Electronic Research Archive

http://www.aimspress.com/journal/ERA

ERA, 31(6): 3169–3186.
DOI: 10.3934/era.2023160
Received: 27 November 2022
Revised: 14 March 2023
Accepted: 14 March 2023
Published: 23 March 2023

Research article

Plot-aware transformer for recommender systems

Suhua Wang[1], Zhen Huang[2], Bingjie Zhang[3], Xiantao Heng[3], Yeyi Jiang[3], Xiaoxin Sun[3,*]

[1] Computer Department, Changchun Humanities and Sciences College, Changchun 130117, China
[2] Department of Computer Science, University of Science and Technology of China, Hefei 230026, China
[3] School of Information Science and Technology, Northeast Normal University, Changchun 130117, China

* **Correspondence:** Email: sunxx772@nenu.edu.cn; Tel: +8643184536338;
Fax: +8643184536338.

Abstract: Plot text is very valuable supporting information in movie recommendations. It has several characteristics: 1) It is rich in content. Each movie often has a document of more than 200 words to describe it, which can give the movie a rich semantic meaning. 2) Objectivity. Plot texts are different from review information. A movie may have thousands of reviews with mixed and conflicting opinions. However, a film has only one plot text, which is fair in tone and does not take a position. Despite its appealing properties and potential for accurate movie portrayal, the lack of a building block for effectively mining plot semantics has led to the marginalization of plot text in the design of movie recommendation algorithms. Therefore, in this paper, we explore the application of the Transformer, currently the best natural language processing module, to learning movie plot texts to help achieve more accurate rating prediction. We propose the "Plot-Aware Transformer" model (PAT) to model the process of "user-movie" rating interaction. We test the PAT model on several movie datasets and demonstrated that the model is competitive. In all tasks, PAT achieves state-of-the-art performance compared to baseline experiments.

Keywords: Plot-Aware Transformer; embedding representation; recommendation algorithms; semantics mining

1. Introduction

The explosive growth of information resources has inevitably caused the problem of information overload. As an information filtering tool for massive data, a recommender system can effectively solve the information overload problem and provide content that meets users' needs in a personalized manner. In addition, as a link between users and information, recommendation systems not only help users discover the information they need but also allow information to be presented to users who are interested in it, thus achieving a mutual benefit for both information producers and information consumers. Currently, recommendation systems have become a hot issue of concern and research in industry and academia, with a wide range of application fields, including e-commerce, social networks, video and music recommendation, etc.

Traditional recommendation methods are mainly classified into content-based recommendation methods, collaborative filtering recommendation methods, and hybrid recommendation methods. The current popular trend in recommendation algorithm design is the hybrid recommendation method that incorporates heterogeneous auxiliary information from multiple sources (including knowledge graphs, social networks, user behavior, images, text, etc.). Although such a design can alleviate the cold-start problem to a certain extent, the auxiliary information often has complex features such as multimodal, unstructured, large-scale, sparse data and uneven distribution, and it still faces serious challenges in processing.

In this paper, we use Transformer to extract features for the plot text information in the movie dataset to obtain more accurate item (movie) representations to help achieve personalized recommendations for specific users.

1.1. Why use Transformer?

Using word frequency statistics, previous researchers have proposed classical word embedding models, such as Word2Vec. However, such trained word models are often rigid "hard" representations. These word embeddings are already fixed vectors and will not be reasonably corrected for the specific contextual environment. For example, consider the following two sentences:

1) "The cake won't fit in the box because it's too big."

2) "The cake won't fit in the box because it's too small."

When humans see two sentences like this, they will know that "it" in the first sentence refers to "the cake" while "it" refers to the "box" in the second sentence. Therefore, the "it" in each sentence is completely different. However, in the traditional word embedding models, only a unique representation of "it" has been trained, so this unique representation cannot accurately represent its true meaning in a specific context.

To better understand and express the meaning of each word in a specific context, Transformer was proposed in the study [1]. Unlike traditional word embedding models, Transformer computes the relationship between each word and all other words within the text sequence using a "self-attentive" mechanism in a specific contextual environment. Different words form an aggregated representation based on the degree of match between them. This representation is characterized by the fact that a particular task that has its textual representation needs to "train itself and use itself" directly on its context (text), instead of simply borrowing a generic, fixed representation trained by a traditional model in a global corpus. In this way, when learning the above two different sentences, Transformer can represent the “it” in each sentence more accurately and capture its true meaning in the specific context.

Given Transformer’s excellent performance in natural language processing, in this paper, we explore the use of Transformer to learn and extract features from movie plot texts to help build a more accurate movie portrait.

First of all, the plot text is chosen as auxiliary information because there is only one plot text for a movie, with no need to consider trade-offs, and this text is large and rich in information. More importantly, the plot text tends to be objective, accurate and comprehensive in describing the movie without uncertainties such as personal emotions and values.

Second, we take the plot text of each movie and generate a review text embedding matrix using the already trained Glove vector library. This is only an initial representation.

This embedding matrix is then fed into the Transformer network, which is trained and adjusted in the current context using a “self-attentive” mechanism to generate a more optimized representation that matches the contextual semantics. As the final representation of the plot, this representation can deeply and accurately represent the semantic information of the corresponding movie.

Finally, user representation and movie representation are subjected to some kind of interactive computation, such as dot product or concatenation, and finally fed into a multilayer perceptron network to predict the user-movie rating after a collaborative filtering computation of the deep network.

The main contributions of this work are as follows.

1. We use Transformer networks and deep feedforward networks to model user and item features and design a general framework called PAT to implement collaborative filtering based on deep non-linear interactions between users and movie plots.

2. We show that the present framework incorporates the semantic information of the plot text well.

3. Extensive experiments are done on three real datasets to demonstrate the effectiveness of PAT.

The paper is organized as follows: Related work is discussed in Section 2; the PAT framework is given in Section 3; experimental results are presented in Section 4; Section 5 provides a summary and expectations for future work.

