Intelligent Automation & Soft Computing
DOI: 10.32604/iasc.2023.036622
Article

Tech Science Press

Multi-Task Deep Learning with Task Attention for Post-Click Conversion Rate Prediction

Hongxin Luo, Xiaobing Zhou*, Haiyan Ding and Liqing Wang

School of Information Science and Engineering, Yunnan University, Kunming, 650500, China
*Corresponding Author: Xiaobing Zhou. Email: zhouxb@ynu.edu.cn
Received: 07 October 2022; Accepted: 06 December 2022

Abstract: Online advertising has gained much attention on various platforms as a hugely lucrative market. In promoting content and advertisements in real life, the acquisition of user target actions is usually a multi-step process, such as impression→click→conversion, which means the process from the delivery of the recommended item to the user's click to the final conversion. Due to data sparsity or sample selection bias, it is difficult for the trained model to achieve the business goal of the target campaign. Multi-task learning, a classical solution to this problem, aims to generalize better on the original task given several related tasks by exploiting the knowledge between tasks to share the same feature and label space. Adaptively learned task relations bring better performance to make full use of the correlation between tasks. We train a general model capable of capturing the relationships between various tasks on all existing active tasks from a meta-learning perspective. In addition, this paper proposes a Multi-task Attention Network (MAN) to identify commonalities and differences between tasks in the feature space. The model performance is improved by explicitly learning the stacking of task relationships in the label space. To illustrate the effectiveness of our method, experiments are conducted on Alibaba Click and Conversion Prediction (Ali-CCP) dataset. Experimental results show that the method outperforms the state-of-the-art multi-task learning methods.

Keywords: Multi-task learning; recommend system; attention; meta-learning

1 Introduction

Conversion Rate (CVR) prediction is a fundamental task in online advertising and recommendation. For example, the predicted CVR is used in Optimized Cost-Per-Click (OCPC) advertising to adjust the bid price per click, achieving a win-win for both the platform and the advertiser [1]. CVR modeling refers to the task of estimating the post-click conversion rate, i.e., pCVR = p (*conversion*|*click*, *impression*). We mainly study the CVR estimation problem. Considering the continuous action information in the user's multi-step transformation process, we adopt Multi-Task Learning (MTL) for modeling. MTL is a typical solution to improve end-to-end transformation in many-step tasks in industry and academia.

By using shared representations between sets of related tasks, MTL can improve data efficiency and potentially lead to faster learning rates for related or downstream tasks. MTL learning techniques have been widely applied in various fields, from Natural Language Processing (NLP) [2] to Computer Vision (CV) [3], bug triage [4], code retrieval [5] and so on. Recent studies have shown that MTL techniques are very suitable for recommender systems [6–10]. Multi-task neural network models share low-level information among different tasks, separate each task's unique features, and finally associate them to obtain the final information. Knowledge transfer mainly relies on inter-task relationships. Therefore, how to properly model the relationship between tasks and how to control the knowledge transfer between tasks is the key to MTL.

Training a custom model from scratch for each task has serious overfitting problems. The pre-trained embedding layer cannot consider the relationship between multiple tasks, and the model cannot generalize well. Hence, the customized network is only suitable for the original distribution space. Deep learning uses large-scale sample instances to "violently" train the network, allowing the model to process a single task effectively. However, this can only alleviate temporary needs, and cannot implement the same set of algorithms to solve various complex tasks. In order to cope with this challenge, we introduce a meta-learning strategy in the training process to train a general model for all tasks that captures complex relationships between tasks.

In this paper, we propose a Multi-task Attention Network (MAN) to control knowledge transfer between tasks dynamically. MAN is an attention mechanism-based method to solve the problem of user sequence modeling and alleviates data imbalances by leveraging shared information across tasks, and captures correlations between sequential tasks by leveraging prior knowledge in MTL. The idea of meta-learning is introduced in the training process, and two-stage training is carried out. First, a general generalization model is obtained through the early stop mechanism. On this basis, the high-level information of the fusion model is continued to be trained to improve the training effect, and it can eliminate data sparsity and sample selection bias problems. The task-sharing experts and task-specific experts in the model enable the model to separate the characteristics of each task and integrate the commonalities among multiple tasks. Compared with existing models, we improve training performance by reducing conflicts by sharing objectives.

