IntTower: the Next Generation of Two-Tower Model for Pre-Ranking System

Xiangyang Li[*]
Peking University
China
xiangyangli@pku.edu.cn

Bo Chen[*]
Huawei Noah's Ark Lab
China
chenbo116@huawei.com

HuiFeng Guo[†]
Huawei Noah's Ark Lab
China
huifeng.guo@huawei.com

Jingjie Li
Huawei Noah's Ark Lab
China
lijingjie1@huawei.com

Chenxu Zhu
Shanghai JiaoTong University
China
zhuchenxv@sjtu.edu.cn

Xiang Long
Beijing University of Posts and Telecommunication
China
xianglong@bupt.edu.cn

Sujian Li
Peking University
China
lisujian@pku.edu.cn

Yichao Wang
Huawei Noah's Ark Lab
China
wangyichao5@huawei.com

Wei Guo
Huawei Noah's Ark Lab
China
guowei67@huawei.com

Longxia Mao
Huawei Technologies Co Ltd
China
maolongxia@huawei.com

Jinxing Liu
Huawei Technologies Co Ltd
China
liujinxing5@huawei.com

Zhenhua Dong
Huawei Noah's Ark Lab
China
dongzhenhua@huawei.com

Ruiming Tang[†]
Huawei Noah's Ark Lab
China
tangruiming@huawei.com

ABSTRACT

Scoring a large number of candidates precisely in several milliseconds is vital for industrial pre-ranking systems. Existing pre-ranking systems primarily adopt the **two-tower** model since the "user-item decoupling architecture" paradigm is able to balance the *efficiency* and *effectiveness*. However, the cost of high efficiency is the neglect of the potential information interaction between user and item towers, hindering the prediction accuracy critically. In this paper, we show it is possible to design a two-tower model that emphasizes both information interactions and inference efficiency. The proposed model, IntTower (short for *Interaction enhanced Two-Tower*), consists of Light-SE, FE-Block and CIR modules. Specifically, lightweight Light-SE module is used to identify the importance of different features and obtain refined feature representations in each tower. FE-Block module performs fine-grained and early feature interactions to capture the interactive signals between user and item towers explicitly and CIR module leverages a contrastive interaction regularization to further enhance the interactions implicitly. Experimental results on three public datasets show that IntTower outperforms the SOTA pre-ranking models significantly and even achieves comparable performance in comparison with the ranking models. Moreover, we further verify the effectiveness of IntTower on a large-scale advertisement pre-ranking system. The code of IntTower is publicly available[1].

[*]Co-first authors with equal contributions. Work done when Xiangyang Li was intern at Huawei Noah's Ark Lab.
[†]Corresponding authors.

CIKM '22, October 17–21, 2022, Atlanta, GA, USA
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9236-5/22/10…$15.00
https://doi.org/10.1145/3511808.3557072

CCS CONCEPTS

• **Information systems** → **Information retrieval**; **Recommender systems**; • **Computing methodologies** → **Machine learning**.

KEYWORDS

Recommender Systems, Pre-Ranking System, Neural Networks

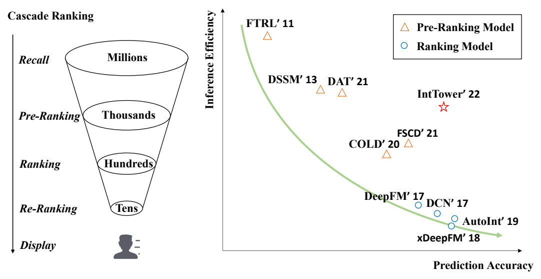
ACM Reference Format:
Xiangyang Li, Bo Chen, HuiFeng Guo, Jingjie Li, Chenxu Zhu, Xiang Long, Sujian Li, Yichao Wang, Wei Guo, Longxia Mao, Jinxing Liu, Zhenhua Dong, and Ruiming Tang. 2022. IntTower: the Next Generation of Two-Tower

[1]https://github.com/archersama/IntTower

Model for Pre-Ranking System. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM '22), October 17–21, 2022, Atlanta, GA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3511808.3557072

1 INTRODUCTION

Existing industrial information services, such as recommender system, search engine, and advertisement system, are multi-stage cascade ranking architecture, which contributes to balancing the efficiency and effectiveness in comparison with the single-stage architecture [22]. Typical cascade ranking system consists of Recall, Pre-Ranking, Ranking, and Re-Ranking stages (Figure 1(a)). The early stages face a massive number of candidates, and thus using simple models (*e.g.*, LR and DSSM [9]) to guarantee low inference latency. On the contrary, the later stages pursue subtly selected items that meet the user's preferences, and hence complex models (*e.g.*, DeepFM [6] and AutoInt [26]) are conducive to improve the prediction accuracy.



(a) Cascade ranking system. (b) Comparison between prediction accuracy and inference efficiency.

Figure 1: The multi-stage cascade ranking architecture and the comparison of model prediction accuracy and inference efficiency.

Pre-ranking stage is in the middle of a cascade link, which is absorbed in preliminarily filtering items (thousands of scale) retrieved from the previous recall stage and generating candidates (hundreds of scale) for the subsequent ranking stage. Therefore, both effectiveness and efficiency need to be carefully considered. Figure 1(b) depicts some representative pre-ranking and ranking models from the perspective of *prediction accuracy* and *inference efficiency*. Compared with the ranking models, pre-ranking models need to score more candidate items for each user request. Therefore, pre-ranking models have higher inference efficiency while weaker prediction performance due to simpler structure.

In the evolution of pre-ranking system, LR (Logistic Regression) [19] is the most basic personalized pre-ranking model, which is widely used in the shallow machine learning era [20, 28, 31]. With the rise of deep learning, many industrial companies deploy various deep models in their commercial systems. The dominant pre-ranking model in industry is two-tower model [9] (*i.e.*, DSSM), which utilizes neural networks to capture the interactive signals within the user/item towers. Moreover, the item representations can be pre-calculated offline and stored in the fast retrieval container. During the online serving, only user representations are required to be calculated in real time while the representations of candidate items can be retrieved directly. These "*user-item decoupling architecture*" paradigm provides sterling efficiency. Besides, COLD [32] and FSCD [18] propose a single-tower structure to fully model feature interaction and further improve the prediction accuracy.

Despite great promise, existing pre-ranking models are difficult to balance model effectiveness and inference efficiency. For the two-tower model, the cost of high efficiency is the neglect of the information interaction between user and item towers. Two towers perform intra-tower information extraction parallelly and independently, and the learned latent representations do not interact until the output layer, which is referred to as "*Late Interaction*" [36], hindering the model performance critically. However, the interactive signals between user features and item features are vital for prediction [30]. Though DAT [35] attempts to alleviate this issue by implicitly modeling the information interaction between the two towers, the performance gain is still limited. As for the single-tower structure pre-ranking models (*i.e.*, COLD and FSCD), although several optimization tricks are introduced for acceleration, the efficiency degradation is still severe (×10).