2. Related work

2.1. Problem definition

Recommender systems have a variety of tasks and scenarios, mainly including sequence recommendation, conversation recommendation [2], session recommendation [3], bundle recommendation [4], click-through rate prediction [5], de-bias [6], fair talk recommendation [7], explainability [8], Counterfactual learning [9], etc. By recommendation technique, it mainly includes contrast learning [4], reinforcement learning [3,10], adversarial learning [7], cross-domain recommendation [11], contextual recommendation [12], debiasing techniques [13], confidence calibration [14], Federation learning [15], knowledge extraction [16], etc.

Recommendations based on textual content as auxiliary information aim at extracting features from user reviews or product description documents to learn the representation of users and products. Recent related works in this area are as follows. Chen et al. [17] proposed the Deformable Convolutional Network (DCN). This network adds an offset layer to the traditional convolutional layer to extend the capability of the convolutional transform model. On this basis, they further proposed a Deformable Convolutional Network Matrix Factorization (DCNMF) recommended model. Guo et al. [18]

proposed a Joint Convolutional Matrix Factorization (JCMF) model, which unifies the framework considering the reviews of goods, the relationships of goods, the social influence of users, and the reviews of users. Gan et al. [19] proposed a Convolutional and Dense layer Matrix Factorization (CDMF) model. This approach uses a convolutional neural network to extract hidden features from item descriptions as document representations and then fuses them with label information through a fully connected layer to generate a comprehensive feature vector. It combines multiple sources of information from item descriptions and labels information for context-aware recommendations. Xu et al. [20] proposed a Social ConvMF model. This model integrates trust relationships and convolutional neural networks into probabilistic matrix factorization. It captures both trust-aware information and contextual information in documents for the social recommendation. Chen et al. [21] proposed a Trailer Inception probability matrix factorization model called Ti-PMF. This model extracts visual information features from movie trailers and combines recurrent convolutional neural network and probability matrix factorization to predict user-movie ratings. Wang et al. [22] proposed a visual recurrent convolutional matrix factorization recommendation scheme based on probabilistic matrix factorization. The scheme extracts movie features from descriptive text and text extracted from movie posters as well as multi-level visual features for a movie recommendation, respectively. Nguyen et al. [23] proposed an attentional probability matrix factorization model that uses neural attention networks to learn the importance of feature interactions and uses Doc2Vec techniques to mine contextual information. Liu et al. [24] proposed a supervised convolutional matrix factorization (Super-ConvMF) recommendation model that combines rating information, item content information and label information into a unified recommendation framework. Zheng et al. [25] proposed an adversarial training framework to learn hybrid recommendation models, in which a generator model is constructed to learn the distribution of paired ranking pairs, while a discriminator is trained to distinguish between generated (fake) and real item pairs. Liu et al. [26] proposed a BiconvMF algorithm based on an improved ConvMF. The algorithm uses two parallel convolutional neural networks to extract depth features from the user review set and the item review set, respectively, and fuses these features into the factorization of the rating matrix to predict user-movie ratings. Cai et al. [27] proposed a constrained neural network probability matrix factorization model (CPMF-NN). This model uses convolutional neural networks to extract item latent features from corresponding documents. In fusing the potential feature vectors, a multilayer perceptron is used to capture the nonlinear structural features of the user-item interaction. Wang et al. [28] extended CNN by introducing a new module called Deep Latent Dirichlet Allocation (DLDA) to capture the counting information of contextual features. The model combines DLDA with CNN to obtain word-driven and context-aware comment representations and finally combines matrix factorization for the recommendation. Wu et al. [29] proposed the Content Embedding Regularization Matrix Factorization (CERMF) model. This model uses convolutional neural networks to generate independent embedding representations for users and items simultaneously. Dual embeddings are then used to regularize the generation of latent features of users and items. Xia et al. [30] proposed a joint deep network-based multi-source feature learning (JDNMFL) framework for QoS prediction (QoS) forecasting. This model is divided into two parts: multi-source feature extraction and feature interaction learning. Zhao et al. [31] proposed a DE-ConvMF model. It has a dual embedding layer in ConvMF that focuses more on item-side information. This dual embedding consists of two parts: a generic embedding layer and a domain embedding layer, which are combined as an embedding layer.

Latest research hotspots in the field of recommendation algorithms include conversation-based recommendations and causal recommendations. Traditional recommendation algorithms lack interaction with users, and it is difficult to grasp users' real-time interests in a timely and effective manner. Conversational Recommender System (CRS) is a new research hotspot in the field of recommendation systems because it can understand users' interests through in-depth interaction with them. The core of the Conversational Recommender System is the online interaction between users and the recommender system, which is to obtain the user's feedback through the dialogue interaction process between the user and the recommender system and integrate the user's feedback into the recommendation model to better understand the user's current interests and improve the accuracy of the recommendation.

Causal learning is the study of how to discover and use causal relationships between variables to make predictions, rather than merely rely on correlations between variables. Causal relationships reveal the nature of events, and changing the "cause" behind an event will often affect the "effect" of the event. In contrast, correlations are often not the essential laws of events, and changing one event may not affect the other. For example, through association mining, we can find a strong correlation between yellow fingers and lung cancer, but there is no clear causal relationship between the two, i.e., painting the fingers of ordinary people yellow does not increase their probability of getting lung cancer. The true "cause" of the yellow finger and lung cancer is smoking, i.e., there is a causal relationship between smoking and lung cancer. Therefore, if non-smokers are allowed to smoke, they are significantly more likely to get lung cancer. The lack of causality analysis in a recommender system may lead to a decrease in the effectiveness of the recommendation or bias of the model.

3. The proposed method

We model recommendation task as a rating prediction problem. This problem is defined as follows: Given a user u and a movie i with the corresponding plot text $T(i) = \{w_1, w_2, \dots w_{|P|}\}$, our task essentially entails learning a function $\hat{r}_{ui} = \Phi(u, T(i))$ to predict the rating of user u on item i.

