Hindawi
Journal of Electrical and Computer Engineering
Volume 2023, Article ID 3162220, 9 pages
https://doi.org/10.1155/2023/3162220

Research Article

Personalized Recommendation Model Based on Improved GRU Network in Big Data Environment

Hui Guo,[1] **Zheng Guo**,[2] **and Zhihong Liu**[3]

[1]*Shanxi Vocational College of Tourism, Taiyuan, Shanxi 030031, China*
[2]*China National Pharmaceutical Group Shanxi Rfl Pharmaceutical Co.,Ltd., Taiyuan, Shanxi 030012, China*
[3]*Zhengzhou College of Finance and Economics, Zhengzhou, Henan 450044, China*

Correspondence should be addressed to Hui Guo; huiguosx2012@163.com

Received 15 April 2022; Revised 18 May 2022; Accepted 2 June 2022; Published 24 March 2023

Academic Editor: Xuefeng Shao

To address the diversity of user preferences and dynamic changes of interests in the personalized recommendation scenario, a personalized recommendation model based on the improved gated recurrent unit (GRU) network in a big data environment is proposed. First, in order to deal with outliers in sequence recommendation, context awareness sequence recommendation is introduced, and the dynamic changes of users' interests are modeled by redefining the update gate and the reset gate of the GRU. Then, the duration information about how long users browse each item is processed and transformed to obtain the duration attention factor of each recommended item. And the duration attention factors and the item information are together used as the input of the proposed model for training and prediction. Finally, the auxiliary loss function is introduced to make up for the shortcomings of the traditional negative logarithmic likelihood function, and a super-parameter is applied to combine the auxiliary loss function with the negative logarithmic likelihood function so as to enhance the relationship between the interest representation and the accuracy of recommendation. Experiments show that the root mean square error (RMSE) of the proposed method in the Criteo dataset and MovieLens-1M dataset is 0.7257 and 0.7869, respectively, and the mean absolute error (MAE) is 0.5147 and 0.5893, respectively, which are better than those of the comparison methods. Therefore, the proposed method significantly outperforms the comparison methods in improving the accuracy of personalized recommendation in the system.

1. Introduction

The recommendation system is a kind of information filtering tools that deals with the problem of information overload, and it can provide users with content that may be of interest to them in a personalized way [1, 2]. In the field of recommendation system, the products or services recommended to users are collectively referred to as items. Recommendation systems predict how interested a user is in a particular target item based on basic characteristics of users, behavioral feedback, and information derived from the attributes of the target item [3]. Unlike search engines that require users to enter keywords, recommendation systems infer users' interests from their implicit feedback and then tailor personalized recommendation services to improve user satisfaction with online service websites [4, 5]. With the rapid development of information technology, the application of recommendation systems in industry has also gained much importance [6]. Nowadays, the study of recommendation systems has been extended to several aspects, including what to recommend (based on the characteristics of users/items), when to recommend (based on time-aware recommendations), where to recommend (based on geolocation recommendations), who to recommend (based on social network recommendations), and why to recommend (explainable recommendations) [7, 8]. Showing why an item is recommended not only helps users understand the rationale for the recommendation, but also helps improve the efficiency, transparency, and credibility of the system [9, 10].

Deep learning is a powerful learning framework that can efficiently process time sequence and high-dimensional sparse data, providing great impetus to recommendation strategies [11–13]. In the traditional research on recommendation algorithms, interpretability and personalization

are often opposed to each other [14, 15]. A simple and easy-to-understand recommendation strategy would be more interpretable, while at the expense of the accuracy of predictions [16, 17]. By contrast, complex models can improve the accuracy of predictions, but the recommendation system cannot provide a reasonable explanation to the user [18]. The application of techniques related to deep learning offers a new solution to this problem. Based on deep learning, recommendation algorithms can be explored with both efficiency and interpretability [19, 20].

To address these problems, a personalized recommendation method based on improved GRU network in a big data environment is proposed. The main contributions of the proposed method are as follows:

(1) The context is classified into four categories: input context, relevance context, static interest context, and transfer context. And the dynamic changes in user interest are modeled by redefining the update gate and the reset gate of the gated recurrent unit (GRU).

(2) The duration information about how long each item is browsed by users is analyzed and transformed to obtain the duration attention factor of each recommended item. Meanwhile, the duration attention factor and item information are jointly used as the input for the improved GRU model, which improves the accuracy of prediction.

(3) The relationship between interest representation and the accuracy of recommendations is enhanced by combining an auxiliary loss function with a negative logarithmic likelihood function using a hyperparameter.

