SSR-MGTI: Self-attention Sequential Recommendation Algorithm Based on Movie Genre Time Interval

Wen Yang[1,2](✉), Ruibo Yue[2], Yawen Chen[3], and Jun Zhao[4]

[1] Hubei Key Laboratory of Intelligent Vision Based Monitoring for Hydroelectric Engineering, China Three Gorges University, Yichang 443002, China
yangwen0720@163.com

[2] College of Computer and Information Technology, China Three Gorges University, Yichang 443002, China

[3] University of Otago, Dunedin 9016, New Zealand
yawen@cs.otago.ac.nz

[4] Hubei Three Gorges Polytechnic, Yichang 443000, China

Abstract. As an important part of the recommendation system, movie recommendation system can recommend movies to users accurately according to their preferences. Traditional movie recommendation systems simply treat user-movie interactions as a time-ordered sequence, without considering the time intervals between movies of the same genre. The genre time interval can reflect the user's preference for a particular genre and determine whether the algorithm can fully capture the user's interests and the time characteristics of the movie, which plays an important role in the accuracy of the movie recommendation. Therefore, in this paper, we propose a Self-Attention Sequential Recommendation algorithm based on Movie Genre Time Interval (SSR-MGTI). Specifically, a multi-head self-attention mechanism is used to model the same genre time interval information. Then, an absolute position is added to the multi-head self-attention mechanism model to solve the problem that multi-head self-attention mechanism does not consider the sequence. In addition, the convolutional neural network is used to convert the model from linear to non-linear and extract local information of user-movie interaction sequences. It is interesting to show that the proposed SSR-MGTI can accurately predict the movie that the user will watch next time. Experimental results on MovieLens and Amazon datasets demonstrate the superiority of our SSR-MGTI over state-of-the-art movie recommendation methods.

Keywords: Movie recommendation system · Genre time interval · Multi-head self-attention mechanism · Convolutional neural network

1 Introduction

Personalized recommendation is one of the most popular recommendation methods at present, which can tailor the recommended content to the users according to their unique preferences. Personalized recommendation algorithm mainly includes collaborative filtering-based recommendation algorithm, content-based recommendation algorithm, and sequential recommendation algorithm. The collaborative filtering-based recommendation algorithm [16] can recommend items according to a certain similarity (similarity between users or similarity between items) through the behavior of groups. The content-based recommendation algorithm [7] only utilizes the basic information (e.g., gender, age) of the user and the user-item interactions to forecast the user's preferences without taking into account the information of other users. The sequential recommendation algorithm [2, 19] attempts to predict the user's next new item by exploiting their historical behavior sequences. The sequential recommendation algorithm is particularly important in movie recommendation, since it can model the relationship between historically watched movies as a dynamic sequence to find the hidden information between movies.

Most of current sequential recommendation algorithms rank movies by interaction timestamps. However, these sequential recommendation algorithms only model the time series, ignoring the temporal information hidden in the timestamp itself. For example, Markov chain [9, 21, 22] assumes that the next movie is related to the previous movies, which only considers the time sequence, without considering time itself. Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) [21] model time series through hidden states. However, both CNN and RNN only compress temporal information into fixed hidden vectors, thus ignoring the temporal relationship between the various movies. The recently emerged "self-attention" mechanism (Self-Attention) can allocate different weights to the information according to their importance [1], but Self-Attention does not take the sequence of the series.

In this paper, we propose a Self-Attention Sequential Recommendation Algorithm based on Movie Genre Time Interval (SSR-MGTI). Specifically, we add absolute position to the multi-head self-attention mechanism to provide the sequential position of the movie. Then, we use the time interval between movies to represent the time information, and model the same genre of time interval of the user-movie interaction sequence to predict the next movie. In order to improve the model's fitting ability and highlight the importance of local preferences, we add CNN to improve the model's prediction ability. Finally, our contributions are summarized as follows:

- We model the same genre of time interval information in the user-movie interaction sequence.
- We use the multi-head self-attention mechanism to train the same genre of time interval by adding the absolute position information of the movie.
- We add the CNN to improve the stability and generalization ability of the model structure and capture the local information of user-movie interaction sequences.
- We carry out extensive experiments on MovieLens and Amazon datasets, which shows that our algorithm outperforms the state-of-the-art algorithms.