To verify the effectiveness of our model, we conduct experiments on the Ali-CCP dataset [6], and compare the results with the state-of-the-art multi-task models, showing that our results are up to the best. The main contributions of this paper include the following points:

1) We propose a Multi-task Attention Network to dynamically learn the relationship between tasks and control knowledge sharing in multi-task learning from the perspective of joint representation learning. So the correlation between user sequence actions can be learned.

2) We define the problems of data sparsity and sample selection bias from the perspective of meta-learning and use meta-learning to solve them.

3) We propose a two-stage training scheme, which is important for improving our performance.

Our work is organized as follows. In Section 2, we summarize related work. Section 3 formally defines our problem and introduces methods for learning feature interactions. In Section 4, we present the experimental results and detailed analysis. Section 5 concludes this paper.

2 Related Work

2.1 CVR Modeling

Accurately predicting user responses, such as click-through rates and conversion rates, is essential in the recommendation, search, and advertising applications. As shown in Fig. 1, click-through rate (CTR) is

designed to predict the probability of a user clicking on an advertisement. At the same time, CVR is to estimate the probability of a user's actions based on the previous one, and there is an order sequence of actions in between. These two tasks can often be formalized as a classic binary classification problem [11]. A major challenge in CVR estimation is the problem of sample selection bias [12]. DeepFM [13] uses Factorization Machines (FM) instead of Logistic Regression (LR) in the wide-area part to automatically learn second-order feature interactions, while DCN [14] uses a cross-network to learn higher-order representations. DUPN [15] learns common user representations across multiple search and recommendation tasks for more effective personalization. MA-RDPG [16] improves the overall performance of ranking strategies in search, recommendation, and advertising through multi-agent reinforcement learning. Weighted Fake Negative (FN) and FN calibration loss functions [17] are proposed to address delayed feedback with continuous training of neural networks in CTR prediction. But they do not exploit the task relationship in the label space. To address this issue, ESMM [6] simultaneously models the posterior view click-through rate and post-view click-through & conversion rate (CTCVR) by employing a feature representation transfer learning strategy across the entire space. HoAFM [18] proposes a high-order attention decomposition machine by considering the interaction of high-order sparse features, and uses a bit-attention mechanism to learn the importance of co-occurring features.

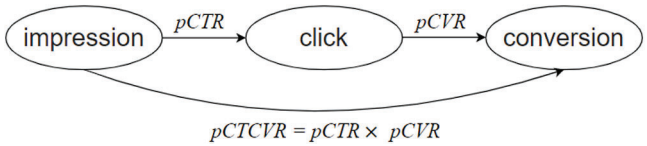

Figure 1: The user first clicks on a product from the search page and then decides whether to buy the product. The aim is to maximize post-view click-through & conversion rate

2.2 Multi-Task Learning (MTL)

MTL provides an effective framework for leveraging task relationships to transfer knowledge and improve model generalization. To exploit the task relationship in MTL, there are recent examples such as the cross-stitch network [19] using linear units to learn the optimal combination of task-specific representations for each task, and a hard parameter sharing structure is thought to help each task better leverage knowledge learned from other tasks, but the natural differences between each task may lead to conflicts in learning shared representations. Furthermore, ESMM [6] proposes a new perspective on multi-task modeling with sequence dependencies to tackle the challenges of extreme data sparsity and sample selection bias. PAL [20] utilizes a two-task model to remove positional bias in advertisement recommendation, which separates view-through and click-through rates. F^3 [21] proposes an intuitive yet effective general framework to concatenate different learners with the filter mechanism to filter out unchanged fixes. Flow2Vec [22] presents a new code embedding approach that preserves interprocedural, context-sensitive and alias-aware value-flows in the low-dimensional vector space to better support subsequent learning tasks. These methods exhibit better feature learning in shared and task-specific representations and achieve better performances than typical MTL methods. However, there are still some limitations, such as the model depends on specific application scenarios and statically captures task relationships.