2. Related Studies

Aiming at the problems of large scale, fast update, and noise in YouTube, Covington et al. (2016) proposed to build a deep learning network with the embedding layer to learn the characteristics of videos and users, which are finally used for personalized recommendations [21]. To deal with sparsity in recommendation algorithms and overgeneralization in deep learning, Cheng et al. (2016) proposed a recommendation algorithm that combined the generalized linear model and the deep learning model [22]. In Ref. [23], a tag recommendation method based on a convolutional neural network was investigated, which extracted and fused the features of images and user interaction information through the convolutional neural network to obtain personalized tag recommendations in order. Guo et al. (2017) proposed a model based on deep matrix decomposition to learn complex and high-dimensional interaction information of users, so as to realize the prediction of click-through rates [24]. He et al. (2017) studied a collaborative filtering recommendation algorithm based on a multilayer perceptron [25]. The algorithm successfully migrated deep learning to collaborative filtering recommendation systems and designed a general model for neural collaborative filtering algorithms. Feng et al. (2019) were dedicated to capturing the dynamic interests embedded behind users' behavior and proposed a personalized recommendation method based on the deep interest network to formulate the evolution of users' interests [26]. Liu et al. (2016) argued that the effective use of temporal and spatial contextual information can help predict the trend in access preferences of users' interest points and can effectively improve the capability of predicting the next interest point [27]. And by extending the recurrent neural network, a novel recommendation model called spatial-temporal recurrent neural networks (ST-RNN) was proposed. To address the issue of mining the actual preferences of users in personalized recommendation, Yin et al. (2019) proposed an attention-based deep learning POI recommendation (ADPR) framework [28]. However, the feature extraction of this recommendation method takes too long and increases the parameters to be trained in each round, so it takes a long time on a large dataset. Zhang et al. (2020) investigated a deep neural model based on transfer learning that incorporated cross-domain knowledge to achieve more accurate personalized recommendations [29]. The above methods can only mine shallow information during the feature extraction, while a large amount of deeper information about features is hard to be extracted, resulting in low accuracy.

3. Personalized Recommendation Model Based on Improved GRU

3.1. Definition of Attentional GRU (aGRU). With increasing interactions between the user and items, user's interests are constantly changing, and user's preferences are influenced by various kinds of factors such as sequential information, time, and locations in addition to the user-item interaction matrix. It is often assumed that there is a relationship between changes in user's interests and his behavior: the early actions of the user have a low impact on the interest preferences, while the recent actions are more indicative of changes in interests. Therefore, when formulating the user's preferences, the recent behavior is given a higher weight, and items with high similarity to his recent interactive items are recommended for him in preference.

Additionally, items that are highly popular do not necessarily represent a user's real interests in recommendation systems. It is inappropriate to assign a high weight to the behavior just because the item with which the user has recently interacted is one of the popular items. In the history list of behaviors, those that are not very distinguishable from the user's personalized interests or not very predictive of the recommendation list are referred to as outliers. The influence of these outliers should be reduced in sequence recommendations.

3.1.1. Traditional GRU. Recurrent neural networks (RNNs) can capture dynamic information in sequential data by periodically connecting nodes in the hidden layer, and they can store, learn, and represent relevant information in contextual windows with random length. Given the current input x_t and the state h_{t-1}, the probability distribution of the

next element in the output sequence of RNN can be written as

$$h_t = g(Wx_t + Uh_{t-1}). \tag{1}$$

The context of the aGRU includes the input context C_t^u, the transfer context C_T^u, the relevance context C_A^u, and the static interest context C_S^u. The impact of these contexts on the recommendations needs to be considered simultaneously in the transfer process of states in the RNN. Thus, the RNN defined in equation (1) is not sufficient to describe the problem proposed in this article. aGRU is a more complex RNN unit whose new hidden state is a linear interpolation between the previous hidden state and the current candidate knowledge.

$$h_t = z_t \widehat{h}_t + (1 - z_t) h_{t-1}. \tag{2}$$

The candidate knowledge can be formulated as

$$\widehat{h}_t = \tanh(Wx_t + U(r_t \odot h_{t-1})). \tag{3}$$

The vectors of the update gate and the reset gate determine which information can be the output of the GRU. The update gate and the reset gate can be defined respectively as

$$z_t = \sigma(W_z x_t + U_z h_{t-1}). \tag{4}$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1}). \tag{5}$$

3.1.2. aGRU. The GRU described by equations (2)–(5) is not fully suitable for the model formulated in this article, thus the aGRU is proposed and the structure of it is shown in Figure 1.

The components of aGRU are defined as follows:

(1) Candidate Knowledge. The structure of the candidate knowledge is shown in Figure 1(a). And the state determined by the current input and the previous hidden state is called candidate knowledge $\widehat{h}_t^u$, which can be expressed as

$$\widehat{h}_t^u = \sigma\left(p_i I_{C_{It}^u} + h_{t-1}^u T_{C_{Tt}^u}\right), \tag{6}$$

where p_i is a k dimensional embedding vector associated with the item i; C_{It}^u is the various input contexts at the moment t; $I_{C_{It}^u}$ is the embedding matrix associated with C_{It}^u; h_{t-1}^u is the hidden state output at the previous moment; C_{Tt}^u is the various transfer contexts at the moment t; $T_{C_{Tt}^u}$ is the embedding matrix associated with C_{Tt}^u; and σ is the nonlinear activation function. In the proposed aGRU, the candidate knowledge $\widehat{h}_t^u$ can be calculated by equation (6), and the update gate determines the proportion of candidate knowledge that can be output as the hidden state at the moment t.