2 Related Work

During the past decades, extensive algorithms based on sequential recommendation have been proposed in the recommendation systems area. In general, existing methods can be categorized into three groups: general sequential recommendation method, deep learning-based sequential recommendation method, and self-attention-based sequential recommendation method.

General sequential recommendation algorithms include sequential pattern mining and Markov chain models. Yap et al. [23] proposed a recommendation framework based on personalized sequential pattern mining, which effectively learned important knowledge of user sequences. The FPMC model [15] proposed by Rendle et al. combines Matrix Factorization with the Markov Chain model, which incorporates both the common Markov chain and the normal matrix factorization model. It introduces modifications to the Bayesian personalized ranking framework recommended for the sequential basket. However, the Markov chain model can only capture the local information of the sequence, ignoring the global information related to the sequence.

In deep learning-based sequential recommendation method, RNN and CNN are most commonly used algorithms. RNN is inherently capable of processing sequence data. In order to solve the long-term dependency problem in RNN, two variants of RNN are generated, namely Long Short-Term Memory Neural Network (LSTM) [12, 17] and Gated Recurrent Unit (GRU) [5]. Duan et al. [6] proposed a new architecture based on LSTM for RNN ignoring collective dependencies due to the monotonous temporal relationship between items. The model adds the “Q-K-V” triplet to the recurrent unit to enhance the memory ability of LSTM, and proposes a “recovery gate” to solve the memory loss problem caused by the “forget gate”. However, RNN is only suitable for long-term sequences. CNN can treat sequence as one-dimensional space and extract features from local sequence convolution. Tang et al. [18] proposed a convolutional sequence embedding model (Caser) to embeds recent sequence items into the “image” of time and latent space, which can use a convolution filter to turn the sequence into a local feature of the image. However, CNN is only good at capturing short-term sequences, which is not suitable for long-term sequences.

In recent years, self-attention mechanism has attracted great attention in the fields of natural language processing and computer vision. Chiang et al. [4] proposed a stacked attention network model, which stacks contextual item attention modules with multi-head attention modules and improves recommendation performance by using additional time information to model contextual items. Kang et al. [13] proposed a sequence model based on self-attention (SAS-Rec), which can be used to balance its sparse and dense data sets. Li et al. [14] proposed time interval-aware self-attention sequential recommendation (TiSAS-Rec), which models time intervals in user interaction sequences and uses the time interval information to predict the next item.

Although all the above methods can use timestamps to model time series, they seldom use the time information of timestamp itself, and do not take into account the time interval characteristics of the same genre. However, the genres of movies watched by users are different, and the time interval between them can better reflect the interests of users. In this paper, we will model the same genre time interval information of movies as the relationship between movies, and add absolute position information to the multi-head self-attention mechanism.

3 Problem Description of Movie Genre Time Interval

Since our recommendation algorithm is based on the movie genre time interval, we give the definition of movie genre time interval firstly in this section, followed by the problem description.

Movie Genre Time Interval (MGTI) refers to the time length between movies with the same category in the user-movie interaction sequence. The time interval between the same type of movies in the user-movie interaction sequence can reflect the user's recent preference for this type of movie. The smaller the time interval between two movies of the same type indicates that the user likes this type of movies more recently.

The MGTI can be modeled as following: Let $U = \{u_i | 1 \leq i \leq N\}$, $V = \left\{v_j | 1 \leq j \leq M\right\}$ and $G = \{g_k | 1 \leq k \leq H\}$ represent the user set, the movie set and the genre set, respectively. Each movie in the movie set has a corresponding timestamp, which can be represented by $T = \left\{t_q | 1 \leq q \leq M\right\}$. For a user u_i, the user-movie interactive sequence can be denoted by $S_i = \left(s_{i1}^{g_1}, s_{i2}^{g_2}, \ldots, s_{ij}^{g_k}, \ldots, s_{iM}^{g_H}\right)$, $i \in [1, N], j \in [1, M], k \in [1, H]$. MGTI can be denoted by $r_{cd}^{u_i} = s_{id}^{g_a} - s_{ic}^{g_b}$, where g_a and g_b represent type set and $g_a \cap g_b \neq \emptyset$. The absolute position sequence refers to the position of the movie in the user-movie interaction sequence, defined as $P = (1, 2, \ldots, M)$. At the time t, the model predicts the next movie based on the previous $t - 1$ movies and $r_{cd}^{u_i}$. The input of our model is a user-movie interaction sequence (S_i), an absolute position of the movie in the user-movie interaction sequence (P) and a genre time interval matrix of user-movie interaction sequence (R^i). The genre time interval matrix of user-movie interaction sequence can be denoted as below.

