Applied Science and Innovative Research
ISSN 2474-4972 (Print) ISSN 2474-4980 (Online)
Vol. 10, No. 2, 2026
www.scholink.org/ojs/index.php/asir

Original Paper

Research on Book Borrowing Prediction Model Based on Transformer

Baiyu Chen[1*]

[1] Xihua University, Chengdu Sichuan Province, 610039, China

[*] Chen Baiyu (1994-), Male, Han ethnicity, master's degree candidate, Assistant Librarian at the Library of Yibin Campus, Xihua University, Research direction: Library and Information Science.

Received: April 28, 2026 Accepted: May 18, 2026 Online Published: May 25, 2026

doi:10.22158/asir.v10n2p15 URL: http://dx.doi.org/10.22158/asir.v10n2p15

Abstract

With the increasing demand for intelligent library services, accurately predicting user borrowing behavior has become a key issue for optimizing resource allocation and improving service quality. Traditional prediction methods have significant shortcomings in long sequence modeling, feature interaction, and cold start scenarios. This paper proposes a book borrowing prediction model based on the Transformer architecture. It dynamically captures the long-term dependency of user behavior through the self-attention mechanism and combines book metadata and temporal context features to achieve multi-dimensional information fusion. Experiments show that it significantly outperforms the baseline model. The study further explores the application potential of the model in dynamic inventory management, personalized recommendation, and other scenarios, providing theoretical support and technical solutions for the intelligent transformation of libraries.

Keywords

Transformer model, Book lending prediction, Model design

1. Introduction

As a core carrier of public cultural services, libraries are confronted with challenges including uneven resource utilization and dynamically changing user demands. Traditional book borrowing prediction methods mainly rely on statistical analysis and simple machine learning models such as linear regression and collaborative filtering, which suffer from the following limitations: insufficient sequential modeling capability—user borrowing behaviors are time-dependent, yet models like RNN and LSTM fail to capture long-term patterns[1]; inadequate mining of feature interactions—the complex relationships among user attributes, book metadata (category, author, publication year) and temporal

context cannot be effectively modeled; cold-start problem—new users or new books cannot be accurately predicted due to the lack of historical data. With the structures of Self-Attention and Multi-Head Attention, the Transformer model demonstrates powerful capabilities in modeling long sequences and feature interactions in the field of natural language processing[2-8]. This paper introduces Transformer into the task of book borrowing prediction, and proposes an end-to-end prediction framework integrating user behavior sequences, book attributes and temporal context, so as to provide new insights for the intelligent management of libraries.

2. Related Work

2.1 Traditional Book Borrowing Prediction Methods

Early studies were mostly based on time-series analysis (e.g., ARIMA) and collaborative filtering algorithms, which mine similarities through user-book rating matrices[9]. Nevertheless, such methods ignore temporal dynamics and fail to handle sparse data.

2.2 Application of Deep Learning in Borrowing Prediction

In recent years, LSTM and GRU have been adopted to model user borrowing sequences. However, their unidirectional recursive structure restricts parallel computing capability, and long-distance dependencies tend to decay. Graph Neural Networks (GNNs) attempt to model interactions via user-book relational graphs, yet they lack sufficient integration of temporal features[10-13].

2.3 Application of Transformer in Recommendation Systems

Transformer has been successfully applied to video recommendation (e.g., the SASRec model) and e-commerce scenarios. Its core advantage lies in dynamically weighting the importance of different time steps through self-attention[14]. On this basis, this paper further integrates book metadata and temporal features to address the specific requirements of library scenarios.

3. Transformer Time-Series Model

In 2017, the paper Attention is All You Need first introduced the Transformer model, proposed by Ashish Vaswani et al.[13]. As a researcher at Google, Ashish Vaswani is the first author of this paper and one of the core inventors of the Transformer model. He has made remarkable contributions to natural language processing and machine learning. The emergence of the Transformer model not only promotes the advancement of NLP technologies, but also offers new tools and research ideas for subsequent scholars and developers, significantly broadening the application scope of artificial intelligence.

The core of the Transformer model is composed of multiple Encoder units and Decoder units. Specifically, the model consists of several key components: multi-head attention module, feed-forward neural network layer, as well as residual connection and layer normalization module. A residual connection and normalization module is arranged between adjacent modules. Residual connections prevent gradient vanishing or exploding during training, whereas normalization accelerates the

convergence of model training. Moreover, an attention mechanism layer is embedded between the encoder and decoder to facilitate their interaction. This section elaborates on major components of the Transformer model, including sequence encoding, multi-head self-attention, feed-forward neural networks, and normalization.

3.1 Sequence Encoding

As shown in the corresponding figure, after the embedding encoding and positional encoding of input sequence data are combined by summation, the input of each encoder is the output of the preceding encoder.

