1^{st} Place Solution of WWW 2025 EReL@MIR Workshop Multimodal CTR Prediction Challenge

Junwei Xu
Shenzhen Key Laboratory of Ubiquitous Data Enabling,
Shenzhen International Graduate School
Tsinghua University
Shenzhen, China
xjw23@mails.tsinghua.edu.cn

Zehao Zhao
Shenzhen International Graduate School
Tsinghua University
Shenzhen, China
zhaozeha23@mails.tsinghua.edu.cn

Xiaoyu Hu
Shenzhen International Graduate School
Tsinghua University
Shenzhen, China
huxiaoyu23@mails.tsinghua.edu.cn

Zhenjie Song
Shenzhen International Graduate School
Tsinghua University
Shenzhen, China
songzj23@mails.tsinghua.edu.cn

Abstract

The WWW 2025 EReL@MIR Workshop Multimodal CTR Prediction Challenge[1] focuses on effectively applying multimodal embedding features to improve click-through rate (CTR) prediction in recommender systems. This technical report presents our 1^{st} place winning solution for Task 2, combining sequential modeling and feature interaction learning to effectively capture user-item interactions. For multimodal information integration, we simply append the frozen multimodal embeddings to each item embedding. Experiments on the challenge dataset demonstrate the effectiveness of our method, achieving superior performance with a 0.9839 AUC on the leaderboard, much higher than the baseline model. Code and configuration are available in our GitHub repository[2] and the checkpoint of our model can be found in HuggingFace[3].

CCS Concepts

• **Information systems → Information retrieval**; • **Computing methodologies → Artificial intelligence**.

Keywords

Click-Through Rate Prediction; Sequential Recommendation; Multimodality; Representation Learning

1 Introduction

Click-through rate (CTR) prediction is a fundamental task in online advertising and recommendation systems, aiming to estimate the probability of users clicking on specific ads or content items. The accuracy of CTR prediction directly affects the revenue of advertising platforms and the user experience. Traditional CTR models predominantly rely on structured tabular data (*e.g.*, user demographics, historical behaviors, and item attributes), using feature interaction paradigms like factorization machines (FMs) [3–5, 7, 10] or advanced deep learning techniques [6, 11–14]. However, the exponential growth of multimodal content (*e.g.*, video covers, content title and audio clips) has become increasingly available in recommender systems, necessitating the development of advanced methods that can effectively leverage the multimodal information.

The Multimodal CTR Prediction (MM-CTR) Challenge at the WWW 2025 EReL@MIR Workshop[2] aims to foster innovation in adopting multimodal embedding features for CTR prediction. In response to industry requirements for low-latency and online inference, the challenge comprises two complementary sub-tasks: Multimodal Item Representation Learning and Multimodal CTR Modeling. The first sub-task focuses on learning multimodal item representations optimized for recommendation scenarios, while the second emphasizes effectively leveraging these frozen multimodal embeddings to further enhance the performance of the CTR model. We participated in the second sub-task, which can be divided into two parts: better utilization of frozen multi-modal embedding features and better CTR prediction modeling.

Although injecting multimodal semantic information into the CTR model was explored, simple concatenation of multimodal embeddings with item embeddings is adopted in our final solution. This was due to the limited time, and we were unable to finish the model optimization and parameter tuning work of the multimodal embeddings part. Inspired by [13], we adopt Transformer for sequential modeling and DCNv2[12] for feature interaction learning. Through extensive parameter tuning, the optimal hyperparameter settings were obtained on the challenge dataset. We achieved the 1^{st} place on the final leaderboard with an AUC score of 0.9839, demonstrating the effectiveness of our method.

2 Methods

2.1 Problem Formulation

Given a set of samples $\mathcal{D} = \{(\mathcal{H}_u, x_{target}, y) | u \in \mathcal{U}, x_{target} \in \mathcal{I}, y \in \{0, 1\})\}$, where $\mathcal{U}$ and $\mathcal{I}$ are the user set and item set. $\mathcal{H}_u = \{x_1, x_2, \ldots, x_N\}$ is the historical interaction sequence of user u, and x_i is the item features (*e.g.*, item ID, item tags and item multimodal embedding) of the i-th clicked item in the user history. y is a binary label indicating whether the user clicked on the target item or not:

$$y = f\left(\mathcal{H}_u, x_{target} | \mathcal{D}, \Theta\right), \tag{1}$$

where Θ is the model parameter.

