

Article

Research and Application of Edge Computing and Deep Learning in a Recommender System

Xiaopei Hao, Xinghua Shan, Junfeng Zhang, Ge Meng * and Lin Jiang

Institute of Computing Technology, China Academy of Railway Sciences Corporation Limited,
Beijing 100081, China; linuxstar@126.com (X.H.)

* Correspondence: mg.clx@163.com

Abstract: Recommendation systems play a pivotal role in improving product competitiveness. Traditional recommendation models predominantly use centralized feature processing to operate, leading to issues such as excessive resource consumption and low real-time recommendation concurrency. This paper introduces a recommendation model founded on deep learning, incorporating edge computing and knowledge distillation to address these challenges. Recognizing the intricate relationship between the accuracy of deep learning algorithms and their complexity, our model employs knowledge distillation to compress deep learning. Teacher–student models were initially chosen and constructed in the cloud, focusing on developing structurally complex teacher models that incorporate passenger and production characteristics. The knowledge acquired from these models was then transferred to a student model, characterized by weaker learning capabilities and a simpler structure, facilitating the compression and acceleration of an intelligent ranking model. Following this, the student model underwent segmentation, and certain computational tasks were shifted to end devices, aligning with edge computing principles. This collaborative approach between the cloud and end devices enabled the realization of an intelligent ranking for product listings. Finally, a random selection of the passengers’ travel records from the last five years was taken to test the accuracy and performance of the proposed model, as well as to validate the intelligent ranking of the remaining tickets. The results indicate that, on the one hand, an intelligent recommendation system based on knowledge distillation and edge computing successfully achieved the concurrency and timeliness of the existing remaining ticket queries. Simultaneously, it guaranteed a certain level of accuracy, and reduced computing resource and traffic load on the cloud, showcasing its potential applicability in highly concurrent recommendation service scenarios.



Citation: Hao, X.; Shan, X.; Zhang, J.; Meng, G.; Jiang, L. Research and Application of Edge Computing and Deep Learning in a Recommender System. *Appl. Sci.* **2023**, *13*, 12541. <https://doi.org/10.3390/app132312541>

Academic Editor: Xiaolin Zheng

Received: 15 September 2023

Revised: 25 October 2023

Accepted: 25 October 2023

Published: 21 November 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: recommender system; knowledge distillation; deep learning; edge computing; teacher–student model

1. Introduction

With progress in the Fourth Industrial Revolution focusing on intelligence, enterprises are increasingly adopting big data technology to settle contradictions between scale production and customization, as well as efficiency improvements and cost control. This reliance on an intellectualized system involves intelligence, information mining, network cooperation, cognitive decision making, and optimal scheduling. To efficiently navigate and automatically recommend valuable information and services to users from massive datasets, an intelligent recommendation system proves vital in combating information overload; it currently plays an important role in business marketing. Deep learning is the most popular data mining method in recent years and has demonstrated remarkable performance in diverse industrial domains, such as computer vision, automatic speech recognition (ASR), natural language processing (NLP), and intelligent recommendation. Evolving from traditional cloud computing technology, edge computing extends powerful computing resources and efficient services to network edge nodes, reducing network bandwidth and latency while enhancing energy efficiency and privacy protection [1]. At present,

deep learning is widely employed in recommendation systems by research institutions and companies, exemplified by Google's "Wide&Deep" recommendation system [2]. This system has designed a model that integrates several modules, including DeepFM, DCN, XDeepFM, and AutoInt. While these models enhance intelligent recommendations to a certain extent, the heightened accuracy of deep learning depends on its broader and deeper network structure. As this complex network structure is constructed to extract feature vectors from extensive and redundant samples, these models possess numerous parameters individually; however, their hardware resources limit the training speed for computing, and they fall short of meeting real-time high-concurrency predictions. In practice, the time consumed by such models progressively lengthens, posing a challenge for their absolute online deployment.

As the world's largest internet railway ticketing system, 12306 boasts 680 million registered users and a daily ticketing capacity exceeding 20 million tickets. The system's daily traffic can reach 53.2 billion user times, equivalent to over 616,000 people browsing tickets per second. Within 12306's ticketing system, the response time for spare ticket queries significantly influences passengers' experience and the effectiveness of the online response system. In this context, a complex deep learning model fails to meet the demands of highly concurrent query requests or operates on a mobile terminal or an embedded device. Consequently, the model requires compression and acceleration to address these challenges. Moreover, a smart ranking model of the spare tickets list on the traveling platform has been developed based on knowledge distillation. This approach aims to reduce the model's complexity, enabling the training of a unified model in different teacher models or a set of them, resulting in more accurate rankings. The knowledge distillation model reduces a model's complexity, enhances its training speed, and improves its generalization ability. At present, spare ticket queries in the aforementioned ticketing system experience high concurrency. If the smart ranking of the spare tickets list was to be processed centrally in the cloud, it would necessitate adjustments to the existing spare ticket query framework and impose high-performance requirements on spare ticket computations, potentially straining the platform's storage and running capacity. Hence, the trained model is segmented based on edge computing, considering computational complexity and data security. Some modules of the model are deployed in terminal devices enabling collaborative computing between the cloud and terminal devices to reduce the cloud's traffic load, computing, and storage resources.