It should be noted that data in the Transformer model does not possess inherent sequential dependencies, which is fundamentally different from the iterative training mechanism of traditional recurrent neural networks. Accordingly, to effectively integrate positional information into input sequences, the Transformer model ingeniously adopts sine and cosine functions with multiple frequencies. The formula for positional encoding is presented in Equation(1).

$$PE(t,2i)=\sin(\frac{t}{10000^{2i/d_{\text{model}}}}),PE(t,2i+1)=\cos(\frac{t}{10000^{2i/d_{\text{model}}}}), \quad (1)$$

In this formula, t denotes the time step, i refers to the dimension index of input features, and model represents the total dimension of input features. Users can customize the number of encoders and encoding length according to training requirements.

3.2 Attention Mechanism

3.2.1 Overview of Attention Mechanism

The attention mechanism is a technique simulating human visual attention, which enables selective focus on critical information while ignoring irrelevant content during information processing. Derived from human visual characteristics, humans selectively filter out unimportant information, especially when processing massive amounts of data.

In cognitive science, constrained by information-processing capacity, humans must efficiently utilize visual information-processing capabilities by focusing intensively on specific elements within the visual field. For instance, when reading a book, people usually only attend to and process a small number of words to be interpreted. The core functions of the attention system are twofold: first, determining which parts of input data should be focused on; second, allocating scarce information-processing resources to key areas.

In deep learning, the attention mechanism is widely applied in various fields including natural language processing, statistical modeling, image analysis, and speech recognition. It effectively optimizes information-processing workflows, achieves rational resource allocation, and thereby improves model performance and efficiency. Taking natural language processing as an example, it enables models to deeply interpret and translate complex long sentences, boosting prediction accuracy; in image detection, it helps models capture key image elements precisely and enhance detection performance.

There are multiple implementations of the attention mechanism. One is saliency-based attention, which

selects target areas according to input information saliency. Another is top-down voluntary attention, also known as focused attention, which selects focused regions based on task objectives and model self-judgment.

3.2.2 Calculation Process of Attention Mechanism

Calculating the attention mechanism involves three steps: similarity computation, attention weight calculation, and weighted summation. The detailed workflow is as follows:

1.Similarity Calculation: Compute the row-wise similarity between all key vectors *Ki* and the query vector *Q*, so as to obtain the similarity score *Si* of each key vector. The calculation formula is as follows:

$$s_i = similarity(Q, K_i) \tag{2}$$

It should be noted that the function denoted by similarity can be additive similarity, dot-product similarity, scaled dot-product similarity, etc.

2.Calculation of Attention Weights

Furthermore, the Softmax function is adopted to normalize the similarity score s_i obtained in the above process, so as to convert it into the attention weight α_i. The calculation formula is as follows:

$$\alpha_i = \frac{\exp(s_i)}{\sum_j exp(s_j)} \tag{3}$$

3.Weighted Summation

Finally, the obtained attention weight α_i is multiplied with the corresponding value vector V_i and summed up to derive the weighted vector *C*. The calculation formula is as follows:

$$C = \sum_i \alpha_i V_i \tag{4}$$

At present, due to its efficient implementation (e.g., matrix multiplication), the attention mechanism is widely applied in deep learning tasks such as speech and audio processing, natural language processing, and computer vision.

3.3 Feed-Forward Neural Network

The Feed-Forward Neural Network (FFN) plays a critical role in realizing nonlinear transformation within the Transformer model. The output of each Multi-Head Attention module serves as the input of the FFN module, which relies on FFN to enhance nonlinear expressive capability. Meanwhile, parameter updates of FFN modules at different layers are independent during training, so as to adapt to feature-learning requirements of respective layers. The calculation formula is given as follows:

$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2, \tag{5}$$

In this formula, W_1, W_2, b_1 and b_2 denote learnable parameters of the model, whose dimensions are determined by the dimension of input data and the user-defined output dimension.

3.4 Residual Connection and Normalization Layer

During the training process of the Transformer model, each sub-layer is followed by a residual connection and normalization module, which play a vital role in improving training stability and model performance. Residual connections preserve the original input via skip connections, alleviating gradient vanishing and preventing weight degradation. The calculation formula is shown in Equation (6).

$$X = X_{\text{input}} + \text{SelfAttention}(Q, K, V). \tag{6}$$

Normalization normalizes elements in the output matrix according to the mean and variance of data in their respective columns, so that each hidden variable follows the standard normal distribution. This process enables faster and more stable convergence of the model, with the calculation formula shown in Equation (7).

$$\begin{aligned} &\mu_j = \frac{1}{m}\sum_{i=1}^{m} x_{ij}, \sigma_j^2 = \frac{1}{m}\sum_{i=1}^{m}(x_{ij} - \mu_j)^2, \\ &\text{LayerNorm}(x_{ij}) = \frac{x_{ij} - \mu_j}{\sqrt{\sigma_j^2 + \varepsilon}}, \end{aligned} \tag{7}$$

It should be noted that μ_j denotes the sample mean and σ_j^2 denotes the sample variance. To prevent the variance from being zero, a regularization factor ($\varepsilon>0$) is introduced into the formula.