[1] https://erel-mir.github.io/challenge
[2] https://github.com/pinskyrobin/WWW2025_MMCTR
[3] https://huggingface.co/pinskyrobin/WWW2025_MMCTR_momo

2.2 Network Structure

Inspired by [13], sequential modeling and feature interaction learning are combined to effectively capture user interest preferences. As shown in Figure 1, it consists of four main components: Embedding Layer, Sequential Feature Learning Module, Feature Interaction Module, and Prediction Layer.

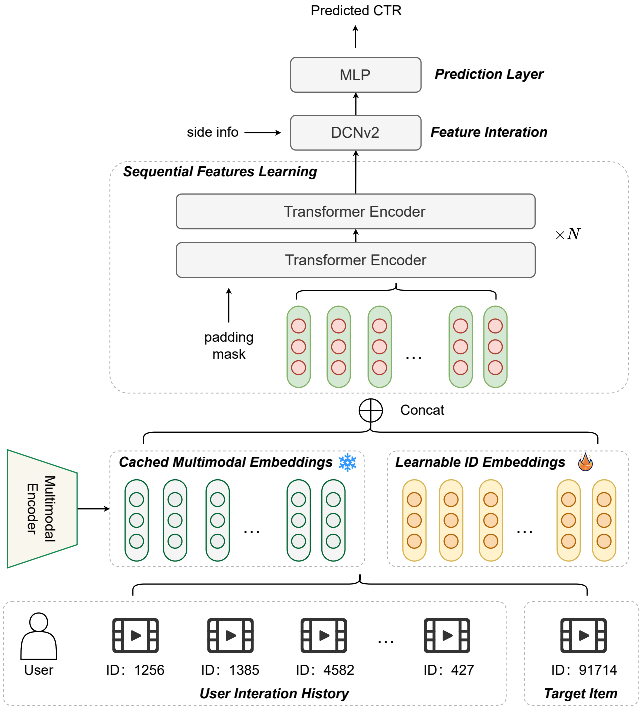

Figure 1: The overall architecture of our model.

2.2.1 Embedding Layer. The Embedding Layer converts different types of input features into dense vector representations. Let $x = \{t_1, t_2, \ldots, t_{|T|}, e_{mm}\}$ be the item features, where t_i is the i-th feature of the item, $|T|$ is the number of features and e_{mm} is the frozen multimodal embedding of the item. The embedding layer maps each feature t_i to a dense vector e_i using an embedding matrix:

$$e_{t_i} = E(t_i) \in \mathbb{R}^{d_e}, \tag{2}$$

where d_e is the embedding dimension. The frozen multimodal embedding e_{mm} is then concatenated with the item embeddings to form the final item representation:

$$e^i_{item} = \left[e^i_{t_1} \parallel e^i_{t_2} \parallel \ldots \parallel e^i_{t_{|T|}} \parallel e^i_{mm}\right], \tag{3}$$

where $\parallel$ denotes the concatenation operation.

2.2.2 Sequential Feature Learning Module. We use Transformer to capture temporal patterns. The target item embedding e_{target} is concatenated with every item embedding in the history sequence to form the input sequence:

$$\widetilde{e^i}_{item} = \left[e^i_{item} \parallel e_{target}\right], \tag{4}$$

where e_{target} is the embedding of the target item.

The length of the history sequence is N. And for those users with fewer than N interactions, the sequence will be padded with zeros. The input sequence is then fed into a Transformer layer with several Transformer Encoders:

$$\begin{aligned} S &= (s_1, s_2, \ldots, s_N) \\ &= \texttt{Transformer}\left(\left[\widetilde{e^1}_{item}, \widetilde{e^2}_{item}, \ldots, \widetilde{e^N}_{item}\right]\right). \end{aligned} \tag{5}$$

The output of the Transformer layer is $S \in \mathbb{R}^{N \times d_t}$, where d_t is the dimension of the Transformer output for each item. While the user's interest cannot be fully represented by the last output, directly using all the outputs of all the items in the history sequence will significantly increase the complexity of the model. Following [13], the latest k outputs are selected as the representation of user's short-term interest preference. And the max pooling operation is adopted to represent the user's long-term interest preference:

$$S_o = \texttt{Flatten}\left(s_1, s_2, \ldots, s_k, \texttt{MaxPool}\left(S\right)\right), \tag{6}$$

where `Flatten` is the flattening operator along the last dimension and `MaxPool` is the max pooling operator.