2. Research Status of the Recommender System

Collaborative filtering, proposed by Su et al. [3], signifies that recommender systems, as an independent discipline, attract extensive attention. These systems revolve around the core concept of leveraging the binary relationship between travelers and products to help users discover projects of interest based on their historical behavior and relevant similarities. Content-based recommendations [4], collaborative filtering recommendations [5], and hybrid recommendations [6] represent three major conventional recommendation algorithms [7]. Moreover, recommendation algorithms find applications across diverse websites, including those focused on books, music, videos, news, movies, maps, etc. In recent years, the surge in the popularity of e-commerce applications, including Amazon, eBay, Staples, Dangdang, Douban, and Taobao, is evident, all adopting e-commerce recommendation systems. The incorporation of recommendation systems has not only generated substantial additional revenue for these internet companies but has also elevated user satisfaction and retention. Over the past few years, with the widespread adoption of deep learning, there has been a yearly escalation in articles concerning recommender systems based on deep learning. Numerous universities, research institutes, and enterprises have extensively explored and applied deep learning-based recommender systems. Consequently, this has evolved into a fervently discussed topic in the realm of recommender systems. Given the challenges in obtaining user profiles from a traditional content-based recommender system, Elkahky et al. [8] extracted user profiles by analyzing their browsing

and searching history, thus enriching the user feature representation. In addition, the deep structured semantic model (DSSM) [9] was also extended by them, introducing a multi-view deep neural network. This enhancement allows the multi-view deep neural network to perform item recommendations for users by leveraging semantic matching between two information entities, namely users and items. In essence, this represents an efficient content-based recommendation approach. Zheng et al. [10] employed a structure akin to the DSSM, seamlessly integrating comment information into the recommender system to alleviate data sparsity and enhance system quality. Building upon this, they introduced the concept of the deep cooperative neural network (DeepCoNN). Wang et al. [11], relying on deep crossing, proposed a deep and cross network (DCN) model for predicting travelers' advertisement click-through rates. The DCN model utilizes both deep and cross networks for input features, generating pertinent information for prediction. Notably, the DCN combines the advantages of a deep neural network (DNN) and deep crossing. In alignment with deep crossing principles, a deep embedding forest (DEF) model was put forward by Zhu et al. [12], which replaced residual units in deep crossing with forest layers. This substitution, along with pre-training, reduced the online prediction time. Covington et al. [13], utilizing multi-source heterogeneous user data, including user profiles, contextual information, historical behavioral data, and item features, proposed a DNN model for recommending YouTube videos. Unlike traditional recommender systems, deep learning-based recommender systems have the potential to automatically learn abstract hidden features of travelers and items through deep learning, incorporating various types of multi-source heterogeneous data. This allows for the modeling of sequence patterns of user behavior, more effectively capturing diverse user preferences and consequently improving the accuracy of recommendations; however, the performance of these deep learning systems relies on more parameters and more profound, broader models.

Consequently, such complex deep learning models fail to meet timeliness requirements in various application scenarios. Moreover, mobile phones and edge devices continually advance, equipped with specific computing power. While preserving the accuracy of deep learning-based recommender systems, research priorities now include model compression, segmentation, and collaborative training. Gou et al. [14] provided a comprehensive survey of knowledge distillation from the perspectives of knowledge categories, training schemes, teacher–student architecture, distillation algorithms, performance comparison, and applications. Zhao et al. [15] identified limitations in the coupled formulation of knowledge distillation and proposed decoupled knowledge distillation (DKD), achieving significant improvements across CIFAR-100, ImageNet, and MS-COCO datasets for image classification and object detection tasks. The concept of knowledge distillation originated with Hinton et al. [16], aiming to use a vast and accurate teacher model to instruct a small but fast student model. Based on this idea, the intention was to select cutting-edge recommender systems such as DeepFM [17], the DCN [18], and xDeepFM [19] as the teacher model, guiding the training of user-defined student models. This approach was first applied in recommending train number lists on a travel platform. Through knowledge distillation, the resultant model reduced the required number of parameters and network runtime, while maintaining accuracy. It has been demonstrated that allowing a small accuracy drop significantly reduces model deployment costs. Finally, the model was segmented to alleviate data traffic, with the segmented models deployed on both cloud and terminal devices. This approach enables high-precision reasoning under delay constraints, particularly suitable for practical production lines as a lightweight front-end deployed model.

3. Recommendation Models Based on Knowledge Distillation and Edge Computing

A traditional deep learning model is typically built via the superposition of multi-layer neural networks. Performance and computational complexity vary among models with different network layers. In addition, significant differences lie in computing resource demands and data sizes at diverse network layers. For deep learning-based recommender systems, achieving robust performance often requires a high-powered computing data

center in the cloud. In the present study, knowledge distillation was combined with deep learning and edge computing to reduce resource demands in the computing center, improve data transmission efficiency, and overcome resource limitations of terminal devices. Model compression could be realized with the help of knowledge distillation, reducing memory costs and boosting computing speed. At the same time, the trained model was segmented, and some of these segmented models could be deployed at terminal nodes leveraging the computational advantages of proximity and timeliness in edge computing. As a result, both the model reasoning latency and energy consumption were reduced.