(2) Update Gate, Reset Gate, and Hidden State. The output structures of the update gate and the hidden state in aGRU are shown in Figure 1(b) and Figure 1(c). The update gate of aGRU is determined by the embedding expression of the item p_i, the hidden state h_{t-1}^u at the previous moment, and the relevance context C_{At}^u, and it can be defined as

$$z_t^u = \sigma(p_i W_k + h_{t-1}^u U_k + C_{At}^u V_k), \tag{7}$$

where W_k, U_k, and V_k are the corresponding weight matrices. C_{At}^u is the relevance context calculated by the attention mechanism at the moment t.

Define the reset gate as: $r_t = I$.

Then the hidden vector h_t^u of the user u at the moment t can be defined as

$$h_t^u = (1 - z_t^u) \cdot \widehat{h}_t^u + z_t^u \cdot h_{t-1}^u, \tag{8}$$

where $\cdot$ represents the element-wise multiplication of the vector. Unlike the common definition of the GRU, the update gate of aGRU depends not only on the current input and the hidden state at the previous moment, but also on the relevance context vector C_{At}^u. When the relevance C_{At}^u between the candidate knowledge $\widehat{h}_t^u$ and the user's static interest context C_S^u is low, h_t^u will mainly originate from h_{t-1}^u. Otherwise, more candidate knowledge $\widehat{h}_t^u$ will be retained in h_t^u.

3.2. Duration Attention Factor. In this article, the sigmoid function is used to normalize the standardized z-score values. The advantages of using the sigmoid function for normalization are as follows:

(1) The sigmoid function can transform the input in the range of $(-\infty, +\infty)$ into the output between $(0, 1)$, meeting the basic requirements of normalization.

(2) The sigmoid function has a larger slope at the position closer to 0, so it is more sensitive to the data closer to the average, while it is relatively insensitive to the data far from the average level. This characteristic has obvious advantages in calculating the duration attention factor.

Thus, after sigmoid normalization, the final expression of the duration attention factor of the user u for the ith item in the browsing sequence can be written as

$$\alpha_{u,i} = \frac{1}{1 + e^{-z_{u,i}}}, \tag{9}$$

where $z_{u,i}$ represents the new browsing duration for the ith item in the browsing sequence of the user u after the z-score transformation.

3.3. The Proposed Recommendation Model. In this section, the duration attention factor is combined with the aGRU to improve the network structure. The structure of the proposed network is shown in Figure 2.

The input data at each moment of the network are the browsing behavior of a certain user at a certain moment. The item information and the browsing duration information about how long a user browses the item can be obtained from the input data at each moment, and the item information is encoded using one-hot encoding method. As the number of items in the dataset in this article is within the

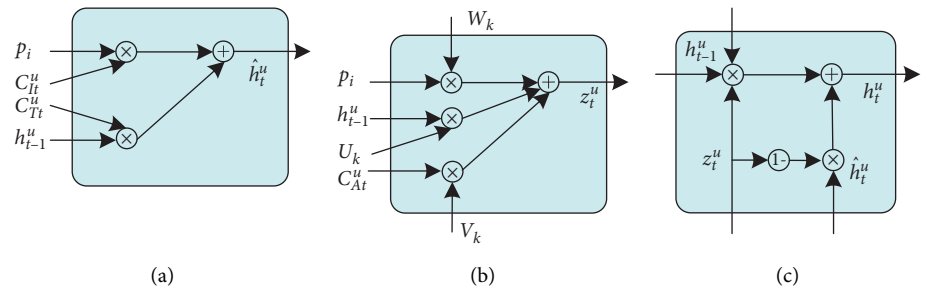

Figure 1: Definition of candidate knowledge, update gate, and hidden state of at the moment t: (a) candidate knowledge, (b) update gate, and (c) the hidden state of at the moment t.

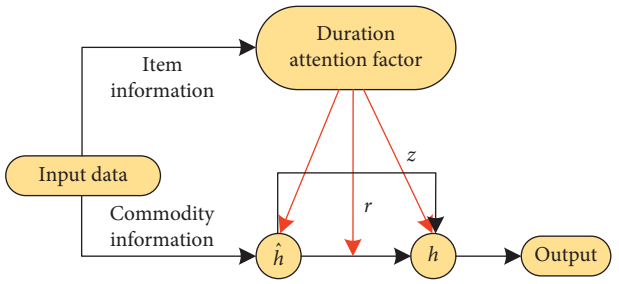

Figure 2: Network structure.

computing capacity of the GPU, and the computational efficiency can meet the requirements of personalized recommendation systems, one-hot coding is used. If there are too many items in the application, embedding operations can be performed in advance to reduce the computation.