Previous work [23,24] attempts to measure differences between tasks based on assumptions about the data generation process. Still, in the real world, the data generation process is often more complex and harder to measure. Existing studies still lack an in-depth exploration of the conflicting nature of shared parameters, nor an in-depth analysis of the incompatibility and synergy of each task. Recent research has shown that tower top modules tend to contain richer and more useful information, and if there is no

information exchange between them, they cannot help tasks improve each other. Multiple tasks will suffer if any of these probabilities cannot be predicted accurately. In this paper, we attempt to adaptively model task relationships in MTL with self-attention networks, dynamically learn the relationships between tasks and control the knowledge transfer between tasks. Our proposed Multi-task Attention Network (MAN) trains a model that can be generalized across multiple tasks from a meta-learning perspective. MAN combines shared and task experts to further capture task relationships and learn transferable knowledge. This enables each task to obtain shared feature representations better, thus enabling MTL methods to achieve better performance.

3 Methodology

3.1 Problem Statement

Impression is represented as a feature set containing user features, item features, and other contextual features. The feature set of impressions is converted into sparse feature vectors (composed of various one-hot encodings), such as user fields, item fields, etc. Consider a sample of impressions $(X,\ Y(click), Z(conv))$, where X is a sparse feature vector and $Y(click) \in \{0, 1\}$, $Z(conv) \in \{0, 1\}$ denote its click label, view after conversion label. $Y \rightarrow Z$ reveals the order dependence of click and transition labels, i.e., when a conversion event occurs, there is always a previous click. Our goal is to maximize the post-view click-through & conversion rate, $pCTCVR = p(y = 1,\ z = 1|x)$. Among them, CTCVR can be decomposed into two goals: CTR and CVR. Post-click conversion rate $= p(z = 1|y = 1,\ x)$, post-click view click-through rate $pCTR = p(z = 1|x)$. Given a sample x, these probabilities fit the formula:

$$\underbrace{p(y = 1, z = 1|x)}_{pCTCVR} = \underbrace{p(y = 1|x)}_{pCTR} \times \underbrace{p(z = 1|y = 1, x)}_{pCVR} \tag{1}$$

Since the original feature vector x is very sparse, the model is prone to overfitting. Therefore, we need to encode the original feature X into an embedding vector to represent the original input features in a small continuous space. Look-up embeddings have been widely used to learn dense representations from raw data for online prediction [25]. We denote all features X as embedding vector E, where $E \in R_k$ and k denotes the embedding dimension.

3.2 Model Structure

In order to better learn the relationship between complex tasks, we introduce a model to obtain the feature information that is beneficial to each task from the original user input, and improve the prediction of the network. The model framework consists of a customized gate control network model, a meta-learning framework, and an attention network, as shown in Fig. 2.

Lower-level representations are jointly extracted/aggregated during knowledge extraction and transformation and routed to higher-level shared experts. Shared knowledge is captured and gradually distributed to specific tower layers, enabling more efficient and flexible joint representation learning and sharing. The CVR task shares the embedding dictionary representation with the CTR task, which follows a feature representation transfer learning paradigm. Since the training samples of the CTR task are relatively abundant, using the parameter sharing mechanism is of great help in alleviating the problem of data sparsity.

3.2.1 Embedding

Given an impression sparse feature vector for the CTR estimation task, the embedding layer converts high-dimensional, sparse features and sequence features into a low-dimensional dense representation, and then concatenates these embeddings to obtain the feature embedding E_{ctr}. Likewise, for the CVR estimation task, we can obtain the feature embedding E_{cvr}. The embedding vector E can be described as

$E_b \in \mathbb{R}^{(dm_b)\times n}$, where n is the maximum length of the sequence and m_b is the field numbers. The embedding matrix is uniformly initialized and then learned during training. By sharing the same embedding vector among all tasks, one can learn the embedding vector with rich positive samples for the former task, on the one hand, to share information and alleviate the class imbalance of the latter task, and on the other hand, to reduce the model parameters.