$$R^i = \begin{bmatrix} r_{11}^i & r_{12}^i & \cdots & r_{1n-1}^i & r_{1n}^i \\ r_{21}^i & r_{22}^i & \cdots & r_{2n-1}^i & r_{2n}^i \\ r_{31}^i & r_{32}^i & \cdots & r_{3n-1}^i & r_{3n}^i \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ r_{n1}^i & r_{n2}^i & \cdots & r_{nn-1}^i & r_{nn}^i \end{bmatrix} \tag{1}$$

4 Multi-head Self-attention Mechanism Based on Movie Genre Time Interval

The overall framework of the model is shown in Fig. 1. This model includes three parts: 1) Embedding Layer: This layer vectorizes the sequence (S_i) and genre time interval matrix (R^i), and the absolute position of the movie in the user-movie interaction sequence (P), and embeds them in a low-dimensional space. 2) Multi-Head Self-Attention Mechanism Layer: This layer focuses on more relevant movies (i.e., genre time interval is shorter) and gives more weight to these movies. By assigning different weights to the movies, the recommendation results can be more personalized. 3) Convolutional Neural Network

Layer: This layer can convert the model from linear to non-linear and capture local information of user-movie interaction sequences. So it can improve the fitting ability of the model and the stability and generalization ability of the model structure.

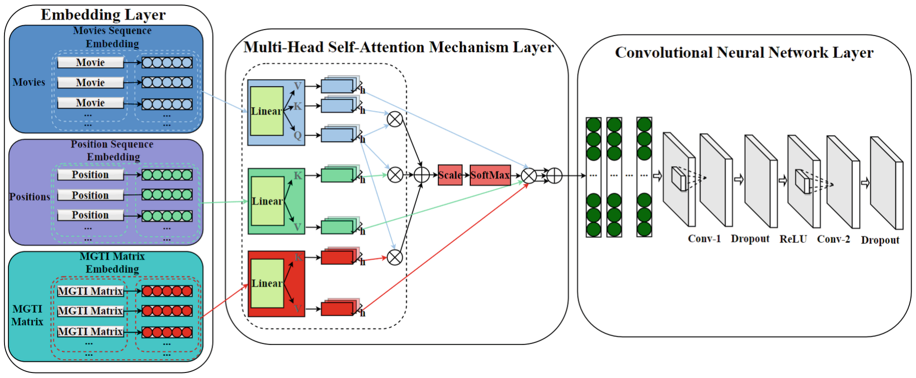

Embedding Layer
Movies Sequence Embedding
Movies
Movie
Position Sequence Embedding
Positions
Position
MGTI Matrix Embedding
MGTI Matrix
Multi-Head Self-Attention Mechanism Layer
Linear
V
K
Q
h
Scale
SoftMax
Convolutional Neural Network Layer
Conv-1
Dropout
ReLU
Conv-2
Dropout

Fig. 1. Overall framework of Multi-Head Self-Attention Mechanism based on Movie Genre Time Interval.

4.1 Embedding Layer

We use the embedding layer to map the user-movie interaction sequence (S_i) to a lower dimensional space. The embedding layer uses the user interaction movie *ID* (the serial number of the movie in the dataset) as a numerical index to create an embedding matrix $E_S \in \mathbf{R}^{\mathbf{c}\times\mathbf{d}}$, where c is the dictionary size and d is the potential dimension. It maps user interaction movie *ID* to fixed-size vectors by embedding matrix (E_S), where S represents user-movie interaction sequence. So we will get the mapped low-dimensional matrix $O_S \in \mathbf{R}^{\mathbf{n}\times\mathbf{d}}$, where n is the maximum length of the sequence.