After encoding the item information, the browsing duration information in the input data at that moment is processed to obtain the duration attention factor corresponding to the user's behavior, and then the duration attention factor is added to each input in the deep learning model as shown in Figure 2. The duration attention factor corresponding to the input at the moment t is $a_{u,i}$. To maintain consistency with the parameters in the original GRU network, the duration attention factor corresponding to the input data of the network at the moment t is all represented by α_t in the expressions of the update gate z_t, the reset gate r_t, and the candidate hidden state $\hat{h}_t$. In the practical training process of the model in this article, bias is introduced for better performance.

With the addition of the duration attention factor, the update gate z_t, the reset gate r_t, and the candidate hidden state $\hat{h}_t$ of the network at the moment t can be written as

$$\begin{aligned} z_t &= \sigma\left(W_z\left(\alpha_t x_t\right) + U_z h_{t-1}\right), \\ r_t &= \sigma\left(W_r\left(\alpha_t x_t\right) + U_r h_{t-1}\right), \\ \hat{h}_t &= \tanh\left(W\left(\alpha_t x_t\right) + U\left(r_t \cdot h_{t-1}\right)\right), \end{aligned} \tag{10}$$

where x_t is the input data of the network at the moment t. W_z, W_r, and W are the $n \times k$ dimensional weight matrices of the model. U_z, U_r, and U are the weight matrices of the dimension. And the values of each matrix will be updated gradually during the training process, where n denotes the number of neural units in the neural network and k is the number of types of items. By adding the duration attention factor to the expressions of the update gate function, reset gate function, and the candidate hidden state, the duration attention factor is involved in each step of the training process for the neural network, giving full play to the duration information it represents.

The hidden state of the network at the moment t can be calculated from the hidden state of the previous moment, the current candidate hidden state, and the current update gate function. Thus, h_t can be calculated as

$$h_t = (1 - z_t)h_{t-1} + z_t \hat{h}_t, \tag{11}$$

The h_t of the network is calculated in the same way as the h_t of the GRU network, and it controls the extent to which the information from h_{t-1} is brought into the hidden state at the current moment through the update gate z_t. The output of the network at each moment can be calculated from the hidden state h_t and the corresponding weight matrix W_0. In the training process, the expected output of the network at the moment t is the vector of items in the training set that the user actually browses at the moment $t + 1$. The actual output y_t of the network at the moment t is a k dimensional vector, and the jth value represents the prediction score of the jth item, where a larger score indicates a higher probability of the item appearing in the user's browsing data at the next moment. The expression for y_t is

$$y_t = \sigma\left(W_0 h_t\right), \tag{12}$$

where W_0 is the weight matrix used to obtain the predicted y_t based on the hidden state h_t in the network, and the value of W_0 is continuously updated during the training process. The goal of training the network is to make the error between the predicted output and the expected output of the neural network as small as possible, and the model is often optimized by the gradient descent algorithm in neural networks. The duration information about how long each item is browsed by users is analyzed and transformed to obtain the duration attention factor of each recommended item. Meanwhile, the duration attention factor and item information are jointly used as the input for the improved GRU model, which improves the accuracy of prediction.

3.4. Objective Function and Optimization. Using GRU can capture the hidden interest in the user sequence, but it is not known whether the obtained hidden state h_t is an accurate

representation of the user's interest. Although whether a user is interested in a new item is triggered by the final interest, that is the loss function L of the deep interest network monitors the final prediction, the historical hidden state h_t should also give feedback to the model during the training.

The auxiliary loss function is proposed to compensate for the shortcomings of the traditional negative logarithmic likelihood function. It takes the positive sample which is the user's real behavior $e(t+1)$ at the next moment and the negative sample instance $\hat{e}(t+1)$ of the user from negative sampling as the input to the auxiliary network to obtain the auxiliary prediction results, and then a logarithmic loss function is applied to obtain the final auxiliary loss. Assuming that there is a user embedding sequence $\{e_b^i, \hat{e}_b^i\}$ with the length N, where e_b^i represents the user's true click sequence and $\hat{e}_b^i$ represents the user's negative sample sequence from negative sampling; T is the length of the user's interaction sequence; n_E is the size of the embedding layer; e_b^i denotes the tth item embedding vector clicked by the user i; and h_t^i denotes the output of the tth hidden layer of the user i in the GRU network, the auxiliary loss function can be formulated as

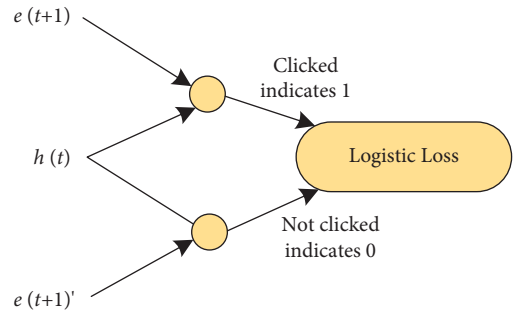

Figure 3: Structure of the auxiliary loss function.