In order to reduce the complexity of the model and improve accuracy, two important steps are proposed in this research:

1. In theory, the search space of the model is greater than that of a network. If the convergence of a network with a smaller search space matches or approximates that of a complex model, the solution spaces of these two networks may overlap. The essence of knowledge distillation can be described as follows: leveraging label features of the population parameter, a teacher–student model is employed to construct a structurally complex teacher model with strong learning ability. This knowledge is then transferred to a student model with a simpler structure and lower learning ability, enhancing its generalization ability. Furthermore, a compact model involving fewer parameters was selected to achieve accuracy comparable to that of a complex model, thus facilitating model compression. Based on knowledge distillation, knowledge gained from the complex teacher model through training was adopted to guide the student model training on a smaller scale. This enables the student model to emulate the teacher model, effectuating knowledge transfer and model compression.
2. The trained student model is formulated through the mutual superposition of multi-layer neural networks. Models with different numbers of network layers exhibit distinct properties and computational complexities. In addition, significant differences were observed in computing resource demands and data sizes across diverse network layers. The student model generated in step (1) was segmented into two sections: one section, with a heavier computational burden, conducted calculations on an edge server, while the other section, involving lighter computations, was processed on the existing terminal device. Additionally, the terminal device collaborated with the edge server system, effectively reducing the computational latency of the corresponding deep model.

3.1. Knowledge Distillation

Knowledge distillation operates within a teacher–student framework, where the practice structure is typically a well-trained model serving as a source of knowledge for learning. The student model gains the teacher model’s knowledge through distillation training, with a potential slight performance loss traded for transferring knowledge from complex teacher models to a simpler student model. Currently, most instances of knowledge distillation focus solely on a single teacher model, overlooking the possibility that a single student model can be supervised by multiple teacher models [20,21]. Alternatively, recognizing the equal significance of multiple teacher models renders it impractical to acquire more valid knowledge based on inherent distinctions in these models. To enhance model accuracy, the knowledge distillation model, typically involving a single teacher model, was expanded to include multiple teacher models. Consequently, a student model can now be trained by several teacher models, enabling the acquisition of valid knowledge aligned with the interior differences among the models. In this study, the teacher model comprised state-of-the-art models such as DeepFM, the DCN, and xDeepFM, while a user-defined neural network model formed the student model. Within this framework, the student model can autonomously learn the importance of different teacher models for a given data sample and integrate this knowledge based on learned diverse degrees of importance. This approach significantly contributes to enhancing the learning capability of the student model [22]. The primary structure of this teacher–student model is presented in Figure 1 below:

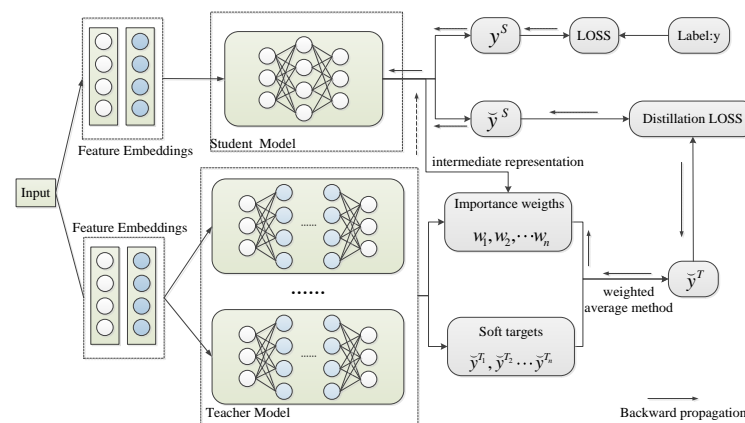


Figure 1. Teacher–student model.

- (1) Different teacher models (quantity: n) were trained. These models were then used as samples in the training set to produce soft labels, \tilde{y}^{Ti} , accordingly. In order to figure out the soft labels, \tilde{y}^{Ti} , $Z(Z = (z_1, z_2 \dots z_t))$ was output through logits from the teacher model and then divided by the T parameter; the outcome obtained was subjected to *Softmax* at last. Here, T is a temperature coefficient. The higher the T parameter, the gentler the corresponding distribution probability tends to be.
- (2) The number of datasets was input into the student model, which was followed via operating it in the same way as the teacher model and obtaining an output of logits. Subsequently, relevant calculations were divided into two steps: First, via division by the same T parameter as that of the teacher model, $\tilde{y}^{Ti} = \text{Softmax}\left(\frac{z_i^T}{T}\right)$, *Softmax*-based computing was conducted, obtaining an output of soft predictions, \tilde{y}^S , and this output was then compared with soft labels of the teacher model. Second, after *Softmax*-based calculations, the predicted values were acquired and then compared with True labels.
- (3) For the corresponding loss function, *KDLoss* was selected, and it was formed via the combination of a relative entropy loss function and a cross entropy loss function [23]. Here, Kullback–Leibler divergence was used to measure the asymmetry of differences in probability distributions A and B, while cross entropy represents a difference value between the predicted sample label and the true sample label. A combination of these loss functions is beneficial to better reveal both differences in and differential values of the predicted samples and the true samples.

$$KDLoss = \sum_{i=1}^n w_i \cdot KL(\tilde{y}^{Ti}, \tilde{y}^S) \cdot \lambda \cdot T^2 + CE(label, y^S) \cdot (1 - \lambda) \quad (1)$$

where *KL* stands for relative entropy, *CE* stands for cross entropy, and w_i stands for the weight of the teacher model, i . Under the circumstance that coefficient λ is 0, $KDLoss = CE(label, y^S)$, no knowledge distillation is performed for the hybrid loss function that adopts a neural network of the cross entropy loss function.