2.2.3 Feature Interaction Module. To explicitly model the interactions between features, we adopt DCNv2[12] as the feature interaction module for its efficiency and effectiveness of modeling high-order feature interactions:

$$\begin{aligned} f_i &= \left[e_{target}, e_{side}, S_o\right], \\ c_{l+1} &= f_i \odot (W_l c_l + b_l) + c_l, \\ d_o &= \mathsf{MLP}_f\left(f_i\right), \end{aligned} \tag{7}$$

where e_{side} is the concatenated embedding of the side features (*e.g.*, like level and view level of the target item). c_l, W_l and b_l are the output, weight and bias of the l-th cross layer. $\odot$ denotes the element-wise multiplication. MLP_l is a 3-layer perceptron with ReLU activation function, representing the deep network part of DCNv2. The parallel structure is adopted in our DCNv2 module, thus the output of the feature interaction module is:

$$f_o = [c_o, d_o], \tag{8}$$

where c_o is the final output of the last cross layer.

2.2.4 Prediction Layer. Since CTR prediction is a binary classification task, we use a 2-layer perceptron and a sigmoid function to predict the probability of the target item being clicked:

$$\hat{y} = \sigma\left(\mathsf{MLP}_p\left(f_o\right)\right), \tag{9}$$

where σ is the sigmoid function.

2.2.5 Loss Function. The loss function is defined as the binary cross-entropy loss:

$$\mathcal{L} = -\frac{1}{|\mathcal{D}_{tr}|}\sum_{i=1}^{|\mathcal{D}_{tr}|}\left[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)\right], \tag{10}$$

where $|\mathcal{D}_{tr}|$ is the number of samples in the training set.

Table 1: Hyperparameters of our model.

Hyperparameter	Grids	Best Value
`learning_rate`	[1e-3, 5e-4, 5e-5, 1e-5]	5e-4
`embedding_dim`	[16, 32, 64, 128]	64
`transformer_dropout`	[0, 0.1, 0.2, 0.3, 0.4]	0.2
`cross_net_dropout`	[0, 0.1, 0.2, 0.3, 0.4]	0.2
k in Eq.6	[0, 2, 4, 8, 16, 24]	16

3 Experiments

3.1 Preparations

3.1.1 Datasets. The dataset[4] provided by the MM-CTR Challenge originates from the recently released MicroLens dataset by Westlake University[8]. It contains 1M users and 91.7K items, with each item featuring rich modalities including text descriptions, images, audio, and raw video information. To obtain the multimodal embeddings, we use the PCA embedding of the concatenated BERT[1] and CLIP[9] embeddings as the multimodal embedding. The whole dataset is divided into training, validation, and test sets, with 3.6M, 10k, and 380k samples respectively.

3.1.2 Environmental Setup. We have released our code and configuration files on GitHub[5]. All of our code is implemented based on FuxiCTR[6]. After cloning the repository, the environment can be replicated using the following commands:

```
conda create -n fuxictr_momo python==3.9
pip install -r requirements.txt
source activate fuxictr_momo
```

All experiments were conducted on a customized GPU with 32GB VRAM (vGPU-32G) through AutoDL[7]. The versions of CUDA and PyTorch are 11.7 and 1.13.1 respectively.

3.1.3 Parameter Settings. We use the standard Adam optimizer with a learning rate of $5e^{-4}$ and a batch size of 128. Embedding dimension is set to 64. Numbers of cross layers and Transformer encoders are set to 3 and 2 respectively. Hidden units of the deep network in DCNv2 and the prediction layer are set to $[1024, 512, 256]$ and $[64, 32]$ respectively. Dropout rate in both Transformer and DCNv2 is set to 0.2. k in Eq.6 is set to 16. Early stopping is applied to avoid model overfitting. The training process will be terminated if the validation AUC score does not improve for 5 consecutive epochs. We carefully tuned specific hyperparameters in our model by grid search, detailed in Table 1.

3.1.4 Evaluation Metrics. Our model is evaluated using the area under the ROC curve (AUC) and the log loss metrics. Both metrics are widely used in CTR prediction tasks. AUC measures the model's ability to distinguish between positive and negative samples, while log loss quantifies the model's prediction accuracy. The higher the AUC and the lower the log loss, the better the model's performance.