$$L = -\frac{1}{N}\left(\sum_{i=1}^{N}\sum_{t}\log\ \sigma\left(h_t^i, e_b^i[[t+1]\right) + \log\left(1-\sigma\left(h_t^i, \hat{e}_b^i[t+1]\right)\right)\right). \tag{13}$$

In order to combine the auxiliary loss function with the general negative logarithmic likelihood function, a new hyper-parameter α is added to the training to balance the interest representation with the accuracy of the recommendations.

$$L' = L_{\text{target}} + \alpha L. \tag{14}$$

With the auxiliary loss function, the output of the hidden layer in each GRU network can have an impact on the final prediction result, which can more accurately represent the interest state of the user after taking the corresponding behavior, and all interest points $[h_1, h_2, \ldots, h_T]$ jointly form the interest evolution sequence of the first layer of the GRU network. Meanwhile, the introduction of the auxiliary loss function reduces the difficulty of the reverse transmission of the GRU and provides more semantic information for the learning of feature embeddings. The structure of the auxiliary loss function is shown in Figure 3. In addition to choosing a suitable objective function, the dropout method is adopted here to avoid overfitting in the model. In this method, a certain number of neural network units are randomly hidden during the training of the model and only the remaining units are activated for training, which effectively improves the robustness of the model.

4. Experiments and Analysis

4.1. Experimental Environment. Data analysis is performed using tool libraries such as Pandas, Numpy, and Matplotlib, and the deep learning model is constructed using Tensorflow, keras, and scikit-learn. The hardware configuration and software environment are shown in Table 1. In this article, the training of the model and the analysis of data are mainly implemented in Python. Python is an easy-to-read, cross-platform and open-source software with rich library resources that can be called directly, which is widely used in deep learning, crawlers, data analysis, and other fields, and that is the reason why it is chosen to implement the model.

4.2. Dataset and Evaluation Metric. Criteo is a benchmark dataset for CTR prediction, shared by Criteo Labs. It includes users' access data in the advertising display system in 1 week. Criteo provides both a training set and a test set, where the training set consists of partial traffic logs from the advertising site, with each row corresponding to one displayed advertisement. The test set has the same distribution as the training set, but it only contains data from the day after the date of the training set. There are click records of 45 million users in the advertising display system. It contains 26 classification feature fields and 13 numeric feature fields. Label denotes whether the target advertisement item has been clicked or not and it is represented by 0 or 1. L1–L13 are all integers and represent counting features. C1–C26 are classification features and the values of these features are hashed to 32 bits for anonymization.

MovieLens-1M is the rating data of movies from users in movie websites, including basic information about users, attributes of movies, and ratings. As the feature fields of this dataset have semantic meaning, it is used in the experiment for interpretability analysis. The specific numerical statistics of the rating dataset are shown in Table 2.

There are various kinds of evaluation metrics to measure the performance of the recommendation algorithms, including accuracy, metrics based on ranking and weighting,

Table 1: Experimental hardware and software environment configuration.

Parameter	Configuration
Memory	32 GB
Processor (CPU)	Intel® core™ i5-8300H
Video memory	6 GB
Graphics card (GPU)	NVIDIA geforce GTX 1060
Operating system	Win 7
Development environment	anaconda3 jupiter notebook
Tensorflow version	1.14.0 (GPU version)
Python version	3.7.3
Keras version	2.2.4

Table 2: Basic numerical statistics of dataset.

Dataset	Number of samples	Number of characteristic fields	Feature quantity
Criteo	45840617	39	998960
MovieLens-1M	739012	7	3529

coverage, diversity, and novelty. Among them, the prediction accuracy metrics represented by RMSE and MAE are widely used in recommendation algorithms, while the famous Netflix competition and Ali in China also often use them to evaluate the algorithms' accuracy.

The RMSE is calculated as

$$\mathrm{RMSE} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(r_i - \hat{r}_i\right)^2}. \tag{15}$$

The MAE is calculated as

$$\mathrm{MAE} = \frac{1}{m}\sum_{i=q}^{m}\left|r_i - \hat{r}_i\right|, \tag{16}$$

where m is the size of the test set, r_i is the rating from users, and $\hat{r}_i$ is the predicted rating.

4.3. Parameter Selection. To obtain the best results for the model in the dataset, parameters are selected using the Criteo dataset and the MovieLens-1M dataset. In order to determine the appropriate dimension of the hidden state in the aGRU of the model, experiments about the number of iterations and the selection of the learning rate are performed on the model under the same conditions, and the results are shown in Figures 4 and 5. In the training of the deep learning model, the number of iterations per batch and the learning rate will directly affect the degree of convergence of the model. In order to optimize the model, a series of tuning on parameters are performed in the model, and RMSE changes with the number of iterations, which is illustrated in Figure 4. It can be found that the model converges when the number of iterations reaches 240.