Similar to the user-movie interaction sequence, we use an embedding layer to map absolute position (P) to a lower dimensional space. The difference is that we will use two different embedding matrices to generate the keys and values in the multi-head self-attention mechanism without requiring additional linear transformations. Because the absolute position is a number, we use it as the numerical index to create embedding matrix $E_P^K \in \mathbf{R}^{\mathbf{c}\times\mathbf{d}}$ and $E_P^V \in \mathbf{R}^{\mathbf{c}\times\mathbf{d}}$ and map absolute position to fixed-size vectors by embedding matrix $\left(E_P^K, E_P^V\right)$, where P, K and V represent absolute position, keys and values of multi-head self-attention mechanism. Therefore, we can get the mapped low-dimensional matrices $O_P^K \in \mathbf{R}^{\mathbf{n}\times\mathbf{d}}$ and $O_P^V \in \mathbf{R}^{\mathbf{n}\times\mathbf{d}}$.

Likewise, we use genre time interval matrix (R^i) as numerical indexes to create embedding matrix $E_R^K \in \mathbf{R}^{\mathbf{c}\times\mathbf{d}}$ and $E_R^V \in \mathbf{R}^{\mathbf{c}\times\mathbf{d}}$ and map genre time interval matrix to fixed-size vectors by embedding matrix $\left(E_R^K, E_R^V\right)$, where R represents genre time interval matrix. Therefore, we can get the mapped low-dimensional matrices $O_R^K \in \mathbf{R}^{\mathbf{n}\times\mathbf{d}}$ and $O_R^V \in \mathbf{R}^{\mathbf{n}\times\mathbf{d}}$.

4.2 Multi-head Self-attention Mechanism Layer

The multi-head self-attention mechanism can give different weights to the movies according to the importance information in the time sequence, which works as following. Firstly, we calculate the attention weight α_{ij} by the following softmax function:

$$\alpha_{ij} = \frac{e^{v^{ij}}}{\sum_{k=1}^{n} e^{v^{ik}}}, \tag{2}$$

where v^{ij} is calculated using low-dimensional matrices $\left(O_S, O_R^K \text{ and } O_P^K\right)$.

$$v^{ij} = \frac{O_S W^Q \left(O_S W^K + O_R^K + O_P^K\right)}{\sqrt{d}}, \tag{3}$$

where $W^Q \in \mathbf{R}^{\mathrm{d} \times \mathrm{d}}$ and $W^K \in \mathbf{R}^{\mathbf{d} \times \mathbf{d}}$ are calculated by a fully connected layer, and W^Q and W^K are the coefficients of query and key. d is the dimension of the hidden layer. $\sqrt{d}$ is used to avoid large values of softmax.

Secondly, according to the attention weight α_{ij}, we calculate the final result of the multi-head self-attention mechanism model, that is, the weighted sum of value:

$$Z_i = \sum_{j=1}^{n} \alpha_{ij} \left(O_S W^V + O_R^V + O_P^V\right), \tag{4}$$

where $W^V \in \mathbf{R}^{\mathbf{d} \times \mathbf{d}}$ is calculated by a fully connected layer and W^V is the value of coefficient.

4.3 Convolutional Neural Network Layer

In order to improve the model's fitting ability and highlight the importance of local preferences, we add CNN to improve the model's prediction ability.

Firstly, we use a layer of 1D convolutional neural network for feature extraction:

$$F_i^1 = W^1 Z_i + b^1 \tag{5}$$

Secondly, The *ReLU* activation function is used after the first layer of convolutional neural network as follows:

$$F_i^2 = ReLU\left(F_i^1\right) \tag{6}$$

Finally, we use a layer of 1D convolutional neural network:

$$F_i^3 = F_i^2 W^2 + b^2 \tag{7}$$

where $W^1 \in \mathbf{R}^{\mathbf{d} \times \mathbf{d}}$ and $W^2 \in \mathbf{R}^{\mathbf{d} \times \mathbf{d}}$ are the parameter matrices of the first and the second convolutional neural network layers, respectively. $b^1 \in \mathbf{R}^{\mathbf{d}}$ and $b^2 \in \mathbf{R}^{\mathbf{d}}$ are the bias terms. F^1, F^2 and F^3 are the output of each layer.