Table 2: Overall performance evaluation.

Model	*w/ multimodal emb.*	AUC	Logloss
Baseline (DIN)	✗	0.9326	0.6485
	✓	0.8577	2.7697
w/ Dice	✗	0.9366	0.6878
	✓	0.8829	2.5390
Ours	✗	0.9729	0.2369
	✓	**0.9776**	**0.2358**
w/o Transformer	✗	0.9741	0.2617
	✓	0.9688	0.3379
w/o DCNv2	✗	0.9023	0.4996
	✓	0.9632	0.3522

3.2 Overall Performance

The baseline model provided by the challenge organizers is a DIN model [14] without Dice activation function. The overall performance comparison of different models with and without multimodal embeddings is summarized in Table 2. AUC improvement is witnessed when we simply replace the activation function with Dice. Our model achieves the best performance across both evaluation metrics, demonstrating its superiority over the baseline.

It is worth noting that the baseline DIN model and our model without Transformer suffer a substantial performance degradation when multimodal embeddings are added, suggesting limited compatibility with multimodal features. We suspect that the frozen multimodal embeddings are not well aligned with the CTR prediction task, making it difficult for the model optimization and better utilization of multimodal information.

Ablation studies reveal the importance of the two key components. Removing the DCNv2 layer (*w/o DCNv2*) sharply degrades performance, while omitting the Transformer module (*w/o Transformer*) also leads to a noticeable decrease in AUC and an increase in log loss, highlighting the complementary roles of sequential feature learning and cross-feature interaction. Overall, the results confirm that our full model optimally integrates multimodal embeddings with Transformer and DCNv2 components to maximize predictive accuracy. And an AUC of 0.9839 on the leaderboard was achieved, ranking 1st in the challenge.

3.3 Parameter Sensitivity Analysis

We conduct parameter sensitivity analysis on various hyperparameters, as shown in Figure 2. Assigning specific values to certain hyperparameters may cause the model collapse phenomenon (*e.g.*, high learning rate), and will not be reported.

Experiments show that the model is sensitive to the learning rate, and a learning rate of $5e^{-4}$ works significantly better than other values. `embedding_dim` and `k` in Eq.6 significantly affect the model capacity and training efficiency, while contributing less to the final performance. Appropriate `dropout` settings have a positive but limited effect on the model performance, whose adjustments were placed at the end to push the performance to the limit.

[4] https://recsys.westlake.edu.cn/MicroLens_1M_MMCTR/
[5] https://github.com/pinskyrobin/WWW2025_MMCTR
[6] https://github.com/reczoo/FuxiCTR
[7] https://www.autodl.com

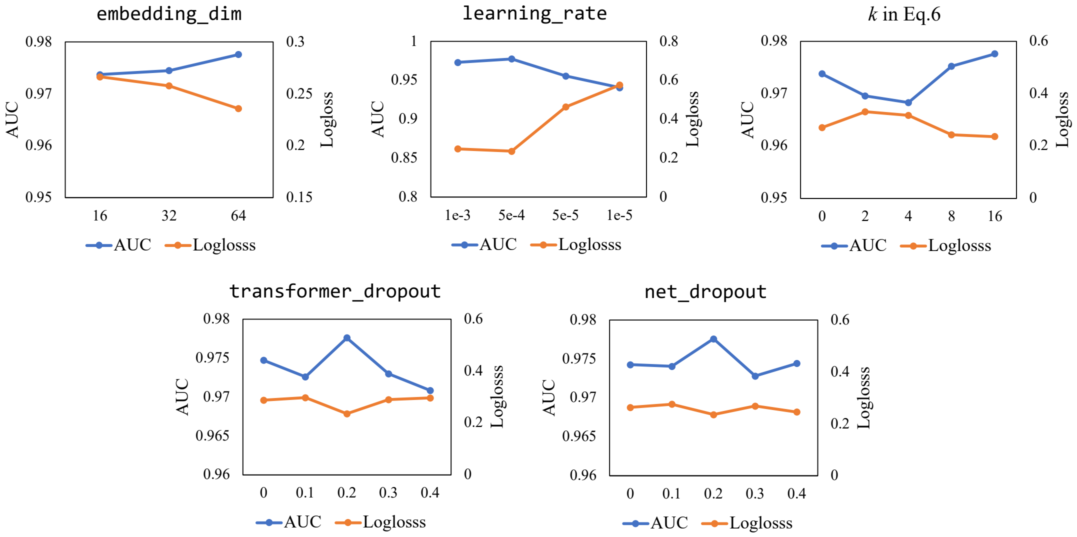

Figure 2: Parameter sensitivity analysis on hyperparameters.

