Modeling the Sequential Dependence among Audience Multi-step Conversions with Multi-task Learning in Targeted Display Advertising

Dongbo Xi[1,*], Zhen Chen[1,*], Peng Yan[1], Yinger Zhang[1,2],
Yongchun Zhu[3,4], Fuzhen Zhuang[5,6], and Yu Chen[1]
[1]Meituan
[2]Zhejiang University, Hangzhou 310027, China
[3]Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS),
Institute of Computing Technology, CAS, Beijing 100190, China
[4]University of Chinese Academy of Sciences, Beijing 100049, China
[5]Institute of Artificial Intelligence, Beihang University, Beijing 100191, China
[6]SKLSDE, School of Computer Science, Beihang University, Beijing 100191, China
{xidongbo,chenzhen06,yanpeng04,chenyu17}@meituan.com,
zhangyinger@zju.edu.cn, zhuyongchun18s@ict.ac.cn, zhuangfuzhen@buaa.edu.cn

ABSTRACT

In most real-world large-scale online applications (e.g., e-commerce or finance), customer acquisition is usually a multi-step conversion process of audiences. For example, an *impression* → *click* → *purchase* process is usually performed of audiences for e-commerce platforms. However, it is more difficult to acquire customers in financial advertising (e.g., credit card advertising) than in traditional advertising. On the one hand, the audience multi-step conversion path is longer, an *impression* → *click* → *application* → *approval* → *activation* process usually occurs during the audience conversion for credit card business in financial advertising. On the other hand, the positive feedback is sparser (class imbalance) step by step, and it is difficult to obtain the final positive feedback due to the delayed feedback of *activation*. Therefore, it is necessary to use the positive feedback information of the former step to alleviate the class imbalance of the latter step. Multi-task learning is a typical solution in this direction. While considerable multi-task efforts have been made in this direction, a long-standing challenge is how to explicitly model the long-path sequential dependence among audience multi-step conversions for improving the end-to-end conversion. In this paper, we propose an Adaptive Information Transfer Multi-task (AITM) framework, which models the sequential dependence among audience multi-step conversions via the Adaptive Information Transfer (AIT) module. The AIT module can adaptively learn what and how much information to transfer for different conversion stages. Besides, by combining the Behavioral Expectation Calibrator in the loss function, the AITM framework can yield more accurate end-to-end conversion identification. The proposed framework is deployed in Meituan app, which utilizes it to real-timely show a banner to the audience with a high end-to-end conversion rate for Meituan Co-Branded Credit Cards. Offline experimental results on both industrial and public real-world datasets clearly demonstrate that the proposed framework achieves significantly better performance compared with state-of-the-art baselines. Besides, online experiments also demonstrate significant improvement compared with existing online models. Furthermore, we have released the source code of the proposed framework at https://github.com/xidongbo/AITM.

CCS CONCEPTS

• **Information systems** → **Computational advertising**; • **Computing methodologies** → **Multi-task learning**.

KEYWORDS

Sequential Dependence; Multi-step Conversions; Multi-task Learning; Targeted Display Advertising

ACM Reference Format:
Dongbo Xi, Zhen Chen, Peng Yan, Yinger Zhang, Yongchun Zhu, Fuzhen Zhuang, and Yu Chen. 2021. Modeling the Sequential Dependence among Audience Multi-step Conversions with Multi-task Learning in Targeted Display Advertising. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3447548.3467071

1 INTRODUCTION

Customer acquisition management can be considered the connection between advertising and customer relationship management to acquire new customers[1]. With the explosive growth of e-commerce, continuous and effective customer acquisition has become one of the biggest challenges for real-world large-scale online applications.

In this paper, we focus on the customer acquisition task with sequential dependence among audience multi-step conversions.

* Corresponding authors: Dongbo Xi and Zhen Chen.

KDD '21, August 14–18, 2021, Virtual Event, Singapore

ACM ISBN 978-1-4503-8332-5/21/08…$15.00
https://doi.org/10.1145/3447548.3467071

[1]https://en.wikipedia.org/wiki/Customer_acquisition_management

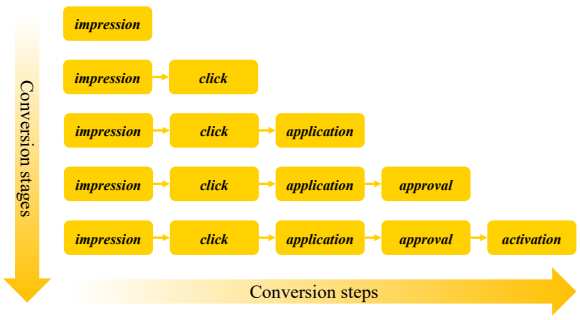

Figure 1: There are five conversion steps with sequential dependence from left to right, and five different conversion stages of audiences from top to bottom. The lower the stage, the more efficient the conversion. In our business, we usually hope audiences to complete the last two stages.

Typically, in the credit card business, the audience multi-step conversion process usually needs to go through *impression* → *click* → *application* → *approval* → *activation* steps. These steps are defined as follows:

- ***impression***: In our business, the *impression* means that the advertising banner is shown to the audience selected according to several ranking metrics, e.g., the Click-Through Rate (CTR).
- ***click***: The *click* means that the shown banner is clicked by the audience, and redirected to the application page.
- ***application***: The *application* means that the audience has filled in the application form and click the application button for a credit card.
- ***approval***: The *approval* means that the credit of the audience has been approved. In our system, this is also a real-time step.
- ***activation***: The *activation* is delayed feedback, and it means that the audience has activated the credit card within a period of time after *approval*. Usually, we consider whether the audience has activated the credit card within 14 days (i.e., activation in T+14). The *activation* feedback label is usually difficult to obtain due to the time-consuming of card mailing and the delayed feedback of audiences, so the class imbalance is more serious.

These conversion steps have sequential dependence, which means that only the former step occurs, the latter step may occur. Based on this constraint, there are five different conversion stages of audiences as shown in Figure 1. Everything else is illegal.

In industry and academia, multi-task learning is a typical solution to improve the end-to-end conversion in the audience multi-step conversion task. Recently, considerable efforts have been done to model task relationships in multi-task learning. One idea is to control how Expert modules are shared across all tasks at the bottom of the multi-task model [14, 16, 19], and Tower modules at the top handle each task separately as shown in Figure 2 (a). However, the Expert-Bottom pattern can only transfer shallow representations among tasks, but in the network close to the output layer, it often contains richer and more useful representations [11, 29], which have been proved to bring more gains [20]. Besides, the Expert-Bottom pattern is not specially designed for tasks with sequential dependence, so these models with the Expert-Bottom pattern can not model the sequential dependence explicitly. Another idea is to transfer probabilities in the output layers of different tasks [2, 3, 15, 23] as shown in Figure 2 (b). Similarly, the Probability-Transfer pattern can only transfer simple probability information via the scalar product, but richer and more useful representations are ignored in the vector space, which results in a great loss of gains. If any one of the probabilities is not predicted accurately, multiple tasks will be affected. Besides, the Probability-Transfer pattern is designed for solving the non-end-to-end post-click conversion rate via training on the entire space to relieve the sample selection bias problem, and these models with Probability-Transfer pattern can not model the sequential dependence well among audience multi-step conversions. Therefore, a long-standing challenge is how to model the sequential dependence among audience multi-step conversions for improving the end-to-end conversion.