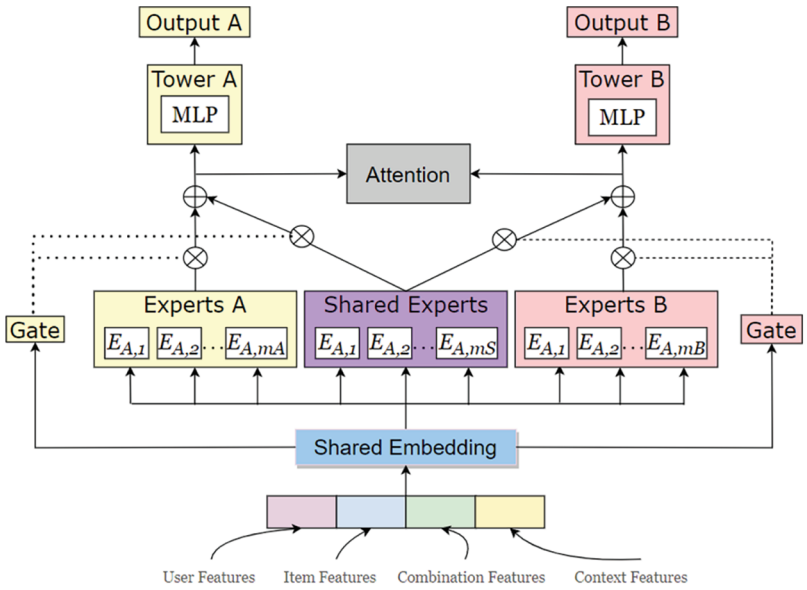

Figure 2: The architecture of Multi-task Attention Network (MAN)

3.2.2 Experts Network

Different experts specialize in different spaces and can capture feature information in different subspaces. Therefore, for a specific task, we choose task-specific experts for learning, and the experts do not interfere with each other. For task-shared features, we use shared experts to learn the representations. Different tasks share the outputs of experts and task-specific experts by combining tasks through a gated network. By combining experts with the gating network, the model can retain the ability of transfer learning and avoid the negative transfer phenomenon. Each of the k subtasks corresponds to a tower network, and each subtask outputs a target that can be expressed as $f(x) = \sum g(x)_i f_i(x)$, where $\sum g(x)_i = 1$, $f_i\ (i = 1, 2, 3, \ldots, n)$ represents n expert networks. Specifically, the gate network g is to generate the probability distribution of each expert, the final output is the weighted sum of all experts.

3.2.3 Multi-Task Attention Network

In order to better utilize richer and more useful information, facilitate the exchange of information between tasks, and help tasks to improve each other, we use attention networks [26–28] to model relevant task information. For the two tasks T_i and T_j, the attention network is used to assign weights to the correlation information of adjacent tasks and the information of the current task, and then the two parts of information are fused to obtain the output C_{ij} with task-related information. The processing is as follows: $C_{ij} = a_t(x_i||x_j)$, where x_i, x_j represent the information transferred from adjacent tasks and the current task, respectively, a_t represents the attention network, and $||$ is the concatenation operation, C_{ij} represents the output of the fusion of the two tasks information. The Query, Key, Value in the attention mechanism is obtained through feed-forward networks, which projects the input information to a new vector representation, then the values of Q, K, V in the attention network are obtained.

3.3 Loss

Each task is equipped with tower networks for converting the final task-specific representation into predicted values. The tower network is further learned to interact by a two-layer Multi-Layer Perceptron (MLP) with ReLU activation. The final logits output the probabilities after the Sigmoid function. The final loss is a linear combination of individual task losses, defined as follows:

$$\mathcal{L}_{MTL} = \sum\nolimits_k w_k \cdot \mathcal{L}_k \tag{2}$$

where W is task-specific weight, and $\mathcal{L}$ represents the loss function. A loss-optimized uncertainty weight [29] is employed in our model, which uses homoscedastic uncertainty to balance the loss for individual tasks. Homoscedastic uncertainty refers to any uncertainty that does not depend on the input data. It is not a model output, but a quantity that remains constant for all input data and varies from task to task. When we execute the optimizer to maximize the Gaussian likelihood objective, homoscedastic uncertainty is taken into account. In the model, the uncertainty loss can be expressed as:

$$\mathcal{L}_{MTL}(W, \sigma_1, \sigma_2) = \frac{1}{2{\sigma_1}^2}\mathcal{L}_1(W) + \frac{1}{2{\sigma_2}^2}\mathcal{L}_2(W) + \log \sigma_1\sigma_2 \tag{3}$$

where $\mathcal{L}_1$ and $\mathcal{L}_2$ represent the losses of the two tasks, respectively. σ_1 and σ_2 are the corresponding noise parameters that balance task-specific losses. During training, the trainable parameters should be updated automatically. In addition, during the training process, we also try to add corresponding weights to the positive and negative samples when calculating the cross-entropy loss, so as to avoid overfitting and make it better to predict the correct category.