To better describe our model in later sections, here, we first list the symbols used in the model and their meanings in Table 1.

Table 1. Summary of terminology used.

Notations	**Description**
u	user representation
i	item (movie) representation
$T(i) = \{w_1, w_2, \dots w_{|P|}\}$	The text sequence of the plot of the movie i . w_n is the nth word.
$|P|$	Plot length (word count) after Padding, experimentally$|P|$ = 300
d	Length of each word vector in the plot
$E \in R^{|P| \times d}$	The embedding matrix composed of all word vectors in the plot sequence, as the initial representation of the corresponding movie
$\hat{r}_{ui}$	The model output, which is the predicted rating of u on i

3.1. *Model architecture*

The general architecture of PAT is shown in Figure 1. In Figure 1, we can see that it follows the popular two-tower embedding and MLP interaction paradigm for recommendation systems, i.e., the relevant features are embedded in a low-dimensional vector before calculating the user's predicted rating for the item and then fed into the MLP. The key of PAT is the addition of the Transformer layer to better learn the item representation by capturing the rich and complex plot sequence signals of the underlying layer. In the next section, we present the key components of PAT in a bottom-up fashion: the embedding layer, the Transformer layer and the MLP interaction layer.

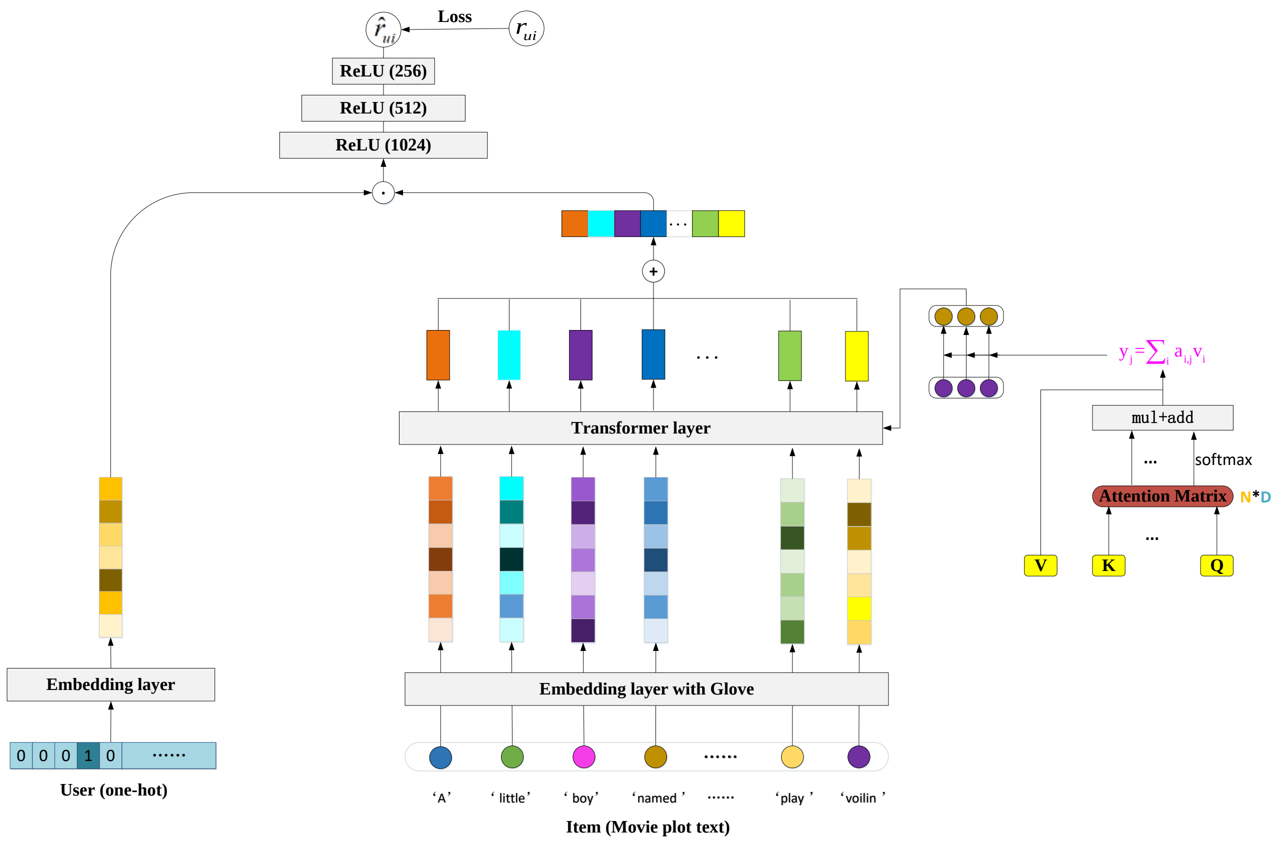

Figure 1. The general architecture of the proposed PAT.

PAT takes a sequence of user and item (movie) episodes text as input. It first embeds these input features as low-dimensional vectors. To better capture the interesting relationship between users and items, a Transformer layer is used to learn a deeper representation of each word in the episode sequence. User embeddings are then connected to the output of the Transformer layer to learn the interaction of hidden features using a three-layer MLP, and a sigmoid function is used to generate the final output. Note that the "location features" are merged into the "sequence item features."

3.2. *Embedding layer*

The first component is the embedding layer, which embeds all the input features into low-dimensional vector representations. In our scenario, there are two kinds of features: user features and movie plot features. We explain the embedding process of both in turn below.

User embedding. User features are represented as one-hot vectors, where the "1" corresponds to the user's ID. We use o_u to represent the one-hot vector of the user as the input layer on the user side. In the embedding layer, it is directly mapped to a low-dimensional vector by a fully connected network. This process is essentially multiplying a sparse high-dimensional vector o_u by a weight matrix W and converting it to a low-dimensional dense embedding vector, i.e.,

$$u = W^T \cdot o_u \tag{1}$$

u is the potential vector used to describe the user.