- (4) In cases where a different teacher model is selected, knowledge at diverse importance levels will be provided for the student model. A teacher model of low efficiency may even mislead the learning results of the student model. Here, w_i refers to the weight of importance of the knowledge contributed by each teacher model, and M refers to the number of teacher models. In particular, the weight, w_i , needs to meet a condition of

$\sum_{i=1}^n w_i = 1$. To realize self-adaption to the knowledge distillation learning framework of a multi-teacher model, w_i is defined as follows:

$$w_i = \frac{e^{(r_i z_{T_i} + b_i)}}{\sum_{i=1}^M e^{(r_i z_{T_i} + b_i)}}, i = 1, \dots, n \quad (2)$$

where r_i and b_i are parameters that should be learned.

3.2. Model Segmentation

The proposed student model, outlined in Figure 2, mainly comprises an input layer, a hidden layer, an aggregation layer, and an output layer. Furthermore, the output layer consists of three features, namely a passenger feature, a train number feature, and a real-time interaction feature. The hidden layer was divided into two parts; passenger feature data and the train number feature data were subjected to embedding in the hidden layer. The embedding structure for the train number feature involves its fusion with the real-time interaction feature. Afterwards, feature fusion was fulfilled by virtue of full connections at the aggregation layer. In the fully connected layer, each neuron was connected to all of the neurons of the previous layer, and there was a certain connection weight between each input feature and each neuron. The input feature space was mapped to the output result space, thereby achieving the complexity and nonlinear fitting ability of the model. For the features fused, multi-level operations were performed. After computations were performed on these operation results, the predicted values were obtained.

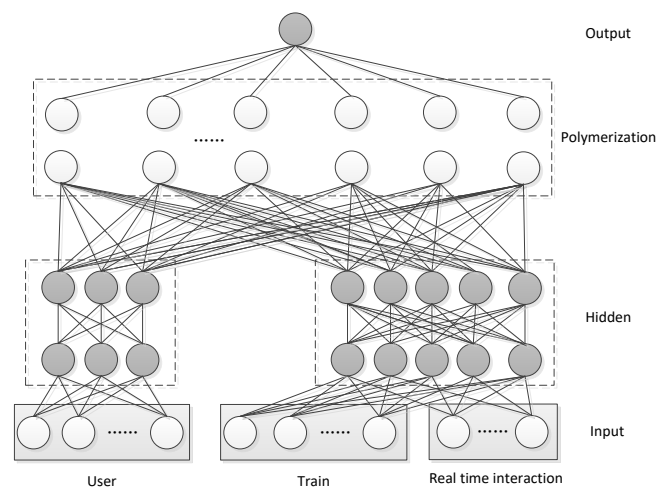


Figure 2. Student model.

For the well-trained student model, resource-intensive computing, extensive bandwidth consumption, and reliance on cloud computing network latency pose challenges. To improve the computing speed of the recommender system model, reduce cloud computing resources, and optimize terminal device resources, a specific algorithm segmented the model into a limited number of parts. After segmentation, diverse network layers were deployed on terminal devices or the cloud based on their data and computational complexity. Terminal devices are mainly in charge of reserving network layers involving real-time interaction data, while network layers with small data sizes but high computational complexity or containing passengers' private data were deployed on the cloud. This approach aims to reduce data traffic between the terminal devices and the cloud, lower traffic loads, and uphold the security of private data. Through collaborative computing of the terminal devices and the cloud, the results computed on the cloud were sent back to the terminal devices. Aggregating data from terminal devices and the cloud occurred on the terminal devices, facilitating real-time computing.

In the present study, the student model was segmented into four modules: passenger feature embedding, train number feature embedding, an aggregation module, and the predicted result output. Based on Edgent, a deep learning model co-inference framework promoting device–edge synergy, different modules and diverse neural network layers were employed to train a regression model estimating the running latency of neural network layers on terminal devices and the cloud. Specific to problems such as data security and data transmission, model segmentation points were carefully designed. Moreover, passenger feature embedding relates to passengers’ massive amounts of essential data and transaction data. To ensure the security of passengers’ private data, it was deployed on the cloud. Train number feature embedding involves fundamental information on train numbers and relevant real-time interaction data. Given the abundant interaction data generated during ticket purchases, this module was deployed on a terminal device to minimize network latency. For this reason, this module was deployed on a terminal device. The aggregation module and predicted result output were also placed on terminal devices, aiming to reduce the transmission of train number feature data [24].