5 Model Prediction

5.1 Prediction Layer

In the multi-head self-attention mechanism layer and the convolutional neural network layer, the increase in the number of model layers will lead to problems of overfitting, gradient disappearing and long training time. So we use layer normalization and *Dropout* regularization techniques to solve these problems:

$$g(x) = x + Dropout(g(LayerNorm(x))) \tag{8}$$

where $g(x)$ is multi-head self-attention mechanism layer or the convolutional neural network layer, and x is the input. We use the layer normalization technique on the input (x). Then we use the *Dropout* technique on the output of the multi-head self-attention mechanism layer or the convolutional neural network layer $(g(x))$. At last, we incorporate the input (x) into this result.

5.2 Loss Function

The binary cross-entropy loss function is commonly used in recommendation systems, which measures the predictive accuracy of the model by calculating the difference between the real label and the predicted label. It allows the model to converge fast and can be updated in real time without retraining the entire model. Therefore, we adopt the binary cross-entropy loss function as following:

$$-\sum_{S^u \in S} \sum_{t \in [1,2,\ldots,n]} \left[\log\left(\sigma\left(r_{o_t,t}\right)\right) + \log\left(1 - \sigma\left(r_{o'_t,t}\right)\right) \right] + \lambda \|\Theta\|_F^2 \tag{9}$$

where r_{o_t} represents positive output, $r_{o'_t}$ represents negative sampling, $\Theta = \left\{O_S, O_P^K, O_P^V, O_R^K, O_R^V\right\}$ is a low-dimensional matrix set of mapping, $\|\cdot\|_F$ represents *Frobenius* norm, and λ represents regularization coefficient.

6 Experimental Evaluation

In this section, we evaluate SSR-MGTI through extensive simulations. Firstly, we present the experimental setup, datasets and evaluation metrics in Sect. 6.1. Then, the results of SSR-MGTI and 7 recommendation baselines (GRU4Rec+ [11], NCF [10], Caser [18], SASRec [13], TiSASRec [14], LSPM [3], SSE-PT [20]) on Movielens and Amazon datasets are presented in Sect. 6.2. Finally, we also show the results of comparison under 3 different hyperparameters settings of SASRec, TiSASRec, and SSE-PT.

6.1 Experimental Configuration

Experimental Setup. All the following experiments were performed on NVIDIA RTX3090Ti GPU, and the code was implemented based on the Pytorch. The dropout rate of the Movielens dataset is 0.2 and the dropout rate of the Amazon dataset is 0.8.

Datasets. We evaluate our method on two datasets from two platforms. Dataset statistics of two datasets after preprocessing are shown in Table 1. MovieLens is the dense dataset which has more average actions with fewer users and movies. Amazon is the sparse dataset which has the fewer actions per user and movie.

- **MovieLens**: This dataset is often used in the recommendation system competition. We will use a version with 1,000,209 anonymous ratings of approximately 3,900 movies made by 6,040 MovieLens users who joined MovieLens in 2000. (MovieLens-1M).
- **Amazon**: This dataset records users' comments on Amazon website. It is the classic dataset of the recommendation system and Amazon has been updating this dataset. We will use the Video_Games dataset from the 2014 release. (Amazon Video_Games)

Table 1. Dataset statistics (after preprocessing)

Dataset	#Users	#movies	avg.actions/user	avg.actions/movie
Movielens	6040	3416	163.50	289.09
Amazon	31013	23715	7.26	9.50

Evaluation Metrics. We use two common Top-N metrics to evaluate the performance of our method: Hit Rate@10 and NDCG@10 [8, 10]. Hit Rate@10 is mainly concerned about whether the movie that users like is recommended, which emphasizes the "accuracy" of prediction. NDCG@10 is more concerned about the "order", which emphasizes whether the recommended movie appears in a higher position in the recommended sequence.

6.2 Results and Analysis

Results on Different Recommendation Methods. We study the performance of our proposed model SSR-MGTI with all baselines on two real-world datasets. Table 2 shows the experimental results of all the methods. It can be observed that:

(1) SSR-MGTI can always achieve the best performance regardless of datasets and evaluation metrics, which can gain 20.13% Hit Rate and 41.06% NDCG improvements on average compared with other methods.
(2) SSR-MGTI has a significant improvement in the NDCG metric on both sparse and dense datasets, which can achieve performance up to 25.65% on dense dataset and up to 56.46% on sparse dataset.