To solve the efficiency-accuracy dilemma, we propose a next generation of two-tower model for pre-ranking system, named ***Int****eraction enhanced Two-***Tower*** (**IntTower**), as illustrated in the Figure 3. The core idea is to enhance the information interaction between user and item towers while keeping the "*user-item decoupling architecture*" paradigm. By introducing fine-grained feature interaction modeling, the model capacity of two-tower can be improved significantly and the sterling inference efficiency can be maintained. Specifically, IntTower first leverages a lightweight *Light-SE* module to identify the importance of different features and obtain refined feature representations. Based on the refined representations, user and item towers leverage multi-layer nonlinear transformation to extract latent representations. To capture the interactive signals between user and item representations, IntTower designs *FE-Block* module and *CIR* module from **explicit** and **implicit** perspectives, respectively. FE-Block module performs fine-grained and early feature interactions between multi-layer user representations and last-layer item representation. Thus, multi-level feature interaction modeling contributes to improving the prediction accuracy while the user-item decoupling architecture enables high inference efficiency. Moreover, CIR proposes a contrastive interaction regularization to further enhance the interactions between user and item representations.

Our main contributions are summarized as follows: (1) We propose IntTower, the next generation of two-tower model for the pre-ranking system, which emphasizes both high prediction accuracy and inference efficiency. (2) IntTower leverages a lightweight Light-SE module to obtain refined feature representations. Based on this, FE-Block module and CIR module are proposed to capture the interactive signals between user and item representations from explicit and implicit perspectives. (3) Comprehensive experiments are conducted on three public datasets to demonstrate the superiority of IntTower over prediction accuracy and inference efficiency. Moreover, we further verify the effectiveness of IntTower on a large-scale advertisement pre-ranking system.

2 RELATED WORK

2.1 Pre-Ranking System

Pre-ranking system is located in middle stage of cascade ranking system and needs to predict thousands of candidate items in a few milliseconds, which is sensitive to both model effectiveness and efficiency. LR model is the simplest personalized pre-ranking model, which has strong fitting capability and is widely used in the shallow machine learning era. Besides, FM model [24] is also popular for the pre-ranking stage, which models the low-order feature interactions using factorized parameters and can be calculated in linear time.

With the rise of deep learning [14, 15], neural network-based models are gradually introduced into industrial pre-ranking systems. The most important one is two-tower model [9] (*i.e.*, DSSM), which designs two-tower to capture the interactive signals within the user/item towers. Moreover, the "*user-item decoupling architecture*" designing paradigm enables sterling inference efficiency. Due to the pre-storage of the item representations, during the online serving, only a single inference is required to obtain user representation for each request. However, the interaction modeling is inadequate for DSSM, hindering the model performance critically. To alleviate this issue, DAT [35] implicitly models the information interaction between the two towers with an Adaptive-Mimic Mechanism. Compared with the DSSM and DAT, our proposed IntTower proposes FE-Block and CIR modules to capture the interactive signals between two-tower from explicit and implicit perspectives, respectively.

To improve the prediction accuracy, COLD [32] and FSCD [18] leverage the single-tower structure to fully model feature interaction. To balance the model efficiency and effectiveness, they adopt optimization tricks (*e.g.*, parallel computation and semi-precision calculation) and complexity-aware feature selection for efficiency acceleration, respectively. However, the single-tower structure determines the need for multiple inferences for each user request given multiple candidate items, which is more time-consuming than two-tower structure.

2.2 Ranking System

For ranking system, whose scale of predicted candidate items is much smaller than that in pre-ranking, user preferences over items need to be learned more accurately. Therefore, models deployed in the ranking system focus on extracting feature interactions with various operations. DNN model is widely used to capture the high-order implicit feature interactions; while the explicit feature interaction modeling is diverse for different models. Wide&Deep [4] utilizes handcrafted cross features to memorize important patterns. PNN [23] and DeepFM [6] use the inner product to capture pairwise interactions. CFM [33] and FGCNN [17] leverages the convolution operation to identify the local patterns and models feature interactions. DCN [30] and EDCN [2] use Cross Network to learn certain bounded-degree feature interactions, while xDeepFM [16] extends to vector-wise level with a Compressed Interaction Network. AutoInt [26] and DIN [39] leverage the Attention Network to model high-order feature interactions and user historical behaviors. However, these ranking models belong to single-tower structure, which have large serving latency and cannot be deployed in the pre-ranking system directly. Instead, our IntTower performs fine-grained and early interaction modeling with two-tower structure, achieving comparable prediction accuracy while higher inference efficiency than the ranking models.

3 PRELIMINARY

The neural network based two-tower, one of the state-of-the-art pre-ranking models, has excellent trade-off between prediction accuracy and inference efficiency. In this section, we first present the details of two-tower, then demonstrate both advantages and disadvantages of applying two-tower model in the pre-ranking system. The architecture is shown in Figure 2, which consists of two parallel sub-networks.

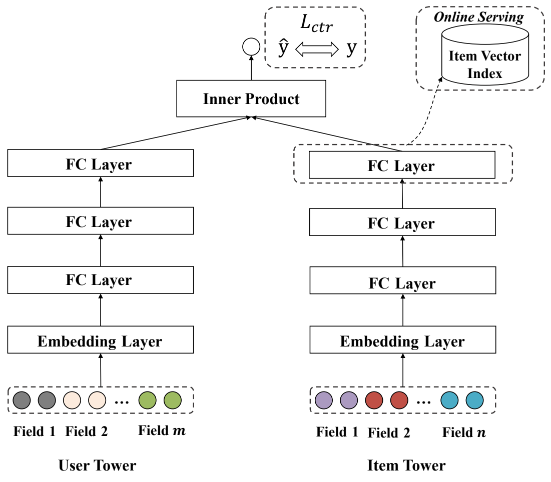


Figure 2: The overview of neural network based two-tower model.

Specifically, the dataset for training pre-ranking models consists of instances $(\mathbf{x}, y)$, where $y \in \{0, 1\}$ indicates the user-item feedback label [25] that means positive signal when equals 1 and negative when it is 0, features $\mathbf{x}$ can be divided into m user-related features and n item-related features, *i.e.*, $\mathbf{x} = [\underbrace{x_1, x_2, \ldots, x_m}_{\text{user-related}}; \underbrace{x_1, x_2, \ldots, x_n}_{\text{item-related}}]$. Then these user-related features and item-related features are fed into the corresponding parallel sub-towers (*i.e.*, user tower and item tower) to obtain user and item representations. Take the user tower as an example. The user-related features are first fed into the embedding layer to obtain user feature embeddings $\mathbf{e} = [\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_m]$ via embedding look-up operation [37], where $\mathbf{e}_i \in \mathbb{R}^d$ denotes the embedding of the i-th feature and d is the embedding dimension. Then the feature embeddings are further processed by multiple FC (fully connected) layers:

$$\mathbf{h}^{i+1} = relu(\mathbf{W}^i \mathbf{h}^i + \mathbf{b}^i), \tag{1}$$

where $\mathbf{W}^i \in \mathbb{R}^{d_{i+1} \times d_i}$, $\mathbf{b}^i \in \mathbb{R}^{d_i}$ are the weight and bias of the i-th FC layer, and the $\mathbf{h}^0 = \mathbf{e}$, d_i is the width of the i-th FC layer and $d_0 = d \times m$. Finally, the user representation can be obtained by: $\mathbf{h}_u = L2Norm(\mathbf{h}^L)$, where L is the depth of FC layers. Similarly, the item representation $\mathbf{h}_v$ can learned via the item tower. Finally, the output of the two-tower model $\hat{y}$ is the inner product of the user representation and item representation:

$$\hat{y} = \mathbf{h}_u^T \mathbf{h}_v. \tag{2}$$

Obviously, the two-tower model adopts "user-item decoupling architecture" designing paradigm, which is more efficient in terms of inference latency and computational resources. The item representations $\mathbf{h}_i$ can be periodically pre-calculated offline and stored in the fast retrieval container. During the online serving stage, for each user request, the user representation can be obtained by one online inference, and the k candidate item representations can be gathered from the retrieval container for prediction (the scale of k is thousands). Therefore, the overall time complexity is $O(N + kM)$, which is smaller than the single-tower structure with $O(kN)$ complexity, where N is the cost of a neural network inference and M is the cost of a vector retrieval and score ($N \gg M$). However, the disadvantage of two-tower model is also apparent. The user/item representations do not interact until the prediction layer, which is referred to "Late Interaction", hindering the model performance critically. To enhance the performance of two-tower model, we propose IntTower, which provides fine-grained and early interaction modeling and maintains excellent efficiency.

4 INTTOWER: INTERACTION ENHANCED TWO-TOWER

In this section, we will give a detailed description of the proposed pre-ranking model IntTower. The core idea behind IntTower is to utilize fine-grained and early feature interactions to improve the performance of the two-tower model. Remarkably, IntTower considers the trade-off between prediction accuracy and inference efficiency, which is vital for the pre-ranking system.

Figure 3 depicts the overall architecture of IntTower. Based on the original Two-Tower model, IntTower introduces three module: Light-SE (Lightweight SENET), FE Block (Fine-grained and Early Feature Interaction Block), and CIR (Contrastive Interaction Regularization). Firstly, for both user tower and item tower, a lightweight Light-SE module is leveraged to identify the importance of different features and obtain refined feature representations. Then, FE-Block module and CIR module are proposed to capture the interactive signals between user and item representations, from the **explicit** and **implicit** perspectives, respectively. Specifically, FE-Block performs fine-grained feature interactions between user and item representation explicitly, and CIR module implements a contrastive interaction regularization to further enhance the interactions implicitly.

4.1 Light Feature Attention Mechanism

For the recommender system, the importance of different features for prediction is diverse. For example, in the e-commerce scenario, it is apparent that `user consumption level` is more important than `gender`. Therefore, it is essential to automatically assign weights for different features to obtain refined feature representations, thus facilitating the subsequent feature interactions. Some previous works, like COLD [32] and FiBiNET [10], use SENET (Squeeze-and-Excitation Net) [8] to identify the feature importance. However, for pre-ranking models, introducing SENET will undoubtedly increase the online serving latency and aggravate the system pressure. Thus, We propose a lightweight version of SENET, named Light-SE with fewer parameters and lower latency. As shown in Figure 4, like SENET, Light-SE also consists of three steps: *Squeeze*, *Excitation* and *Reweighing* step.

Squeeze. This step is used to extract the "statistical information" of feature embedding. Specifically, we use the mean pooling operation to squeeze original user/item embeddings $\mathbf{e} = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_m]$ into vector $\mathbf{z} = [z_1, z_2, \ldots, z_m]$, where z_i is a scalar value that reflects the global statistic information of the i-th user feature, shown as:

$$z_i = f_{sq}(\mathbf{e}_i) = \frac{1}{d} \sum_{t=1}^{d} \mathbf{e}_i^d. \tag{3}$$

Excitation and Re-Weight. The next step is to learn the importance of each feature based on the statistical vector $\mathbf{z}$. Previous works [1, 34] has shown that using two FC layers of SENET is not necessary. Thus, here we use a more lightweight approach with a single FC layer to obtain the feature importance. Besides, we replace the Sigmoid with Softmax function to take the relative relationship of different feature importances into consideration comprehensively. Formally, the weight vector $\mathbf{k} \in \mathbb{R}^m$ can be calculated as follows:

$$\mathbf{k} = f_{ex}(\mathbf{z}) = softmax(\mathbf{W}\mathbf{z} + \mathbf{b}), \tag{4}$$

where $\mathbf{W} \in \mathbb{R}^{m \times m}$ and $\mathbf{b} \in \mathbb{R}^m$ are the transform matrix and bias vector, respectively. The last step is the re-weight step, which does the multiplication between the original feature embedding $\mathbf{e}$ and the weight vector $\mathbf{k}$, and outputs the refined user feature embedding $\tilde{\mathbf{e}}$, which can be calculated as follows:

$$\tilde{\mathbf{e}} = f_{rw}(\mathbf{k}, \mathbf{e}) = [k_1 \cdot \mathbf{e}_1, k_2 \cdot \mathbf{e}_2, \ldots, k_m \cdot \mathbf{e}_m]. \tag{5}$$

Based on the refined user and item feature representations, the user and item tower can better capture feature interactions.

4.2 Fine-grained and Early Feature Interaction

As mentioned in Section 3, the two-tower model performs "*Late Interaction*" modeling and the user/item representations do not interact until the prediction layer (E.q.(2)). This approach reduces the online inference latency, yet makes compromise on accuracy. However, effectively modeling feature interactions contributes to improving prediction accuracy [2, 38]. Inspired by the ColBERT [36], we propose FE-Block with "*Early Interaction*", which improves the performance of the two-tower model with fine-grained interactions and is friendly enough for computing power costs. FE-Block consists of multiple layers of interaction modeling, each of which contains two steps: *Projection* and *Interaction* step. It's worth noting that, in order to keep the "user-item decoupling architecture" and save a single vector for each item thus accelerating the online inference procedure, FE-Block performs fine-grained feature interactions between multi-layer user representations and last-layer item representation.

Projection. This step is to map the user/item representations into the latent space and extract more informative knowledge. As Figure 3 shows, FE-Block extracts the latent information from each FC layer for user tower, while the last FC layer only for item tower. For user/item tower, each representation is transformed into H head

