Journal of Computer Technology and Software

ISSN:2998-2383

Vol. 3, No. 7, 2024

Time-Series Load Prediction for Cloud Resource Allocation Using Recurrent Neural Networks

Yun Zi

Georgia Institute of Technology, Atlanta, USA

yzi9@gatech.edu

Abstract: This study proposes a container scheduling optimization system based on recurrent neural network (RNN) to improve the efficiency of cloud computing platform in resource management. By using the RNN model to analyze and predict historical load data, this system can accurately estimate future resource requirements and dynamically adjust the container scheduling strategy to achieve efficient resource allocation and utilization. Experimental results show that compared with other comparison models such as support vector regression (SVR), decision tree regression (DTR), multi-layer perceptron (MLP) and long short-term memory network (LSTM), the RNN model performs best in load prediction tasks, with high prediction accuracy and good generalization ability. This system can not only effectively reduce resource waste, but also improve server utilization, providing a reliable solution for container management in actual production environments. Future research can introduce graph neural network (GNN) or self-attention mechanism on the existing basis to further improve the prediction performance of the system. In addition, expanding cross-platform container scheduling optimization methods and creating a more intelligent resource management system will also be an important research direction in the future.

Keywords: **Container scheduling, recurrent neural network, load prediction, cloud computing**

1. Introduction

Cloud computing has experienced substantial and far-reaching adoption across a diverse range of sectors, including healthcare, finance, and high-performance computing for deep learning. In healthcare [1-3], cloud computing enables real-time data sharing, enhances telemedicine services, and supports large-scale medical research through efficient data processing and storage solutions. The financial sector benefits from cloud platforms that enhance data security, provide scalable resources for fraud detection and support complex financial modeling [4-6]. In deep learning and artificial intelligence, cloud computing offers the high computational power necessary for model training, allowing for accelerated processing speeds and scalable resource management, which are critical for advancements in large language models and AI research [7-10]. With the rapid development of cloud computing, as enterprise needs to grow and diversify, how to efficiently manage and schedule container resources has become a key issue. Although traditional container scheduling methods have been able to achieve reasonable resource allocation to a certain extent, it is often difficult to make accurate predictions when facing complex and dynamically changing loads, resulting in uneven resource allocation, low server utilization, and even resource waste. With the rapid development of deep learning technology [11], applying deep learning to container scheduling optimization, predicting future resource requirements by analyzing historical load data and resource usage patterns, and dynamically optimizing container deployment strategies has become an important means to improve the performance of cloud computing platforms.

Based on this, this paper proposes a container scheduling optimization system based on deep learning, which analyzes historical data and predicts future resource requirements through a deep learning model to achieve efficient container scheduling. This system uses the RNN time series prediction model to accurately predict the peak and trough periods of workloads, automatically schedule container deployment at the right time, and dynamically adjust resource configuration to cope with sudden traffic [12]. Unlike traditional static scheduling methods, this system has the ability of real-time learning and adaptive adjustment. It can not only respond quickly to fluctuations in resource demand, but also reduce latency and effectively avoid excessive or insufficient use of resources, thereby maximizing server utilization and minimizing resource waste. In addition, the introduction of deep learning models enables the scheduling system to deeply explore and analyze the implicit relationship of load data, thereby optimizing scheduling decisions [13]. For example, by identifying periodic changes and sudden peaks in the load, the system can reserve resources in advance or unload some tasks

to achieve a balance of resource demand. The system can continuously self-optimize through deep learning algorithms, so that the scheduling strategy can be adjusted over time, realizing the transition from passive scheduling to active scheduling. This scheduling system can not only meet peak demand, but also reduce redundant configuration of resources when the load is low, thereby improving the responsiveness and resource utilization efficiency of the entire cloud platform.