Ablation Study. We conduct experiments by removing position, CNN, dropout and layernorm separately to demonstrate the role of each component of our model. Table 3 shows the performance of the two datasets with the best set of hyperparameters.

For the Movielens dataset, removing the added components clearly shows that the recommendation performance has declined, especially for the dropout component. The

Table 2. Performance of different recommendation methods. The best performance in each row is boldfaced (higher is better), and the second best method in each row is underlined. Improvements are shown in the last column.

Dataset	Metric	GRU4Rec +	NCF	Caser	SASRec	TiSASRec	LSPM	SSE-PT	SSR-MGTI	Improvement
Movielens	Hit Rate@10	0.6522	0.6954	0.7517	0.8174	0.8311	0.8303	0.8371	**0.8760**	**13.24%**
	NDCG@10	0.4334	0.5193	0.5011	0.5786	0.6108	0.6240	0.6160	**0.6970**	**25.65%**
Amazon	Hit Rate@10	0.3971	0.6642	0.4474	0.7551	0.7327	0.6157	0.7466	**0.7909**	**27.01%**
	NDCG@10	0.2321	0.4632	0.2661	0.5425	0.5256	0.4210	0.5448	**0.6695**	**56.46%**

Table 3. Ablation analysis (NDCG@10) on two datasets. Performance better than the default version is boldfaced. '↓' indicates performance drop.

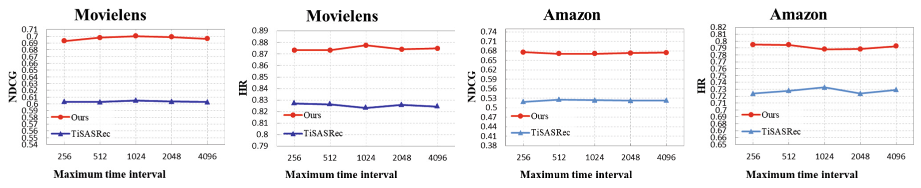
Dataset	Default	Remove Position	Remove CNN	Remove Dropout	Remove Layernorm
Movielens	0.6970	0.6846 ↓	0.6865 ↓	0.6521 ↓	0.6890 ↓
Amazon	0.6695	**0.6795**	**0.6714**	0.5530 ↓ (overfitting)	0.6410 ↓

position, CNN and layernorm components can modify the model to some extent. Adding the absolute position of the user-movie interaction sequence can be combined with the relative position of the type time interval to better capture the connection between movies. Adding the CNN component can not only convert the linear model into a non-linear model, but also extract short-term preferences. Removing the layernorm component will reduce the generalization ability of the model and may also cause gradient disappearance and gradient explosion problems, so it is lower than the default model metric. Removing the dropout component will cause the recommendation metric of the model to drop sharply, which shows that the dropout component greatly affects the recommendation performance of the model. It can also be seen from the output results of the model that the evaluation metric fluctuates around 0.6500, which indicates that the model overfitting problem is not obvious on the dense dataset.

For the Amazon dataset, removing the dropout and layernorm components will cause a severe performance drop, especially removing dropout will cause overfitting problems (evaluation metric gradually decrease). Removing the layernorm component will seriously hurts model performance for sparse datasets. Removing the position and CNN components is more suitable for sparse datasets. Since the sparse data interaction sequence is less, the absolute position plays a little role and there is little difference between the short-term feature and the long-term feature. Adding position and CNN may easily increase the noise of the data, so the recommendation performance of the model will be improved.

Comparison of 3 Different Hyperparameters Settings. We compared 3 hyperparameters (i.e., maximum genre time interval, maximum sequence length n and number of heads of attention) based on the SASRec, TiSASRec, SSE-PT and SSR-MGTI models, and the maximum movie genre time interval was only compared with TiSASRec.