Item embedding. The embedding of movies is a bit more complicated. The original plot texts of different movies have different lengths, and for the convenience of model learning, we remove the deactivated words and then unify the plot texts of all movies into 300 words by padding. The movies are also embedded differently from the users. We obtain the Glove vector model by querying the trained Glove vector library. The Glove word vector model is derived from [32]. In contrast to another word vector model, Word2Vector [33], Glove successfully exploits the global information of the corpus and is a word characterization tool based on global word frequency statistics. The publicly available Glove model provides 50-, 100-, 200- and 300-dimensional vectors of about 400,000 words. If we query 300 words in a movie text sequence from the 50-dimensional Glove, the output of the embedding layer is an embedding matrix $E_s \in R^{300\times 50}$.

In addition to the above word features of the plot text, we also added the positional features of each word. In this way, we use two types of features to represent a movie, "sequence word features" and "positional features", where "sequence word features" is matrix E_s. Then, for each word in the plot text, we concatenate the sequence word features and positional features to create an embedding matrix $E \in R^{|P|\times d}$, where d is the dimension of the embedding and $|P|$ is the length of the plot text, i.e., the number of words in the plot sequence. We use $e_k \in R^d$ to denote the embedding of the kth item in the given movie plot sequence.

Position embedding. In [1], the authors propose a positional embedding method to capture the order information in sentences. Similarly, the order is also present in plot text sequences. Therefore, we add "position" as an input feature for each item of the underlying layer and project it as a low-dimensional vector.

3.3. *Transformer layer*

The $E \in R^{|P|\times d}$ obtained in subsection 3.2 initially represents the plot features of the movie (there are $|P|$ words in the plot text and d features for each word).

To apply the self-attentive mechanism to better learn the plot text context, we transform the initial plot matrix E into three auxiliary matrices through a cubic linear mapping, as shown in Eqs 2–4.

$$Q = EW^Q \tag{2}$$

$$K = EW^K \tag{3}$$

$$V = EW^V \tag{4}$$

where the mapping matrices are W^Q, W^K and $W^V \in R^{d\times d}$. $E$is the embedding matrix of all words in the plot sequence. These three auxiliary matrices represent the three elements in the attention calculation: Q for queries, K for keys and V for values. Then, we feed Q, K and V into the self-attention layer, as shown in Eq 5.

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d}})V \tag{5}$$

Following [1], we then extend the single self-attention to multi-headed attention.

$$head_i = \text{Attention}(EW_i{}^Q, EW_i{}^K, EW_i{}^V) \tag{6}$$

$$i\text{=MH}(E) = [head_1, head_2, \ldots, head_h]W^H \tag{7}$$

h represents the number of heads, and all the head matrices are stitched together and then linearly mapped through W^H to form the final representation of the movie at i.

3.4. *User-Item interaction layer and output*

To model the users' predicted rating process for movies, we concatenate the above generated user representation u and item representation i $(u \oplus i)$, where $\oplus$ represents the concatenation operation of the two vectors, and feed them together into a multilayer perceptron network (MLP). In order to better fit the complex user-item interaction process, we set deeper nonlinear neural layers in the MLP. As shown in Eq 8,

$$\begin{aligned} h_0 &= u \oplus i \\ h_1 &= Relu(W_1{}^T h_0 + b_1) \\ h_{L-1} &= Relu(W_{L-1}{}^T h_{L-2} + b_{L-1}) \\ \hat{r}_{ui} &= Relu(W_L{}^T h_{L-1} + b_L) \end{aligned} \tag{8}$$

where W_k, b_k and a_k denote the weight matrix, bias vector and activation function of the$k$ layer of the MLP, respectively. Relu is the activation function. $\hat{r}_{ui}$ is the rating of the movie i by the user u as predicted by our method.

3.5. *Loss function*

Predicting users' ratings of items is a regression problem, and under the prior condition that the prediction error satisfies a Gaussian distribution, we use the following mean squared error (MSE) function.

$$LOSS = \frac{1}{|R|} \sum_{(u,i,r_{ui}) \in R} (r_{ui} - \hat{r}_{ui})^2 + \lambda ||W||^2 \tag{9}$$

R denotes the set of all rating entries observed in the dataset, the label r_{ui} is the real rating of the movie i by the user u, and $\hat{r}_{ui}$ is the rating predicted by our PAT model. Equation 9 is the objective function to be minimized, and in our experiments we use Adam's algorithm for training optimization.

4. Experiments

In this section, we evaluate the experimental performance of PAT on two real-world datasets. Our experimental results demonstrate that PAT greatly improves recommendation accuracy compared to other text-based recommendation algorithms.

4.1. *Experiment settings*

4.1.1. Datasets and evaluation metrics

We adopted two datasets that are widely used in the field of recommendation algorithms: MovieLens 1M (ML-1M) and MovieLens 10M (ML-10M), which include explicit ratings of movies by users (from 1 to 5). We used a web crawler to collect the plot files corresponding to all movies in MovieLens from the IMDB website. Similar to [34], we did the following preprocessing on all movie plot files: 1) Removing deactivated words, 2) Calculating the TF-IDF value for each word, 3) Selecting the top 8000 words of TF-IDF to form a vocabulary list, 4) Removing words that are not in the glossary, 5) Padding each movie text into 300 words. After pre-processing, the statistical results and features of these datasets are summarized in Table 2:

Table 2. Statistics of the datasets (# represents "The number of").

Dataset	#users	#items	#plots	#ratings	density
ML-1M	6,040	3,706	3,706	1,000,209	4.65%
ML-10M	69,878	10,073	10,073	9,945,875	1.41%

We used the root mean square error (RMSE), commonly used in rank prediction, to evaluate the results of our experiments. The RMSE is used to measure the deviation between the observed value and the true value. For a user u and item i in the test set T, let r_{ui} be the actual rating of the item i by the user u and $\hat{r}_{ui}$ be the rating given by the recommendation algorithm. Then the RMSE is defined as

$$RMSE = \frac{\sqrt{\sum u,i \in T (r_{ui} - \hat{r}_{ui})^2}}{|T|} \tag{10}$$

The smaller the value of RMSE is, the better the experimental result, indicating that the predicted rating is closer to the real rating.