Along this line, we propose an Adaptive Information Transfer Multi-task (AITM) framework to model the sequential dependence among audience multi-step conversions. Specifically, due to the sequential dependence among audience multi-step conversions, the former conversion step (task) can bring useful information to the latter step (task). For example, if an audience has clicked the banner, then he/she may apply for the credit card. Conversely, if an audience doesn't click the banner, he/she certainly will not apply for the credit card. Based on this, different conversion stages of different audiences need to transfer different information from the former step to the latter step, and as mentioned above, the vector space close to the output layer often contains richer and more useful information. Therefore, we let the model adaptively transfer information in the vector space close to the output layer via the Adaptive Information Transfer (AIT) module. Another advantage of the AIT module is that it can alleviate the class imbalance of the latter task with the help of the information from the former task, which has richer positive samples. Also, because of the sequential dependence, the former task should have a higher end-to-end conversion probability than the latter for the same audience. Therefore, we design a Behavioral Expectation Calibrator in the loss function. On the one hand, it makes the model results more satisfy the real production constraint, on the other hand, it provides more accurate end-to-end conversion identification. To summarize, the contributions of this paper are threefold:

- The proposed AIT module can adaptively learn what and how much information to transfer for different conversion stages of different audiences for improving the performance of multi-task learning with sequential dependence.
- Combining the Behavioral Expectation Calibrator in the loss function, offline experimental results on both industrial and public real-life datasets clearly demonstrate that the proposed framework achieves significantly better performance compared with state-of-the-art baselines.
- Online experiments also demonstrate significant improvement compared with existing online models, and the source code of the proposed framework has also been released.

2 RELATED WORK

Multi-task learning (MTL) has led to successes in many applications of machine learning, from natural language processing and

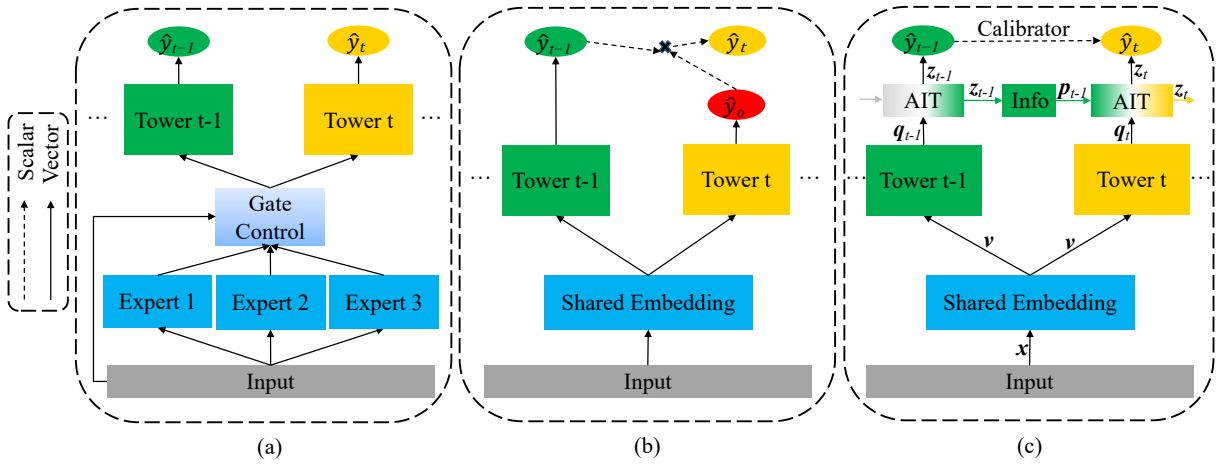

Figure 2: (a) Expert-Bottom pattern. (b) Probability-Transfer pattern. The $\hat{y}_o$ is the non-end-to-end post-click conversion rate and the multi-task loss function only acts on the $\hat{y}_{t-1}$ and $\hat{y}_t$ in the original paper. (c) The proposed Adaptive Information Transfer Multi-task (AITM) framework. For simplicity, only two adjacent tasks are shown in the figure.

speech recognition to computer vision and drug discovery [17]. In this section, we present the main multi-task learning works related to our work in two-fold: the Expert-Bottom pattern and the Probability-Transfer pattern.

As shown in Figure 2 (a), the main idea of the Expert-Bottom pattern is to control how Expert modules are shared across all tasks at the bottom of the multi-task model [14, 16, 19], and the Tower modules at the top handle each task separately. Since complex problems may contain many sub-problems each requiring different experts [1], some Mixture-of-Experts (MoE) models have been proposed one after another [1, 8, 18]. Inspired by the idea, Ma et al. introduced the MoE to the multi-task learning and proposed the Multi-gate Mixture-of-Experts (MMoE) [14] model by the gating networks assembling the experts for different tasks. Zhao et al. explored a variety of soft-parameter sharing techniques such as MMoE to efficiently optimize for multiple ranking objectives for Video recommendation [31]. Tang et al. proposed a Progressive Layered Extraction (PLE) [19] model to separate task-shared experts and task-specific experts explicitly. The Mixture of Sequential Experts (MoSE) model [16] has also been proposed to model sequential user behaviors in multi-task learning. However, the top Tower modules, which often contain richer and more useful information, can not help the tasks to improve each other due to there is no information exchange among them.

Another idea to model task relationships in multi-task learning is to transfer probabilities in the output layers of different tasks [2, 3, 15, 23] as shown in Figure 2 (b). Ma et al. proposed an Entire Space Multi-task Model (ESMM) [15] to transfer probabilities in the output layers by post-impression click-through rate (CTR) multiplying post-click conversion rate (CVR) equals post-impression click-through&conversion rate (CTCVR). Further, more tasks are decomposed for probability transfer in ESM^2 [23]. The Neural Multi-Task Recommendation (NMTR) [2, 3] has also been proposed to extend the Neural Collaborative Filtering (NCF) [7] to multi-task learning and relate the model prediction probability of each task in a cascaded manner. However, as mentioned in Section 1, the Probability-Transfer pattern can only transfer simple probability information via the scalar product, but richer and more useful representations are ignored in the vector space, which results in a great loss. Besides, if any one of the probabilities is not predicted accurately, multiple tasks will be affected.

Other efforts have also utilized the tensor factorization [28], tensor normal priors [13], attention mechanism [12, 30], and so on to solve the multi-task learning. Nevertheless, these above efforts are not specially designed for tasks with sequential dependence, and they can not model the sequential dependence well among audience multi-step conversions.

3 THE MTL RANKING SYSTEM IN MEITUAN APP

In this section, we give an overview of the MTL ranking system in Meituan app. As shown in Figure 3, in our credit card business, we model four tasks except for the passive *impression* step. Among them, the *approval* and *activation* are the main tasks, and the *click* and *application* are the auxiliary tasks. That is because if the audience has only completed the *click* and *application* steps, but the *approval* step has not been completed, then it will cause a waste of resources (e.g., the computing and traffic resources). Because different audiences have different values to different businesses, the traffic that is useless to the credit card business may be useful to other businesses. For this kind of audience, we might as well give the traffic to other businesses that may promote the audience conversion. Therefore, we mainly focus on the last two end-to-end conversion tasks, i.e., *impression*→ *approval* and *impression*→ *activation*. Besides, because the last two tasks have fewer positive samples and the *activation* is delayed feedback, the first two auxiliary tasks with more positive samples can alleviate the class imbalance problem via the Adaptive Information Transfer module.

