A Novel Tensor Decomposition Networks for Recommender Systems via Adaptive Moment Estimation Method

Xunpeng Xia (191550053@st.usst.edu.cn)
University of Shanghai for Science and Technology
Rongfu Zhang
University of Shanghai for Science and Technology
Xufeng Yao
Shanghai University of Medicine and Health Sciences
Gang Huang
Shanghai Key Laboratory of Molecular Imaging
Tiequn Tang
University of Shanghai for Science and Technology

Article

Keywords:

Posted Date: May 18th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1638590/v1

A Novel Tensor Decomposition Networks for Recommender Systems via Adaptive Moment Estimation Method

Xunpeng Xia[1,*], Rongfu Zhang[1], Xufeng Yao[2], Gang Huang[3,4], and Tiequn Tang[1]

[1]School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai, 200093, China
[2]College of Medical Imaging, Shanghai University of Medicine and Health Sciences, Shanghai, 201318, China
[3]Shanghai Key Laboratory of Molecular Imaging, Jiading District Central Hospital Affiliated Shanghai University of Medicine and Health Sciences, Shanghai, 201318, China
[4]Shanghai Key Laboratory of Molecular Imaging, Zhoupu Hospital, Shanghai University of Medicine and Health Sciences, Shanghai, 201318, China
[*]191550053@st.usst.edu.cn

ABSTRACT

The tag, text, and other multiple auxiliary information are acquired in the social networks. The multi-source auxiliary information is heterogeneous information. A large amount of heterogeneous information would result in extremely difficult to analyze. This paper proposes a deep learning network for multi-heterogeneous information processing. The main idea of the our deep learning network is threefold: (1) using tensor decomposition algorithms to process standard encoded data; (2) mining potential factor from heterogeneous data using multi-layer learning networks. Symmetric non-negative potential factor optimization algorithm can effectively predict the missing values of high dimensional sparse data; (3) adaptive moment estimation optimization algorithm is used to replace the traditional first-order optimization algorithm for stochastic gradient descent(SGD) process. The aggregation of heterogeneous data is essentially a minimization problem of multiple parameters. The rate of the model convergence can be improved by adaptive moment estimation in the training stage. Simultaneously, it can solve the problem of parameter optimization of large-scale data and processing of non-stationary targets. Finally, experimental results on public vaildation datasets are given to verigy the effectiveness of our proposed blend network.

Please note: Abbreviations should be introduced at the first mention in the main text no abbreviations lists. Suggested structure of main text (not enforced) is provided below.

1 Introduction

Currently, recommender system is a very active area of research which has a wide range of applications in industry. For example, literature[1] introduces the related technologies and improved algorithms of existing recommendation systems. In this section, the recommendation algorithms used by several advanced recommender systems related to the work of this paper will be analyzed, and the problems existing in the current research on recommender systems will be briefly described. In the social network, these information usually comprise a huge amount of entities object[2,3], where each object has a undirected relationship denoting special non-negative feature. However primarily to great difficulties in obtaining complete relationships between each object, these object information contain much missing information, described by high-dimensional, symmetric and spares tensor[4,5]. For example, recommender systems consisting of quite a few nodes can be a typical tensor, where each pair tag denotes the non-negateve relative distance between each elements[6]. In spite of the extreme sparsity in these tensor, they contain rich attribute information with a variety of mode, such as the preference information of user and the tag information of item in the recommender system. If reliable predictions for the missing data based on the few known information, it is a important method for extract the interactive relationship features between entities tensor data[7].

The main elements of the social tagging system used in the recommendation system include User, Item, Tag,which defined as: $Y = (U, I, T, S)$ U is User; I is Item; T is Tag; S is the interaction between the User, Item and Tag $S \in U \times I \times T$[8]. However, current recommendation algorithms[9] mostly study the relationship between User and Item, such as matrix factorization model. Matrix decomposition can effectively explore the corresponding features of User and Item and the potential relationship between them. Therefore, in the standard matrix model, the recommendation task can be abstractly expressed as the process of parsing two User-Item matrices with partial default values. Therefore, although the current recommendation

algorithm has good practicability, it cannot directly deal with the three-dimensional relationship of the social tagging system. In order to use the tag information of tags to improve the performance of the recommendation system, Symeonidis et al.[10] first proposed to complete the high-dimensional data in the form of tensors, and obtained the prediction score through the tensor decomposition algorithm. And Rendle et al.[11] improved it and proposed the PITF (Pairwise Interaction Tensor Factorization) algorithm. The algorithm is a Tucker decomposition with special linear time complexity, which mainly considers the interaction between pairs of third-order data, and has better performance in recommendation performance. In addition, to solve the model training problem, the selection of optimization algorithm is different from the traditional algorithm. The additive gradient descent(AGD) optimization algorithm is a widely method in deep learning owing to its easy manipulation[12]. As introducted by Luo et al.[5], for an symmetric, high-dimensional, and sparse (SHiDS) matrices, the computational cost of the non-negative latent factor model with AGD is only dependent on linear manner. In spite of the AGD algorithm enjoys a low computation cost, it suffers from a low convergence rate because it easily traps into the local optimum[13]. To tackle this problem[12, 14], the stochastic gradient descent(SGD) algorithm and its variants like mini-batch gradient descent(MBGD) algorithm have been proposed, which main idea is to stochastically select a mini part of instances at each training stage until it reaches the convergence[13]. In order to deal with a large amount of heterogeneous data, an improved Adam optimization algorithm is used in this paper. A large amount of heterogeneous multi-source data is processed through the fusion of the two networks.