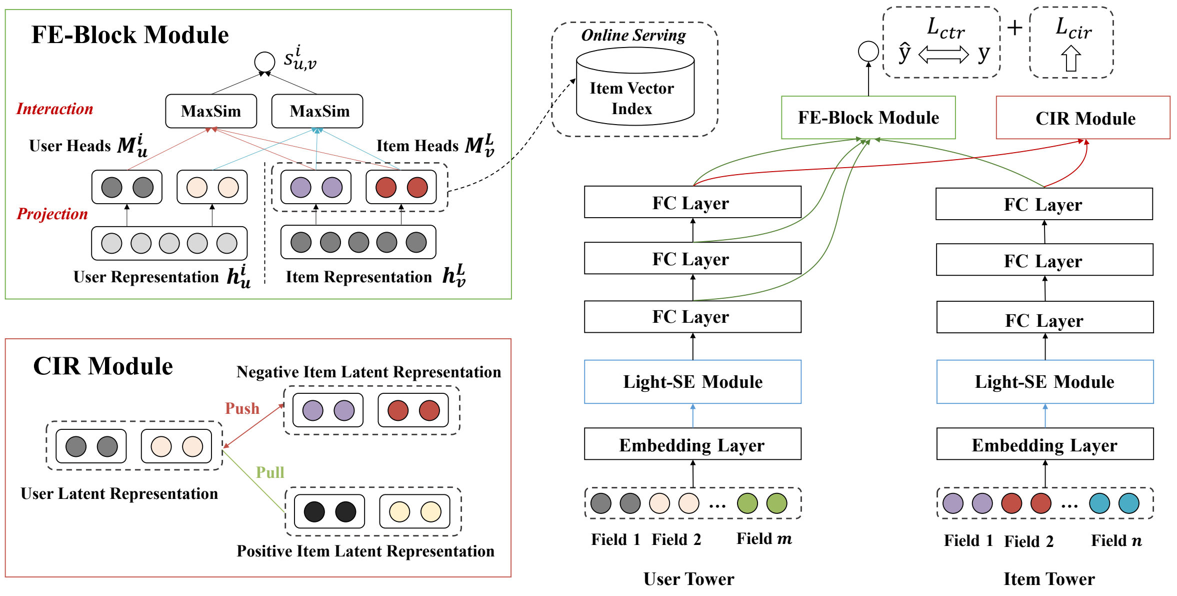


Figure 3: The architecture of IntTower.

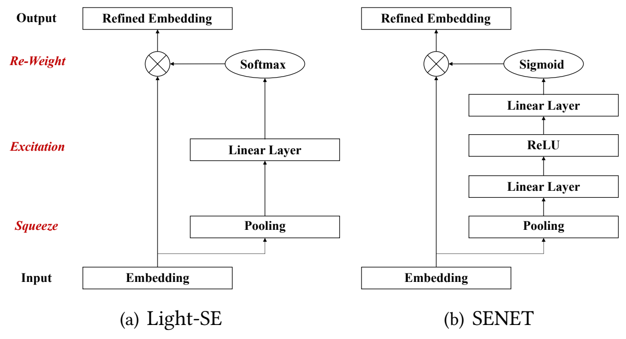


Figure 4: The comparison of Light-SE and SENET.

sub-spaces to extract information from different aspects. Specifically, the i-th layer user representation $\mathbf{h}^i_u$ is mapped into H sub-spaces and the h-th sub-space is represented as:

$$\mathbf{m}^{i,h}_u = \mathbf{W}^{i,h}_u \mathbf{h}^i_u + \mathbf{b}^{i,h}_u, \qquad h = 1, 2, \ldots, H \tag{6}$$

where $\mathbf{W}^{i,h}_u \in \mathbb{R}^{d_i \times p}$ and $\mathbf{b}^{i,h}_u \in \mathbb{R}^p$ are the transform matrix and bias vector for the h-head sub-space in the i-th user layer respectively, p is the dimension of sub-space. $\mathbf{m}^{i,h}_u \in \mathbb{R}^p$ is the i-th layer user latent representation under the h-th sub-space. It is noteworthy that, the H head transformation can be concatenated for parallel computing and the result is $\mathbf{M}^i_u \in \mathbb{R}^{pH}$.

Similarly, the last layer item latent representation under the h-th sub-space can be represented as:

$$\mathbf{m}^{L,h}_v = \mathbf{W}^{L,h}_v \mathbf{h}^L_v + \mathbf{b}^{L,h}_v, \qquad h = 1, 2, \ldots, H \tag{7}$$

and the concatenated result is presented as $\mathbf{M}^L_v \in \mathbb{R}^{pH}$.

Interaction. After extracting informative knowledge from different sub-space, FE-Block applies fine-grained feature interactions to compute the relevance score $S^i_{u,v}$ between the i-th layer user latent representation and the last layer item latent representation. The relevance score $S^i_{u,v}$ is conducted as a sum of maximum similarity, calculated as follows:

$$S^i_{u,v} = \sum_{h_u=1}^{H} \max_{h_v \in 1,2,\ldots,H} \{(\mathbf{m}^{i,h_u}_u)^T \mathbf{m}^{L,h_v}_v\}. \tag{8}$$

Note that the interaction mechanism in this step does not have any extra parameters, which is suitable for online serving. Finally, we calculate and sum the relevance score for each user layer to capture comprehensive multi-layer interactive signals:

$$\hat{y} = \sum_{i=1}^{L} S^i_{u,v}. \tag{9}$$

In short, FE-Block module implements fine-grained and early user-item feature interaction explicitly, which improves the expression capability of the two-tower model significantly. More importantly, the "user-item decoupling architecture" paradigm can be maintained, ensuring high inference efficiency. The multi-head item latent representation $\mathbf{M}^L_v$ can be pre-stored for online serving. Besides, FE-Block replaces the original cosine similarity scoring approach (E.q.(2)) with fine-grained sum of maximum cosine similarity (E.q.(8-9)), which is parameter-free and easy to deploy in the pre-ranking system.

4.3 Contrastive Interaction Regularization

FE-Block enhances the feature interactions of two-tower model via an explicit approach. In this subsection, we propose a contrastive interaction regularization module by applying contrastive learning to implicitly strengthen the information interaction between the two towers. In FE-Block, the multi-head latent representation $\mathbf{M}^L_u$ and $\mathbf{M}^L_v$ are the final user and item representation, respectively. To

Table 1: Statistics of datasets

Dataset	Users	Items	User Field	Item Field	Samples
MovieLens-1M	6,040	3,952	5	3	1,000,000
Amazon(Electro)	192,403	630,001	2	4	1,689,188
Alibaba	1,061,768	785,597	9	6	26,557,961

further strengthen interaction regularization modeling, we introduce self-supervised signals to shorten the distance between user and positive items.

Hence, we leverage the positive instances to construct positive pairs $(\mathbf{M}_u^L, \mathbf{M}_v^L)$, and $(\mathbf{M}_u^L, \mathbf{M}_{v'}^L)$ are negative pairs when $v \neq v'$. Following previous work [3], we employ InfoNCE [7] to maximize the consistency of positive pairs and minimize the consistency of negative pairs:

$$\mathcal{L}_{cir} = -\frac{1}{Q} \sum_{(u,v)\in Q} log \frac{exp(sim(\mathbf{M}_u^L, \mathbf{M}_v^L)/\tau)}{\sum\limits_{(u',v')\in N} exp(sim(\mathbf{M}_u^L, \mathbf{M}_{v'}^L)/\tau)}, \quad (10)$$

where τ is a temperature factor and $sim(\cdot)$ measures the similarity between two vectors, which is calculated by $sim(\mathbf{M}_u^L, \mathbf{M}_v^L) = (\mathbf{M}_u^L)^T \mathbf{M}_v^L / (\|\mathbf{M}_u^L\| \cdot \|\mathbf{M}_v^L\|)$. Q is the total number of positive instances and N is the total number of samples. By introducing contrastive interaction regularization, the information interaction between user and positive items can be effectively modeled.