Meituan Co-Branded Credit Cards are issued in cooperation with different banks, and different banks are in different stages of

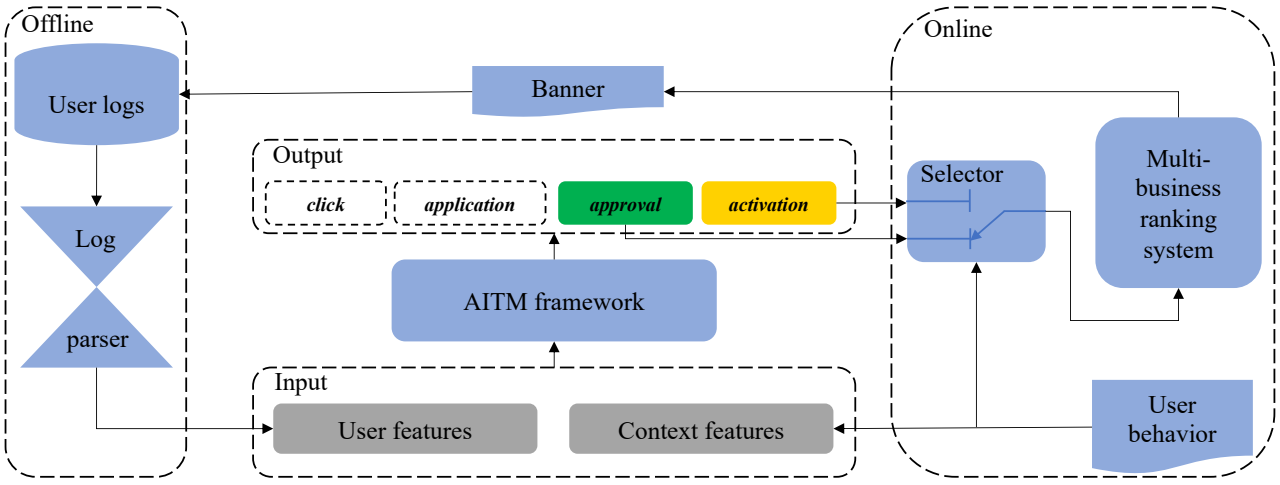

Figure 3: The MTL Ranking System in Meituan app.

business development, so they have different requirements for the *approval* and *activation*. Start-up banks often want to issue more credit cards to quickly occupy the market, while mature banks want to increase the activation rate to achieve rapid profits. Therefore, there is a selector in our system to output different conversion objectives for different banks. The multi-task framework can deal with different business requirements well.

Besides, because different businesses in Meituan all need the traffic to acquire customers for their own business, and the sensitivities of different audiences to different businesses are different, so we can not simply divide the traffic into different businesses. We need a ranking mechanism to maximize the overall gain. The multi-business ranking system ranks the different business scores according to the Equation:

$$score = weight \times \hat{y}, \tag{1}$$

where $\hat{y}$ is the predicted conversion probability for each business, and the $weight$ includes the value of the audience itself, the value of the business itself, and the value of the audience to the business. The business banner with the highest score is shown to the audience.

4 METHODOLOGY

In this section, we first formulate the problem, then we present the details of the proposed framework AITM as shown in Figure 2 (c).

4.1 Problem Formulation

Given the input feature vector $\boldsymbol{x}$, assuming the audience needs T steps to complete the final conversion after *impression* (In Figure 1, $T = 4$). In each conversion step t, if the audience completes the conversion step, the label y_t is 1, otherwise it is 0. The sequential dependence means that $y_1 \geq y_2 \geq \cdots \geq y_T$ ($y_t \in \{0, 1\}, t = 1, 2, \cdots, T$). The multi-task framework needs to predict the end-to-end conversion probability $\hat{y}_t$ of each conversion step t based on the input features $\boldsymbol{x}$:

$$\hat{y}_t = p(y_1 = 1, y_2 = 1, \cdots, y_t = 1 | \boldsymbol{x}). \tag{2}$$

4.2 Adaptive Information Transfer Multi-task (AITM) framework

As shown in Figure 2 (c), given the input feature vector $\boldsymbol{x}$, we embed each entry x_i ($x_i \in \boldsymbol{x}$, $1 \leq i \leq |\boldsymbol{x}|$) to a low dimension dense vector representation $\boldsymbol{v}_i \in \mathbb{R}^d$, where d is the dimension of embedding vectors. The output of the Shared Embedding module is the concatenation of all embedding vectors:

$$\boldsymbol{v} = [\boldsymbol{v}_1; \boldsymbol{v}_2; \cdots; \boldsymbol{v}_{|\boldsymbol{x}|}], \tag{3}$$

where $[\cdot;\cdot]$ denotes the concatenation of two vectors. By sharing the same embedding vectors among all tasks, on the one hand, the framework could learn the embedding vectors with rich positive samples of the former tasks to share information and alleviate the class imbalance of the latter tasks, and reduce the model parameters on the other hand.

Given T tasks, the output of the Tower of each task $t (1 \leq t \leq T)$ is defined as:

$$\boldsymbol{q}_t = f_t(\boldsymbol{v}), \tag{4}$$

where the $f_t(\cdot)$ function is the Tower, $\boldsymbol{q}_t \in \mathbb{R}^k$ and k is the output dimension of the Tower. It should be mentioned that designing a different Tower is not the focus of this paper as we aim at designing an Adaptive Information Transfer module to model the sequential dependence. In fact, our approach is a general framework, and any advanced models (e.g., NFM [6], DeepFM [5], AFM [27], and even the sequence models NHFM [26], DIFM [24]) can be easily integrated into our framework to act as the Tower, making the proposed AITM general and flexible.

For two adjacent tasks $t-1$ and t, the output of the AIT module of the task t is computed as:

$$\boldsymbol{z}_t = AIT(\boldsymbol{p}_{t-1}, \boldsymbol{q}_t), \tag{5}$$

$$where\ \boldsymbol{p}_{t-1} = g_{t-1}(\boldsymbol{z}_{t-1}), \tag{6}$$

$\boldsymbol{z}_{t-1} \in \mathbb{R}^k$ is the output of the AIT module of the task $t-1$, $g_{t-1}(\cdot)$ is the function to learn what information to transfer between the tasks $t-1$ and t, and $\boldsymbol{p}_{t-1} \in \mathbb{R}^k$ is the learned transfer information.