The optimization algorithm also has broad adaptability and can be extended to a variety of cloud computing scenarios, covering the scheduling needs of private clouds, public clouds, and complex hybrid cloud environments. Whether it is a private cloud deployment within an enterprise, a public cloud platform for the public, or a hybrid cloud architecture that combines the advantages of both, the container scheduling optimization system based on deep learning can respond flexibly to provide diversified cloud computing applications. Scenarios provide efficient resource scheduling support. By deploying deep learning models and combining them with dynamic adjustment strategies, the system can adjust scheduling decisions in real time when resource demand fluctuates greatly, helping cloud computing service providers quickly adapt to changes in different businesses, allocate resources reasonably, and balance loads to ensure system security. reliability and operational efficiency. In addition, the algorithm's refined resource management capabilities also bring significant operating cost reductions. By accurately controlling and dynamically adjusting resources, the container scheduling system can avoid over-allocation and idle waste of resources, which not only effectively reduces operation and maintenance costs, but also improves resource utilization. This resource optimization strategy can greatly improve user experience and bring the response speed and stability of cloud services to a higher level. At the same time, the improvement of service quality and system competitiveness has enabled the cloud computing platform to occupy a more favorable position in the fierce market competition and meet users' needs for high-performance and high-stability services. It is foreseeable that with the further application of this system on cloud computing platforms, cloud computing service providers will be able to significantly improve their market competitiveness while providing better services, injecting new vitality into the healthy development of the cloud computing industry.

The container scheduling optimization system based on deep learning not only significantly improves the resource management efficiency of the cloud computing platform, but can also adapt to increasingly complex and changing user needs, bringing greater adaptability and scalability to the cloud computing system [14]. Through this system, the cloud platform can realize flexible scheduling of resources during peak load periods, avoid resource bottlenecks, and reduce the configuration of redundant resources during low load periods, thus greatly improving resource utilization. With the continuous development and innovation of deep learning technology, this system not only shows strong performance advantages in the current environment but is also expected to become an indispensable core module in cloud computing platforms in the future. In the future, the system can further optimize the scheduling strategy and combine it with an adaptive learning mechanism, so that it can still make efficient and intelligent resource allocation decisions when facing more diverse application scenarios and higher real-time requirements. It is foreseeable that this intelligent container management method will provide strong support for the intelligent development of cloud computing, promote the development of cloud computing platforms in a more efficient and intelligent direction, and meet the stringent resource management requirements of various enterprises and applications [15-18]. Requirements to lay a solid foundation for promoting the innovative development of cloud computing.

2. Method

The method in this paper mainly designs a load prediction model for container scheduling based on Recurrent Neural Network (RNN) to improve the resource utilization efficiency of cloud computing platforms. RNN is a deep learning model suitable for processing time series data. It can capture contextual information in sequence data and is particularly suitable for tasks such as load prediction that require consideration of historical data [19]. In this system, we use RNN to model historical load data and predict future resource requirements, thereby achieving dynamic and reasonable container scheduling. The overall network structure is shown in Figure 1.

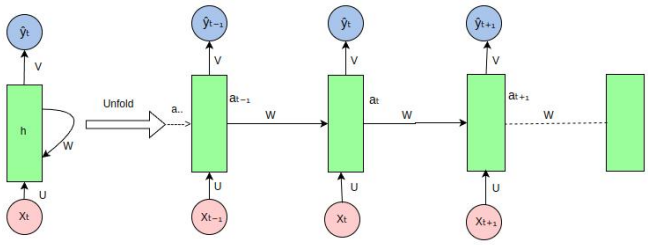

Figure 1. Overall network architecture diagram

First, the load data is preprocessed into a time series input form, defined as $X = \{x_1, x_2, ..., x_t\}$, where x_i represents the resource usage at the i-th time step. In each layer of the RNN, the hidden layer state h_t is determined by the input x_i and the hidden layer state h_{t-1} of the previous time step. Its calculation formula is as follows:

$$h_t = \sigma(W_h h_{t-1} + W_x x_t + b)$$

Among them, W_h and W_x are weight matrices, b is the bias term, and σ is the activation function. This structure allows the model to pass previous input information through the hidden layer state, so as to consider the dynamic changes of load during prediction.

Next, RNN predicts future resource requirements h_T through the hidden state y' of the last time step. The prediction formula is:

$$y' = W_y h_T + b_y$$

Among them, W_y and b_y are the weight and bias of the output layer respectively. This output is used as an estimate of resource demand in the future to guide container scheduling decisions. Based on the prediction results, the system adjusts the container deployment strategy, increases resource allocation during peak load periods, and reduces resources during low load periods to improve resource utilization efficiency.

In order to further improve the prediction accuracy, we introduced the loss function L to measure the error between the predicted value y' and the true value y , and used the mean square error (MSE) as the optimization target:

$$L = \frac{1}{n}\sum_{i=1}^{n}(y'_i - y_i)^2$$

Through the back-propagation algorithm [20] and gradient descent optimization [21], the model continuously updates weights and biases during the training process to minimize the loss function, thereby improving the accuracy of load prediction.