From the research above, it can be concluded that most traditional recommendation algorithms use shallower models for prediction, and still rely on a lot of manual intervention during feature extraction, so it is difficult to effectively learn the deep implicit representation between users and items. Moreover, the traditional recommendation algorithm assumes that user interests and item attributes are statically related, which does not conform to the current complex and changeable actual situation. By integrating multi-source heterogeneous data, such as social relations, user information, item attributes and tags and other auxiliary information, deep learning models can effectively learn more abstract and higher-dimensional users, items to be recommended and tags. At the same time, the multi-layer neural network structure model can also more effectively extract the nonlinear structural features of the interaction between users and items[15]. Compared with the deep learning model, the traditional recommendation[16] method has the advantages of simple implementation and strong interpretability of the prediction results. Therefore, combining deep learning with existing traditional recommendation algorithms can integrate the advantages of both algorithms. In order to verify the hybrid recommendation algorithm proposed in this paper, two movie datasets, Movielens-1M and Nexfix-3M are used for training, and the performance is compared with related recommendation algorithms to verify that the performance of the hybrid recommendation algorithm precises better in the social tag recommendation system.

2 Hybrid recommendation based on tensor decomposition and deep learning

There are corresponding advantages and disadvantages for any single recommendation algorithm, and by integrating different recommendation algorithms to obtain mixed recommendation results, it can show better performance. The usual mixing mechanisms are roughly divided into three types, namely pre-fusion, mid-fusion, and post-fusion. In this paper, the post-fusion mixing strategy is used to combine the recommendation results generated by the two recommendation algorithms, tensor decomposition and deep learning, with weighted mixing, cascade and feature combination to produce the final recommendation results

To improve the recommendation effect, tensor decomposition and deep learning models are mixed. As shown in Figure 1, this model is composed of two parts, tensor decomposition and deep learning. They correspond to two different inputs respectively. The input of the tensor decomposition part is the scoring data, and the input of the deep learning part is the auxiliary data information corresponding to the third order of users, items and labels. The two parts are trained to give scores or corresponding features respectively, and then the results of the two parts are mixed to obtain a comprehensive score.

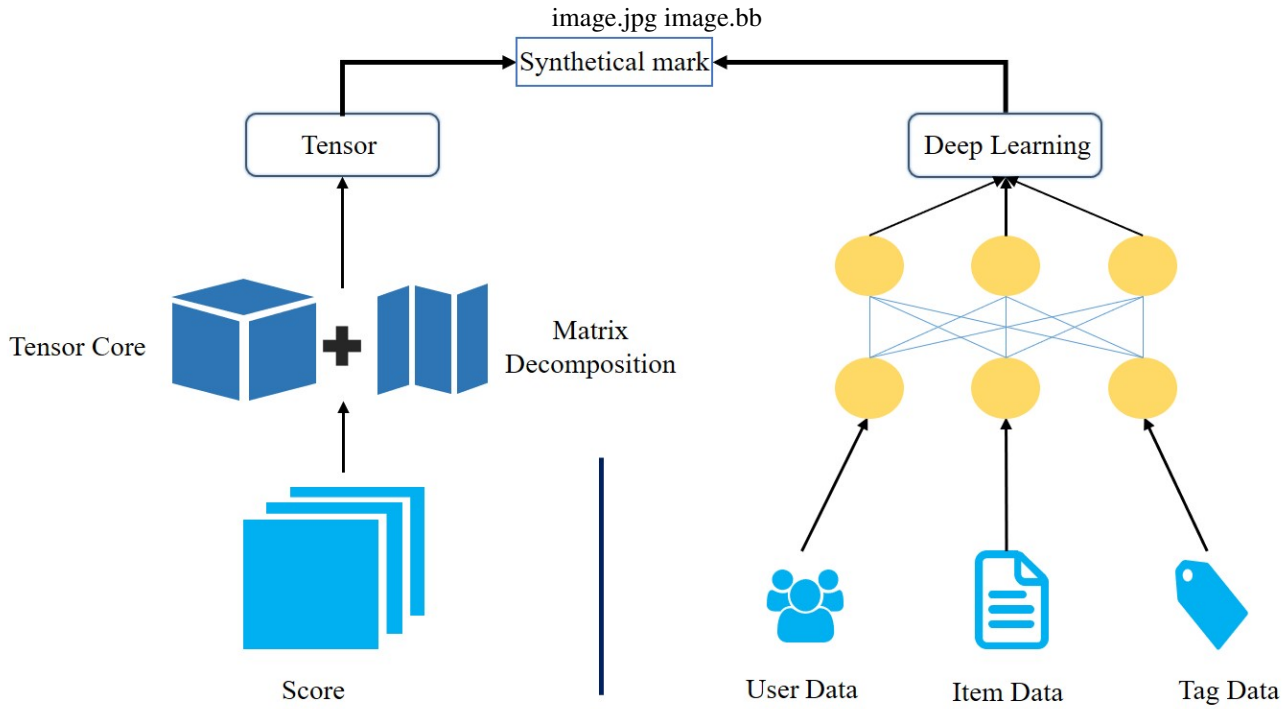

Figure 1. Schematic diagram of tensor decomposition and deep learning hybrid recommendation model

3 Tensor decomposition part

The tensor decomposition part belongs to the factorization machine[9], which learns the interaction between higher-order data by decomposing and reorganizing the dataset. The traditional collaborative filtering calculates the similarity between the features as the recommendation basis, but the effect is poor when the data set is extremely sparse. The use of the factorization machine model can effectively obtain the relationship between the features. Compared with Collaborative filtering algorithm has a certain improvement. Theoretically, the factorization machine can learn the interaction relationship between higher-order features, but due to the limitation of the data set, this paper only deals with the third-order feature relationship, which is called the internal relationship between users, items and labels.

3.1 Tensor Decomposition Algorithm

Y_{ijk} represents the target tensor, X_{ijk} representing the approximate representation tensor of the target tensor.

$$Y_{ijk} \approx X_{ijk} \tag{1}$$

The tensor decomposition model used in this paper is the Tucker decomposition model, which is a high-order principal component analysis method, which decomposes a target tensor into the form of a core tensor and a product of three matrices. The decomposition model as follow:

$$Y_{ijk} \approx \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} U_{ip} T_{jq} I_{kr} S_{ijk} \tag{2}$$

Among them, U_{ip}, T_{jq} and I_{kr} are three decomposition matrices (usually orthogonal matrices), generally defined as principal components in each dimension.S_{ijk} is the core tensor, representing the underlying interrelationships in each dimension.