In Figure 5, different learning rates are used to verify the effect of the number of iterations on the RMSE. As can be seen from Figure 5, for both Criteo and MovieLens-1M

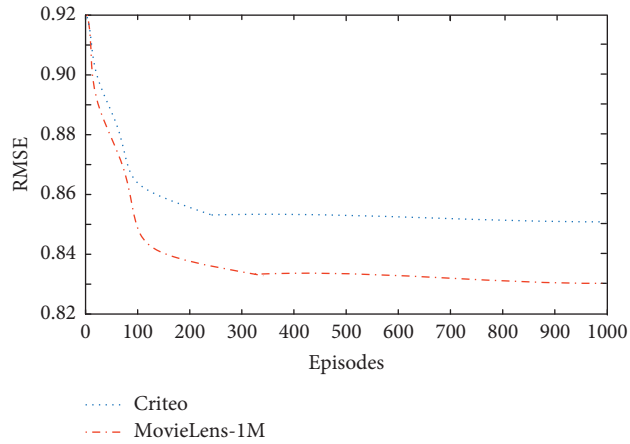

Figure 4: Effects of the number of iterations on RMSE.

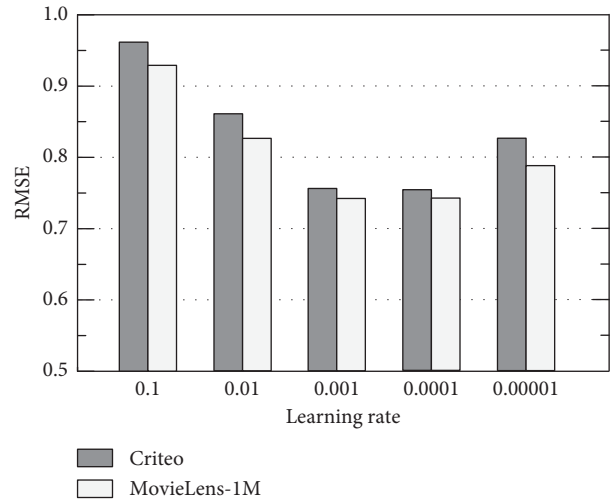

Figure 5: Effects of different learning rates on RMSE.

datasets, convergence is obtained when the learning rate is 0.0001, and the value of error is minimized. Thus, the model reaches the optimal condition.

4.4. Comparison with Other Methods. To demonstrate the advantages of the proposed personalized recommendation method, it is compared with the methods in Ref. [28] and Ref. [29] under the same experimental conditions. RMSE and MAE are used to measure the effectiveness in recommendation of each model, and the results are shown in Tables 3 and 4. As shown in Table 3, the RMSE of the proposed method is 0.7257 and the MAE is 0.5147 under the Criteo dataset, both of which are lower than those of the comparison methods. For the MovieLens-1M dataset, which has a sparse data volume, the proposed method also achieves better recommendation performance. As depicted in Table 4, the RMSE of the proposed method in the MovieLens-1M dataset is 0.7869 and the MAE is 0.5893, both of which are lower than those of the compared personalized recommendation methods. This is because the proposed method

Table 3: Comparison of different methods under the Criteo dataset.

Method	RMSE	MAE
Ref. [28]	0.8167	0.6349
Ref. [29]	0.8138	0.6274
Proposed method	0.7257	0.5147

Table 4: Comparison of different methods under the MovieLens-1M dataset.

Method	RMSE	MAE
Ref. [28]	0.8630	0.6741
Ref. [29]	0.8454	0.6680
Proposed method	0.7869	0.5893