The AIT module is designed to adaptively allocate the weights of the transferred information $\boldsymbol{p}_{t-1}$ and original information $\boldsymbol{q}_t$:

$$z_t = \sum_{\boldsymbol{u} \in \{\boldsymbol{p}_{t-1}, \boldsymbol{q}_t\}} w_u h_1(\boldsymbol{u}), \tag{7}$$

where w_u is the weight which is formulated as:

$$w_u = \frac{exp(\hat{w}_u)}{\sum_u exp(\hat{w}_u)}, \ \hat{w}_u = \frac{< h_2(\boldsymbol{u}), h_3(\boldsymbol{u}) >}{\sqrt{k}}, \tag{8}$$

where $< \cdot, \cdot >$ represents the dot product. $h_1(\cdot)$, $h_2(\cdot)$, and $h_3(\cdot)$ represent the feed-forward networks to project the input information to one new vector representation. There are lots of ways to design $h_1(\cdot)$, $h_2(\cdot)$, and $h_3(\cdot)$. In this paper, we use a simple single-layer MLP (Multi-Layer Perceptron) [4] as $h_1(\cdot)$, $h_2(\cdot)$, and $h_3(\cdot)$. The idea of this kind of attention mechanism is similar to self-attention [22], the $h_1(\cdot)$, $h_2(\cdot)$ and, $h_3(\cdot)$ first learn Value, Query, Key from the same input $\boldsymbol{u}$, respectively. Then, we compute the similarity between Query ($h_2(\cdot)$) and Key ($h_3(\cdot)$) according to Equation (8). Finally, the Value ($h_1(\cdot)$) is weighted via the similarity according to Equation 7. This kind of attention mechanism has been proved to be more effective in the previous works [24, 26, 32].

For the first task without the former task, the out of the AIT module is initialized to:

$$\boldsymbol{z}_1 = \boldsymbol{q}_1. \tag{9}$$

The prediction probability of each task t is:

$$\hat{y}_t = sigmoid(MLP(\boldsymbol{z}_t)), \tag{10}$$

where the MLP is used to project the $\boldsymbol{z}_t$ to the output space.

4.3 Behavioral Expectation Calibrator and Joint Opratortimization for MTL

For classification tasks, we need to minimize the *cross-entropy* loss of all tasks:

$$\mathcal{L}_{ce}(\theta) = -\frac{1}{N} \sum_{t=1}^{T} \sum_{(\boldsymbol{x}, y_t) \in \mathcal{D}}^{N} ((y_t \log \hat{y}_t + (1 - y_t) \log(1 - \hat{y}_t)), \tag{11}$$

where N is the number of samples in the entire sample space $\mathcal{D}$, y_t is the label of the t-th task and θ is the parameter set in the MTL framework.

Besides, because of the sequential dependence, the former task should have a higher end-to-end conversion probability than the latter for the same audience, i.e., $\hat{y}_{t-1} \geq \hat{y}_t$. We design a Behavioral Expectation Calibrator to minimize the following objective. On the one hand, it makes the model results more satisfy the real production constraint, on the other hand, it provides more accurate end-to-end conversion identification:

$$\mathcal{L}_{lc}(\theta) = \frac{1}{N} \sum_{t=2}^{T} \sum_{\boldsymbol{x} \in \mathcal{D}}^{N} max(\hat{y}_t - \hat{y}_{t-1}, 0). \tag{12}$$

If $\hat{y}_t > \hat{y}_{t-1}$, the $\mathcal{L}_{lc}(\theta)$ will output a positive penalty term, otherwise output 0.

The final loss function $\mathcal{L}(\theta)$ of the AITM combines the two components to a unified multi-task learning framework:

$$\mathcal{L}(\theta) = \mathcal{L}_{ce}(\theta) + \alpha \mathcal{L}_{lc}(\theta), \tag{13}$$

where α controls the strength of the Behavioral Expectation Calibrator component.

Table 1: Summary statistics for the datasets. "%Positive" represents the percentage of positive samples in the train set over each task.

Dataset	#Task	#Train	#Validation	#Test	%Positive(%)
Industrial	4	20M	3M	26M	23.29/1.84/1.30/1.00
Public	2	38M	4.2M	43M	3.89/0.02

The framework is implemented using TensorFlow[2] and trained through stochastic gradient descent over shuffled mini-batches with the Adam [10] update rule.

5 EXPERIMENTS

In this section, we perform experiments to evaluate the proposed framework against various baselines on both industrial and public real-world datasets. We first introduce the datasets, evaluation protocol, and baseline methods. Finally, we present our experimental results and analysis.

5.1 Datasets

- **Industrial dataset:** The industrial dataset contains all samples that are shown a banner of Meituan Co-Branded Credit Cards over a continuous period of time. We divide the training, validation, and test sets in chronological order. We downsample the *activation* negative samples for each bank to keep the proportion of positive samples to be 1% overall except for the test set. Because it is necessary to evaluate the performance of the model on the test set that meets the real data distribution. Four tasks (i.e., *click*, *application*, *approval*, *activation*) are contained in the dataset.
- **Public dataset:** The public dataset is the Ali-CCP (Alibaba Click and Conversion Prediction) [15] dataset[3]. We use all the single-valued categorical features. Two tasks (i.e., *click*, *purchase*) are contained in the dataset. We randomly take 10% of the train set as the validation set to verify the convergence of all models.

For these two datasets, we filter the features whose frequency less than 10. The statistics of these datasets are shown in Table 1.

5.2 Evaluation Protocol

In the offline experiments, to evaluate the performance of the proposed AITM framework and the baselines, we follow the existing works [14, 15, 19, 23] to use the standard metric: **AUC** (Area Under ROC). In ranking tasks, AUC is a widely used metric to evaluate the ranking ability. The mean and standard deviation (std) are reported over five runs with different random seeds. In the online A/B test, we use the **end-to-end conversion rate** to evaluate the performance more intuitively. On all datasets, we report the AUC of end-to-end tasks, which are directly optimized in their loss functions. Besides, we only report the metrics on the focused main tasks (i.e., the *approval* and *activation* tasks) over the industrial dataset.

[2]https://www.tensorflow.org/
[3]https://tianchi.aliyun.com/datalab/dataSet.html?dataId=408

Table 2: The AUC performance (mean±std) on the industrial and public datasets. The Gain means the mean AUC improvement compared with the LightGBM. Underlined results indicate the best baselines over each task. "*" indicates that the improvement of the proposed AITM is statistically significant compared with the best baselines at p-value < 0.05 over paired samples t-test, and "" indicates that the p-value < 0.01.**

Model	Industrial dataset				Public dataset			
	approval AUC	*activation* AUC	Gain		*click* AUC	*purchase* AUC	Gain	
LightGBM	0.8392±0.0011	0.8536±0.0035	-	-	0.5837±0.0005	0.5870±0.0038	-	-
MLP	0.8410±0.0010	0.8602±0.0014	+0.0018	+0.0066	0.6048±0.0013	0.5806±0.0035	+0.0211	-0.0064
ESMM	0.8443±0.0028	0.8691±0.0025	+0.0051	+0.0155	0.6022±0.0020	0.6291±0.0061	+0.0185	+0.0421
OMoE	0.8438±0.0022	0.8714±0.0009	+0.0046	+0.0178	**0.6049±0.0020**	0.6405±0.0041	**+0.0212**	+0.0535
MMoE	0.8444±0.0026	0.8705±0.0009	+0.0052	+0.0169	0.6047±0.0017	0.6420±0.0031	+0.0210	+0.0550
PLE	0.8518±0.0006	0.8731±0.0016	+0.0126	+0.0195	0.6039±0.0014	0.6417±0.0013	+0.0202	+0.0547
AITM	**0.8534±0.0011****	**0.8770±0.0005***	**+0.0142**	**+0.0234**	0.6043±0.0016	**0.6525±0.0024****	+0.0206	**+0.0655**