4 Experiment

4.1 Datasets

The performance of our proposed method is evaluated on the public dataset Ali-CCP [6]. The overall description of Ali-CCP data is $\{(x_i, y_i \rightarrow z_i)\}|_{i=1}^{N}$, and the format of the sample is $(X, Y \rightarrow Z)$, which is from the custom domain $(x, y \rightarrow z)$, where X is the feature space, Y and Z are the label spaces, and N is the total number of samples. x represents the feature vector of the observed sample, which is usually a high-dimensional sparse vector divided into multiple domains, such as user domain, commodity domain, etc. Both y and z are binary labels (0 or 1), where $y = 1$ represents a click event for the sample, and $z = 1$ represents a conversion event for the sample. Data processing method follows the previous work of AITM [30].

4.2 Baseline Models

- **OMoE** [8]: OMoE builds on the shared expert base model by sharing a gate across all tasks to integrate the output of experts.
- **MMoE** [8]: The difference from OMoE is that the original unique gate is changed to gates of the number of tasks to integrate the output of experts respectively.
- **ESMM** [6]: This model is designed to be trained on the entire space to address non-end-to-end post-click conversion rates. With the help of two auxiliary tasks, CTR and CTCVR, ESMM subtly addresses the challenges of sample selection bias and data sparsity encountered in practical applications of CVR modeling.
- **PLE** [10]: Progressive Hierarchical Extraction (PLE) distinguishes inter-task shared information from specific information by constructing task-sharing experts and task-specific experts.
- **AITM** [30]: AITM improves the performance of sequence-dependent multi-task learning by constructing an adaptive information transfer multi-task framework to simulate sequential

dependencies among transitions across many steps. This method achieves state-of-the-art results under different task dependencies.

Experimental Settings: We set the hidden dimension of each expert to be [256, 128]. We use 3 layers [128, 16, 32] MLP as a tower network to predict each task. For OMoE, MMoE, PLE, we use 8 experts, where 4 shared experts and 4 task-specific experts for fair comparison. Since ESMM and AITM do not use expert networks, we follow the original settings for the rest of the parameter settings. For our model, we set *expert_per_task* to 4 and *shared_expert_num* to 4. In these six models, the activation function of the expert is set to ReLU. All models are optimized using Adam Optimizer [31] with batch size 2048, setting weight decay to 1e-6 and learning rate to 0.001. Preliminary optimal parameters are obtained through grid search, and the optimal parameters are determined on this basis. For feature input, we use the same embedding module for consistency and fairness. All experiments are repeated 5 times and the average results are reported. We also try to use the focal loss proposed in [32], but it does not improve our results.

4.3 Meta-Learning

Meta-learning is helpful in achieving fast learning to alleviate overfitting, and can quickly learn general knowledge of tasks even in a limited sample space [33–35]. We treat the overall training algorithm from a meta-learning perspective. First, we prepare N training tasks, support set and query set corresponding to each task, and test tasks. The test task is used to evaluate the effect of the parameters learned by meta-learning. Both training and test sets are sampled from our original sample space. The parameters to initialize meta-learning network are θ. The meta-learning network is the network that will eventually be applied to the new test task, and the "prior knowledge" is stored in the network. Then we perform iterative "pre-training" to obtain the parameters of the meta-learning network, through which the parameters of the meta-learning network are "fine-tuning" using the support set in the test task. Finally, the effect of meta-learning is evaluated using the query set of the test task. See Algorithm 1 for the detailed training process.

Algorithm 1: Training Multi-task Attention Network from a meta-learning perspective.