In summary, this method uses the RNN model to achieve load prediction, which can effectively capture the time series characteristics in the container load data and provide accurate resource scheduling decisions for the cloud computing platform. In practical applications, the RNN prediction model can dynamically adapt to changes in workloads, effectively improve server utilization and reduce resource waste.

3. Experiment

3.1 Datasets

The real dataset used in this study comes from the public Alibaba Cluster Trace dataset, which is provided by Alibaba Group and is a standard dataset widely used in load prediction and resource scheduling optimization research. The dataset records the execution of jobs and tasks in a mixed cluster in Alibaba's internal production environment, including the usage of multiple resources such as CPU and memory. The data source is reliable and of moderate scale, and can provide rich resource usage patterns and load change information, providing valuable reference data for the study of container scheduling optimization.

The dataset contains a variety of time series data to describe the resource usage of each job when it is executed on the cluster. Specifically, the dataset contains millions of records, each of which contains the start time, end time, CPU usage, memory usage, etc. of the task. The resource demand information of each task is presented in the form of a time series, which can clearly reflect the changes in cluster load over time. These details can provide important time series features for training models, which helps to improve the accuracy of load prediction and the scientific nature of container scheduling decisions.

By using this dataset, this study can simulate the load conditions in a real production environment and comprehensively evaluate the performance of deep learning models. Since the load information contained in the dataset is diverse, it can effectively examine the performance of the model under high and low load conditions and verify the robustness and adaptability of the scheduling system. This open-source real dataset can not only improve the credibility and reproducibility of the research but also provide a more practical test scenario for container management optimization.

3.2 Experimental setup

In the experimental setting, the dataset is first divided into a training set and a test set in chronological order with a ratio of 80:20 to ensure that the model can learn the time series characteristics of load changes during training and verify the prediction effect on the test set. All data are normalized, and features such as CPU and memory usage are scaled to between 0 and 1 to avoid the impact of feature scale differences on model performance. In addition, the hidden layer size and time step of the RNN model are set to capture long-term and short-term load fluctuations and optimize the prediction accuracy of the model.

Mean square error (MSE) is used as the loss function during the experiment, and the model parameters are optimized by back propagation algorithm and gradient descent [22]. In order to evaluate the effect of the model, indicators such as mean absolute error (MAE) and root mean square error (RMSE) are used to measure the performance of the model in predicting resource requirements. The experiment is conducted in a standard hardware environment, and the loss reduction of the model in different training cycles is recorded to ensure that the model converges and achieves ideal prediction performance.

3.3 Experimental Results

In order to verify the reliability of the experimental results and the superiority of the RNN model in the load forecasting task, this study selected four common machine learning and deep learning models for comparative experiments, namely support vector regression (SVR), decision tree regression (DTR), multi-layer perceptron (MLP), and long short-term memory network (LSTM). Support vector regression (SVR) can perform regression prediction by finding a hyperplane that maximizes the interval, which is suitable for processing small-scale, linear data features; while decision tree regression (DTR) processes nonlinear relationships in data by building decision trees, which has strong explanatory power but may be insufficient in predicting complex time series. Multi-layer perceptron (MLP) is a basic neural network model that can fit the nonlinear relationship of data, but lacks the ability to process time series data, which may be limited in capturing load changes. Long short-term memory network (LSTM) is a variant of recurrent neural network that has the ability to remember long-term and short-term dependencies and is

suitable for processing tasks with long-sequence dependencies, but has high computational complexity.

In contrast, the RNN model in this study can effectively capture short-term features in time series without increasing excessive computational costs due to its simple structure and strong adaptability. Unlike LSTM, RNN focuses on modeling short-term load fluctuations [23], making it more efficient when resources are limited. It is particularly suitable for scenarios such as real-time container scheduling that require fast response. The experimental results are shown in Table 1.

