3% in terms of MAE) on the two datasets. Comparing MF and FM, We infer that the second-order feature interactions make outstanding contributions to the prediction. Besides, learning higher-order interactions is also effective because DeepFM performs better than FM in terms of RMSE, MAE and R2. Therefore, learning both low- and higher-order feature interactions can greatly improve the accuracy of prediction.

Lastly, according to the comparison DFIAM and AFM with other models, the method with attention mechanism make a remarkable contribution to the better performance. We conducted the part carefully in Section 4.5.

4.4 Feature Contribution(RQ3)

The auxiliary information has an essential contribution to our model. In this section, we aim to explore the influence of the auxiliary information and validate the effect of multi-faceted features on the system. Firstly, We divide all auxiliary features into three groups as mentioned above, which are user group (containing user's age, occupation, etc.), program group (containing type and name) and interaction group (including time, device). Then we test the DFIAM model by removing one feature group each time. And the result is shown in Table 6. It can be seen that DFIAM with extracted features performs better than without auxiliary information. Therefore, We observe that all auxiliary features, whether user and group-profiling, positively contribute to our TV recommendation model. What's more, the performance is better when only one group was taken out than all groups. Therefore we conclude that the auxiliary information greatly contributes to improving the performance of TV recommendation.

Table 6 Performance of different groups

Field	MovieLens			Company * Data		
	RMSE	**MAE**	**R2**	**RMSE**	**MAE**	**R2**
Auxiliary data	**0.1359**	**0.1062**	**0.6287**	**0.0848**	**0.0542**	**0.8972**
User-interaction	0.1380	0.1081	0.6190	0.0896	0.0595	0.8946
User-program	0.1364	0.1085	0.6106	0.0868	0.0547	0.8970
Program-interaction	0.1363	0.1063	0.6265	0.0875	0.0557	0.8966
Only User	0.1381	0.1080	0.6183	0.0914	0.0600	0.8894
Only program	0.1370	0.1070	0.6222	0.0895	0.0562	0.8969
Only interaction	0.1392	0.1085	0.6106	0.0902	0.0572	0.8929
No auxiliary data	0.1393	0.1091	0.6099	0.0935	0.0622	0.8853

4.5 Impact of the Attention Network (RQ4)

The attention mechanism of DFIAM plays an essential role in improving the accuracy of prediction. In this section, we focus on analyzing the effect of the attention network on model DFIAM. And We test the DFIAM model on MovieLens and Company * Data with attention unit and without attention unit. Figure 5 shows the performance of comparison. DFIAM with attention network (denoted as solid lines) performs better than without attention (marked as dotted lines) on both datasets, regardless of RMSE, MAE, or R2. In short, we conclude that attention score can increase weights of relative feature and field interactions in different situations to enhance the capacity of TV content recommendation.

4.6 Effects of Data Sparsity (RQ5)

We now explore how the data sparsity influence the performance of our model DFIAM. Specially, the data sets are generated by filtering users with low watching frequency,which is set as {5,10,20,30,40,50} on Company* Data and {5,30,40,50,60,70} on MovieLens with the sparsity of {99.31%, 99.25%, 98.89%, 98.59%, 98.20%,

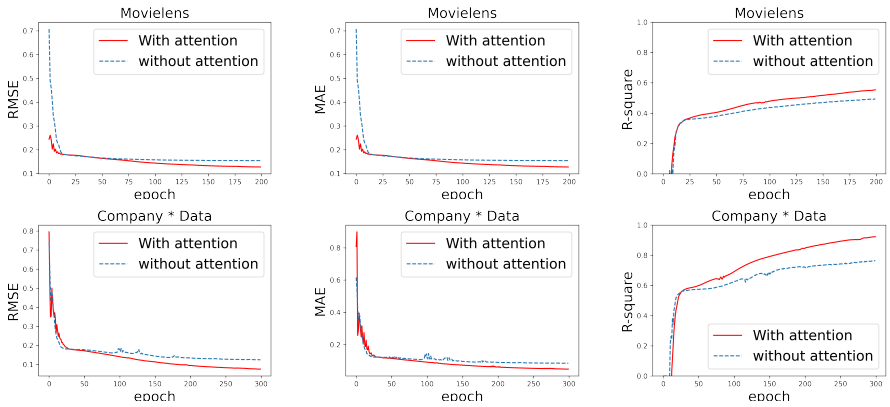

Fig. 5 The Comparison of DFIAM with attention unit and without attention unit

97.57%} and {95.53%,95.18%,94.70%,94.23%,,93.77%,93.37%},respectively.The result is shown in Table 7 , which illustrate that reducing sparsity, to some extent, is beneficial to improve the performance of our model.