In order to ensure the stability of the evaluation model and truly compare the performances of all methods fairly, we adopted cross-validation to avoid the limitations and particularities of fixed partition data sets. In the experiment, we carried out 5-fold cross-validation on the data set and calculated the average value of all folds to obtain cross-validation error.

4.1.2 Baseline experiments

We chose the following baseline approach for comparison with PAT.

NCF [35]. NCF is a classic deep learning recommendation algorithm that trains embedding representations of users and items through neural networks and further predicts users' ratings on items through MLP. Note, the original paper predicts whether the user likes or dislikes the item; we rewrote the code and changed the task to predict the user's rating on the item.

U-AutoRec / I-AutoRec [36]. AutoRec is a self-encoder based collaborative filtering model that learns user and item representations by reconstructing known user-item scores. These representations are incidentally used to calculate additional unknown user-item scores.

ConvMF [34]. ConvMF learns the textual content of the movie using convolutional networks and feeds the extracted features, along with the user representation, into a probabilistic matrix factorization model to predict the rating.

U-CFN/ U-CFN++/ V-CFN/ V-CFN [37]. CFN is an upgrade and expansion of AutoRec, adding edge information and denoising techniques to the self-encoder, which has solved the sparse and cold start problems of pure scoring math.

DPGMF [38]. DPGMF feeds the movie file into a convolutional neural network to extract abstract features, which are the representations of the item. Then, they and the embedding representation of the user's one-hot vector are fed into the multi-layer perceptron network to model the prediction score of the user on the item.

Table 3 summarizes the information used by all the baseline methods. Among them, NCF and AutoRec series (including U-AutoRec and I-AutoRec) only use ratings information, and do not use other auxiliary information. Among CFN series (including: U-CFN, U-CFN++, I-CFN, I-CFN++), U-CFN and I-CFN only use ratings, U-CFN++ and I-CFN++ use both ratings and side information, side information includes the user's age, gender, genre and movie category (action, thriller, etc.). ConvMF+ and DPGMF use ratings and plot text information in exactly the same way we do. Therefore, ConvMF+ and DPGMF are major baselines for comparison with our method.

Table 3. Information used by baseline methods.

Method	**MovieLens 1M**			**MovieLens 10M**		
	Ratings	Plot	Side	Ratings	Plot	Side
NCF	✓			✓		
U-AutoRec	✓			✓		
I-AutoRec	✓			✓		
ConvMF+	✓	✓		✓	✓	
U-CFN	✓			✓		
U-CFN++	✓		✓	✓		✓
I-CFN	✓			✓		
I-CFN++	✓		✓	✓		✓
DPGMF	✓	✓		✓	✓	
PAT (Ours)	✓	✓		✓	✓	

4.1.3 Parameter setting

Our model is implemented with Python 3.9.12 and Keras 2.7.0 on GeForce GTX1080 GPU. We used mini-batch Adam as the optimization algorithm. The configuration of various hyperparameters of the model is listed in Table 4.

Table 4. PAT hyperparameters configuration.

Configuration of PAT			
Glove embedding size	[50,100,200,300]	**Batch size**	256
head number	4	**Batch-Normalization**	default
sequence length	300	**epochs**	1
transformer block	1	**queue capacity**	1024
MLP Shape	1024*512*256	**learning rate**	0.01

4.2. Performance comparison

The experimental results are shown in Table 5. From this we can observe the advantages of PAT compared to baseline.

We will now compare the performance of PAT with other recommended methods. For embedding-based methods (e.g. ConvMF+, DPGMF etc.), the size of the embedding determines the capability, so we set the size of embedding to 200 in all methods to make a fair comparison. In the following Section 4.3, we will change the size of the embedding for each method in order to compare. Table 5 shows the comparison of rating prediction performance between PAT and other competitors on different datasets. We obtain the following key observations:

1) Overall, our PAT model achieves the best performance. Specifically, on both datasets, our model achieved the lowest RMSE values. On MovieLens1M, compared to the best method in the baseline, our experiment showed a 6.75% decrease in RMSE. On MovieLens 10M, this is a decrease of 5.37%. From this, we believe that Transformer can effectively extract useful information from movie documents to achieve more accurate rating prediction than other models.
2) Compared to NCF and AutoRec, our PAT showed the most significant improvement in RMSE: Compared with NCF, PAT decreased by 19.76 and 20.88% at ML-1M and ML-10M, respectively. Compared with U-AutoRec, PAT decreased by 9.37 and 8.98% at ML-1M and ML-10M, respectively. Also these two types of baseline methods are the least effective among all comparison models. We believe that the main reason for this is that NCF and AutoRec don't use any other information than scoring data. The sparsity of rating data causes the semantic representation of users and items trained by the model to be rough and inaccurate, affecting the effectiveness of predicted ratings.
3) Compared to ConvMF+ and CFN series, PAT also achieves an average 10% relative decline in RMSE on both datasets (Please refer to Table 5 for the specific percentages). We believe that the depth and refinement of the model are the primary reason for PAT's advantage in this comparison. Although there are deep learning models, ConvMF+ and the CFN series are simple in structure and only have three to four layers in depth, which are not enough to mine the semantics of auxiliary information well.
4) Finally, when compared to the DPGMF method, which also uses the plot text to design the recommendation algorithm, PAT achieves an error decrease of about 6% on both datasets. We believe that the method of learning the plot text is the main reason for this gap. DPGMF uses

a traditional convolutional network to learn the textual content of items, while PAT uses a more complex Transformer network. The computerized mechanism of CNN can only learn local features and cannot learn long-term dependencies. However, Transformer uses the attention mechanism to capture global information. In addition, CNN are naturally suited to image processing and are not well suited to information from serial data such as sound, text, time, and so on. While Transformer was designed for natural language processing, in Transformer's multi-headed attention structure, the model can learn relevant information in different representation subspaces for different tasks.