5.3 Baseline Methods

We compare the proposed method with the following competitive and mainstream models:

- **LightGBM** [9]: LightGBM is a gradient boosting framework that uses tree based learning algorithms. LightGBM is being widely-used in many winning solutions of machine learning competitions[4].
- **MLP** [4]: We use the base structure of our AITM framework as the single task model. It is a Multi-Layer Perceptron.
- **ESMM** [15, 23]: The ESMM and ESM^2 with Probability-Transfer pattern are designed for solving the non-end-to-end post-click conversion rate via training on the entire space to relieve the sample selection bias problem.
- **OMoE** [14]: The OMoE with Expert-Bottom pattern integrates experts via sharing one gate among all tasks.
- **MMoE** [14]: The MMoE with Expert-Bottom pattern is designed to integrate experts via multiple gates in the Gate Control as shown in Figure 2 (a).
- **PLE** [19]: The Progressive Layered Extraction (PLE) with Expert-Bottom pattern separates task-shared experts and task-specific experts explicitly. This is the state-of-the-art method under different task correlations.

5.4 Performance Comparison

5.4.1 ***Offline Results.*** In this subsection, we report the AUC scores and gains of all models on the offline test set. As mentioned in Section 3, we only focus on the last two main end-to-end conversion tasks on the industrial dataset. The results of *approval* AUC and *activation* AUC are shown in Table 2. From these results, we have the following insightful observations:

- The MLP obtains 0.0018 and 0.0066 AUC gains on two tasks, respectively, compared with the tree-based model LightGBM, which indicates the fitting ability of neural network models on large-scale datasets.
- Compared with the single-task models LightGBM and MLP, the multi-task models ESMM, OMoE, MMoE, PLE and AITM obtain more gains by introducing the multi-task information in the neural networks.
- The Probability-Transfer pattern-based ESMM achieves a relatively small improvement due to only simple probability information is transferred between adjacent tasks.
- The Expert-Bottom pattern-based models obtain further performance improvement by controlling the shared information among different tasks. However, neither of the one-gate and multi-gate models is a clear winner on this dataset.
- The PLE obtains the best performance among these baselines on the two tasks via separating task-shared experts and task-specific experts explicitly.
- Our AITM achieves significant improvement compared with various state-of-the-art baseline models, which shows the AIT module is effective and could bring more gains on sequential dependence tasks.

The results on the public dataset are also shown in Table 2. From these results, we have the following findings, which are not the same as above:

- Serious class imbalance on the *purchase* task (the proportion of positive samples is 0.02% as shown in Table 1) leads to the poor performance of the two single-task models, i.e., the LightGBM and MLP.
- The MLP obtains similar performance improvement compared with multi-task models on the *click* task. In other words, the multi-task models seem to have no significant improvement on the *click* task. This may be because there are only two tasks in this dataset, and no other task can provide more information before the *click* task. 3.89% positive samples in the *click* task are relatively abundant as shown in Table 1.
- The Expert-Bottom pattern shows better performance than the Probability-Transfer pattern on the *purchase* task with serious class imbalance. Besides, our AITM can explicitly use the rich positive sample information of the former *click* task to alleviate the class imbalance of the current *purchase* task and achieve the best performance. On the other hand, it also shows the generalization ability of the proposed AITM.

5.4.2 ***Online Results.*** The proposed framework is trained offline and regularly updated. The pre-trained model is deployed in

[4]https://github.com/microsoft/LightGBM

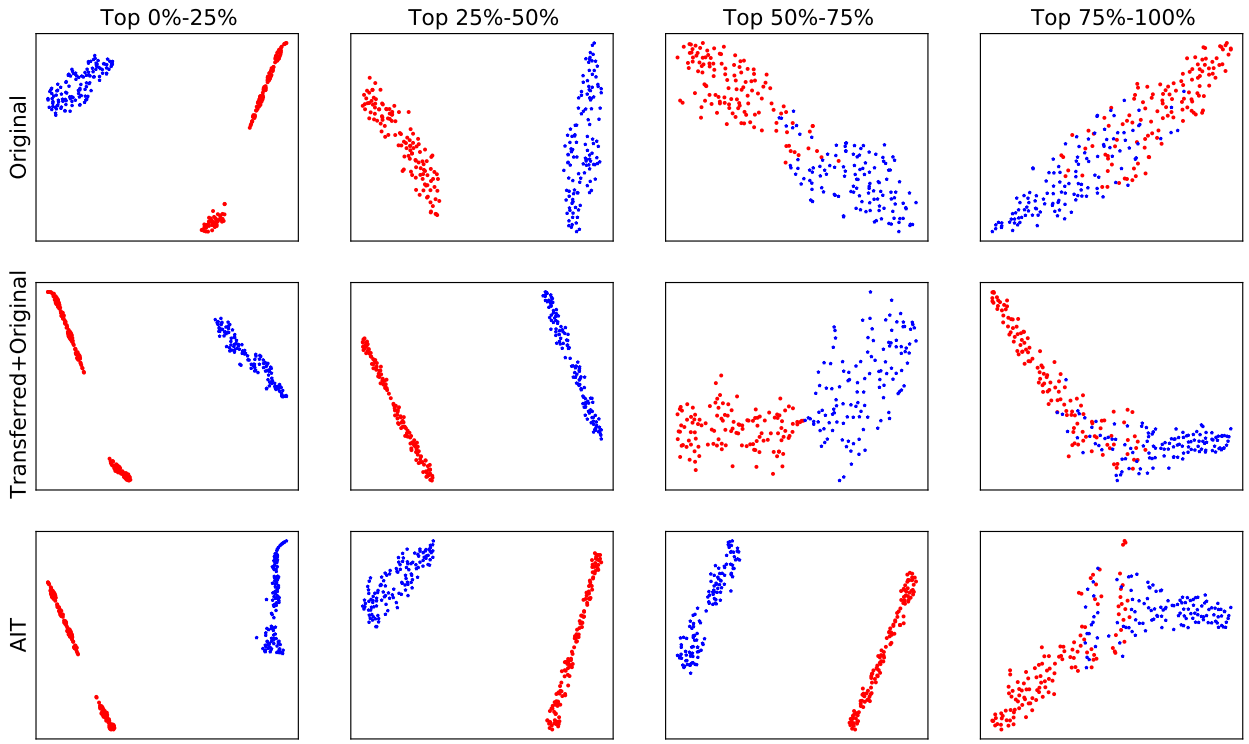

Figure 4: The t-SNE visualization at different conversion score rankings of the original information q_t, transferred plus original information $p_{t-1} + q_t$ and the information z_t learned by the AIT on the *activation* task.

Table 3: Online A/B test results.