The error calculation of tensor decomposition is expressed by the difference between the target tensor and the approximate tensor. In order to prevent overfitting, a regular term is added, and the loss function is expressed by formula (2):

$$L = argmin \sum_{ijk} (y + ijk - (s_{ijk} \times e_i^{ut} \times e_j^{ti} \times e_k^{iu}))^2 + \lambda |e_i^{ut} + e_j^{ti} + e_k^{iu}| \tag{3}$$

e_i^{ut}, e_j^{ti} and e_k^{iu} are the feature implicit factors in the three decomposition matrices respectively. S_{ijk} is the core tensor. λ is the normalization coefficient.

As shown in the tensor decomposition part in Figure 1, the scoring data in the dataset is decomposed by tensor, and the corresponding feature factors are extracted. Then, the feature factors obtained by the decomposition operation are input into the fully connected layer for recombination to obtain the prediction score of the tensor part. The calculation formula is shown in formula (3):

$$y_{TD} = Relu(w_U^T S_u^{User}, w_I^T S_i^{Item}, w_L^T S_l^{Tag} | w_U, w_I, w_L) \tag{4}$$

In the formula(3), S_u^{User}, S_i^{Item} and S_l^{Tag} are correspond to the three characteristic factors of users, items and labels, respectively. w_U^T, w_I^I and w_L^T are the weights of users, objects and labels, respectively. *Relu* is used as the activation function for the fully connected layer undergoing reorganization.

3.2 Tensor decomposition partial prediction score

As shown in Figure 2, the deep learning part processes multi-source heterogeneous auxiliary information such as user information, movie tag information and movie title information. Among them, the user information and movie tags are structured text data using a recurrent neural network to extract features, and the movie name information is textual information, so a text convolution network is used to extract features. Then input the movie label feature extracted by the recurrent neural network and the movie name feature extracted by the text convolutional network into the secondary fully connected layer to obtain the movie feature, and finally match the user feature with the movie feature to give the prediction score of the deep learning part.

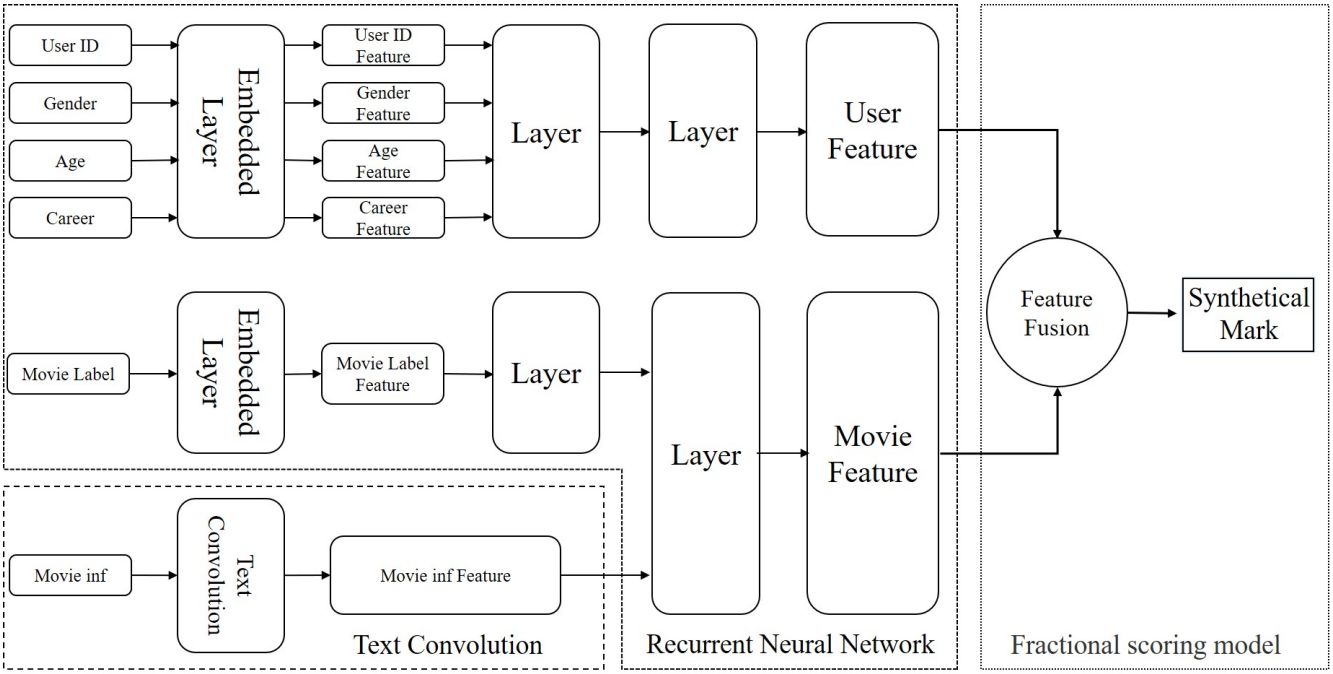

Figure 2. Part of the flow chart of deep learning

4 Deep Learning Part

The deep learning part is based on recurrent neural networks. First, these sparse and high-dimensional raw data are converted into low-dimensional, dense real-valued vectors through an embedding layer. Then the value in the low-dimensional dense embedding layer vector is passed to the hidden layer, and each hidden layer is updated according to formula (4):

$$a^{(l+1)} = f(W^{(l)} a^{(l)} + b^{(l)}) \tag{5}$$

In the above formula, f is the activation function, L is the number of hidden layers. $W^{(l)}$, $a^{(l)}$ and $b^{(l)}$ are the weight value, input value and bias of the corresponding level of the model, respectively.

4.1 Recurrent Neural Networks

The construction of the recurrent neural network is shown in Figure 2. The recurrent neural network is used to extract structured information such as user information and movie genres. User data and movie type data are passed into the embedding layer for preprocessing. The processing of movie types requires an additional preprocessing step, because a movie may correspond to multiple movie types, and the embedding matrix will index an n-dimensional matrix, so the matrix must be summed and turned into a one-dimensional vector.