Table 5. Overall comparison of RMSE.

Method	**MovieLens 1M**		**MovieLens 10M**	
	RMSE↓	PAT impr.	RMSE ↓	PAT impr.
NCF	0.9381	19.76%	0.8997	20.88%
U-AutoRec	0.8740	13.88%	0.8670	17.90%
I-AutoRec	0.8305	9.37%	0.7820	8.98%
ConvMF+	0.8549	11.95%	0.7930	10.24%
U-CFN	0.8574	12.21%	0.7954	10.51%
U-CFN++	0.8572	12.19%	0.7880	9.67%
I-CFN	0.8321	9.54%	0.7767	8.36%
I-CFN++	0.8316	9.49%	0.7754	8.20%
DPGMF	<u>0.8072</u>	6.75%	<u>0.7522</u>	5.37%
PAT (Ours)	**0.7527**	–	**0.7118**	–

The best results are shown in bold, and the second best results are underlined.

In order to test whether there is a significant difference in the RMSE between our PAT model and DPGMF, the most competitive baseline model, we conducted a t-test in the experiment. Algorithm 1 shows the detailed procedures for performing a paired sample t-test when the population variance is unknown and absolute errors (sample means and variances) are used:

```
Algorithm 1: T-test for determining the level of significance of the difference between PAT and baseline on RMSE indicator.
Input: The null hypothesis $H_0$, the alternative hypothesis $H_1$
Output: Accept or reject the null hypothesis $H_0$
    1: Define the null and alternative hypotheses. The null hypothesis $H_0$ states that there is no difference between the means of the two paired samples, while the alternative hypothesis $H_1$ states that there is a difference between the means.
    2: Calculate the differences $(\bar{x}_1 - \bar{x}_2)$ between each pair of observations, and calculate the mean ($\mu_1$ and $\mu_2$) and standard deviation ($s_1$ and $s_2$) of the differences.
    3: Calculate the t-statistic using the formula:
```

$$t = \frac{(\bar{x}_1 - \bar{x}_2) - (\mu_1 - \mu_2)}{s_p\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$$

where

$$s_p = \sqrt{\frac{s_1^2(n_1 - 1) + s_2^2(n_2 - 1)}{n_1 + n_2 - 2}}$$

where n_1 and n_2 are the numbers of pairs, and the hypothesized difference is typically set to zero for a two-tailed test.

4: Determine the degrees of freedom (df) for the t-distribution using the formula:

$$df = n_1 + n_2 - 2$$

5: Determine the critical value of t for the desired level of significance ($\partial=0.05$) and degrees of freedom ($df = 26$) using a t-distribution table or calculator.

6: Compare the calculated t-value with the critical t-value. If the calculated t-value is greater than the critical t-value, reject the null hypothesis and conclude that there is a significant difference between the means of the two paired samples. If the calculated t-value is less than the critical t-value, fail to reject the null hypothesis and conclude that there is no significant difference between the means.

7: Report the results of the test, including the calculated t-value, degrees of freedom, critical t-value, and conclusion.

In the experiment, we set the significance level to 0.05 and the degrees of freedom to 26. After querying "critical values of t for two-tailed tests", the critical value of t is 2.0555, and the t-value calculated by our experimental results is 2.1062. The calculation results show that the final t-value is greater than the critical value of t (2.1062 > 2.0555). This proves that our model is significantly different from the baseline model in terms of RMSE indicators. Therefore, the performance improvement of our model is effective.

4.3. Impact of embedding vector size (RQ2)

Feature representation enters the embedding era thanks to machine learning. Researchers are accustomed to embedding the representation of everything in artificial intelligence. Although the embedding vector is still difficult to explain theoretically, there is no doubt that each dimension of embedding is trying to express some features of things. In this paper, we query the Glove word model to obtain the embedding representation of each word in the movie scenario. Based on the Glove embedding, the three fully connected networks further generate query vector, key vector and value vector, which is used as the input layer of Transformer. Glove embedding, as the input to the whole model, undoubtedly has a key impact on the final performance of the model. How many dimensions is the best representation? We performed ablation experiments on the length of embedding in this subsection. The glove word model offers four dimensions: 50, 100, 200 and 300. In turn, we feed the vector of these dimensions to the model as input. Figure 2 shows the trend of RMSE generated by different dimensions and the comparison with several major competitor models.

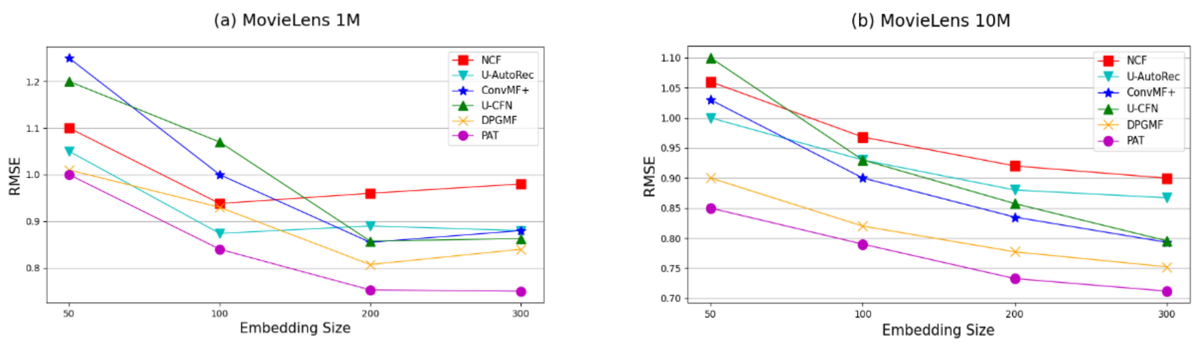

Figure 2. Impact of different embedding sizes on RMSE on ML-1M and ML-10M.