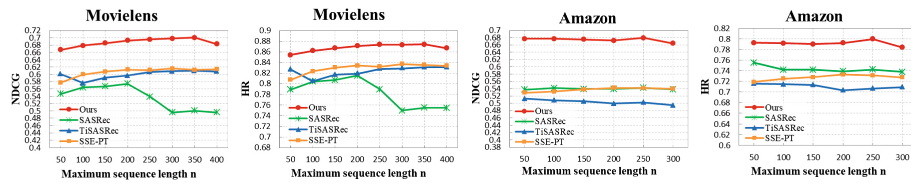
(1) Influence of the maximum genre time interval. The maximum genre time interval refers to the maximum value of the set genre time interval. Since maximum genre time interval may replace the computed movie genre time interval for training, it has a great impact on the recommendation performance. Figure 2 shows the effect of maximum time interval between TiSASRec and SSR-MGTI on the two datasets. From Fig. 2, we can see that SSR-MGTI can always achieve better performance than TiSASRec under different datasets, which can gain 7.41% Hit Rate and 22.16% NDCG improvements on average. This is because our method adds movie genre features to the time interval, which enables the model to accurately capture temporal information between movie genres in user-movie interaction sequences.



(a) (b) (c) (d)

Fig. 2. Effect of maximum genre time interval on ranking performance

(2) Influence of maximum sequence length *n***.** The maximum sequence length *n* refers to the length of the user-movie interaction sequence, which determines the number of data that can be added to the model for training. From the results, we can see that SSR-MGTI gains 7% Hit Rate and 18.99% NDCG improvements on average in the Movielens dataset. As shown in Fig. 3 (a) and (b), the recommendation performance of the TiSASRec, SSE-PT and SSR-MGTI models increases with the increase of *n*. But the SASRec rises firstly and then declines. It may because SASRec uses fewer features. So a large number of 0 are filled with *n* increases, resulting in a decline in recommendation performance. For the Amazon dataset, SSR-MGTI gains 8.95% Hit Rate and 28.04% NDCG improvements on average. As shown in Fig. 3 (c) and (d), the recommendation performance of the four models decreases slightly with the increase of *n*.



(a) (b) (c) (d)

Fig. 3. Effect of maximum sequence length *n* on ranking performance

(3) Influence of the number of heads of multi-head self-attention. Since number of heads of multi-head self-attention can enable the network to capture the user's interests from multiple aspects, it has a great impact on the recommendation performance. As shown in Fig. 4 (a) and (b), the recommendation performance of the SASRec and SSE-PT model increases significantly. But the recommendation performance of the TiSASRec and SSR-MGTI models firstly increases then decreases. This is because when the number of heads is too large, a large number of parameters will be generated to cause overfitting problems. For the Movielens dataset, SSR-MGTI gains 8.21% Hit Rate and 21.74% NDCG improvements on average. For the Amazon dataset (Fig. 4 (c) and (d)), SSR-MGTI gains 7.37% Hit Rate and 25.12% NDCG improvements on average.

It can be seen that the performance of SSE-PT is slightly improved. But the performance of SSR-MGTI, TiSASRec and SASRec decreases as the number of heads increases. This is because the dataset is relatively sparse, the hidden information of the data is relatively fewer.

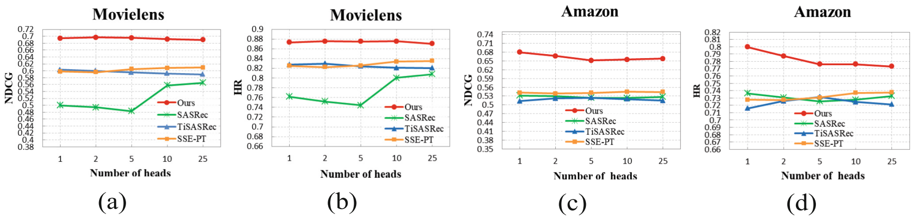


Fig. 4. Effect of the number of heads of multi-head self-attention on ranking performance

7 Conclusion

In this paper, we proposed a Self-Attention Sequential Recommendation Algorithm based on Movie Genre Time Interval (SSR-MGTI). Firstly, we give the definition of Movie Genre Time Interval (MGTI), based on which a multi-head self-attention mechanism is modeled. Then, we add the absolute position of the movie in the user-movie interaction sequence to make up for the deficiency of the multi-head self-attention mechanism. In addition, we use a convolutional neural network to convert the model to non-linear and extract local information of user-movie interaction sequences. The experiment results show that our proposed recommendation scheme can achieve 20.13% and 41.06% improvement in HR@10 and NDCG@10 respectively over other state-of-the-art schemes in terms of dense (MovieLens) and sparse (Amazon) datasets.