Model	Gain	
	approval	*activation*
MLP vs LightGBM	+16.95%	+17.55%
AITM vs MLP	+25.00%	+42.11%

Meituan app by the TF Serving[5], to real-timely show a banner to the audience with a high end-to-end *approval* or *activation* conversion rate for Meituan Co-Branded Credit Cards. Due to business competition, user experience, and delayed feedback (T+14) of the *activation* task, we can not deploy all models to the online system. With the development of our business, we have successively deployed LightGBM, MLP, and AITM to the online system. These models serve tens of millions of traffic every day. A/B test is carried out for every two models with the same traffic for two consecutive weeks (It takes four weeks for all feedback to be received for every two models). The online A/B test results are shown in Table 3. Compared with LightGBM, the *impression* → *approval* conversion rate of MLP increases by 16.95% and the *impression* → *activation* conversion rate increases by 17.55%. The AITM further increases the *impression* →*approval* and *impression* → *activation* conversion rate by 25.00%, 42.11% compared with MLP, respectively. Now, the AITM has provided real-time prediction for all traffic in our system.

[5]https://github.com/tensorflow/serving

Besides, our system is computationally efficient, and the TP999, TP9999 of the real-time prediction is less than 20ms, 30ms in the system every day, respectively, which can meet the requirement of real-time solutions.

5.5 Ablation Study

In this subsection, we perform the ablation study of the AIT module and the number of the tasks.

Firstly, we randomly sample 500 *activation* positive and negative samples in the test set, respectively. The *activation* prediction scores of positive samples are ranked in descending order, while those of negative samples are in ascending order. We plot the original information $\boldsymbol{q_t}$, transferred plus original information $\boldsymbol{p_{t-1} + q_t}$ and the information $\boldsymbol{z_t}$ learned by the AIT on the *activation* task in Figure 4 via the t-SNE (t-distributed Stochastic Neighbor Embedding [21]). From the visualization, we can obtain the following inspiring observations (Similar results can also be observed in the *approval* task, we list it in the appendix due to the space limitation):

- When the prediction scores of the AITM are very confident (see the Top 0% – 50% in Figure 4), the three components (i.e., the Original, Transferred+Original, and AIT) can accurately identify positive and negative samples.
- With the confidence of the prediction scores of the AITM decreases (see the Top 50% – 100% in Figure 4), it is difficult to identify positive and negative samples only using the original information. The transferred plus original information

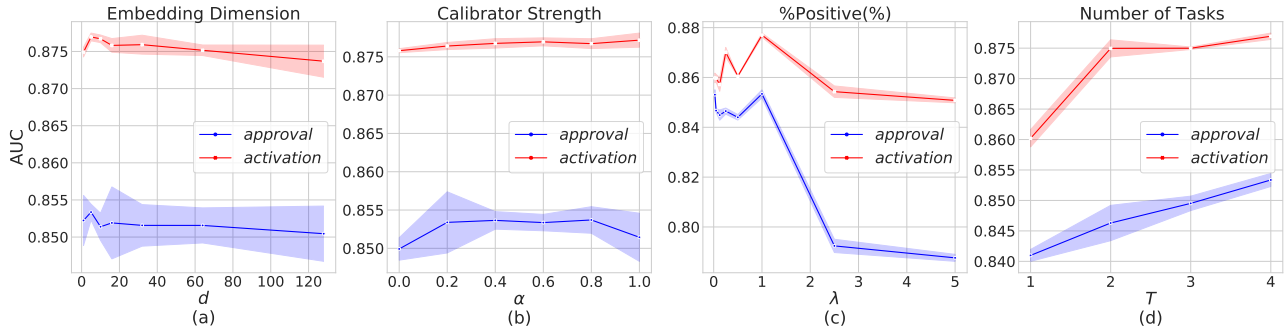

Figure 5: The mean AUC performance in different experiment settings, the shaded part represents the standard deviation. (a) The impact of the embedding dimension d. (b) The impact of the strength α of the Behavioral Expectation Calibrator. (c) The impact of the proportion λ of positive samples. (d) Ablation Study of the number T of tasks.

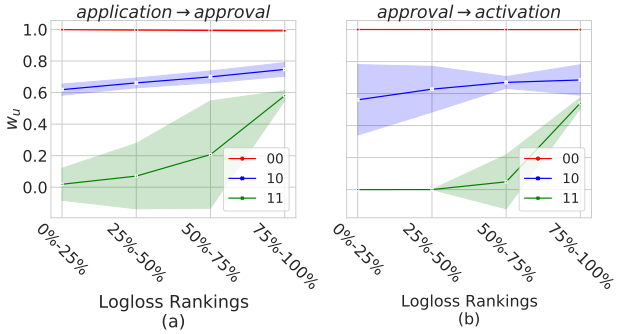

Figure 6: The mean weight w_u of the transferred information $\boldsymbol{p}_{t-1}$ over different conversion stages (00/10/11) at different logloss rankings, the shaded part represents the standard deviation. (a) The tasks *application* → *approval*. (b) The tasks *approval* → *activation*.

improves the performance compared with only the original information, which indicates that information transfer could improve the performance of tasks with sequential dependence in multi-task learning.

- The AIT module could adaptively learn what and how much information to transfer among audience multi-step conversions via the multi-task framework, so the AIT further improves the performance compared with the transferred plus original information under low confidence.

Besides, we study the impact of the number of tasks as shown in Figure 5 (d). We perform the experiments over tasks *activation* (*approval*), *approval* → *activation*, *application* → *approval* → *activation*, *click* → *application* → *approval* → *activation*, respectively. More tasks with more positive sample information and transferred information greatly improve the performance.

5.6 Case Study

In order to understand how much information the AIT module transfers for different conversion stages, we extract the weight w_u in Equation (8) of the transferred information $\boldsymbol{p}_{t-1}$ and show it in Figure 6. We first randomly sample 40,000 test samples. In Figure 6 (a), we divide the samples into three groups according to the *application* and *approval* labels of 00/10/11, and rank the top 500 samples in each group according to the logloss of each sample in ascending order. Figure 6 (b) is the same except for the tasks are *approval* and *activation*. From Figure 6, we have the following interesting findings:

- Because when the label of the former task is 0, the label of the latter task must also be 0, we can see that at this time the former task transfers very strong information to the latter task (the weight is close to 1 of the red lines in Figure 6).
- When the label of the former task is 1, the label of the latter task is uncertain. When the label of the latter task is 1, there is little information is transferred from the former task (the green lines in Figure 6), which indicates that the latter task mainly identifies positive samples based on the task itself.
- When the label of the former task is 1, with the prediction becomes worse (the logloss rankings from 0%-25% to 75%-100%), the weight of the transferred information gradually increases (the blue and green lines in Figure 6), which indicates that the prediction result of the latter task is misled by the former task.

From the above results, we could see that the AIT module can learn how much information to transfer between two adjacent tasks.

5.7 Hype-parameter Study

In order to study the impact of hype-parameters and the stability on the performance of the AITM, we perform the hyper-parameter study.