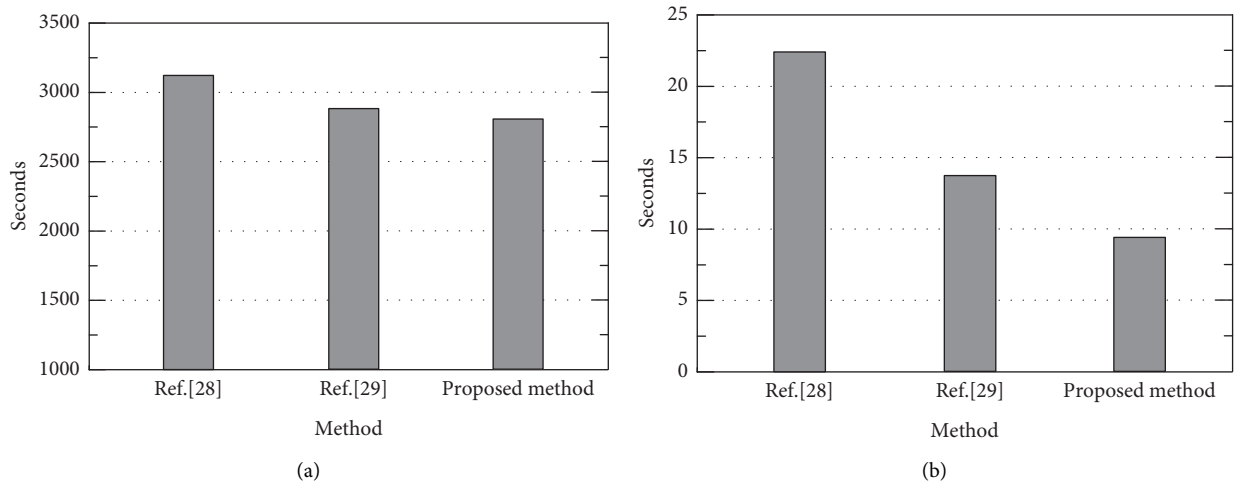

Figure 6: Comparison of time consumption of different methods in the Criteo and MovieLens-1M datasets: (a) Criteo and (b) MovieLens-1M.

analyses the duration information about how long users browse each item, transforms it to obtain the duration attention factor corresponding to each recommended item, and uses the duration attention factor and the item information as the input of the proposed model, so as to improve the accuracy of prediction. Although the comparison methods take into account the influence of both contextual and auxiliary information, they do not consider the effect of the duration information of each item, and therefore the recommendation performance is less effective.

Also, the time consumed to run each model is analyzed in this article, as shown in Figure 6. For the Criteo dataset, the feature extraction of methods proposed in Ref. [28] and Ref. [29] takes much time. They increase the number of parameters that need to be trained in each round, so they take a longer time in large datasets. The proposed model can extract the data features faster and takes less time. Even in the MovieLens-1M dataset which is with less data, the proposed model still takes less time than the methods proposed in Ref. [28] and Ref. [29]. This proves that the proposed method improves the efficiency of personalized recommendation in a big data environment.

5. Conclusion

A personalized recommendation model based on an improved GRU network is proposed to address the problem of diverse user preferences and dynamic changes of interests in the personalized recommendation scenario in the big data environment. The duration information about how long users browse each item is analyzed and transformed to obtain the duration attention factor corresponding to each item, which is used together with the item information as the input of the proposed model for training and prediction of the model. A hyper-parameter is introduced to combine the auxiliary loss function with the negative logarithmic likelihood function to enhance the relationship between interest representation and recommendation accuracy. The

experimental results show that the proposed method improves the capability of the system in personalized recommendation.

In the future, the research will consider how to integrate heterogeneous information sources into interpretable recommender systems. In addition, the research in this article is based on existing static datasets, whereas recommendation systems are engineered with dynamic user data, and how to capture these features will be the focus of the future research.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Acknowledgments

This work was supported by the fund project of Shanxi Provincial Education Department (no. J2020392).

References

[1] D. Smolyakov, A. Korotin, P. Erofeev, P. Artem, and B. Evgeny, "Meta-learning for Resampling Recommendation systems," in *Proceedings of the Eleventh International Conference on Machine Vision (ICMV 2018)*, pp. 472–484, SPIE, Munich, Germany, March 2019.

[2] M. Robillard, R. Walker, and T. Zimmermann, "Recommendation systems for software engineering," *IEEE software*, vol. 27, no. 4, pp. 80–86, 2009.

[3] A. Felfernig, M. Jeran, G. Ninaus, F. Reinfrank, S. Reiterer, and M. Stettinger, "Basic approaches in recommendation systems," *Recommendation Systems in Software Engineering*, Springer, Berlin, Heidelberg, pp. 15–37, 2014.

[4] A. H. N. Rafsanjani, N. Salim, and A. R. Aghdam, "Recommendation systems: a review," *International Journal of Computational Engineering Research*, vol. 3, no. 5, pp. 47–52, 2013.

[5] S. S. Khanal, P. W. C. Prasad, A. Alsadoon, and A. Maag, "A systematic review: machine learning based recommendation systems for e-learning," *Education and Information Technologies*, vol. 25, no. 4, pp. 2635–2664, 2020.

[6] M. K. Khribi, M. Jemni, and O. Nasraoui, "Automatic personalization in E-learning based on recommendation systems," *Intelligent and Adaptive Learning Systems*, vol. 3, no. 6, pp. 19–33, 2012.

[7] P. Thiengburanathum, S. Cang, and H. Yu, "Overview of Personalized Travel Recommendation systems," in *Proceedings of the 2016 22nd International Conference on Automation and Computing (ICAC)*, pp. 415–422, IEEE, Colchester, UK, September 2016.

[8] N. E. Hernández, D. C. A. Bartolomé, S. P. Chamoso, F. D. L. P. Pintado, and J. M. C. Rodríguez, "A Machine Learning Platform for Stock Investment Recommendation systems," in *Proceedings of the International Symposium on Distributed Computing and Artificial Intelligence*, pp. 303–313, Springer, Cham Germany, August 2020.