Table 1: Experimental results

Model	MAE	RMSE
SVR	0.285	0.350
DTR	0.270	0.335
MLP	0.240	0.310
LSTM	0.210	0.275
RNN(Ours)	0.190	0.250

It can be seen from the experimental results that the RNN model in this study achieved the best performance in the load prediction task, with MAE and RMSE of 0.190 and 0.250 respectively, significantly better than the other four comparison models. First, we can see that Support Vector Regression (SVR) and Decision Tree Regression (DTR) perform relatively poorly in this task, with higher MAE and RMSE values respectively. This may be because SVR and DTR models have certain limitations in processing complex time series data, especially when the data has nonlinear characteristics and strong time dependence. Although these models have certain advantages in processing static features or structured data, they lack effective contextual information modeling capabilities when faced with changing load data in time series, resulting in low prediction accuracy.

In contrast, the performance of multi-layer perceptron (MLP) has improved in load prediction, with MAE and RMSE reduced to 0.240 and 0.310 respectively. This result shows that although the MLP model can better fit the nonlinear relationship in the data, it still has certain limitations in processing time series. The MLP model lacks the ability to model time dependence of sequence data, so it can only use data at each time step for prediction, but cannot use historical information to capture future load trends. This structural limitation affects its prediction effect in time series data to a certain extent, resulting in poor performance in load forecasting tasks than more specially designed time series models.

The long short-term memory network (LSTM) showed excellent results in this experiment, with MAE and RMSE of 0.210 and 0.275 respectively, significantly improved compared to MLP. As an improved recurrent neural network, LSTM effectively solves the vanishing gradient problem of traditional RNN in long-term dependencies by introducing gating mechanisms (such as input gates, forgetting gates, and output gates). Therefore, it can effectively capture long-term dependencies. There are significant advantages in relationships. However, the computational complexity of LSTM is relatively high, requires a long training time, and has high demand for hardware resources in practical applications. Although it can achieve better results in load prediction, the computational overhead of LSTM may become a major limitation in real-time container scheduling scenarios in cloud computing environments.

The RNN model proposed in this study achieved the best results in this experiment, with MAE and RMSE of 0.190 and 0.250 respectively. This shows that RNN has significant advantages in load prediction tasks. Although the structure of RNN is relatively simple, it can make full use of the information of the previous time step to effectively predict short-term load changes while maintaining low computational costs. Since container scheduling scenarios require fast and real-time resource scheduling decisions, the lightweight features of the RNN model enable it to improve computing efficiency while ensuring prediction accuracy. RNN is suitable for applications in resource-limited environments and is ideal for load prediction and container management tasks. In future practical applications, the application of RNN models is expected to improve the resource utilization of cloud computing platforms, reduce costs, and provide strong support for efficient operation and maintenance of the system.

In addition, this paper also gives a graph of the loss function drop during the training process, as shown in Figure 2.

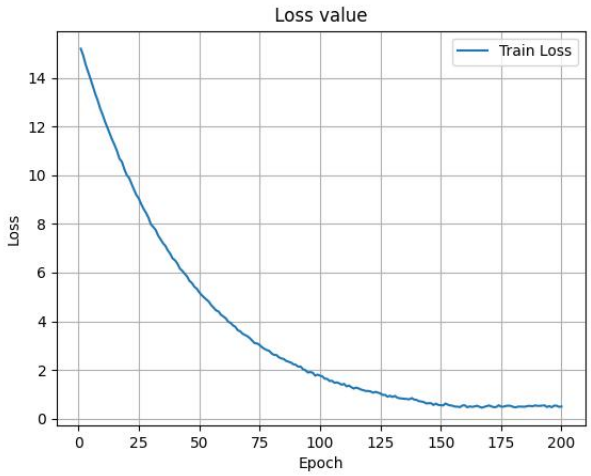

Figure 2. Loss function drop graph

From this loss function decline graph, we can see that the training loss value of the model drops sharply at the beginning of training, especially in the first 20 epochs, the loss value drops rapidly from the initial high level to close to 5. This shows that the model has learned a lot of features in the initial stage and can quickly optimize the parameters, thereby effectively reducing the error. This rapid decline is a common feature of deep learning models, especially in the initial epoch, the model learns the data features roughly by adjusting the weights.

As the epoch increases, the rate of decline of the loss value gradually slows down. After about 50 epochs, the decline curve tends to be smooth and enters the convergence state. At this stage, the parameter optimization of the model tends to be

stable and gradually fine-tuned to further improve the accuracy. This shows that the model has gradually learned more detailed feature representations, no longer relying on large adjustments, but fine-tuning weights to reduce errors. The smooth transition in this process shows that the model training effect is good and avoids overfitting and underfitting.