As observed in Figure 2, compared to the baseline model, PAT achieves smaller error and better effect at the embedding of all sizes. On MovieLens 1M, the minimum error can be obtained by using 200 dimensional embeddings. On MovieLens 10M, the better effect can be obtained by using 300 dimensional embeddings. From the overall trend, the longer the embedding length is, the better the performance of the model. When the length increases, the performance of model remains constant even if the error cannot keep decreasing. This demonstrates that our model is able to robustly extract valid semantic information from the noisy representation.

4.4. Impact of sequence length N

Additionally, we studied the effect of the text's maximum sequence length N on the recommendation performance and efficiency of the model. The recommendation performance of various maximum lengths N on MovieLens 1M and MovieLens 10M is shown in Table 6. On the two datasets, when the text was padded to 100 and 200 dimensions, we observed that the error was relatively large and the trend was still downward. When the text was padded to 300 dimensions, the error started to converge and stabilize. Namely, when we continue to increase the length of the text sequence to 400 and 500 dimensions, the error does not drop significantly. We consider that the selection of N depends on the average length of the plot texts in the dataset. If N is too large, it will result in a large number of zeros in the text representation, which will in turn affect the performance of the model. This is because these zeros are only to fill the blank positions and do not contain semantic information.

Table 6. Performance with different text sequence lengths N.

	100	200	300	400	500	100	200	300	400	500
MovieLens 1M	0.8253	0.7846	0.7527	0.7520	0.7609	0.8004	0.7588	0.7118	0.7133	0.7125

5. Conclusions and future work

In order to effectively mine rich semantics information in plot text, to help overcome the weak recommendation problem caused by sparse ratings, we propose a Transformer-based plot-aware deep recommendation framework, called PAT. Its feature is that the framework uses Transformer to learn rich semantic text content from the input layer. Experiments have demonstrated that our approach achieves good results in rating prediction task. Experimental results show that our proposed PAT method has a great improvement in performance compared with all baseline experiments. On MovieLens-1M, PAT RMSE decreased by 6.75% compared to the best baseline DPGMF. On MovieLens-10M, PAT's RMSE dropped 5.37%.

There are several directions to explore for the next step of this paper. The current user-side representation process is still too simple, and we will add graph neural networks on the user side to generate more accurate user representations. In addition, to identify and eliminate the prevalence bias in the dataset, causal inference can be integrated into our model. We can overcome the negative impact of the "Matthew" effect on recommendations by utilizing it.

Acknowledgments

This study was supported by the Jilin Provincial Development and Reform Commission (contract number 2022C046-5), the 2022 Jilin Provincial Education Department Project (contract number JJKH20221172KJ), the 2023 Jilin Provincial Education Department Project (contract number JJKH20231514KJ) and Long term Major Cultivation Project of Philosophy and Social Sciences of Northeast Normal University.

Conflict of interest

The authors declare that there is no conflict of interest.

References

1. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, et al., Attention is all you need, *Adv. Neural Inf. Process. Sys. 30*, **2017** (2017).
2. X. Wang, K. Zhou, J. R. Wen, W. X. Zhao, Towards unified conversational recommender systems via knowledge-enhanced prompt learning, in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, (2022), 1929–1937. https://doi.org/10.1145/3534678.3539382
3. A. Montazeralghaem, J. Allan, Learning relevant questions for conversational product search using deep reinforcement learning, in *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, (2022), 746–754. https://doi.org/10.1145/3488560.3498526
4. Y. Ma, Y. He, A. Zhang, X. Wang, T. S. Chua, CrossCBR: Cross-view contrastive learning for bundle recommendation, preprint, arXiv:220600242.
5. K. Wu, W. Bian, Z. Chan, L. Ren, S. Xiang, S. Han, et al., Adversarial gradient driven exploration for deep click-through rate prediction, in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, (2022), 2050–2058. https://doi.org/10.1145/3534678.3539461
6. Q. Dai, H. Li, P. Wu, Z. Dong, X. Zhou, R. Zhang, et al., A generalized doubly robust learning framework for debiasing post-click conversion rate prediction, in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, (2022), 252–262. https://doi.org/10.1145/3534678.3539270
7. T. Wei, J. He, Comprehensive fair meta-learned recommender system, in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, (2022), 1989–1999. https://doi.org/10.1145/3534678.3539269