4. Experimental Process and Experimental Result Analysis

To validate the accuracy and concurrency of the recommendation model, passengers’ ticket purchase data over the last five years were randomly extracted for experimentation. The precision rates, recall rates, F1 scores, and NDCG of corresponding testing results were averaged and compared with recommendation results and concurrencies obtained by classic algorithms, such as FM [25], FFM [26], Wide&Deep [27], DeepFM [28], and the DCN. This study employed the traditional recommendation algorithm FM and the deep learning-based recommendation algorithm Wide&Deep as baselines for testing and evaluation [19,29].

4.1. Experimental Datasets

As shown in Table 1 below, the experimental dataset consists of four categories of data, namely the ticket purchase data, user information, train information, and real-time interaction.

Table 1. Experimental datasets.

Dataset	Instances	Fields	Positive Ratio
Ticket purchase data	240 M	38	86%
User info	25 M	40	92%
Train info	3000	20	95%
Real-time interaction	125 M	18	74%

4.2. Experimental Process

As illustrated in Table 1 below, the experimental dataset comprises four categories of data: ticket purchase data, user information, train information, and real-time interaction.

(1) Training Stage

The most advanced algorithms at present, DeepFM, DCN, and xDeepFM, were used as teacher models. Random seeds were varied for five experiments to ensure statistical significance. In addition, cross entropy served as the loss function to train all teacher models. Subsequently, based on ensemble knowledge distillation (EKD), the teacher model was initially trained to recognize the target dataset and extract key attributes from samples. Afterward, the ticket purchase dataset was selected for *Softmax*-based computing based on an outcome achieved by dividing the output of logits in the teacher model by the T-parameter, thus producing the values of soft labels. The dataset was then input into the student model, repeating operations of the teacher model and acquiring an output of logits. Afterwards, calculations were performed in two steps: First, the output of logits was divided by the same T-parameter as that of the teacher model, followed by *Softmax*-based computation, generating soft predictions that were further used as outputs to be compared

with soft labels. Second, after *Softmax*-based computing, predicted values were obtained and then compared with True labels. The total loss function, *KD Loss*, was acquired by adding the loss functions of the above two steps together. After determining the loss function, the gradient decreased, and the corresponding parameters needed to be updated. Finally, the availability of the trained model was analyzed in comparison with FM, FFM, Wide&Deep, DeepFM, and the DCN using the same dataset.

(2) Real-Time Computing Stage

Real-time computing is designed to assess the concurrency and timeliness of the deployed recommendation system. Following the segmentation of the student model, these segments were deployed on both the cloud and terminal devices. A recommendation list was generated through collaborative computing based on passengers' demands. Simultaneously, other models were deployed on the cloud to conduct concurrency testing. The same cloud resources and terminal devices used during training were employed to analyze their concurrencies and computational efficiency.

4.3. Experimental Results and Analysis

A change in the temperature reflects the degree to which the student model pays attention to negative labels. When the temperature is low, the information carried by negative categories is relatively reduced, resulting in less attention to negative categories. A lower probability of negative categories corresponds to reduced attention. When the temperature is high, the probability value of negative categories will relatively increase, and the information carried by negative categories will be relatively amplified. The student model will pay more attention to negative labels. This study selected the temperature {1, 3, 7} for a comparative analysis, and separately trained these student models with different temperature, T , values under fixed samples. The experimental results revealed that when the temperature, T , was taken as 1, 3, and 7, the average inference precision of the student model was 43.2%, 52.8%, and 48.7%, respectively. Therefore, we chose 3 as the temperature parameter.

After training with the proposed KD model and other algorithms, including FM, FFM, Wide&Deep, DeepFM, and the DCN, a train number recommendation list was generated and compared with the actual purchases made by passengers. The model's performance was then evaluated based on precision, recall rates, and F1 scores. The comparison results are presented in the figures below.

Evidently, the recommendation results of the five algorithms tested exhibit varying indices. As shown in Figures 3–5, the deep learning-based recommendation model outperforms other models like FM and FFM in terms of precision, recall rates, and F1 scores. Particularly in F1 scores, the proposed KD recommendation model demonstrates certain advantages among the evaluated recommender systems but still lags behind deep learning models, which is especially the case for the DCN.

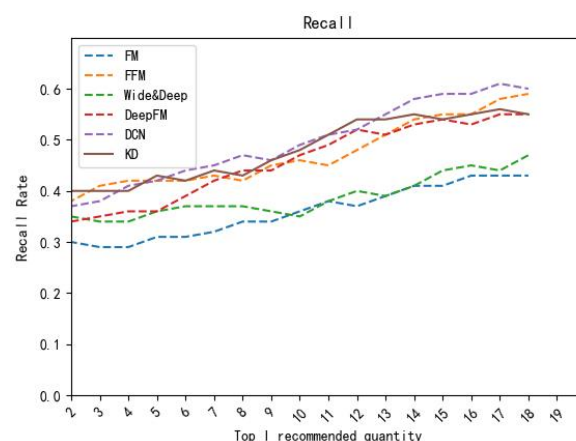


Figure 3. Recall comparison.