4.4 Training

We use *Logloss* to calculate the loss between the model prediction scores and the true labels, which is defined as follows:

$$\mathcal{L}_{ctr} = -\frac{1}{N} \sum_{i=1}^{N} (y_i log(\hat{y_i}) + (1 - y_i) log(1 - \hat{y_i})), \quad (11)$$

where y_i and $\hat{y_i}$ are the ground truth of i-th instance and score $\hat{y_i}$.

The ultimate purpose of the learning process is to minimize $\mathcal{L}_{ctr}$ and $\mathcal{L}_{cir}$, the final loss function is formulated as:

$$\mathcal{L} = \mathcal{L}_{ctr} + \lambda_1 \mathcal{L}_{cir} + \lambda_2 \|\Theta\|_2, \quad (12)$$

where λ_1, λ_2 are hyper-parameters and Θ is the model parameters.

5 EXPERIMENTS

In this section, we describe experiments in detail, including datasets, evaluation metrics, comparisons with SOTA baseline models and corresponding analyses. The experiment results on three public large-scale datasets demonstrate the effectiveness of IntTower on the task of CTR prediction. The code of IntTower is publicly available[1].

5.1 Experimental Setting

5.1.1 Datasets. We use three public available real-world datasets, whose statistics are summarized in Table 1. According to previous work [10, 26], we split all samples randomly into two parts: 80% for training and the rest for test.

5.1.2 Evaluation Metrics. Following previous work [10, 26, 39], we use three popular metrics to evaluate the performance . **Logloss** Logloss is a widely used metric in binary classification to measure the distance between two distributions. A lower bound of 0 for Logloss indicates that the two distributions are perfectly matched, and a smaller value indicates a better performance. **AUC** The area under the ROC curve (AUC) measures the probability that the model will assign a higher score to a randomly selected positive item than to a randomly selected negative item. **RelaImpr** RelaImpr metric [39] measures the relative improvement with respect to models, which is defined as follows:

$$RelaImpr = (\frac{AUC(measure\ model) - 0.5}{AUC(base\ model) - 0.5} - 1) \times 100\% \quad (13)$$

5.1.3 Competing Models. We compare IntTower with the following models, and for fairness, we classify the compared models into two classes. (A) Pre-Ranking Models. (B) Ranking Models. The pre-ranking models include **LR** [19], **Two-Tower** [9], **DAT** [35], **COLD** [32], and the ranking models include **Wide&Deep** [4], **DeepFM** [6], **DCN** [30], **AutoInt** [26].

5.1.4 Implementation Details. We implement the proposed IntTower and other compared baselines with Pytorch [21] and empirically set the feature embedding dimension d to 32, and the batch size to 2048. Besides, for all deep models, we set the number of hidden layers L as 3 and the number of hidden units as $[300, 300, 128]$. For IntTower, we use three FE-Blocks by default, where the number of sub-spaces is equal to the number of features in the user tower, *i.e.*, $H = m$ and the dimension of each sub-space (head size) is set to 64. All models are trained by the Adam [13] optimizer with the initial learning rate setting to 0.001. Batch Normalization [11] and Dropout [27] is applied to avoid overfitting. To ensure a fair comparison, other hyper-parameters such as training epochs are adjusted individually for all models to obtain the best results. We conduct experiments with 2 Tesla 3090 GPUs.

5.2 Result of Model Accuracy

We compare the overall performance with some SOTA pre-ranking and ranking models, results are summarized in Table 2. Generally speaking, the prediction accuracy of ranking models is much higher than that of the pre-ranking models by a large margin [18, 32]. To fully demonstrate the superior prediction accuracy of IntTower, we compare it with some ranking models and are delighted to find that the performance is *comparable* and even *exceeded* in some cases. Noted that, for the concern of fairness, we repeat each experiment 10 times to obtain the best results. From which, we obtain the following observations: (1) Among the pre-ranking models, LR achieves the worst performance in comparison with the neural network-based models. Two-Tower leverages the FC layers to capture the feature interactions within the user/item towers independently. Though DAT implicitly models the information interaction between the two towers, the performance gains are limited. COLD with single-tower structure outperforms other pre-ranking models while the increased inference burden is also noticeable. (2) Compared with the pre-ranking models, ranking models (*e.g.*, DeepFM and AutoInt) achieve significant improvement by a large margin. Ranking models with single-tower structure attribute to capturing higher-order feature interactions between user and item, which is vital for CTR prediction. However, these models cannot be deployed in the pre-ranking system without massive engineering optimization. Therefore, they are more suitable for serving in the ranking

Table 2: Performance comparison of different models. The base model of RelaImpr is Two-Tower, which is a popular neural network-based pre-ranking model. Boldface denotes the highest score and underline indicates the best result of the pre-ranking baselines. ⋆ represents significance level p-value < 0.05 of comparing IntTower with the best pre-ranking baselines.

Stage	Model	MovieLens			Amazon			Alibaba		
		AUC	Logloss	RelaImpr	AUC	Logloss	RelaImpr	AUC	Logloss	RelaImpr
Pre-Ranking	LR	0.8501	0.4845	-5.30%	0.7952	0.4677	-14.9%	0.6496	0.2350	-5.85%
	Two-Tower	0.8697	0.4559	0%	0.8469	0.4313	0%	0.6579	0.2292	0%
	DAT	0.8712	0.4556	0.40%	0.8480	0.4278	0.31%	0.6598	0.2279	0.56%
	COLD	$\underline{0.8836}$	$\underline{0.3297}$	3.75%	$\underline{0.8633}$	$\underline{0.3402}$	4.72%	$\underline{0.6816}$	$\underline{0.2248}$	14.28%
Ranking*	Wide&Deep	0.8820	0.3344	3.32%	0.8615	0.3409	4.20%	0.6814	0.2250	14.15%
	DeepFM	0.8920	0.3211	6.03%	0.8643	0.3405	5.05%	0.6820	0.2247	14.53%
	DCN	0.8964	0.3151	7.22%	0.8665	0.3366	5.65%	0.6831	0.2244	15.22%
	AutoInt	0.8948	0.3192	6.79%	0.8686	0.3351	6.25%	**0.6867**	**0.2238**	**17.49%**
	IntTower (ours)	**0.8974***	**0.3128***	**7.49%**	**0.8696***	**0.3309***	**7.91%**	0.6827*	0.2245*	14.97%

* **Generally speaking, the prediction accuracy of ranking models is much higher than that of the pre-ranking models by a large margin [18, 32]. To fully demonstrate the superior prediction accuracy of IntTower, we compare it with some SOTA ranking models and are delighted to find that the performance is *comparable* and even *exceeded* in some cases.**

stage with fewer candidates. (3) IntTower outperforms SOTA pre-ranking models by a large margin and even some ranking models (*e.g.*, Wide& Deep and DeepFM), indicating superior prediction performance. Even in the face of industrial SOTA ranking models DCN and AutoInt, IntTower still achieves a comparable performance. We attribute this great improvement to the effective interaction modeling between user and item towers from the explicit and implicit perspectives. Moreover, IntTower keeps the "user-item decoupling architecture" paradigm, which is highly efficient , making our model suitable for the pre-ranking system.