Firstly, considering the embedding dimension d, we vary the embedding dimension as $[1, 5, 10, 16, 32, 64, 128]$, the results are shown in Figure 5 (a). We can see that the performance of the AITM is not very sensitive to the embedding dimension. The embedding dimension is related to the complexity and capability of the model. Usually, smaller embedding dimension may fit the data distribution insufficiently, especially if the numbers of samples and features are large. While a larger embedding dimension increases the complexity of the model and requires more samples and features to fit, a proper embedding dimension can achieve the best performance

[25]. Making a trade-off between model complexity and capability, we finally set $d = 5$ as the embedding dimension in all experiments.

Secondly, we study the impact of the strength α of the Behavioral Expectation Calibrator as shown in Figure 5 (b). There are performance fluctuations (the seesaw phenomenon) among four different tasks. However, the Behavioral Expectation Calibrator brings the improvement of the overall performance. We finally set $\alpha = 0.6$ as the weight in all the experiments.

Thirdly, we study the impact of the proportion λ of positive samples in the industrial dataset. We downsample the *activation* negative samples to keep the proportion λ of positive samples to be $[0.025\%, 0.05\%, 0.125\%, 0.25\%, 0.5\%, 1\%, 2.5\%, 5\%]$ in the train set, respectively, and report the AUC performance on the entire test set in Figure 5 (c). On the one hand, if audiences do not apply for the credit card at present, it does not mean that they will not apply for the card in the future, so we can not use too many negative samples for training. On the other hand, it can be seen that when λ is too large, the performance of the model drops sharply. This is because too much *activation* negative sample information is lost. Besides, excessive downsampling of the *activation* negative samples also leads to the loss of *approval* positive samples. We finally downsample the *activation* negative samples to keep the proportion λ of positive samples to be 1%. This setting is applied to all models.

Combining the performance in Table 2 and Figure 5, we can see that even without the best parameters, the AITM is still superior to other baselines in most cases. In other words, the performance of the AITM stays stable in a large range of values of hyper-parameters and is not very sensitive to the hyper-parameters.

6 CONCLUSION

In this paper, we proposed an Adaptive Information Transfer Multi-task (AITM) framework to model the sequential dependence among audience multi-step conversions. The proposed Adaptive Information Transfer (AIT) module combining the Behavioral Expectation Calibrator in the loss function could learn what and how much information to transfer for different conversion stages for improving the performance of multi-task learning with sequential dependence. Offline and online experimental results demonstrate significant improvement compared with state-of-the-art baseline models.

7 ACKNOWLEDGMENTS

Fuzhen Zhuang is supported by the National Natural Science Foundation of China under Grant Nos. U1836206, U1811461. Besides, we thank Zhenhua Zhang, Kun Chen, Chang Qu, Qiu Xiong, and KangMing Yu for their support and valuable suggestions.

REFERENCES

[1] David Eigen, Marc'Aurelio Ranzato, and Ilya Sutskever. 2014. Learning factored representations in a deep mixture of experts. In *ICLR*.
[2] Chen Gao, Xiangnan He, Dahua Gan, Xiangning Chen, Fuli Feng, Yong Li, Tat-Seng Chua, and Depeng Jin. 2019. Neural multi-task recommendation from multi-behavior data. In *ICDE*. 1554–1557.
[3] Chen Gao, Xiangnan He, Danhua Gan, Xiangning Chen, Fuli Feng, Yong Li, Tat-Seng Chua, Lina Yao, Yang Song, and Depeng Jin. 2019. Learning to Recommend with Multiple Cascading Behaviors. *TKDE* (2019).
[4] Matt W Gardner and SR Dorling. 1998. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric environment* (1998).
[5] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. In *IJCAI*.
[6] Xiangnan He and Tat-Seng Chua. 2017. Neural factorization machines for sparse predictive analytics. In *SIGIR*.
[7] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *TheWebConf*. 173–182.
[8] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation* 3, 1 (1991), 79–87.
[9] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. In *NeurIPS*. 3146–3154.
[10] Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*, Vol. 5.
[11] Yixuan Li, Jason Yosinski, Jeff Clune, Hod Lipson, and John E Hopcroft. 2016. Convergent learning: Do different neural networks learn the same representations?. In *ICLR*.
[12] Shikun Liu, Edward Johns, and Andrew J Davison. 2019. End-to-end multi-task learning with attention. In *CVPR*. 1871–1880.
[13] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and S Yu Philip. 2017. Learning multiple tasks with multilinear relationship networks. In *NeurIPS*. 1594–1603.
[14] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. 2018. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *KDD*. 1930–1939.
[15] Xiao Ma, Liqin Zhao, Guan Huang, Zhi Wang, Zelin Hu, Xiaoqiang Zhu, and Kun Gai. 2018. Entire space multi-task model: An effective approach for estimating post-click conversion rate. In *SIGIR*. 1137–1140.
[16] Zhen Qin, Yicheng Cheng, Zhe Zhao, Zhe Chen, Donald Metzler, and Jingzheng Qin. 2020. Multitask Mixture of Sequential Experts for User Activity Streams. In *KDD*. 3083–3091.
[17] Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098* (2017).
[18] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *ICLR*.
[19] Hongyan Tang, Junning Liu, Ming Zhao, and Xudong Gong. 2020. Progressive Layered Extraction (PLE): A Novel Multi-Task Learning (MTL) Model for Personalized Recommendations. In *RecSys*. 269–278.
[20] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. 2014. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474* (2014).
[21] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research* 9, 86 (2008), 2579–2605.
[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*. 5998–6008.
[23] Hong Wen, Jing Zhang, Yuan Wang, Fuyu Lv, Wentian Bao, Quan Lin, and Keping Yang. 2020. Entire Space Multi-Task Modeling via Post-Click Behavior Decomposition for Conversion Rate Prediction. In *SIGIR*. 2377–2386.
[24] Dongbo Xi, Bowen Song, Fuzhen Zhuang, Yongchun Zhu, Shuai Chen, Tianyi Zhang, Yuan Qi, and Qing He. 2021. Modeling the Field Value Variations and Field Interactions Simultaneously for Fraud Detection. In *AAAI*.
[25] Dongbo Xi, Fuzhen Zhuang, Yanchi Liu, Jingjing Gu, Hui Xiong, and Qing He. 2019. Modelling of Bi-Directional Spatio-Temporal Dependence and Users' Dynamic Preferences for Missing POI Check-In Identification. In *AAAI*, Vol. 33. 5458–5465.
[26] Dongbo Xi, Fuzhen Zhuang, Bowen Song, Yongchun Zhu, Shuai Chen, Dan Hong, Tao Chen, Xi Gu, and Qing He. 2020. Neural Hierarchical Factorization Machines for User's Event Sequence Analysis. In *SIGIR*. 1893–1896.
[27] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. 2017. Attentional factorization machines: Learning the weight of feature interactions via attention networks. In *IJCAI*.
[28] Yongxin Yang and Timothy Hospedales. 2017. Deep multi-task representation learning: A tensor factorisation approach. In *ICLR*.
[29] Matthew D Zeiler and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In *ECCV*. 818–833.
[30] Jiejie Zhao, Bowen Du, Leilei Sun, Fuzhen Zhuang, Weifeng Lv, and Hui Xiong. 2019. Multiple Relational Attention Network for Multi-task Learning. In *KDD*. 1123–1131.
[31] Zhe Zhao, Lichan Hong, Li Wei, Jilin Chen, Aniruddh Nath, Shawn Andrews, Aditee Kumthekar, Maheswaran Sathiamoorthy, Xinyang Yi, and Ed Chi. 2019. Recommending what video to watch next: a multitask ranking system. In *RecSys*. 43–51.
[32] Yongchun Zhu, Dongbo Xi, Bowen Song, Fuzhen Zhuang, Shuai Chen, Xi Gu, and Qing He. 2020. Modeling Users' Behavior Sequences with Hierarchical Explainable Network for Cross-domain Fraud Detection. In *TheWebConf*. 928–938.