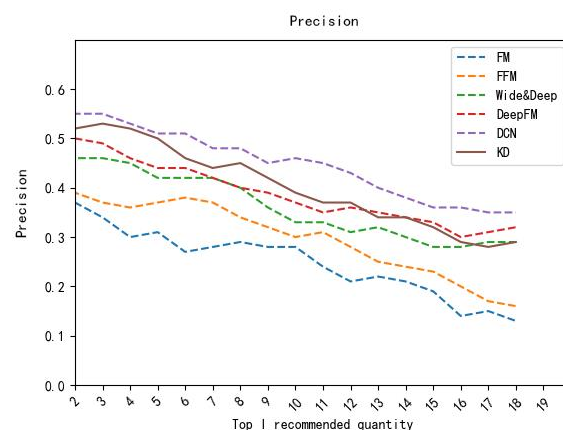


Figure 4. Precision comparison.

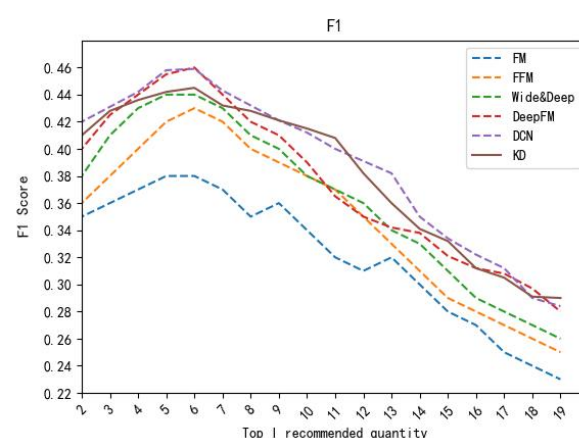


Figure 5. F1 comparison.

The testing results of NDCG are given in Table 2. It can be observed that the deep learning-based recommendation model outperforms other models such as FM and FFM in NDCG. Among them, the DCN performed the best, followed by KD.

Table 2. The NDCG comparison.

Model	NDCG@4	NDCG@8	NDCG@12	NDCG@16	NDCG@20
FM	0.0379	0.0571	0.0728	0.0531	0.0312
FFM	0.0631	0.8391	0.1257	0.0452	0.0429
Wide&Deep	0.0931	0.1231	0.1498	0.0943	0.0782
DeepFM	0.1432	0.1674	0.1982	0.1321	0.0912
DCN	0.1627	0.2027	0.2219	0.1337	0.0937
KD	0.1523	0.1841	0.2087	0.1186	0.0821

Concurrency testing results are illustrated in Figure 6. Upon the deployment of Wide&Deep, DeepFM, the DCN, and KD models on identical cloud and terminal devices, the KD model exhibits significantly superior dynamic computing time and concurrency compared to other models, including Wide&Deep and DeepFM. Moreover, the cost of model deployment is notably reduced.

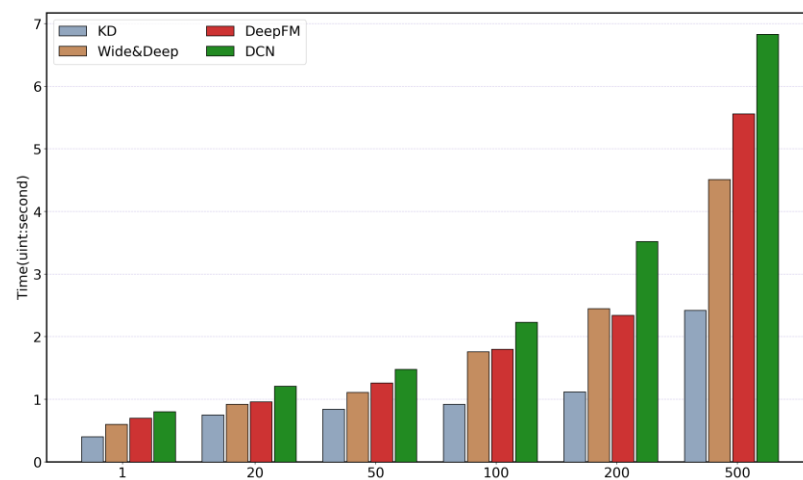


Figure 6. Concurrent access comparison.

In summary, while the accuracy of the KD model experiences a moderate decrease, the model deployment cost is significantly reduced. Simultaneously, concurrency and the efficiency of real-time computing are enhanced.

5. Conclusions

Given the rapid advancements in information technologies and Internet products, the issue of information overload is escalating. In the realm of rail transit, passengers' incessant demands for relevant and valid information are on the rise. Recommender systems are assuming an increasingly pivotal role during the digital transformation in various fields. In comparison to traditional recommendation algorithms, deep learning-based recommender systems have the capability to (1) fuse various types of multi-source heterogeneous data; (2) extract high-level features of big data; (3) automatically perform embedding for features of passengers and products; (4) acquire a valid representation of data space or features; and (5) effectively reflect the potential preferences of passengers and characteristics of products. In this way, the performance of the relevant recommendation models can be improved; however, as the model performance advances the structure of the deep learning network becomes more complex. Model training and computing require significant network, storage, CPU, and memory overhead, posing significant constraints on deep learning applications in resource-limited environments or high-concurrency real-time recommendation scenarios. To boost the precision of the recommender system, reduce the complexity of the deep learning model, and improve the concurrency as well as timeliness of relevant computing, classic deep learning-based recommendation models, such as DeepFM, the DCN, and xDeepFM, were used as teacher models. These were employed to construct a multi-teacher model based on the principles of knowledge distillation, subsequently used to train the user-defined student model. This approach, undertaken to ensure recommendation accuracy, facilitated the structural compression of deep learning models.