5.3 Result of Model Efficiency

5.3.1 Training Efficiency and Model Size. For CTR prediction, the model training efficiency is crucial as it will affect the model update frequency and further affect the prediction accuracy. In this part, we calculate the parameters and running time of different models over the MovieLens and Amazon, shown in Table 3.

Not surprisingly, LR has the smallest parameters and lowest training time due to its shallow structure. Compared with the Two-Tower, the increased parameters and training time of IntTower is negligible. Besides, the single-tower structure models have more parameters and higher training time than IntTower. IntTower employs Light-SE feature attention mechanism and the FE-Block introduces only a few parameters to model fine-grained feature interaction.

5.3.2 Serving Efficiency. In industrial multi-stages cascade systems, each stage is subject to strict latency limits, *e.g.*, 10~20 milliseconds. In this case, pre-ranking system is typically designed as low-latency system because the size of the candidates is thousands for each user request, which is much more than that of the ranking stage. Therefore, high serving efficiency is indispensable for a pre-ranking model. In this subsection, we discuss the model serving efficiency.

In Figure 5, we show the inference latency of different models over different candidate sizes. For Two-Tower and IntTower, we store the obtained item representations into faiss [12], and during inference, we retrieve the corresponding representations from faiss for prediction. The potential of IntTower can be clearly illustrated in Figure 5. Compared with the Two-Tower with the highest efficiency, the increased latency of IntTower is acceptable given different candidate scales. Benefiting from the "user-item decoupling architecture" paradigm, the time complexity of both Two-Tower and IntTower is

Table 3: Training efficiency comparison of different models in terms of Model Size and Running Time in an epoch. For the model with two-tower structure , we only calculate the parameters of the user tower since only the user tower is required for online inference.

	MovieLens		Amazon	
Model	Params	Run Time	Params	Run Time
LR	5.24×10^4	11s	5.10×10^5	27s
Two-Tower	5.03×10^5	15s	6.26×10^6	35s
COLD	6.83×10^5	18s	8.58×10^6	42s
DCN	6.84×10^5	19s	8.58×10^6	43s
AutoInt	6.95×10^5	19s	8.59×10^6	42s
IntTower	5.67×10^5	16s	6.29×10^6	40s

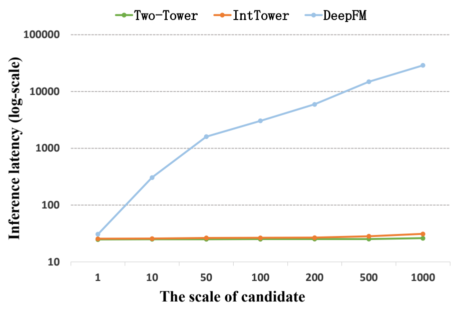
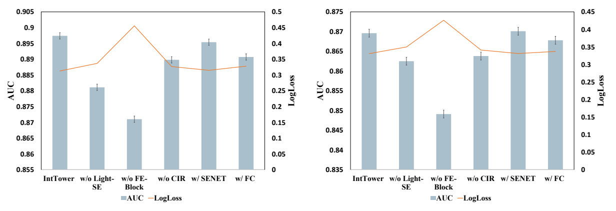


Figure 5: Inference latency for different candidate set sizes. We use MovieLens, the number of users is 1×10^5, test on a Tesla 3090 Gpu.

$O(N + kM)$, where k is the number of candidates, N is the cost of a neural network inference and M is the cost of a vector retrieval and score prediction. Therefore, IntTower has superior online serving efficiency and is suitable for pre-ranking systems. Instead, for the single-tower model (*e.g.*, DeepFM), each user-item pair require a neural network inference, resulting in $O(kN)$ complexity ($N \gg M$).

5.4 Ablation Study

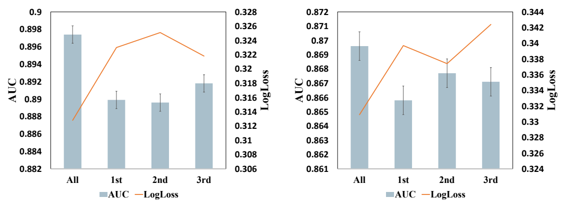


(a) MovieLens (b) Amazon

Figure 6: The impact of different components in IntTower.

5.4.1 *The Impact of Different Components.*

In this section, we conduct ablation experiments on the IntTower to better understand the importance of different components. We first remove the three modules of IntTower; then we replace the Light-SE with SENET, and replace the FE-Block with a FC layer via $FC(\mathbf{h}_u^i || \mathbf{h}_v^L)$. From Figure 6, we observe the following results: (1) Each component plays an active role in the performance of the IntTower, and the removal of any component will result in a degradation of performance. (2) FE-Block is the most important part of IntTower. Removing it will lead to a huge performance drop, which indicates that the explicit feature interaction modeling between user and item towers is statistically significant for two-tower. (3) After replacing Light-SE with SENET, the performance decreases on MovieLens and slightly improves AUC on Amazon. The above phenomena indicates that Light-SE has fully comparable performance to SENET but less parameters. (4) After replacing FE-Block with a FC layer, the performance decreases but still achieves great improvement compared with the two-tower model, demonstrating the importance of interaction modeling. However, FC layer is not friendly to online serving due to extra parameters. Instead, the *Interaction* step in FE-Block is parameter-free, which is suitable for online deployment.



(a) MovieLens (b) Amazon

Figure 7: The performance of different user tower layers.

5.4.2 *The Impact of FE-Block in Different Layers.*

In IntTower, we deploy multiple FE-Blocks to capture interactive signals between multiple user layers $\{\mathbf{h}_u^1, \mathbf{h}_u^1, \ldots, \mathbf{h}_u^L\}$ and the last item layer $\mathbf{h}_v^L$. In order to further study the influence of the user layer in FE-Block, we perform experiments over MovieLens and Amazon datasets by using a single FE-Block and summarize the experimental results in Figure 7. We can find some observations as follows: (1) Deploying a single FE-Block still achieves significant performance, demonstrating the effectiveness of the fine-grained feature interaction modeling. (2) Performing multiple FE-Blocks over multiple user layers can obtain significant improvement since different level feature interactions can be captured synthetically. Remarkably, the effect of only one FE-Block is also quite well, so we can reduce the number of FE-Blocks to further reduce online latency.

Table 4: Performance *w.r.t.* the number of heads in FE-Block.