[9] A. C. Rivera, M. Tapia-Leon, and S. Lujan-Mora, "Recommendation systems in education: a systematic mapping study," in *Proceedings of the International Conference on Information Technology & Systems (ICITS 2018)*, pp. 937–947, Springer, Cham Germany, January 2018.

[10] F. Jia and Y. Shi, "Library Management System Based on Recommendation system," in *Proceedings of the International Conference on Information Computing and Applications*, pp. 488–497, Springer, Berlin, Heidelberg, 2013.

[11] H. Ko, S. Lee, Y. Park, and A. Choi, "A survey of recommendation systems: recommendation models, techniques, and application fields," *Electronics*, vol. 11, no. 1, pp. 141–147, 2022.

[12] F. Yang, "A hybrid recommendation algorithm-based intelligent business recommendation system," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 21, no. 6, pp. 1317–1322, 2018.

[13] B. Xiang, Z. Zhang, H. Dong, Q. Wu, and L. Hu, "Research of mobile Recommendation System Based on Hybrid Recommendation technology," in *Proceedings of the 2013 3rd International Conference on Consumer Electronics, Communications and Networks*, pp. 508–512, IEEE, Xianning, China, November 2013.

[14] A. Werner-Stark and Z. Nagy, "A Heuristic Method to Recommendation Systems," in *Proceedings of the 2020 6th International Conference on Information Management (ICIM)*, pp. 200–204, IEEE, London, UK, March 2020.

[15] N. Kurmashov, K. Latuta, and A. Nussipbekov, "Online Book Recommendation system," in *Proceedings of the 2015 Twelve International Conference on Electronics Computer and Computation (ICECCO)*, pp. 1–4, IEEE, Madurai, India, December 2015.

[16] J. K. Kim, H. K. Kim, H. Y. Oh, and Y. U. Ryu, "A group recommendation system for online communities," *International Journal of Information Management*, vol. 30, no. 3, pp. 212–219, 2010.

[17] S. S. Anand and B. Mobasher, *Contextual recommendation [C]//Workshop on Web Mining*, pp. 142–160, Springer, Berlin, Heidelberg, 2006.

[18] R. Baral, S. S. Iyengar, X. Zhu, T. Li, and P. Sniatala, "HiRecS: a hierarchical contextual location recommendation system," *IEEE Transactions on Computational Social Systems*, vol. 6, no. 5, pp. 1020–1037, 2019.

[19] B. Bhattacharya, I. Burhanuddin, A. Sancheti, and K. Satya, "Intent-aware Contextual Recommendation system," in *Proceedings of the 2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 1–8, IEEE, New Orleans, LA, USA, 2017.

[20] M. Aharon, E. Hillel, A. Kagian, R. Lempel, H. Makabee, and R. Nissim, "Watch-it-next: a contextual TV recommendation system," in *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 180–195, Springer, Cham Germany, August 2015.

[21] P. Covington, J. Adams, and E. Sargin, "Deep Neural Networks for Youtube recommendations," in *Proceedings of the 10th ACM Conference on Recommender Systems*, pp. 191–198, New York, NY, USA, September 2016.

[22] H. T. Cheng, L. Koc, J. Harmsen, S. Tal, and C. Tushar, A. Hrishi, A. Glen, C. Greg et al., Wide & Deep Learning for Recommender systems," in *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, pp. 7–10, Jun 2016.

[23] H. T. H. Nguyen, M. Wistuba, J. Grabocka, L. R. Drumond, and L. Schmidt-Thieme, "Personalized Deep Learning for Tag

recommendation," in *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 186–197, Springer, Cham Germany, April 2017.

[24] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "DeepFM: a factorization-machine based neural network for CTR prediction," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, vol. 9, no. 3, pp. 111–121, ACM, Huawei, China, August 2017.

[25] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T. S. Chua, "Neural Collaborative filtering," in *Proceedings of the 26th International Conference on World Wide Web*, pp. 173–182, August 2017.

[26] Y. Feng, F. Lv, W. Shen et al., "Deep session interest network for click-through rate prediction," vol. 4, no. 1, pp. 23–32, 2019, https://arxiv.org/abs/1905.06482.

[27] Q. Liu, S. Wu, L. Wang, and T. Tan, "Predicting the next location: a recurrent model with spatial and temporal contexts," in *Proceedings of the Thirtieth AAAI conference on artificial intelligence*, pp. 112–121, aaai press, Huawei, China, 2016.

[28] J. Yin, Y. Li, Z. Liu, J. Xu, B. Xia, and Q. Li, "ADPR: An Attention-Based Deep Learning point of Interest Recommendation Framework," in *Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, IEEE, Budapest, Hungary, July 2019.

[29] H. Zhang, S. Wei, X. Hu, Y. Li, and J. Xu, "On accurate POI recommendation via transfer learning," *Distributed and Parallel Databases*, vol. 38, no. 3, pp. 585–599, 2020.