Additionally, the student model was segmented based on edge computing principles, allowing certain computations to be migrated to a mobile terminal for collaborative computing between a data center and a terminal device. This effort generated a recommendation list, improving data transmission efficiency, reducing traffic load, optimizing cloud resource allocation, and enhancing the timeliness as well as concurrency of the recommendation model. Finally, the proposed model underwent testing for train number list recommendation on the 12306 ticketing system. A comprehensive analysis found that the proposed model performs excellently in terms of precision, recall rates, F1 scores, timeliness, and high concurrency. In practical applications, this research methodology proves feasible and holds significant value in model deployment as well as real-time production line applications. In authentic learning scenarios, students not only learn from correct answers and teacher guidance but also through interactions with their peers, a crucial aspect in increasing their

learning ability. In future research, it is advisable to explore the integration of multiple student models within the knowledge distillation framework to enhance the efficiency of these models. Simultaneously, the authors intend to conduct further investigations into the application of deep learning models within an edge computing architecture, aiming to enhance the prediction accuracy and efficiency of residual ticket queries in the 12306 ticketing system. It is worth noting that the 12306 ticketing system encompasses not only train ticket sales but also various travel services such as hotels, tourism, and catering; however, practical challenges may arise due to the potential isolation of information by different data holders, leading to a data isolation problem. Subsequent research efforts should be directed towards addressing and resolving this issue [30].

Author Contributions: Conceptualization: X.H.; methodology: X.S.; software: X.H.; validation: X.H., L.J. and J.Z.; formal analysis: X.S.; investigation: X.H. and G.M.; resources: X.S.; data curation: J.Z.; writing—original draft preparation: X.H.; writing—review and editing: X.H. and G.M.; visualization: X.H. and L.J.; supervision: X.S. and J.Z.; project administration: X.H.; funding acquisition: X.H. and G.M. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by scientific research projects of China Academy of Railway Sciences Co., Ltd. (grant no. 2023YJ136), and by the Science and Technology Research Project of Beijing-Shanghai High Speed Railway Co., Ltd. (grant no. Beijing-Shanghai Scientific Research-2022-7).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available in article.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper. All authors are employees of China Academy of Railway Sciences Co., Ltd., who provided funding and technical support for the work. The authors declare that this study received funding from Science and Technology Research Project of Beijing-Shanghai High Speed Railway Co., Ltd. The funder had no role in the design of the study; in the collection, analysis, or interpretation of data, in the writing of the manuscript, or in the decision to publish the results.

References

- Shi, W.; Cao, J.; Zhang, Q.; Li, Y.; Xu, L. Edge computing: Vision and challenges. *IEEE Internet Things J.* **2016**, *3*, 637–646. [\[CrossRef\]](#)
- Shi, S.; Zhang, M.; Lu, H.; Liu, Y.; Ma, S. Wide & deep learning in job recommendation: An empirical study. In *Asia Information Retrieval Symposium*; Springer: Cham, Switzerland, 2017; pp. 112–124.
- Su, X.; Khoshgoftaar, T.M. A survey of collaborative filtering techniques. *Adv. Artif. Intell.* **2009**, *2009*, 421425. [\[CrossRef\]](#)
- Mooney, R.J.; Roy, L. Content-based book recommending using learning for text categorization. In *DL '00: Proceedings of the Fifth ACM Conference on Digital Libraries*; ACM Digital Library: New York, NY, USA, 2000; pp. 195–204.
- Breese, J.S.; Heckerman, D.; Kadie, C. Empirical analysis of predictive algorithms for collaborative filtering. *arXiv* **2013**, arXiv:1301.7363.
- Balabanović, M.; Shoham, Y. Fab: Content-based, collaborative recommendation. *Commun. ACM* **1997**, *40*, 66–72. [\[CrossRef\]](#)
- Verbert, K.; Manouselis, N.; Ochoa, X.; Wolpers, M.; Drachsler, H.; Bosnic, I.; Duval, E. Context-Aware Recommender Systems for Learning: A Survey and Future Challenges. *IEEE Trans. Learn. Technol.* **2012**, *5*, 318–335. [\[CrossRef\]](#)
- Elkahky, A.M.; Song, Y.; He, X. A Multi-View Deep Learning Approach for Cross Domain User Modeling in Recommendation Systems. In *Proceedings of the 24th International Conference on World Wide Web, Florence, Italy, 18–22 May 2015*; pp. 278–288.
- Yadav, N.; Singh, A.K.; Pal, S. Improved self-attentive Musical Instrument Digital Interface content-based music recommendation system. *Comput. Intell.* **2022**, *38*, 1232–1257. [\[CrossRef\]](#)
- Liu, K.; Xue, F.; Guo, D.; Wu, L.; Li, S.; Hong, R. MEGCF: Multimodal Entity Graph Collaborative Filtering for Personalized Recommendation. *ACM Trans. Inf. Syst.* **2023**, *41*, 1–27. [\[CrossRef\]](#)
- Hussain, J.S.I.; Ghazali, R.; Javid, I.; Hassim, Y.M.M.; Khan, M.H. A Hybrid Solution for The Cold Start Problem in Recommendation. *Comput. J.* **2023**, *8*, bxad088. [\[CrossRef\]](#)
- Zhu, J.; Shan, Y.; Mao, J.C.; Yu, D.; Rahmanian, H.; Zhang, Y. Deep Embedding Forest: Forest-based Serving with Deep Embedding Features. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, 13–17 August 2017*; pp. 1703–1711.
- Covington, P.; Adams, J.; Sargin, E. Deep Neural Networks for YouTube Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, 15–19 September 2016*; pp. 191–198.