Require: p(T): given a training dataset $D_{[c]}$

Require: α, β: step size hyperparameters

Require: $f\theta$: the general model

1. randomly initialize θ.
2. **while** not converge **do**:
3. Sample batch of tasks T$i \sim$ p(T)
4. **for** all Ti do:

   Ti contains two disjoint sets $D^{a}_{[c]}$, $D^{a}_{[c]}$

   evaluate loss $L_a(\theta)$ with $D^{a}_{[c]}$

   compute updated parameter $\theta_{[c]} = \theta - \alpha \frac{\vartheta L_a(\theta)}{\vartheta \theta}$

   evaluate loss $L_b(\theta_{[c]})$ with $D^{b}_{[c]}$
7. **end for**
8. Update $\theta = \theta - \beta \sum T_i \epsilon \{T_1, \ldots, T_n\} \frac{\vartheta L_b\left(\theta_{[c]}\right)}{\vartheta \theta}$
9. **end while**

4.4 Performance Comparison

The results show that our proposed model outperforms the mainstream models on all metrics tested offline. Among the base models we compared, only ESMM performed slightly worse on the CTR task, which may be due to the model implicitly modeling the relationship between tasks and relying too much on the performance of the previous task leading to the seesaw phenomenon. From the detailed data information, we can find that the samples have serious class imbalance classes, and from the results in Table 1, we can find that the click task appears as the first task in these MTL baseline models, and the MTL model does not significantly improve its performance. We explicitly model the relationship between tasks through an attention network, so our model can achieve better performance in this case and good generalization performance even in the case of sample class imbalance. We print out the final weights for model training and find that the weights assigned to the two tasks are [0.0041, 0.0765]. From the learning results of the model, it can be seen that the reason for this result is due to the problem of sample selection bias. The completion process from click event to conversion event is rare, so the model learns different weights to balance the training process, indicating that our method is effective in solving the above problems. Compared with the state-of-the-art models, our model achieves 3.52% and 7.29% improvements over the LightGBM [36] baseline model on the CTR and CVR tasks, respectively. Our model achieves the best results Ali-CCP dataset, proving the effectiveness of our model.

Table 1: The performances of each model on Ali-CCP. The Gain indicates the improvement of the results of each model compared to the LightGBM baseline. Our experiments are repeated 5 times, and the results report the mean and standard deviation

Model	CTR AUC	CVR AUC	Gain	
LightGBM [36]	0.5837 ± 0.0005	0.5870 ± 0.0038	–	–
OMoE [8]	0.6049 ± 0.0020	0.6405 ± 0.0041	+0.0212	+0.0535
MMOE [8]	0.6047 ± 0.0017	0.6420 ± 0.0031	+0.0210	+0.0550
ESMM [6]	0.6022 ± 0.0020	0.6291 ± 0.0061	+0.0185	+0.0421
PLE [10]	0.6039 ± 0.0014	0.6417 ± 0.0013	+0.0202	+0.0547
AITM [30]	0.6043 ± 0.0016	0.6525 ± 0.0024	+0.0206	+0.0655
OURS	**0.6225 ± 0.0003**	**0.6599 ± 0.0047**	+0.0352	+0.0729

4.5 Ablation Study

To verify the effectiveness of our method, we further compare our model with its own variants, evaluating our various modules, and the results are shown in Table 2. “Without meta” means to remove the training method of meta-learning, “Without att” means to remove the attention network, “Without meta-att” means to remove the embedding layer and directly train the original feature data. The results show that under the same setting, our main method outperforms several other variants in both CTR and CVR estimation. The addition of meta-learning enables fast adaptation tasks. After training for 3 epochs, the AUC of CTR and CVR can reach 0.61 and 0.64, respectively, which is helpful in alleviating overfitting. The attention network module promotes the exchange of information between tasks, so that the auxiliary information of related tasks can be used between tasks to improve the model’s performance. Experimental results show that considering the task hierarchy in the process of using multi-task learning to solve the CVR modeling problem can further improve performance.

Table 2: Results of ablation study of our model on the Ali-CCP dataset

Model variants	CVR AUC	CTR AUC
MAN	**0.6225**	**0.6599**
Without meta	0.6118	0.6443
Without att	0.6211	0.6539
Without meta-att	0.6042	0.6401

5 Conclusion

In this paper, we propose a Multi-task Attention Network to learn knowledge transfer between tasks and use an attention network to learn weight assignment and knowledge sharing between complex tasks. The correlations between tasks are captured by leveraging prior knowledge in MTL. The shared experts and task-specific experts in the model can well separate the characteristics of each task and integrate the commonalities between multiple tasks, avoiding the phenomenon of negative transfer. In the training process, a two-stage training meta-learning strategy is introduced to train a general model for all tasks that can capture the complex relationships between tasks. Experimental results show that our method achieves state-of-the-art results.