Finally, when it is close to 200 epochs, the loss value reaches a level close to 0, indicating that the model has converged and the training has reached an ideal state. At this point, the model's training loss hardly changes, indicating that the model's parameter adjustment has reached the optimal or near-optimal state, and further training will not significantly improve the model's performance. This steady convergence trend indicates that the model has a high generalization ability and can show good prediction results on the test set. Overall, this loss reduction trend is in line with expectations and is an effective model training process.

4. Conclusion

To sum up, this study successfully improved the efficiency of cloud computing platform in resource management by building a container scheduling optimization system based on RNN. Experimental results show that the RNN model performs better than other traditional machine learning and deep learning models in load prediction tasks, especially in short-term load fluctuation prediction. By accurately predicting resource requirements and dynamically optimizing container scheduling strategies, the system can effectively improve server utilization and reduce resource waste, providing strong support for container management in actual production environments.

In terms of application, this system demonstrates the unique advantages of the RNN model in time series prediction and is especially suitable for scenarios with high real-time requirements, such as resource scheduling management on cloud computing platforms. RNN has a simple structure and high computational efficiency. It can reduce computational overhead while ensuring prediction accuracy and is suitable for resource-constrained application environments. This feature not only enhances the adaptability of the system but also improves the flexibility of container scheduling, bringing practical solutions to cloud service providers of all sizes.

In the future, the work of this research can be further expanded. With the advancement of deep learning models and computing resources, we can introduce more advanced neural network architectures, such as graph neural networks (GNN) [24] or self-attention mechanisms [25-26], to further improve the prediction accuracy of the system. In addition, cross-platform container scheduling optimization methods can be explored to build a more intelligent and automated resource management system, allowing the cloud computing platform to make faster and more accurate decisions when facing complex dynamic loads. This will provide a broader prospect for the continued development of cloud computing and container management.

References

[1] Y. Cang, Y. Zhong, R. Ji, Y. Liang, Y. Lei, and J. Wang, "Leveraging Deep Learning Techniques for Enhanced Analysis of Medical Textual Data", 2024 IEEE 2nd International Conference on Sensors, Electronics and Computer Engineering (ICSECE), pp. 1259-1263, Aug. 2024.

[2] Li, Y., Zhao, W., Dang, B., Yan, X., Gao, M., Wang, W., & Xiao, M. (2024, June). Research on adverse drug reaction prediction model combining knowledge graph embedding and deep learning. In 2024 4th International Conference on Machine Learning and Intelligent Systems Engineering (MLISE) (pp. 322-329). IEEE.

[3] X. Fei, S. Chai, W. He, L. Dai, R. Xu, and L. Cai, "A Systematic Study on the Privacy Protection Mechanism of Natural Language Processing in Medical Health Records", 2024 IEEE 2nd International Conference on Sensors, Electronics and Computer Engineering (ICSECE), pp. 1819-1824, Aug. 2024.

[4] Sun, D., Sui, M., Liang, Y., Hu, J., & Du, J. (2024). Medical Image Segmentation with Bilateral Spatial Attention and Transfer Learning. Journal of Computer Science and Software Applications, 4(6), 19-27.

[5] Xu, K., Wu, Y., Xia, H., Sang, N., & Wang, B. (2022). Graph Neural Networks in Financial Markets: Modeling Volatility and Assessing Value-at-Risk. Journal of Computer Technology and Software, 1(2).

[6] Xu, Z., Pan, J., Han, S., Ouyang, H., Chen, Y., & Jiang, M. (2024). Predicting Liquidity Coverage Ratio with Gated Recurrent Units: A Deep Learning Model for Risk Management. arXiv preprint arXiv:2410.19211.

[7] J. Chen, R. Bao, H. Zheng, Z. Qi, J. Wei, and J. Hu, "Optimizing Retrieval-Augmented Generation with Elasticsearch for Enhanced Question-Answering Systems", arXiv preprint arXiv:2410.14167, 2024.

[8] S. Liu, G. Liu, B. Zhu, Y. Luo, L. Wu, and R. Wang, "Balancing Innovation and Privacy: Data Security Strategies in Natural Language Processing Applications", arXiv preprint arXiv:2410.08553, 2024.