8. E. Gholami, M. Motamedi, A. Aravindakshan, Using session partial actions, preprint, arXiv:220913015.
9. Y. Liu, J. N. Yen, B. Yuan, R. D. Shi, P. Yan, C. J. Lin, Practical counterfactual policy learning for Top-K recommendations, in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, (2022), 1141–1151. https://doi.org/10.1145/3534678.3539295
10. J. Chen, W. Fan, G. Zhu, X. Zhao, C. Yuan, Q. Li, et al., Knowledge-enhanced black-box attacks for recommendations, in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, (2022), 108–117. https://doi.org/10.1145/3534678.3539359
11. S. Ishikawa, Y. J. Chung, Y. Hirate, Dynamic collaborative filtering Thompson Sampling for cross-domain advertisements recommendation, preprint, arXiv:220811926.
12. S. Oh, A. Bhardwaj, J. Han, S. Kim, R. A. Rossi, S. Kumar, Implicit session contexts for next-item recommendations, in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, (2022), 4364–4368. https://doi.org/10.1145/3511808.3557613
13. D. R. Turnbull, S. McQuillan, V. Crabtree, S. Zhang, Exploring popularity bias in music recommendation models and commercial steaming services, preprint, arXiv:220809517.
14. M. Naghiaei, H. A. Rahmani, M. Aliannejadi, N. Sonboli, Towards confidence-aware calibrated recommendation, in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, (2022), 4344–4348. https://doi.org/10.1145/3511808.3557713
15. S. Luo, Y. Xiao, Y. Liu, C. Li, L. Song, Towards communication efficient and fair federated personalized sequential recommendation, preprint, arXiv:220810692.
16. Y. Zhang, Z. Chan, S. Xu, W. Bian, S. Han, H. Deng, KEEP: An industrial pre-training framework for online recommendation via knowledge extraction and plugging, in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2022, 3684–3693. https://doi.org/10.1145/3511808.3557106
17. H. Chen, J. Fu, L. Zhang, S. Wang, K. Lin, L. Shi, Deformable convolutional matrix factorization for document context-aware recommendation in social networks, *IEEE Access*, **7** (2019), 66347–66357. https://doi.org/10.1109/ACCESS.2019.2917257
18. L. Guo, Y. Han, H. Jiang, X. Yang, X. Wang, X. Liu, Learning to make document context-aware recommendation with joint convolutional matrix factorization, *Complexity*, **2020** (2020), 1–15. https://doi.org/10.1155/2020/1401236
19. M. Gan, Y. Ma, K. Xiao, CDMF: A deep learning model based on convolutional and dense-layer matrix factorization for context-aware recommendation, **2019** (2019).
20. G. Xu, L. He, M. Hu, Document context-aware social recommendation method, in *2019 International Conference on Computing, Networking and Communications (ICNC)*, (2019), 787–791. https://doi.org/10.1109/ICCNC.2019.8685666
21. H. Chen, Z. Li, Z. Wang, Z. Ni, J. Li, G. Xu, et al., Edge data based trailer inception probabilistic matrix factorization for context-aware movie recommendation, *World Wide Web*, **25** (2021), 1–20. https://doi.org/10.1007/s11280-021-00974-4
22. Z. Wang, H. Chen, Z. Li, K. Lin, N. Jiang, F. Xia, VRConvMF: Visual recurrent convolutional matrix factorization for movie recommendation, *IEEE Trans. Emerg. Topics Comput. Intell.*, **6** (2021), 519–529. https://doi.org/10.1109/TETCI.2021.3102619

23. M. Nguyen, J. Yu, Q. Bai, S. Yongchareon, Y. Han, Attentional matrix factorization with document-context awareness and implicit API relationship for service recommendation, in *Proceedings of the Australasian Computer Science Week Multiconference*, (2020), 1–10. https://doi.org/10.1145/3373017.3373034
24. H. Liu, C. Ling, L. Yang, P. Zhao, Supervised convolutional matrix factorization for document recommendation, *Int. J. Comput. Intell. Appl.*, **17** (2018), 1850018. https://doi.org/10.1142/S1469026818500189
25. X. Zheng, D. Dong, An adversarial deep hybrid model for text-aware recommendation with convolutional neural networks, *Appl. Sci.*, **10** (2019), 156. https://doi.org/10.3390/app10010156
26. P. Liu, J. Du, Z. Xue, A. Li, Bi-convolution matrix factorization algorithm based on improved ConvMF, in *Intelligent Networked Things: 5th China Conference, CINT 2022, Urumqi, China, August 7–8, 2022, Revised Selected Papers*, Springer, (2023), 122–134. https://doi.org/10.1007/978-981-19-8915-5_11
27. G. Cai, N. Chen, Constrained probabilistic matrix factorization with neural network for recommendation system, in *Intelligent Information Processing IX. IIP 2018. IFIP Advances in Information and Communication Technology*, Springer, (2018), 236–246. https://doi.org/10.1007/978-3-030-00828-4_24
28. Q. Wang, S. Li, G. Chen, Word-driven and context-aware review modeling for recommendation, in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, (2018), 1859–1862. https://doi.org/10.1145/3269206.3269258
29. H. Wu, Z. Zhang, K. Yue, B. Zhang, R. Zhu, Content embedding regularized matrix factorization for recommender systems, in *2017 IEEE International Congress on Big Data (BigData Congress)*, (2017), 209–215. https://doi.org/10.1109/BigDataCongress.2017.36
30. Y. Xia, D. Ding, Z. Chang, F. Li, Joint deep networks based multi-source feature learning for QoS prediction, *IEEE Trans. Serv. Comput.*, **15** (2021), 2314–2327. https://doi.org/10.1109/TSC.2021.3050178
31. J. Zhao, Z. Liu, H. Chen, J. Zhang, Q. Wen, Hybrid recommendation algorithms based on ConvMF deep learning model, in *2019 International Conference on Wireless Communication, Network and Multimedia Engineering (WCNME 2019)*, Atlantis Press, (2019), 151–154. https://doi.org/10.2991/wcnme-19.2019.36
32. J. Pennington, R. Socher, C. D. Manning, Glove: Global vectors for word representation, in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, (2014), 1532–1543.
33. T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, preprint, arXiv:1301.3781.
34. D. Kim, C. Park, J. Oh, S. Lee, H. Yu, Convolutional matrix factorization for document context-aware recommendation, in *Proceedings of the 10th ACM Conference on Recommender Systems*, (2016), 233–240. https://doi.org/10.1145/2959100.2959165
35. X. He, L. Liao, H. Zhang, L. Nie, X. Hu, T. S. Chua, Neural collaborative filtering, in *Proceedings of the 26th International Conference on World Wide Web*, (2017), 173–182. https://doi.org/10.1145/3038912.3052569
36. S. Sedhain, A. K. Menon, S. Sanner, L. Xie, Autorec: Autoencoders meet collaborative filtering, in *Proceedings of the 24th International Conference on World Wide Web*, (2015), 111–112. https://doi.org/10.1145/2740908.2742726

37. F. Strub, R. Gaudel, J. Mary, Hybrid recommender system based on autoencoders, in *Proceedings of the 1st workshop on deep learning for recommender systems*, (2016), 11–16. https://doi.org/10.1145/2988450.2988456
38. X. Sun, H. Zhang, M. Wang, M. Yu, M. Yin, B. Zhang, Deep plot-aware generalized matrix factorization for collaborative filtering, *Neural Process. Lett.*, **52** (2020), 1983–1995. https://doi.org/10.1007/s11063-020-10333-5