Funding Statement: Our work was supported by the research project of Yunnan University (Grant No. 2021Y274), Natural Science Foundation of China (Grant No. 61862064).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] H. Zhu, J. Jin, C. Tan, F. Pan, Y. Zeng *et al.,* "Optimized cost per click in taobao display advertising," in *Proc. SIGKDD*, Boston, MA, USA, pp. 2191–2200, 2017.

[2] K. Hashimoto, C. Xiong, Y. Tsuruoka and R. Socher, "A joint many-task model: Growing a neural network for multiple nlp tasks," in *Proc. EMNLP*, Copenhagen, Denmark, pp. 1923–1933, 2017.

[3] Y. Lu, A. Kumar, S. Zhai, Y. Cheng, T. Javidi *et al.,* "Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification," in *Proc. CVPR*, Honolulu, HI, USA, pp. 5334–5343, 2017.

[4] T. Aung, Y. Wan, H. Huo and Y. Sui, "Multi-triage: A multi-task learning framework for bug triage," *Systems and Software*, vol. 184, no. 12, pp. 111133, 2022.

[5] Y. Wan, J. Shu, Y. Sui, G. Xu, Z. Zhao *et al.,* "Multi-modal attention network learning for semantic source code retrieval," in *Proc. ASE*, San Diego, California, USA, pp. 13–25, 2019.

[6] X. Ma, L. Zhao, G. Huang, Z. Wang, Z. Hu *et al.,* "Entire space multi-task model: An effective approach for estimating post-click conversion rate," in *Proc. SIGIR*, Ann Arbor, MI, USA, pp. 1137–1140, 2018.

[7] R. Jacobs, M. Jordan, S. Nowlan and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Computation*, vol. 3, no. 1, pp. 79–87, 1991.

[8] J. Ma, Z. Zhao, X. Yi, J. Chen, L. Hong *et al.,* "Modeling task relationships in multi-task learning with multi-gate mixture-of-experts," in *Proc. SIGKDD*, London, United Kingdom, pp. 1930–1939, 2018.

[9] J. Zhao, B. Du, L. Sun, F. Zhuang, W. Lv *et al.,* "Multiple relational attention network for multi-task learning," in *Proc. SIGKDD*, Anchorage, AK, USA, pp. 1123–1131, 2019.

[10] H. Tang, J. Liu, M. Zhao and X. Gong, "Progressive layered extraction (ple): A novel multi-task learning (mtl) model for personalized recommendations," in *Proc. RecSys*, Virtual Event, Brazil, pp. 269–278, 2020.

[11] H. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra *et al.,* "Wide & deep learning for recommender systems," in *Proc. DLRS*, Boston, MA, USA, pp. 7–10, 2016.

[12] B. Zadrozny, "Learning and evaluating classifiers under sample selection bias," in *Proc, ICML*, Banff, Canada, pp. 114, 2004.

[13] H. Guo, R. Tang, Y. Ye, Z. Li and X. He, "Deepfm: A factorization-machine based neural network for CTR prediction," in *Proc, IJCAI*, Melbourne, Australia, pp. 1725–1731, 2017.

[14] R. Wang, B. Fu, G. Fu and M. Wang, "Deep & cross network for ad click predictions," in *Proc. ADKDD*, Halifax, NS, Canada, pp. 1–7, 2017.

[15] Y. Ni, D. Ou, S. Liu, X. Li, W. Ou *et al.,* "Perceive your users in depth: Learning uni versal user representations from multiple e-commerce tasks," in *Proc. SIGKDD*, London, United Kingdom, pp. 596–605, 2018.

[16] J. Feng, H. Li, M. Huang, S. Liu, W. Ou *et al.,* "Learning to collaborate: Multi scenario ranking via multi-agent reinforcement learning," in *Proc. WWW*, Lyon, France, pp. 1939–1948, 2018.

[17] S. Ktena, A. Tejani, L. Theis, P. Myana, D. Dilipkumar *et al.,* "Addressing delayed feedback for continuous training with neural networks in CTR prediction," in *Proc. RecSys*, Copenhagen, Denmark, pp. 187–195, 2019.