14. Gou, J.; Yu, B.; Maybank, S.J.; Tao, D. Knowledge distillation: A survey. *Int. J. Comput. Vis.* **2021**, *129*, 1789–1819. [[CrossRef](#)]
15. Zhao, B.; Cui, Q.; Song, R.; Qiu, Y.; Liang, J. Decoupled knowledge distillation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 11953–11962.
16. Hinton, G.; Vinyals, O.; Dean, J. Distilling the knowledge in a neural network. *arXiv* **2015**, arXiv:1503.02531.
17. Guo, H.; Tang, R.; Ye, Y.; Li, Z.; He, X. DeepFM: A factorization-machine based neural network for CTR prediction. *arXiv* **2017**, arXiv:1703.04247.
18. Shen, X.; Dai, Q.; Mao, S.; Chung, F.-L.; Choi, K.-S. Network together: Node classification via cross-network deep network embedding. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 1935–1948. [[CrossRef](#)] [[PubMed](#)]
19. Lian, J.; Zhou, X.; Zhang, F.; Chen, Z.; Xie, X.; Sun, G. xDeepFM: Combining Explicit and Implicit Feature Interactions for Recommender Systems. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 1754–1763.
20. Yu, L.; Li, Y.; Weng, S.; Tian, H.; Liu, J. Adaptive multi-teacher softened relational knowledge distillation framework for payload mismatch in image steganalysis. *J. Vis. Commun. Image Represent.* **2023**, *95*, 103900. [[CrossRef](#)]
21. Jeon, E.S.; Choi, H.; Shukla, A.; Turaga, P. Leveraging angular distributions for improved knowledge distillation. *Neurocomputing* **2023**, *518*, 466–481. [[CrossRef](#)]
22. Ding, H.; Chen, K.; Huo, Q. Improving Knowledge Distillation of CTC-Trained Acoustic Models with Alignment-Consistent Ensemble and Target Delay. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2020**, *28*, 2561–2571. [[CrossRef](#)]
23. Hershey, J.R.; Olsen, P.A. Approximating the Kullback Leibler Divergence Between Gaussian Mixture Models. In Proceedings of the 2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP '07, Honolulu, HI, USA, 15–20 April 2007; Volume 4, pp. IV-317–IV-320.
24. Li, E.; Zhou, Z.; Chen, X. Edge Intelligence: On-Demand Deep Learning Model Co-Inference with Device-Edge Synergy. In Proceedings of the 2018 Workshop on Mobile Edge Communications, Budapest, Hungary, 20 August 2018; pp. 31–36.
25. Rendle, S. Factorization Machines. In Proceedings of the 2010 IEEE International Conference on Data Mining, Sydney, NSW, Australia, 13–17 December 2010; pp. 995–1000.
26. Sun, J.; Zhao, L.; Liu, Z.; Li, Q.; Deng, X.; Wang, Q.; Jiang, Y. Practical differentially private online advertising. *Comput. Secur.* **2022**, *112*, 102504. [[CrossRef](#)]
27. Da, F.; Peng, C.; Wang, H.; Li, T. A risk detection framework of Chinese high-tech firms using wide & deep learning model based on text disclosure. *Procedia Comput. Sci.* **2022**, *199*, 262–268.
28. Xu, J.; Hu, Z.; Zou, J. Personalized Product Recommendation Method for Analyzing User Behavior Using DeepFM. *Korea Inf. Process. Soc.* **2021**, *17*, 369–384.
29. Wang, R.; Shivanna, R.; Cheng, D.; Jain, S.; Lin, D.; Hong, L.; Chi, E. Dcn v2: Improved deep & cross network and practical lessons for web-scale learning to rank systems. In Proceedings of the Web Conference 2021, Ljubljana, Slovenia, 19–23 April 2021; pp. 1785–1797.
30. Chen, C.; Zhou, J.; Zheng, L.; Wu, H.; Lyu, L.; Wu, J.; Wu, B.; Liu, Z.; Wang, L.; Zheng, X. Vertically federated graph neural network for privacy-preserving node classification. *arXiv* **2020**, arXiv:2005.11903.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.