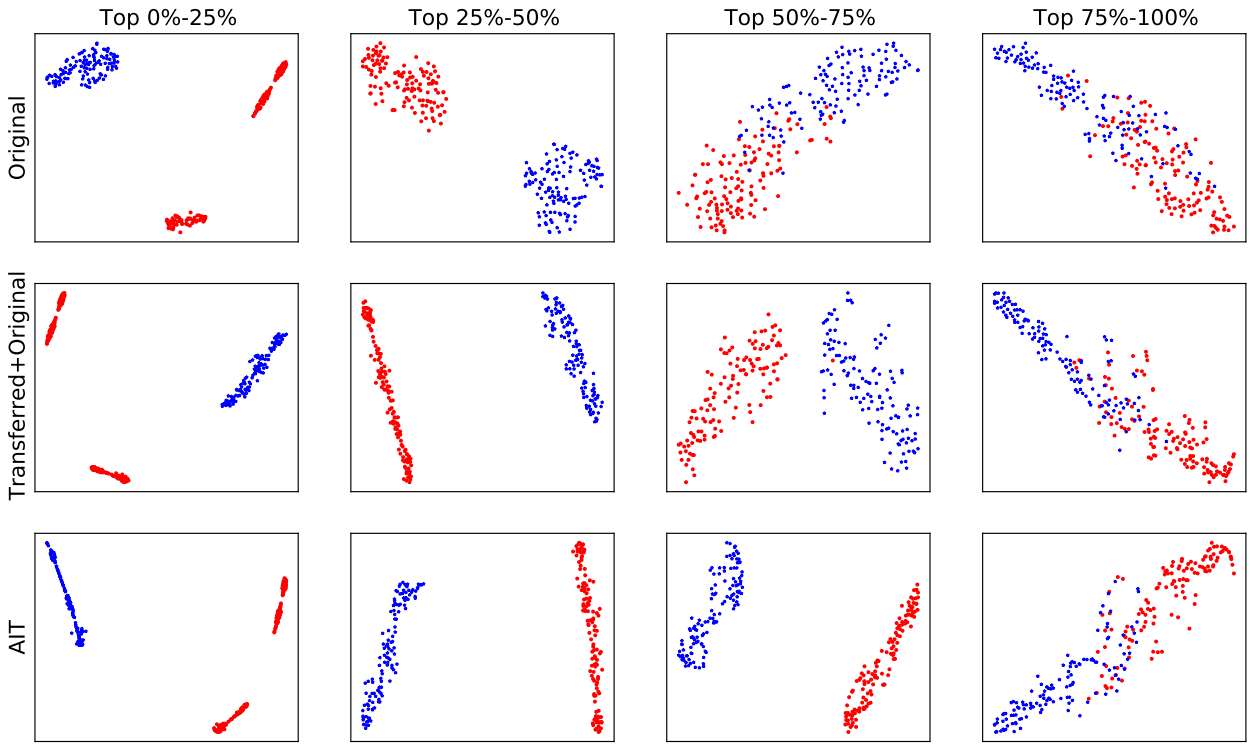

Figure 7: The t-SNE visualization at different conversion score rankings of the original information q_t, transferred plus original information $p_{t-1} + q_t$ and the information z_t learned by the AIT on the *approval* task.

Table 4: The summary of the hyper-parameters in multi-task models, which includes ESMM, OMoE, MMoE, PLE and AITM. The T is the number of tasks. Except for these listed, other parts of these models that involve MLP are all single-layer.

Hyper-parameter	Value
Optimizer	Adam
Batch size	2000
Learning rate	1e-3
L2 regularization	1e-6
Embedding dimension	5
Dimensions of layers in the MLP-Expert of Expert-Bottom pattern	[64,32,16]×2×T
Dimensions of layers in the MLP-Tower of Probability-Transfer pattern	[128,64,32]×T
Dropout rate in each layers	[0.1,0.3,0.3]
Activation function in MLP	Relu

A APPENDIX

A.1 Reproducibility Information

For the LightGBM model, the learning rate, feature fraction, bagging fraction, bagging frequency, max bin, number of leaves, boosting type are set as 0.1, 0.9, 0.7, 5, 2000, 70, gbdt, respectively, which are chosen according to the grid search on the validation set.

For the neural network-based models, to verify the generalization ability of different models and compare them fairly, different neural network-based models use the same common hyper-parameters on two datasets considering the empirical values and computational efficiency. For the industrial dataset, we downsample the *activation* negative samples to keep the proportion λ of positive samples at 1% except for the test set. For both industrial and public datasets, we use: embedding dimension $d = 5$, the output dimension of the Tower is $k = 32$, the strength of the Behavioral Expectation Calibrator is $\alpha = 0.6$. We only fine-tune the hyper-parameters of d, α, and λ according to grid search on the validation set. Besides, the Tower $f_t(\cdot)$ is a three-layer MLP with dimension $[128, 64, 32]$, the $g_t(\cdot)$, $h_1(\cdot)$, $h_2(\cdot)$ and $h_3(\cdot)$ are all single-layer MLP with dimension 32. For a fair comparison, we try our best to ensure that the main architecture of different multi-task models is consistent. The summary of different multi-task models is shown in Table 4. We conduct experiments of all models with NVIDIA Tesla V100 GPU with 16G memory.

A.2 More Experiments

These inspiring observations, which are described in Subsection 5.5 and shown in Figure 4 on the *activation* task, can also be observed on the *approval* task. Firstly, we randomly sample 500 *approval* positive and negative samples in the test set, respectively.

The *approval* prediction scores of positive samples are ranked in descending order, while those of negative samples are in ascending order. Then, we plot the original information $\boldsymbol{q}_t$, transferred plus original information $\boldsymbol{p}_{t-1} + \boldsymbol{q}_t$ and the information $\boldsymbol{z}_t$ learned by the AIT on the *approval* task via the t-SNE. We show the results in Figure 7.

A.3 Data Collection and Privacy Protection

The industrial dataset contains all samples that are shown a banner of Meituan Co-Branded Credit Cards over a continuous period of time. In the traditional credit card business, the *approval* step is usually not real-time. However, in our online credit card application, Meituan and the card-issuing bank make two-level real-time risk judgment, which makes the *approval* step is almost real-time. Besides, the *activation* step requires the user to have received the mailed credit card and go to the bank to activate it or make an appointment with a salesman to activate it at home. Therefore, the samples used are shown a banner at least 14 days ago, so as to ensure the accuracy of the *activation* label. The features used include context features and user statistics features. We don't use any features that can locate specific users and involve user privacy. The accurate end-to-end conversion identification can disturb users as little as possible and improve the user experience.