References

1. Ashish, V.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30, p. I (2017)
2. Chang, J., et al.: Sequential recommendation with graph neural networks. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 378–387 (2021)
3. Chen, J., Jiang, L., Sun, H., Ma, C., Liu, Z., Zhao, D.: LSPM: joint deep modeling of long-term preference and short-term preference for recommendation. In: Gedeon, T., Wong, K., Lee, M. (eds.) Neural Information Processing: 26th International Conference, ICONIP 2019, Sydney, NSW, Australia, 12–15 December 2019, Proceedings, Part IV. CCIS, vol. 1142, pp. 237–246. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-36808-1_26
4. Chiang, J.H., Ma, C.Y., Wang, C.S., Hao, P.Y.: An adaptive, context-aware, and stacked attention network-based recommendation system to capture users' temporal preference. IEEE Trans. Knowl. Data Eng. **35**(4), 3404–3418 (2022)
5. Cho, K., et al.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078 (2014)
6. Duan, J., Zhang, P.F., Qiu, R., Huang, Z.: Long short-term enhanced memory for sequential recommendation. World Wide Web **26**(2), 561–583 (2023)
7. Fkih, F.: Similarity measures for collaborative filtering-based recommender systems: review and experimental comparison. J. King Saud Univ.-Comput. Inf. Sci. **34**(9), 7645–7669 (2022)
8. He, R., Kang, W.C., McAuley, J.: Translation-based recommendation. In: Proceedings of the Eleventh ACM Conference on Recommender Systems, pp. 161–169 (2017)
9. He, R., McAuley, J.: Fusing similarity models with Markov chains for sparse sequential recommendation. In: 2016 IEEE 16th International Conference on Data Mining (ICDM), pp. 191–200. IEEE (2016)
10. 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)
11. Hidasi, B., Karatzoglou, A.: Recurrent neural networks with top-k gains for session-based recommendations. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 843–852 (2018)
12. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
13. Kang, W.C., McAuley, J.: Self-attentive sequential recommendation. In: 2018 IEEE International Conference on Data Mining (ICDM), pp. 197–206. IEEE (2018)
14. Li, J., Wang, Y., McAuley, J.: Time interval aware self-attention for sequential recommendation. In: Proceedings of the 13th International Conference on Web Search and Data Mining, pp. 322–330 (2020)
15. Rendle, S., Freudenthaler, C., Schmidt-Thieme, L.: Factorizing personalized Markov chains for next-basket recommendation. In: Proceedings of the 19th International Conference on World Wide Web, pp. 811–820 (2010)
16. Shen, J., Zhou, T., Chen, L.: Collaborative filtering-based recommendation system for big data. Int. J. Comput. Sci. Eng. **21**(2), 219–225 (2020)
17. Sundermeyer, M., Schlüter, R., Ney, H.: LSTM neural networks for language modeling. In: Thirteenth Annual Conference of the International Speech Communication Association (2012)
18. Tang, J., Wang, K.: Personalized top-n sequential recommendation via convolutional sequence embedding. In: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, pp. 565–573 (2018)

19. Wang, D., Xu, D., Yu, D., Xu, G.: Time-aware sequence model for next-item recommendation. Appl. Intell. **51**, 906–920 (2021)
20. Wu, L., Li, S., Hsieh, C.J., Sharpnack, J.: SSE-PT: sequential recommendation via personalized transformer. In: Proceedings of the 14th ACM Conference on Recommender Systems, pp. 328–337 (2020)
21. Xu, C., et al.: Long-and short-term self-attention network for sequential recommendation. Neurocomputing **423**, 580–589 (2021)
22. Yan, C., Wang, Y., Zhang, Y., Wang, Z., Wang, P.: Modeling long-and short-term user behaviors for sequential recommendation with deep neural networks. In: 2021 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. IEEE (2021)
23. Yap, G.E., Li, X.L., Yu, P.S.: Effective next-items recommendation via personalized sequential pattern mining. In: Lee, S.G., Peng, Z., Zhou, X., Moon, Y.S., Unland, R., Yoo, J. (eds.) Database Systems for Advanced Applications: 17th International Conference, DASFAA 2012, Busan, South Korea, 15–19 April 2012, Proceedings, Part II 17, vol. 7239, pp. 48–64. Springer, Cham (2012). https://doi.org/10.1007/978-3-642-29035-0_4