Table 7 Performance of different sparsity

Sparsity	MovieLens			Sparsity	Company* Data		
	RMSE	**MAE**	**R2**		**RMSE**	**MAE**	**R2**
95.53%	0.1359	0.1062	0.6287	99.31%	0.0848	0.0542	0.8972
95.18%	0.1351	0.1050	0.6247	99.25%	0.0850	0.0565	0.8929
94.70%	0.1360	0.1053	0.6292	98.89%	0.0832	0.0543	0.8915
94.23%	0.1342	0.1043	0.6350	98.59%	0.0832	0.0527	0.8936
93.77%	0.1331	0.1034	0.6398	98.20%	0.0809	0.0501	0.9032
93.37%	0.1335	0.1037	0.6376	97.57%	0.0807	0.0511	0.9045

4.7 Effect of Field Interactions(RQ6)

In this section, we focus on analyzing the effect of field interactions on our model DFIAM. Generally, more feature combinations can mine more helpful information and may have better performance. Therefore, we take advantage of the features and their combinations. Besides, organizing the feature representations of the same field can reduce the complexity of our model. To explore the effect of field interactions, we aim to test the model with only feature interactions on MovieLens and Company* Data. Because our model has two sub-module, we add the field interactions to each submodule separately. Lastly, we apply field interactions on both submodules.

As shown in Table 8, the performance of only using feature interactions is better than others on RMSE, MAE and R2. However, the only feature interaction is more time-consuming. All in all, we choose the model with field interactions only applying in attention DMF as our DFIAM model according to effectiveness and efficiency.

Table 8 Performance comparison with field interactions. "Both" in the table means both submodules apply the field interactions."Attention DMF" means only this submodule uses field interaction.

Applying submodule	MovieLens				Company* Data			
	RMSE	**MAE**	**R2**	**Time(s)**	**RMSE**	**MAE**	**R2**	**Time(s)**
Both	0.1510	0.1184	0.5436	2986	0.0917	0.0586	0.8787	1933
Attention DMF	0.1359	0.1062	0.6287	3376	0.0848	0.0542	0.8972	2537
Attention FNN	0.1428	0.1117	0.5920	3669	0.0856	0.0514	0.8761	2886
None	0.1298	0.1008	0.6878	4211	0.0798	0.0505	0.9081	3569

5 Conclusion

In this paper, we propose a novel model called Deep Factorization Integrated Attention Mechanism(DFIAM) for personalized program recommendation on smart TV platforms, which aims to take advantage of multi-faceted features from auxiliary information, learning lower- and higher-order interactions to improve the performance. Besides, we integrate the attention mechanism into our model to assign different weights for feature and field interactions in different situations. Furthermore, embedding layer is divided into feature embedding layer and field embedding layer for attention FNN component and attention DMF component, respectively. Lastly, we conducted extensive experiments on two real-world datasets (MovieLens and Company* Data) and the experimental results demonstrated our proposed method outperforms mainstream models in smart TV recommendation.

In the future, because the clicks and rates fail to represent users' preferences fully, we will explore the multi-task framework, which can integrate users' different behavior to predict users' diverse interests. Moreover, it is interesting to extend the smart TV recommendation to multi-media recommendation with multi-modal data, such as the program summary, user review, video and audio, containing rich semantic information.

Acknowledgements This work was supported by the Key Research and Development Program of Zhejiang Province, China(Grant No.2019C03138).Dingguo Yu is the corresponding author(yudg@cuz.edu.cn).

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. Asabere, N.Y., Acakpovi, A.: Roppsa: Tv program recommendation based on personality and social awareness. Mathematical Problems in Engineering **2020** (2020)
2. Cheng, H.T., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhye, H., Anderson, G., Corrado, G., Chai, W., Ispir, M., et al.: Wide & deep learning for recommender systems. In: Proceedings of the 1st workshop on deep learning for recommender systems, pp. 7–10 (2016)
3. F., MAXWELL, HARPER, JOSEPH, A., KONSTAN: The movielens datasets: History and context. Acm Transactions on Interactive Intelligent Systems **5**(4), 1–19 (2015)
4. Guo, H., Tang, R., Ye, Y., Li, Z., He, X.: Deepfm: a factorization-machine based neural network for ctr prediction. arXiv preprint arXiv:1703.04247 (2017)