User feature processing uses a single set of fully connected layers for feature extraction to obtain user feature h_i^u. The calculation method is as formula (5):

$$h_i^u = f(w_i a^{(l)} + b^l) \tag{6}$$

In the above formula, w_i is the weight value of the user network layer "l". $a^{(l)}$ is the input value. $b^{(l)}$ is the bias and f is the activation function.

The movie type and movie name processing share a set of fully connected layers. First, the movie type features are extracted through the fully connected layer, and then the movie name features extracted from the convolutional network are input into the secondary fully connected layer to obtain the complete movie features. Calculate The way is as formula (6):

$$a^{(t)} = f(w_k a^{t-1}) \tag{7}$$

In the above formula, w_k is the weight value of the "t-1" layer of the user network. a is the input value. b is the bias, and f is the activation function.

The movie genre feature and movie name feature P_k are input into the secondary fully connected layer to obtain movie features h_j^m. The calculation method is as formula (7):

$$h_j^m = f(w_j(a^{(t)} + P_k) + b^{(b)}) \tag{8}$$

In the above formula, w_j is the weight value of the user network layer "t" . a is the input value. b is the bias. f is the activation function. P_k is obtained by formula (9). After obtaining the features of the user and the movie, the two features are matched to obtain the predicted score, which will be described in detail in Section 4.3.

4.2 Text Convolutional Neural Network Model

The movie name information contains many keywords with movie attribute categories, such as "war", "comedy", "suspense", etc. These names play an important role in the judgment of movie attributes. However, these text information is unstructured data and cannot be processed directly. It needs to be converted into word vectors through Word2vec, and then embedded into the network structure, so that each movie name has a unique identifier vector, so as to complete the preprocessing of movie name information. Then, the preprocessed identifier vector is combined with the convolutional neural network to extract the feature information of the movie name.

The construction of the convolutional neural network model is shown in Figure 3. The first layer of the network is the word embedding layer, which consists of an embedding matrix composed of each processed word vector. The second layer is the convolution layer, which uses multiple feature extractors of different sizes (window sizes) to perform convolution operations on the embedding matrix. The window size refers to the number of words covered by the feature extractor each time. The extractor in this paper covers 3 to 5 words each time, and performs feature extraction on the information to be input in the word embedding layer. The third layer is the pooling layer, which extracts the maximum value of the feature values generated by each feature extractor in the upper layer to achieve representative feature extraction, and finally inputs the fully connected layer to generate the feature of the movie name.

In Natural Language Processing (NLP) tasks, the input words are expressed using the word embedding layer. we assuming that $X_i \in R^n$ is the corresponding sentence information in the n-dimensional word vector, its form is expressed as formula (8):

$$X_{1:n} = x_1 \oplus x_2 \oplus \ldots \oplus x_n \tag{9}$$

In the formula, $\oplus$ is the concatenation operator. Usually, $X_{i:i+n}$ is equivalent to concatenating the i to $i+n$ words, and then expressing the corresponding sentence in this form to complete the preliminary processing of the phonetic information. With this arrangement, one-dimensional text content can be transformed into a two-dimensional matrix data structure. Then set the

work.jpg work.bb

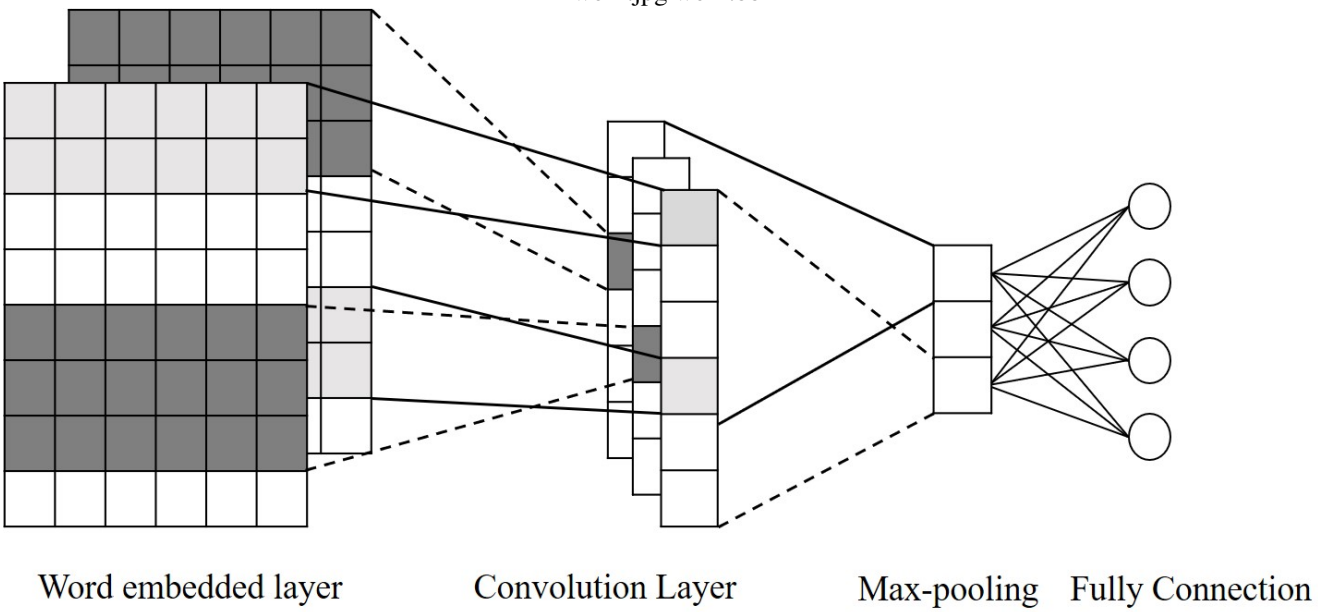

Figure 3. Schematic diagram of convolutional neural networks for sentence

number of feature extractors and determine the size of the feature extractor window as k. The processed text information is sequentially collected to generate new features. The expression function of the feature P_k in $X_{k:k+n}$ as follow:

$$P_k = f(WX_{k:k+n} + b) \tag{10}$$

In this paper, we use the hyperbolic tangent function as a nonlinear function of f. b is the bias.