User Head	Item Head	MovieLens		Amazon	
		AUC	Logloss	AUC	Logloss
1	1	0.8957	0.3160	0.8669	0.3513
1	2	0.8968	0.3144	0.8653	0.3397
1	4	**0.8994**	**0.3110**	0.8671	0.3368
1	8	0.8953	0.3215	0.8660	0.3366
1	16	0.8987	0.3126	0.8689	0.3328
1	32	0.8936	0.3184	**0.8694**	**0.3322**
2	2	**0.9016**	**0.3077**	0.8672	0.3356
4	4	0.8944	0.3215	0.8668	0.3379
8	8	0.8961	0.3158	0.8679	0.3344
16	16	0.8920	0.3217	**0.8705**	**0.3307**
32	32	0.8953	0.3166	0.8701	0.3309
64	64	0.8974	0.3128	0.8696	0.3309

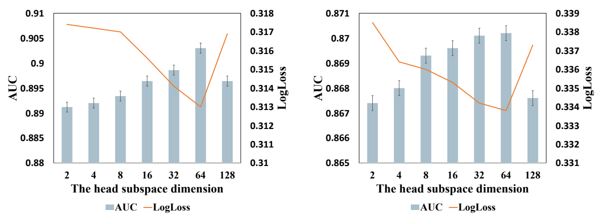
5.5 Hyper-parameter Study

5.5.1 *The impact of the number of heads.*

In order to investigate the effect of the number of heads, experiments are implemented over MovieLens and Amazon datasets and summarized in Table 4. From the results we can get the following observations: (1) When increasing the number of heads, the overall performance shows a trend of increasing first and then decreasing, indicating that increasing a certain number of heads helps boost performance. (2) For the MovieLens dataset, the optimal number of heads is achieved with a small number $\{2, 2\}$. We speculate the reason is that the model with more heads tend to overfit on the MovieLens dataset. Instead, the optimal number for Amazon dataset is much larger ($\{16, 16\}$).

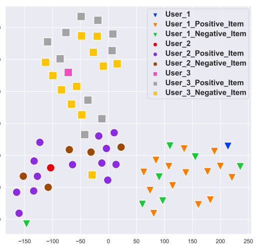
5.5.2 *The impact of the head sub-space dimension.*

Next, we investigate the impact of the sub-space dimensions in FE-Block and the results are depicted in Figure 8. We can observe that, as the sub-space dimension increases, the model performance increases first and then decreases. The optimal performance can be achieved when the sub-space dimension reaches 64. The reason is that enlarging the dimension attributes to capture informative interactions due to more parameters. However, more parameters also tend to overfit, thus degrading the performance.

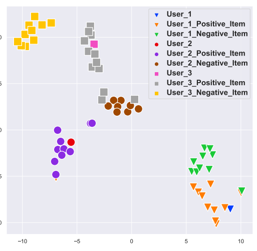


(a) MovieLens (b) Amazon

Figure 8: Performance *w.r.t.* the head subspace dimension .



(a) Two-Tower



(b) IntTower

Figure 9: Visualization of user and item representations. Points with the same shape represent a user and the corresponding items.

5.6 Visualization of IntTower

To deeply investigate the properties of original Two-Tower and IntTower, we visualize the user and item representations by projecting into a two-dimensional space with t-SNE [29]. Points with the same shape correspond to a user and the corresponding positive and negative items in Amazon Dataset. As shown in Figure 9, for Two-Tower, the positive and negative items are around the user and mixed together, which is difficult to distinguish. By contrast, for IntTower, user and positive items are almost in the same cluster, while negative items are far away from the user. This phenomenon demonstrates that IntTower can better infer users' preferences and distinguish between positive and negative items.

6 APPLICATION: ADVERTISEMENT PRE-RANKING SYSTEM

In this section, we deploy IntTower in a large-scale industrial pre-ranking system to validate its effectiveness.

6.1 System Overview

The overview of pre-ranking system in a large-scale advertisement platform is presented in Figure 10. When a user request arrives, it triggers the recall, pre-ranking, and ranking recommendation pipeline. Pre-ranking system scores the candidate ads retrieved from the recall stage and generates top-k high-quality ads for the subsequent ranking stage. Specifically, pre-ranking system consists of two core modules: *Offline Training* and *Online Serving*.

Offline training is periodically performed over the latest user behavior logs to update the model parameters. Take the two-tower model as an example, features are divided into user-related x_{user} and ad-related x_{ad}, which are fed into user tower and ad tower for training, respectively. Upon the finish of model training, the user tower model is deployed on the online server for prediction while the ad tower model is utilized to infer ads representation vectors $\mathbf{e}_{ad}$, which will be stored in the AD Vectors Index Server for the online serving. During the online serving phase, a user request will trigger user and contextual feature gathering, and these features will be concatenated into a user feature vector x_{user}. Then, feature vector x_{user} is fed into the user tower model for inferring user representation $\mathbf{e}_{user}$. A candidate ad ID list returned from the recall stage is used to retrieve a group of ad representations $\{\mathbf{e}_{ad\ 1}, \mathbf{e}_{ad\ 2}, \dots, \mathbf{e}_{ad\ k}\}$ from the AD Vectors Index Server. Finally, the online scoring server performs k prediction to obtain predicted scores $Pr(y_i|\mathbf{e}_{user}, \mathbf{e}_{ad\ i})$ for each candidate ad i, and a sorted ad ID list is organized according to the pre-defined ranking function (*e.g.*, eCPM) before sending to the ranking stage.

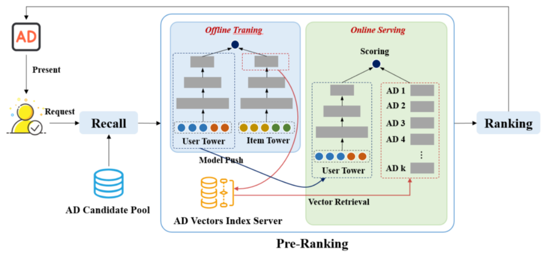


Figure 10: Overview of pre-ranking model in advertisement system.

Table 5: The performance comparison in a advertisement system.

Stage	Model	AUC	Logloss	RelaImpr
Pre-Ranking	Two-Tower	0.7550	0.0536	0%
	IntTower	0.7603	0.0523	2.08%
Ranking	DeepFM	0.7591	0.0525	1.61%
	EDCN	0.7612	0.0518	2.43%

6.2 Dataset and Compared Models

We collect and sample 8 consecutive days of user behavior records from a large-scale advertisement platform, where millions of daily active users interact with advertisements and tens of millions of user logs are generated. The user and contextual related features x_{user} include user profile features (*e.g.*, gender), user behavior features (*e.g.*, list of ads clicked by users), user statistics features (*e.g.*, number of ads clicked by users), as well as contextual features (*e.g.*, time). Additionally, the ad related features x_{ad} include ad task features (*e.g.*, task id) and ad statistics features (*e.g.*, number of clicks on the ad). For the categorical features, the embeddings are obtained by embedding look-up, while the numerical feature embeddings are learned via the AutoDis [5].

We compare IntTower with pre-ranking and ranking models, respectively. For the pre-ranking model, we choose a highly-optimized DNN-based Two-Tower model. For the ranking model, we choose DeepFM [6] and EDCN [2], which are deployed in the large-scale advertisement business system.

6.3 Performance Comparison

The performance comparison on the large-scale advertising dataset is presented in Table 5, where we can observe that IntTower outperforms Two-Tower model by a significant margin in terms of AUC and Logloss, even surpasses ranking model DeepFM and achieving a performance close to EDCN. By enhancing the information interaction between user and item towers, IntTower achieves significant performance improvement compared with the original Two-Tower model. Moreover, as presented in Figure 5, the inference latency of IntTower is several orders of magnitude lower than the single-tower structure models. Therefore, IntTower has comparable prediction accuracy while higher inference efficiency than the ranking models, which fits perfectly with industrial pre-ranking systems.