[18] Z. Tao, X. Wang, X. He, X. Huang and T. Chua, "Hoafm: A high-order attentive factorization machine for CTR prediction," *Information Processing & Management*, vol. 57, no. 6, pp. 102076, 2020.

[19] I. Misra, A. Shrivastava, A. Gupta and M. Hebert, "Cross-stitch networks for multi-task learning," in *Proc. CVPR*, Las Vegas, NV, USA, pp. 3994–4003, 2016.

[20] H. Guo, J. Yu, Q. Liu, R. Tang and Y. Zhang, "Pal: A position-bias aware learning framework for CTR prediction in live recommender systems," in *Proc. RecSys*, Copenhagen, Denmark, pp. 452–456, 2019.

[21] Y. Sui, X. Cheng, G. Zhang and H. Wang, "Flow2vec: Value-flow-based precise code embedding," in *Proc. ACM Program. Lang*, Sanya, China, pp. 1–27, 2020.

[22] H. Hong, J. Zhang, Y. Zhang, Y. Wan and Y. Sui, "Fix-filter-fix: Intuitively connect any models for effective bug fixing," in *Proc. EMNLP*, Online and Punta Cana, Dominican Republic, pp. 3495–3504, 2021.

[23] R. Caruana, "Multi-task learning," *Machine Learning*, vol. 28, no. 1, pp. 41–75, 1997.

[24] Y. Yang and T. Hospedales, "Deep multi-task representation learning: A tensor factorization approach," in *Proc. ICLR*, Toulon, France, pp. 12, 2017.

[25] F. Pan, S. Li, X. Ao, P. Tang and Q. He, "Warm up cold-start advertisements: Improving ctr predictions via learning to learn id embeddings," in *Proc. SIGIR*, Paris, France, pp. 695–704, 2019.

[26] D. Xi, B. Song, F. Zhuang, Y. Zhu, S. Chen *et al.,* "Modeling the field value variations and field interactions simultaneously for fraud detection," in *Proc. AAAI*, Virtual Event, pp. 14957–14964, 2021.

[27] D. Xi, F. Zhuang, B. Song, Y. Zhu, S. Chen *et al.,* "Neural hierarchical factorization machines for user's event sequence analysis," in *Proc. SIGIR*, Virtual Event, China, pp. 1893–1896, 2020.

[28] Y. Zhu, D. Xi, B. Song, F. Zhuang, S. Chen *et al.,* "Modeling usersâ behavior sequences with hierarchical explainable network for cross-domain fraud detection," in *Proc. WWW*, Taipei, Taiwan, pp. 928–938, 2020.

[29] A. Kendall, Y. Gal and R. Cipolla, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," in *Proc. CVPR*, Salt Lake City, UT, USA, pp. 7482–7491, 2018.

[30] D. Xi, Z. Chen, P. Yan, Y. Zhang, Y. Zhu *et al.,* "Modeling the sequential dependence among audience multi-step conversions with multi-task learning in targeted display advertising," in *Proc. SIGKDD*, Virtual Event, Singapore, pp. 3745–3755, 2021.

[31] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, San Diego, CA, USA, pp. 11245, 2015.

[32] T. Lin, P. Goyal, R. Girshick, K. He and P. Dollár, "Focal loss for dense object detection," in *Proc. ICCV*, Venice, Italy, pp. 2980–2988, 2017.

[33] Y. Zhu, R. Xie, F. Zhuang, K. Ge, Y. Sun *et al.,* "Learning to warm up cold item embeddings for cold-start recommendation with meta scaling and shifting networks," in *Proc. SIGIR*, Virtual Event, Canada, pp. 1167–1176, 2021.

[34] Q. Sun, Y. Liu, T. Chua and B. Schiele, "Meta-transfer learning for few-shot learning," in *Proc. CVPR*, Long Beach, CA, USA, pp. 403–412, 2019.

[35] C. Finn, P. Abbeel and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. ICML*, Sydney, NSW, Australia, pp. 1126–1135, 2017.

[36] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen *et al.,* "Lightgbm: A highly efficient gradient boosting decision tree," in *Proc. NIPS*, Long Beach, CA, USA, pp. 3149–3157, 2017.