Feature matching between user features and movie features

User and movie feature matching is shown in Figure (2). Through feature matching, the user features and movie features given by the front-layer network are integrated to give the prediction score of the deep learning part. In the experiment, the user part features and movie part features obtained in the previous section are input into the fully connected layer, and then the classification function Softmax is used to obtain the predicted score probability value. The formula for calculating the prediction score of the deep learning part is:

$$\begin{aligned} P(y|x) &= Softmax(w^T_{User} h^u_i + w^T_{Movie} h^m_j + b) \\ y_{Deep} &= argmax P(y|x) \end{aligned} \tag{11}$$

In the above formula, y_{Deep} is the rating level, corresponding to 1-5 rating levels respectively, $P(y|x)$ is the probability value of each predicted rating level, $w^T_{User} h^u_i$ and $w^t_{Movie} h^m_j$ are the product of user features and movie features and their corresponding model weight values, and b is the bias. Finally, the rating level corresponding to the highest $P(y|x)$ probability is taken as the predicted rating of the depth part, which denoted as y_{Deep}.

5 Tensor Decomposition and Deep Learning Hybrid

The hybrid model of tensor decomposition and deep learning is a complex mechanism. This paper uses three hybrid mechanisms to mix tensor decomposition and deep learning results respectively.

Weighted Mixing: The prediction scores of the two parts are added together to obtain a global prediction score. The formula is as follows:

$$y_{Score} = W(y_{TD} + y_{Deep}) + b \tag{12}$$

In the formula above, $y_{score} \in (0,5]$ is the global score of the system given by method 1, W is the weight, and b is the bias. This method is the most simple and convenient when analyzing multivariate models. Not only can the function be predicted and derived, but the results can also be tested for residuals to verify the accuracy of the model. However, the assumptions of this model are strict, and it is necessary to analyze all the independent variables that cause the change of the dependent variable. Otherwise, the diversity of influencing factors and the unmeasurability of some factors will limit the weighted model in some cases, resulting in poor performance.

Cascading: The scores of the two parts are added and passed to the secondary fully-connected layer, and each parameter is feature-matched through the corresponding activation function in the new first-level fully-connected layer to give the final prediction score. The prediction model is as follows:

$$\begin{aligned} y_{Score} &= f(s_1) \\ s_1 &= w_1 y_{TD} + w_2 y_{Deep} + b \end{aligned} \tag{13}$$

In the formula, $y_{score} \in (0,5]$ is the global score of the system, and f is the activation function. This paper uses two activation functions (*Relu* function and $Softplus$ function).

Relu activation function: The full name of Rectified linear unit (*Relu*) is one of the common activation functions in neural networks. The function formula is as follows:

$$f(s_1) = \begin{cases} s_1, & s_1 \geq 0 \\ 0, & Otherwise. \end{cases} \tag{14}$$

s_1 in the formula is calculated in the formula above, the activation function can simplify the calculation, improve the operation speed, and the derivative does not contain complex mathematical operations. When the input is positive, the derivative is not zero and gradient learning can be performed. Since the scoring input of this model is greater than or equal to zero, it will not enter the negative input area of the *Relu* function to avoid the weight update being hindered.

$Softplus$ activation function: $Softplus$ is an alternative to *Relu*, which can be regarded as the output range of *Relu* smooth function $(0,\infty)$, and the function formula is as follows:

$$f(s_1) = ln(1 + e^{s_1}) \tag{15}$$

As can be seen from the formula, the derivative of $Softplus$ is continuous and non-zero, which can prevent the silencing of neurons. However, the function is not centered at zero, and the derivative may be less than 1 result in the gradient disappears.

Feature combination: This method is different from the above two methods. In this method, multiple decomposition matrices and hidden layer features of deep learning are directly fused. The features are directly calculated as shown in Figure (4), and then the probability of occurrence corresponding to the five scoring levels is obtained through the Softmax hierarchical function. The model is as follows:

$$\begin{aligned} P(y|x) &= softmax(W^T s_2 + b) \\ s_2 &= e_i^u \times e_j^t \times e_k^i \times S_{ijk} + h_i^u \times h_j^m \\ y_{Score} &= argmax P(y|x) \end{aligned} \tag{16}$$

In the formula, W is the weight and b is the bias. $y_{score} \in (0,5]$, corresponding to 1-5 scoring levels respectively, and the scoring level with the largest occurrence probability is selected as the global score, denoted as y_{score}. Then take the logarithm of the value $P(y|x)$ to calculate the Softmax loss. Calculated as follows:

$$Loss = -logP(y|x) \tag{17}$$

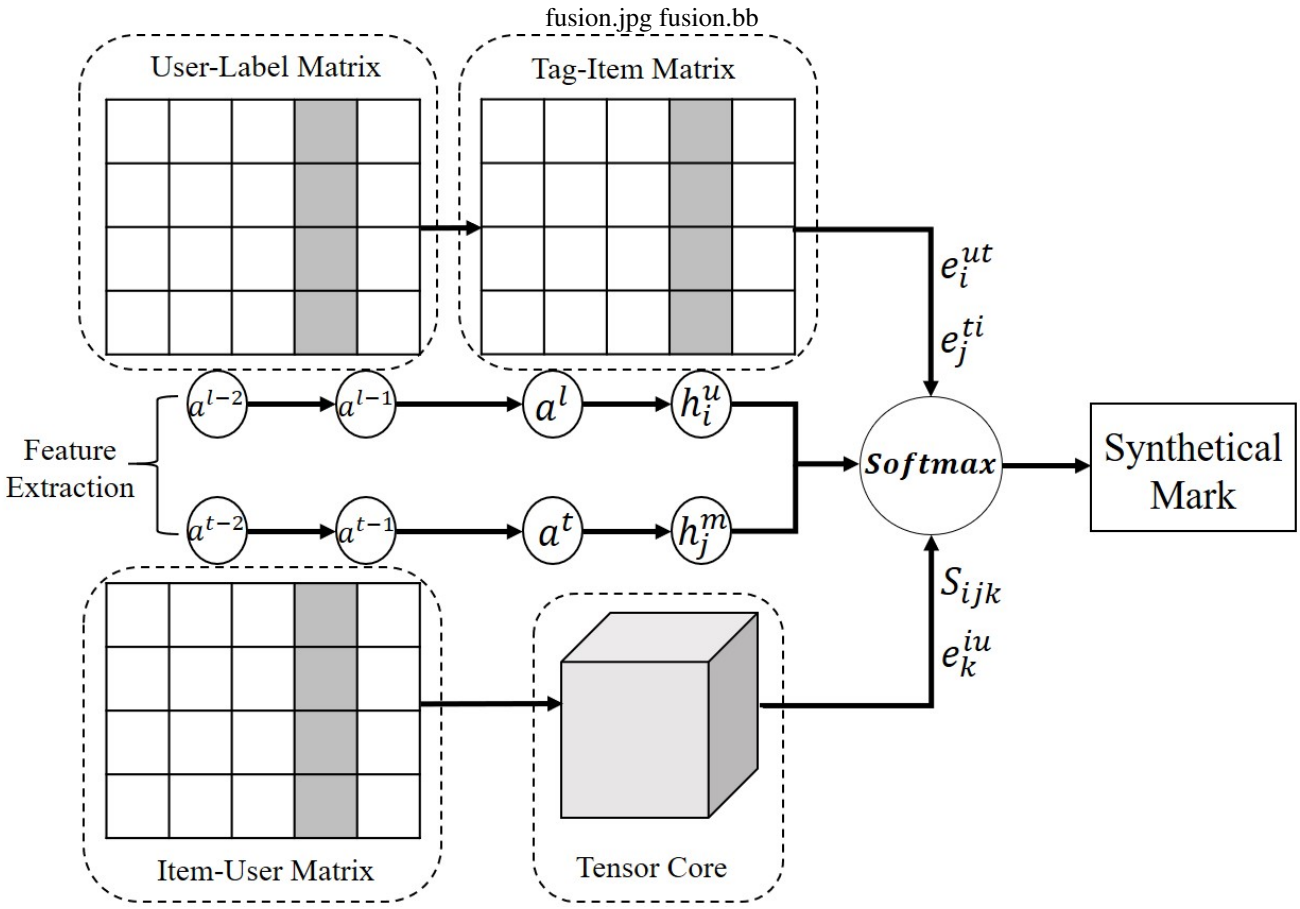

Figure 4. Schematic diagram of feature combined with blending mechanism

6 Experimental results

The experimental environment of this paper is Python3 in Anaconda, TensorFlow creates the entire algorithm data flow graph for the underlying network framework, and builds a deep learning network to extract features from the user, movie name and movie tag datasets. Then use the Tensorly toolkit to implement the *Tucker* decomposition of the dataset. Movielens and Netflix datasets were used in experiment.

The dataset is randomly divided into the test set and the training set according to the proportion, and then the evaluation and verification are completed according to the cross-validation principle. A total of 16,000 iterations of training are carried out in 5 Epochs. Each Epoch is 3109 and 777 batches respectively, and the result is output every 20 iterations. The calculation result of the loss is shown in Figure 5

6.1 Comparison of hybrid mechanisms between tensor decomposition and deep learning

In order to quantitatively compare the mixing mechanisms used by various recommender systems, this section will experimentally compare the three mixing mechanisms introduced above with three benchmark models. The Movielens-1M and Netfix-3M datasets are respectively used

As shown in Table (1) and Table (2), using the feature combination mechanism performs better on all four evaluation methods and using two datasets. For example, when using the Netfix3M dataset, using the feature combination mechanism improves the MRR performance by 3.47% and the MAP performance by 3.56% compared with the other three baseline algorithms. Compared with the first two cascaded and weighted hybrid methods, the MRR is improved by 1.03%, and the MAP performance is improved by 1.09%. The model can effectively integrate the tensor decomposition and the relevant features learned by the deep learning model, and use the classification function to generate the probability of the five grades that the user may give to a certain movie. score. The first two hybrid mechanisms directly integrate the two parts of the score, and will give a score that does not conform to the scoring data set, so the given score will inevitably have errors. Therefore, the feature combination mechanism is similar to rounding the score, so that the generated predicted value is more in line with

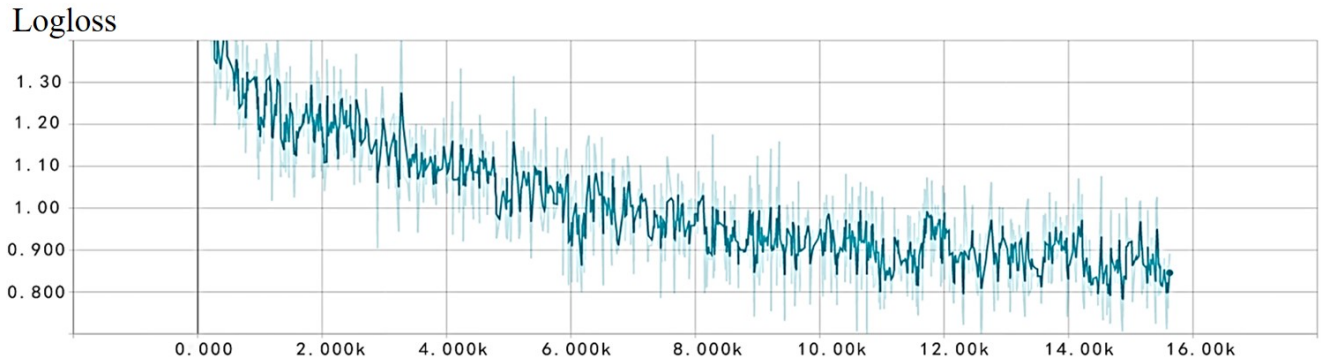

Figure 5. Visualization results of model training loss values

Table 1. Movie recommendation results (Movielens-1M)

Methods	Precision@5	NDCG@5	MRR	MAP
BPR	0.2664	0.2761	0.4324	0.3549
PRFM	0.2699	0.2894	0.4484	0.3885
LambdaFM	0.2953	0.3117	0.4611	0.4014
Weight Hybrid	0.2624	0.2735	0.4463	0.3902
Cascade+Relu	0.2753	0.2851	0.4457	0.4015
Cascade+Softplus	0.2903	0.2865	0.4573	0.4056
Feature Fusion	0.3086	0.2978	0.4645	0.4132

the actual real score value, thereby improving the performance of the entire recommendation result.