7 CONCLUSION

In this paper, we propose the Interaction enhanced Two-Tower model (IntTower) to boost the performance of two-tower model by capturing information interaction between user and item towers and preserving high inference efficiency with "user-item decoupling architecture" paradigm. The IntTower includes Light-SE, FE-Block and CIR modules. Specifically, Light-SE module can identify the importance of different features and obtain refined feature representations. FE-Block module performs fine-grained and early feature interactions to capture the interactive signals between user and item towers explicitly while CIR module leverages a contrastive interaction regularization to further enhance the interactions implicitly. Comprehensive experiments show that IntTower outperforms the SOTA pre-ranking models significantly. Moreover, we further verify the effectiveness of IntTower on a large-scale advertisement pre-ranking system.

REFERENCES

[1] Yue Cao, Jiarui Xu, Stephen Lin, Fangyun Wei, and Han Hu. 2019. Gcnet: Non-local networks meet squeeze-excitation networks and beyond. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*. 0–0.

[2] Bo Chen, Yichao Wang, Zhirong Liu, Ruiming Tang, Wei Guo, Hongkun Zheng, Weiwei Yao, Muyu Zhang, and Xiuqiang He. 2021. Enhancing Explicit and Implicit Feature Interactions via Information Sharing for Parallel Deep CTR Models. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 3757–3766.

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of ICML*. PMLR, 1597–1607.

[4] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*. 7–10.

[5] Huifeng Guo, Bo Chen, Ruiming Tang, Weinan Zhang, Zhenguo Li, and Xiuqiang He. 2021. An embedding learning framework for numerical features in ctr prediction. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2910–2918.

[6] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. *arXiv preprint arXiv:1703.04247* (2017).

[7] Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of AISTATS*. JMLR Workshop and Conference Proceedings, 297–304.

[8] Jie Hu, Li Shen, and Gang Sun. 2018. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 7132–7141.

[9] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*. 2333–2338.

[10] Tongwen Huang, Zhiqi Zhang, and Junlin Zhang. 2019. FiBiNET: combining feature importance and bilinear feature interaction for click-through rate prediction. In *Proceedings of the 13th ACM Conference on Recommender Systems*. 169–177.

[11] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*. PMLR, 448–456.

[12] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data* 7, 3 (2019), 535–547.

[13] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. https://doi.org/10.48550/ARXIV.1412.6980

[14] Xiangyang Li, Xiang Long, Yu Xia, and Sujian Li. 2022. Low Resource Style Transfer via Domain Adaptive Meta Learning. https://doi.org/10.48550/ARXIV.2205.12475

[15] Xiangyang Li, Yu Xia, Xiang Long, Zheng Li, and Sujian Li. 2021. Exploring Text-transformers in AAAI 2021 Shared Task: COVID-19 Fake News Detection in English. (2021). https://doi.org/10.48550/ARXIV.2101.02359

[16] Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. 2018. xdeepfm: Combining explicit and implicit feature interactions for recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 1754–1763.

[17] Bin Liu, Ruiming Tang, Yingzhi Chen, Jinkai Yu, Huifeng Guo, and Yuzhou Zhang. 2019. Feature generation by convolutional neural network for click-through rate prediction. In *The World Wide Web Conference*. 1119–1129.

[18] Xu Ma, Pengjie Wang, Hui Zhao, Shaoguo Liu, Chuhan Zhao, Wei Lin, Kuang-Chih Lee, Jian Xu, and Bo Zheng. 2021. Towards a Better Tradeoff between Effectiveness and Efficiency in Pre-Ranking: A Learnable Feature Selection based Approach. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2036–2040.

[19] H Brendan McMahan, Gary Holt, David Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, et al. 2013. Ad click prediction: a view from the trenches. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1222–1230.

[20] Dongchen Miao and Fei Lang. 2017. A recommendation system based on text mining. In *2017 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*. IEEE, 318–321.

[21] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.). Curran Associates, Inc., 8024–8035. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[22] Jiarui Qin, Jiachen Zhu, Bo Chen, Zhirong Liu, Weiwen Liu, Ruiming Tang, Rui Zhang, Yong Yu, and Weinan Zhang. 2022. RankFlow: Joint Optimization of Multi-Stage Cascade Ranking Systems as Flows. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[23] Yanru Qu, Han Cai, Kan Ren, Weinan Zhang, Yong Yu, Ying Wen, and Jun Wang. 2016. Product-based neural networks for user response prediction. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 1149–1154.

[24] Steffen Rendle. 2010. Factorization machines. In *2010 IEEE International conference on data mining*. IEEE, 995–1000.

[25] Matthew Richardson, Ewa Dominowska, and Robert Ragno. 2007. Predicting clicks: estimating the click-through rate for new ads. In *Proceedings of the 16th international conference on World Wide Web*. 521–530.

[26] Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. 2019. Autoint: Automatic feature interaction learning via self-attentive neural networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 1161–1170.

[27] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15, 1 (2014), 1929–1958.

[28] Nguyen Thai-Nghe, Lucas Drumond, Artus Krohn-Grimberghe, and Lars Schmidt-Thieme. 2010. Recommender system for predicting student performance. *Procedia Computer Science* 1, 2 (2010), 2811–2819.

[29] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, 11 (2008).

[30] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & cross network for ad click predictions. In *Proceedings of the ADKDD'17*. 1–7.

[31] Yaozheng Wang, Dawei Feng, Dongsheng Li, Xinyuan Chen, Yunxiang Zhao, and Xin Niu. 2016. A mobile recommendation system based on logistic regression and gradient boosting decision trees. In *2016 international joint conference on neural networks (IJCNN)*. IEEE, 1896–1902.

[32] Zhe Wang, Liqin Zhao, Biye Jiang, Guorui Zhou, Xiaoqiang Zhu, and Kun Gai. 2020. Cold: Towards the next generation of pre-ranking system. *arXiv preprint arXiv:2007.16122* (2020).

[33] Xin Xin, Bo Chen, Xiangnan He, Dong Wang, Yue Ding, and Joemon M Jose. 2019. CFM: Convolutional Factorization Machines for Context-Aware Recommendation.. In *IJCAI*, Vol. 19. 3926–3932.

[34] Haolan Xue, Chang Liu, Fang Wan, Jianbin Jiao, Xiangyang Ji, and Qixiang Ye. 2019. Danet: Divergent activation for weakly supervised object localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 6589–6598.

[35] Yantao Yu, Weipeng Wang, Zhoutian Feng, and Daiyue Xue. 2021. A Dual Augmented Two-tower Model for Online Large-scale Recommendation. (2021).

[36] Matei Zaharia. 2020. ColBERT: E icient and E ective Passage Search via Contextualized Late Interaction over BERT. (2020).

[37] Weinan Zhang, Tianming Du, and Jun Wang. 2016. Deep learning over multi-field categorical data. In *European conference on information retrieval*. Springer, 45–57.

[38] Weinan Zhang, Jiarui Qin, Wei Guo, Ruiming Tang, and Xiuqiang He. 2021. Deep Learning for Click-Through Rate Estimation. In *Proceedings of the International Joint Conference on Artificial Intelligence*. 4695–4703.

[39] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 1059–1068.