4. Experimental Analysis

4.1 Source of Training Dataset

The dataset designed in this paper is collected from access data of the library access-control system, which is stored locally directly and preprocessed to form the final dataset.

4.2 Dataset Preprocessing

Before feeding data into machine learning models, an essential procedure known as data preprocessing is required, which includes data cleaning, transformation and arrangement. This step is critical for improving the accuracy and efficiency of prediction models.

4.3 Data Cleaning

Data cleaning is a key step in data preprocessing. It aims to correct anomalous, missing and inconsistent data in the dataset, so as to improve data quality and guarantee the accuracy of subsequent analysis and modeling. Differences among datasets lead to varied cleaning processes and procedures; thus, there is no universal strategy for data cleaning.

4.3.1 Removal of Duplicate and Irrelevant Records

Generally, records to be removed from the dataset are mainly duplicate or irrelevant observations. When merging datasets from multiple forms or sources during data collection, duplicate records or those inconsistent with research objectives may be generated, which should be primarily eliminated.

4.3.2 Imputation of Missing Values

Missing values are occasionally encountered in data processing. Such values are difficult for algorithms

to accept during model training and may cause data distortion in visualization. Therefore, observations containing missing values are usually deleted, yet this method may result in information loss and even reduce dataset validity. In addition, the mean value of two adjacent data points can be calculated to impute missing values, which still carries risks of data integrity loss.

4.3.3 Screening and Removal of Outliers

One or more mismatched outliers may occur during data analysis, which can be caused by multiple factors. Comprehensive analysis of their generation mechanisms is required before screening, since the existence of outliers does not necessarily indicate erroneous results. If outliers are verified as input errors or irrelevant to the dataset, they can be directly removed. If their validity is confirmed, the underlying analysis principle is proven correct.

4.4 Dataset Splitting

Dataset splitting is a vital step in machine learning, which divides the original dataset into multiple subsets for model training, validation and testing respectively. Two core principles must be followed: first, each subset should retain representative features of the original dataset to ensure the model learns the true data distribution; second, a reasonable splitting ratio should be selected according to dataset size, complexity and specific model requirements.

Typically, the training set accounts for the largest proportion, as sufficient data is required for the model to learn and optimize internal parameters. The validation set is relatively small-scaled, mainly used to tune hyperparameters for optimal model performance during training. The test set takes the smallest proportion, serving to evaluate the model's generalization ability on unseen data, i.e., its prediction capability for unknown samples.

In this paper, a dataset splitting strategy is proposed by comprehensively considering algorithm complexity and dataset representativeness. Specifically, the first 80% of the dataset is selected as the training set for model training, while the remaining 20% is taken as the test set to evaluate the model's performance in temporal correlation prediction and prediction accuracy. With this strategy, we intend to verify the practical performance of the model and further improve its prediction accuracy.

4.5 Prediction Service

The dataset is utilized for model training.

The training set data is fed into the encoder module, which performs linear transformation and positional encoding on the training data. The calculation formula for the combined output of linear transformation and positional encoding is as follows:

$$X_{IE} = (X_{IN} \cdot W_0 + b_0) + P \tag{8}$$

In this formula, $X_{IN} \in i^{T \times d}$ denotes a matrix with T rows and d columns; X_{IE} refers to the result obtained by combining linear transformation and positional encoding, which takes the form of $X_{IE} \in i^{T \times k}$, namely a matrix with T rows and k columns. $W_0 \in i^{d \times k}$ represents the linear transformation matrix in the first-round training across all time periods, with a dimension of d rows and

k columns; $b_0 \in i^k$ denotes the bias term of the first stage. $P \in i^{T\times k}$ is defined as the positional encoding matrix with *T* rows and *k* columns. The initial values of *P* and W_0 are generated by a random number generator. Subsequently, the results of linear transformation and positional encoding undergo fully-connected processing and normalization, and are then fed into the attention module. The normalized output data X_{FF} after full connection is presented as follows:

$$X_{FF} = Norm(X_{IE} + ReLU(X_{IE} \cdot W_1 + b_1) \cdot W_1 + b_2) \tag{9}$$

According to Formula (6.2), $W_1 \in i^{d\times k}$ is the secondary-training linear transformation matrix covering all time periods, consisting of *d* rows and *k* elements. b_1 and b_2 are two vectors in i^k representing the preset values of the second and third stages respectively. ReLU (Rectified Linear Unit) appearing on the right-hand side of this equation is a common nonlinear activation function, whose definition is given as follows:

$$ReLU(x) = \begin{cases} 0, if\ (x \le 0) \\ x, if\ (x > 0) \end{cases} \tag{10}$$