5. He, X., Chua, T.S.: Neural factorization machines for sparse predictive analytics. In: Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval, pp. 355–364 (2017)
6. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. In: Proceedings of the 26th international conference on world wide web, pp. 173–182 (2017)
7. He, X., Zhang, H., Kan, M.Y., Chua, T.S.: 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, pp. 549–558 (2016)
8. Hong, F., Huang, D., Chen, G.: Interaction-aware factorization machines for recommender systems. Proceedings of the AAAI Conference on Artificial Intelligence **33**, 3804–3811 (2019)
9. Juan, Y., Zhuang, Y., Chin, W.S., Lin, C.J.: Field-aware factorization machines for ctr prediction. In: Proceedings of the 10th ACM Conference on Recommender Systems, pp. 43–50 (2016)
10. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
11. Koren, Y.: Factorization meets the neighborhood: A multifaceted collaborative filtering model. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24-27, 2008, pp. 426–434 (2008)
12. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**(8), 30–37 (2009)
13. Li, W., Xu, B.: Aspect-based fashion recommendation with attention mechanism. IEEE Access **8**(8), 141814–141823 (2020)
14. Li, Y., Liu, M., Yin, J., Cui, C., Nie, L.: Routing micro-videos via a temporal graph-guided recommendation system. In: Proceedings of the 27th ACM International Conference on Multimedia, pp. 1464–1472 (2019)
15. Lian, J., Zhou, X., Zhang, F., Chen, Z., Xie, X., Sun, G.: xdeepfm: Combining explicit and implicit feature interactions for recommender systems. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1754–1763 (2018)
16. Ni, Y., Ou, D., Liu, S., Li, X., Ou, W., Zeng, A., Si, L.: Perceive your users in depth: Learning universal user representations from multiple e-commerce tasks. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 596–605 (2018)
17. Phelan, O., Mccarthy, K., Smyth, B.: Using twitter to recommend real-time topical news. In: Proceedings of the third ACM conference on Recommender systems, RecSys 2009, New York, NY, USA, October 23-25, 2009, pp. 385–388
18. Pyo, S., Kim, E., Kim, M.: Automatic and personalized recommendation of tv program contents using sequential pattern mining for smart tv user interaction. Multimedia Systems **19**(6), 527–542 (2013)
19. Rendle, S.: Factorization machines. In: 2010 IEEE International Conference on Data Mining, pp. 995–1000. IEEE (2010)
20. Rendle, S., Schmidt-Thieme, L.: Pairwise interaction tensor factorization for personalized tag recommendation. In: Proceedings of the third ACM international conference on Web search and data mining, pp. 81–90 (2010)
21. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th international conference on World Wide Web, pp. 285–295 (2001)
22. Song, W., Shi, C., Xiao, Z., Duan, Z., Xu, Y., Zhang, M., Tang, J.: Autoint: Automatic feature interaction learning via self-attentive neural networks. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, pp. 1161–1170 (2019)
23. Véras, D., Prota, T., Bispo, A., Prudêncio, R., Ferraz, C.: A literature review of recommender systems in the television domain. Expert Systems with Applications **42**(22), 9046–9076 (2015)
24. Wang, D., Zhang, X., Yu, D., Xu, G., Deng, S.: Came: Content-and context-aware music embedding for recommendation. IEEE Transactions on Neural Networks and Learning Systems (2020)

25. Wen, H., Zhang, J., Wang, Y., Lv, F., Bao, W., Lin, Q., Yang, K.: Entire space multi-task modeling via post-click behavior decomposition for conversion rate prediction. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 2377–2386 (2020)
26. Xiao, J., Ye, H., He, X., Zhang, H., Wu, F., Chua, T.S.: Attentional factorization machines: Learning the weight of feature interactions via attention networks. arXiv preprint arXiv:1708.04617 (2017)
27. Xue, H.J., Dai, X., Zhang, J., Huang, S., Chen, J.: Deep matrix factorization models for recommender systems. In: Twenty-Sixth International Joint Conference on Artificial Intelligence, vol. 17, pp. 3203–3209. Melbourne, Australia (2017)
28. Zhang, H., Shen, F., Liu, W., He, X., Luan, H., Chua, T.S.: Discrete collaborative filtering. In: Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, pp. 325–334 (2016)
29. Zhang, W., Du, T., Wang, J.: Deep learning over multi-field categorical data. In: European Conference on Information Retrieval, pp. 45–57. Springer (2016)
30. Zhao, Z., Hong, L., Wei, L., Chen, J., Nath, A., Andrews, S., Kumthekar, A., Sathiamoorthy, M., Yi, X., Chi, E.: Recommending what video to watch next: a multitask ranking system. In: Proceedings of the 13th ACM Conference on Recommender Systems, pp. 43–51 (2019)
31. Zhou, X., Yue, X., Li, Y., Josang, A., Cox, C.: The state-of-the-art in personalized recommender systems for social networking. Artificial Intelligence Review **37**(2), 119–132 (2012)