6.2 Performance Comparison of Tensor Deep Mixed Model and Single Model

In order to study the performance of the model in movie recommendation, this paper extracts and draws the loss value generated during the model training process. As shown in Figure (6), the tensor decomposition model performs the worst in terms of accuracy and convergence speed in sparse datasets. The deep learning model tends to be stable after a large amount of data training, and the convergence speed is the fastest. After synthesizing tensor and depth, the hybrid model reduces the error by 41.1% compared with deep learning, so the hybrid model has a certain improvement in accuracy and convergence speed compared with the other two independent models.

6.3 Comparison of Tensor Deep Hybrid Recommendation Model and Traditional Recommendation Algorithms

In the comparative experiment, the performance of the traditional recommendation algorithm in the extremely sparse dataset such as Movielens-1M is compared, and the evaluation index is MAE. It can be seen from Table 3 that the loss value of the tensor depth mixture model is the lowest, which is 2.8% smaller than that of the SVD model, and 34.0% smaller than that of the User CF. Therefore, when the tensor depth mixture model and the mainstream recommendation algorithm model are trained with the same dataset, the error of the mixture model is lower, and the recommended results are more accurate.

6.4 Comparison of Tensor Deep Hybrid Recommendation Model and Other Advanced Hybrid Recommendation Models

As shown in Table (4), the other four comparison models selected in this experiment all use the commercial data set based on CTR prediction, the number of training samples exceeds 50 billion, and through the online practical and post-debugging of a large number of users, so in The performance of AUC and Logloss is slightly better than that of the hybrid recommendation model in this paper (the Wide & Deep[17] model performs better on the acquisition line, which is 2.9% higher than the AUC of the local test). The performance of the model proposed in this paper is lower than that of DeepFM[18] model, the AUC area is reduced by 0.57%, and the Logloss error is increased by 3.01%. Both models use a factorization machine to reduce manual intervention, and are trained end-to-end to integrate the results of the two parts to improve prediction accuracy. Therefore, based on the existing model, later research can not only improve by increasing the scale of the dataset and the number of iterations, but also introduce a CTR prediction mechanism to enrich user behavior information and further improve the model.

Table 2. Movie recommendation results (Netfix-3M)

Methods	Precision@5	NDCG@5	MRR	MAP
BPR	0.2548	0.2576	0.3829	0.3484
PRFM	0.2645	0.2575	0.4022	0.3712
LambdaFM	0.2984	0.2993	0.4316	0.4043
Weight Hybrid	0.2832	0.2735	0.4257	0.3904
Cascade+Relu	0.2793	0.2804	0.4268	0.4022
Cascade+Softplus	0.2802	0.2857	0.4375	0.4056
Feature Fusion	0.2905	0.2879	0.4403	0.4103

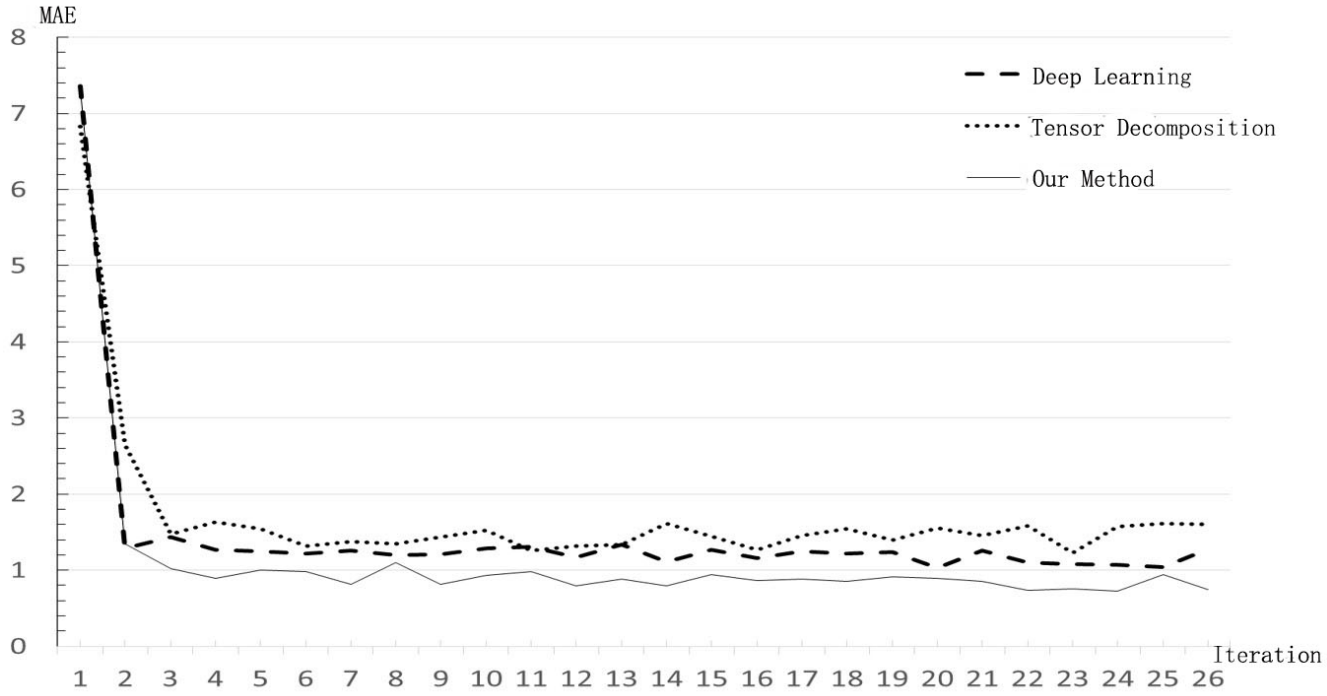

Figure 6. Comparison of loss value between single recommendation and hybrid recommendation