4 Conclusion

Although model ensemble is not allowed in the challenge, the superior performance could still be presented by the simple yet effective model architecture. FuxiCTR provides efficient configuration for tuning our model, saving us a lot of time. Due to the limited time, we simply concatenate the multimodal embeddings with item embeddings. Aligning multi-modal embeddings with the downstream CTR task is the key to further improving the performance of the model. In the future, we will explore quantization methods to transform the frozen multimodal embeddings into semantic and learnable item embeddings. How to effectively utilize prior knowledge of the multimodal embeddings to guide the learning of collaborative ID embeddings is also a worthy research direction.

References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. doi:10.18653/v1/N19-1423

[2] Junchen Fu, Xuri Ge, Xin Xin, Haitao Yu, Yue Feng, Alexandros Karatzoglou, Ioannis Arapakis, and Joemon M. Jose. 2025. The 1st EReLMIR Workshop on Efficient Representation Learning for Multimodal Information Retrieval. arXiv:2504.14788 [cs.IR] https://arxiv.org/abs/2504.14788

[3] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence* (Melbourne, Australia) *(IJCAI'17)*. AAAI Press, 1725—1731.

[4] Xiangnan He and Tat-Seng Chua. 2017. Neural Factorization Machines for Sparse Predictive Analytics. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Shinjuku, Tokyo, Japan) *(SIGIR '17)*. Association for Computing Machinery, New York, NY, USA, 355—364. doi:10.1145/3077136.3080777

[5] Yuchin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. 2016. Field-aware Factorization Machines for CTR Prediction. In *Proceedings of the 10th ACM Conference on Recommender Systems* (Boston, Massachusetts, USA) *(RecSys '16)*. Association for Computing Machinery, New York, NY, USA, 43—50. doi:10.1145/2959100.2959134

[6] Wang-Cheng Kang and Julian McAuley. 2018. Self-Attentive Sequential Recommendation . In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE Computer Society, Los Alamitos, CA, USA, 197–206. doi:10.1109/ICDM.2018.00035

[7] 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* (London, United Kingdom) *(KDD '18)*. Association for Computing Machinery, New York, NY, USA, 1754—1763. doi:10.1145/3219819.3220023

[8] Yongxin Ni, Yu Cheng, Xiangyan Liu, Junchen Fu, Youhua Li, Xiangnan He, Yongfeng Zhang, and Fajie Yuan. 2023. A Content-Driven Micro-Video Recommendation Dataset at Scale. *arXiv preprint arXiv:2309.15379* (2023).

[9] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 139)*, Marina Meila and Tong Zhang (Eds.). PMLR, 8748–8763. https://proceedings.mlr.press/v139/radford21a.html

[10] Steffen Rendle. 2010. Factorization Machines. In *Proceedings of the 2010 IEEE International Conference on Data Mining (ICDM '10)*. IEEE Computer Society, USA, 995—1000. doi:10.1109/ICDM.2010.127

[11] 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* (Beijing, China) *(CIKM '19)*. Association for Computing Machinery, New York, NY, USA, 1161—1170. doi:10.1145/3357384.3357925

[12] Ruoxi Wang, Rakesh Shivanna, Derek Cheng, Sagar Jain, Dong Lin, Lichan Hong, and Ed Chi. 2021. 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) *(WWW '21)*. Association for Computing Machinery, New York, NY, USA, 1785–1797. doi:10.1145/3442381.3450078

[13] Xue Xia, Pong Eksombatchai, Nikil Pancha, Dhruvil Deven Badani, Po-Wei Wang, Neng Gu, Saurabh Vishwas Joshi, Nazanin Farahpour, Zhiyuan Zhang, and Andrew Zhai. 2023. TransAct: Transformer-based Realtime User Action Model for Recommendation at Pinterest. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Long Beach, CA, USA) *(KDD '23)*. Association for Computing Machinery, New York, NY, USA, 5249–5259. doi:10.1145/3580305.3599918

[14] 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* (London, United Kingdom) *(KDD '18)*. Association for Computing Machinery, New York, NY, USA, 1059–1068. doi:10.1145/3219819.3219823