X_{FF} generates three input matrices *K, V* and *Q* for the attention module, where $\mathrm{K} \in i^{T\times k}$, $\mathrm{V} \in i^{T\times k}$, and $\mathrm{Q} \in i^{T\times k}$ are all matrices with *T* rows and *k* columns. Subsequently, the Softmax activation function is applied to the three input matrices *K, V* and *Q* to obtain the output matrix *C* of the attention module, where $\mathrm{C} \in i^{T\times k}$ is also a matrix with *T* rows and *k* columns. The calculation formula for the output matrix *C* of the attention module is as follows:

$$C = Attention(K, Q, V) = softmax\left(\frac{Q \cdot K^T}{\sqrt{h}}\right) \cdot V \tag{11}$$

Softmax is an activation function specifically applied to multi-class classification tasks. It converts a series of unordered real values into a probability distribution where each value ranges between 0 and 1 and the sum of all values equals 1. In other words, for a real-valued vector $x=(x_1,x_2,\ldots,x_n)$ containing *n* elements, the expression of the Softmax function is shown as follows.

$$\mathrm{softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} (i = 1, 2, \ldots, n) \tag{12}$$

In this expression, *exp(x)* denotes the exponential function e^x, *i* refers to the *i-th* element of the output, and *j* corresponds to the *j-th* component of the input. Therefore, the Softmax function converts each component of the input vector *x* into a set of real values between 0 and 1 whose sum equals 1. These real values can also be used to describe the probability ratio of *x* belonging to different categories.

In the formula, *h* represents the number of multi-head attention heads adopted, and 8 attention heads are designed in this study. Subsequently, the output matrix *c* of the attention module is normalized, and the predicted value *e* of the fully-connected layer is output through the ReLU activation function to obtain environmental information of the next time period.

4.6 Prediction Evaluation Metrics

The prediction performance is shown in Table 1.

Table 1. Prediction Trend Data

Prediction	1	2	3	4	5	6
Accuracy	90%	91%	89%	90%	91%	90%

References

[1] Xia, X., Yan, E. L., & Li, X. W. (2024). A Review on the Application of Transformer in Time Series Forecasting. *Information Technology and Informatization*, *2024*(3), 124-128.

[2] Yule, G. U. (1927). On a method of investigating periodicities in distributed series, with special reference to Wolfer's sunspot numbers. *Philosophical Transactions of the Royal Society of London Series A*, *226*, 267-298.

[3] Kim, K. (2003). Financial time series forecasting using support vector machines. *Neurocom-puting*, *55*(1-2), 307-319.

[4] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735-1780.

[5] Heimes, F. O. (2008). Recurrent neural networks for remaining useful life estimation, in: 2008 international conference on prognostics and health management. *IEEE, 2008*, 1-6.

[6] Huang, N. E., & Shen, S. (2005). The Hilbert-Huang transform and its applications. *World Scientific*.

[7] Wu, Y. X., & Wen, X. (2016). Short-term Prediction of Book Borrowing Volume Based on ARIMA Model. *Statistics & Decision*, *2016*(23), 83-86. https://doi.org/10.13546/j.cnki.tjyjc.2016.23.051

[8] Box, G. E. P., & Jenkins, G. M. (1970). Time series analysis forecasting and control. *Journal of Time Series Analysis*, *3*(3228).

[9] Li, H. (2012). *Statistical Learning Methods*. Beijing: Tsinghua University Press.

[10] Liu, M. (2011). Research on Sigmoid Kernel Function in Support Vector Machine. Xidian University.

[11] Guan, J. (2024). *Research on Book Borrowing Volume Prediction Method Based on RNN-CNN Model*. Nanjing University of Information Science and Technology. https://doi.org/10.27248/d.cnki.gnjqc.2023.000448

[12] Chen, Y. Y., Zhao, B. K., Wang, H., et al. (2024). Prediction of Time-Series InSAR Surface Deformation Based on LSTM Model. *Yangtze River*, *55*(3), 146-152. https://doi.org/10.16232/j.cnki.1001-4179.2024.03.020

[13] Vaswani, A., Shazeer, N., Parmar, N., et al. (2017). Attention is all you need. Advances in neural information processing systems, 30.

[14] Tian, A. B., Wei, J. J., & Xiao, J. B. (2024). Research on Network Traffic Prediction Based on Transformer. *Information Technology*, *2024*(4), 156-160. https://doi.org/10.13274/j.cnki.hdzj.2024.04.025