Table 3. Comparison of error between traditional recommendation algorithm and tensor & deep learning blending model

Methods	MAE
Item CF	0.2548
User CF	0.2645
Slope One	0.2984
KNN(k=5)	0.2832
SVD	0.2793
Our Method	0.2802

Table 4. Performance comparison between advanced hybrid recommendation model and tensor & deep learning blending model

Methods	AUC	Logloss
LR & DNN	0.8673	0.02634
FM & DNN	0.8661	0.02640
DeepFM	0.8715	0.02618
Wide& Deep	0.7280	–
Our Method	0.8665	0.02697

7 Conclusion

Data in social tagging systems is extremely sparse and missing. Although the tensor form can completely represent the relationship between the three in the labeling system, and give the corresponding recommendation results. However, the performance in extremely sparse datasets is poor and the accuracy is not high, so a deep learning network is introduced to extract direct features from auxiliary information through deep learning to make up for the shortcomings of a single recommendation algorithm.

Through the above several sets of experiments and the performance on the two datasets, it can be shown that the hybrid recommendation algorithm based on tensor decomposition and deep learning proposed in this paper has strong performance, and the integration of tensor decomposition model and deep learning network is also effective. Has strong robustness.

Author contributions statement

Xunpeng Xia and Rongfu Zhang designed the study and wrote the paper. Xunpeng Xia and Rongfu Zhang managed the literature searches and analyses. Xufeng Yao, Gang Huang, and Tiequn Tang revised the figures, and references and even conducted a series of additional experiments for the manuscript. All authors contributed to and approved the final manuscript.

Additional information

My datasets generated or analysed during the current study are available in the Netflix and Movielens repository. Netflix:https://www.netflixprize.com/ Movielens: http://files.grouplens.org/datasets/movielens/ The datasets used or analysed during the current study are available from the corresponding author on reasonable request.

References

1. Huang, L. W., Jiang, B. T., Shou-Ye, L. V., Liu, Y. B. & De-Yi, L. I. Survey on deep learning based recommender systems.
2. Ghahramani, Z. Probabilistic machine learning and artificial intelligence. *Nature* **521** (2015).
3. He, X. *et al.* Fast matrix factorization with nonuniform weights on missing data. *IEEE Transactions on Neural Networks Learn. Syst.* 1–14 (2019).
4. Luo, X. *et al.* Generating highly accurate predictions for missing qos data via aggregating nonnegative latent factor models. *IEEE Transactions on Neural Networks & Learn. Syst.* **27**, 524–537 (2015).
5. Luo, X., Sun, J., Wang, Z., Li, S. & Shang, M. Symmetric and nonnegative latent factor models for undirected, high-dimensional, and sparse networks in industrial applications. *IEEE Transactions on Ind. Informatics* **13**, 3098–3107 (2017).
6. Luo, X., Wu, H., Yuan, H. & Zhou, M. C. Temporal pattern-aware qos prediction via biased non-negative latent factorization of tensors. *IEEE Transactions on Cybern.* **PP**, 1–12 (2019).
7. Yang, L., Cao, X., Jin, D., Wang, X. & Meng, D. A unified semi-supervised community detection framework using latent space graph regularization. *IEEE Trans Cybern* **45**, 2585–2598 (2015).
8. Liao, Z. F., Li, L., Liu, L. M. & Li, Y. Z. A tripartite decomposition of tensor for social tagging. *Chin. J. Comput.* **35**, 2625 (2012).
9. Liu, H., Fang, S., Zhang, Z., Li, D. & Wang, J. Mfdnet: Collaborative poses perception and matrix fisher distribution for head pose estimation. *IEEE Transactions on Multimed.* **PP**, 1–1 (2021).
10. Symeonidis, P., Nanopoulos, A. & Manolopoulos, Y. Tag recommendations based on tensor dimensionality reduction. In *Proceedings of the 2008 ACM Conference on Recommender Systems, RecSys 2008, Lausanne, Switzerland, October 23-25, 2008* (2008).
11. Rendle, S. & Schmidt-Thieme, L. Pairwise interaction tensor factorization for personalized tag recommendation. In *Proceedings of the Third International Conference on Web Search and Web Data Mining, WSDM 2010, New York, NY, USA, February 4-6, 2010* (2016).
12. Song, Y., Li, M., Luo, X., Yang, G. & Wang, C. Improved symmetric and nonnegative matrix factorization models for undirected, sparse and large-scaled networks: A triple factorization-based approach. *IEEE Transactions on Ind. Informatics* **PP**, 1–1 (2019).
13. Luo, X., Zhou, M. C., Li, S. & Shang, M. S. An inherently nonnegative latent factor model for high-dimensional and sparse matrices from industrial applications. *IEEE* (2018).

14. Li, M., Song, Y., Wang, C. & Chu, Y. An improved non-negative latent factor model via momentum-based additive gradient descent method.

15. Alfarhood, M. & Cheng, J. Deep learning-based recommender systems.

16. Molaei, S., Havvaei, A., Zare, H. & Jalili, M. Collaborative deep forest learning for recommender systems. *IEEE Access* **PP**, 1–1 (2021).

17. Guo, H., Tang, R., Ye, Y., Li, Z. & Dong, Z. Deepfm: An end-to-end wide & deep learning framework for ctr prediction.

18. Guo, H., Tang, R., Ye, Y., Li, Z. & He, X. Deepfm: A factorization-machine based neural network for ctr prediction. In *Twenty-Sixth International Joint Conference on Artificial Intelligence* (2017).