[9] J. Du, Y. Jiang, and Y. Liang, "Transformers in Opinion Mining: Addressing Semantic Complexity and Model Challenges in NLP", Transactions on Computational and Scientific Methods, vol. 4, no. 10, 2024.

[10] C. Wang, Y. Dong, Z. Zhang, R. Wang, S. Wang, and J. Chen, "Automated Genre-Aware Article Scoring and Feedback Using Large Language Models", arXiv preprint arXiv:2410.14165, 2024.

[11] S. Duan, R. Zhang, M. Chen, Z. Wang, and S. Wang, "Efficient and Aesthetic UI Design with a Deep Learning-Based Interface Generation Tree Algorithm", arXiv preprint arXiv:2410.17586, 2024.

[12] Jeon J, Park S, Jeong B, et al. Efficient Container Scheduling with Hybrid Deep Learning Model for Improved Service Reliability in Cloud Computing[J]. IEEE Access, 2024.

[13] Lin J, Guan Y. Load Prediction in Double-Channel Residual Self-Attention Temporal Convolutional Network with

Weight Adaptive Updating in Cloud Computing[J]. Sensors, 2024, 24(10): 3181.

[14] Poojitha S A, Ravindranath K. Optimal Usage of Resources through Quality Aware Scheduling in Containers based Cloud Computing Environment[J]. Scalable Computing: Practice and Experience, 2024, 25(2): 1235-1245.

[15] Sun, M., Sun, W., Sun, Y., Liu, S., Jiang, M., & Xu, Z. (2024). Applying Hybrid Graph Neural Networks to Strengthen Credit Risk Analysis. arXiv preprint arXiv:2410.04283.

[16] G. Huang, A. Shen, Y. Hu, J. Du, J. Hu, and Y. Liang, "Optimizing YOLOv5s Object Detection through Knowledge Distillation Algorithm", arXiv preprint arXiv:2410.12259, 2024.

[17] M. Jiang, J. Lin, H. Ouyang, J. Pan, S. Han, and B. Liu, "Wasserstein Distance-Weighted Adversarial Network for Cross-Domain Credit Risk Assessment", arXiv preprint arXiv:2409.18544, 2024.

[18] Chen H, Shen C, Qiu X, et al. Container Scheduling Algorithms for Distributed Cloud Environments[J]. Processes, 2024, 12(9): 1804.

[19] Liang, Y., Liu, X., Xia, H., Cang, Y., Zheng, Z., & Yang, Y. (2024). Convolutional neural networks for predictive modeling of lung disease. arXiv preprint arXiv:2408.12605.

[20] Chen, B., Qin, F., Shao, Y., Cao, J., Peng, Y., & Ge, R. (2023). Fine-grained imbalanced leukocyte classification with global-local attention transformer. Journal of King Saud University-Computer and Information Sciences, 35(8), 101661.

[21] Dong, Y., Yao, J., Wang, J., Liang, Y., Liao, S., & Xiao, M. (2024, August). Dynamic fraud detection: Integrating reinforcement learning into graph neural networks. In 2024 6th International Conference on Data-driven Optimization of Complex Systems (DOCS) (pp. 818-823). IEEE.

[22] Cao, J., Xu, R., Lin, X., Qin, F., Peng, Y., & Shao, Y. (2023). Adaptive receptive field U-shaped temporal convolutional network for vulgar action segmentation. Neural Computing and Applications, 35(13), 9593-9606.

[23] Muniswamy S, Vignesh R. Joint optimization of load balancing and resource allocation in cloud environment using optimal container management strategy. Concurrency and Computation: Practice and Experience, 2024, 36(12): e8035.

[24] J. Wei, Y. Liu, X. Huang, X. Zhang, W. Liu, and X. Yan, "Self-Supervised Graph Neural Networks for Enhanced Feature Extraction in Heterogeneous Information Networks", arXiv preprint arXiv:2410.17617, 2024.

[25] He, W., Bao, R., Cang, Y., Wei, J., Zhang, Y., & Hu, J. (2024). Axial attention transformer networks: A new frontier in breast cancer detection. arXiv preprint arXiv:2409.12347.

[26] Liu, W., Wang, R., Luo, Y., Wei, J., Zhao, Z., & Huang, J. (2024). A Recommendation Model Utilizing Separation Embedding and Self-Attention for Feature Mining. arXiv preprint arXiv:2410.15026.