Using Multi-Task Learning Models for Click Through Rate and Conversion Rate Predictions

Paper ID: 3142

Abstract

Click Through Rate (CTR) and Conversion Rate (CVR) predictions are key challenges in different bidding models of online advertising. Current literature mainly focuses on predicting either CTR or CVR by modeling various kinds of feature learning. In this paper, we consider optimizing both CTR and CVR predictions through deep neural networks (DNNs) from multi-task learning (MTL) perspective. To achieve the goal, we propose two basic MTL models, *feature sharing* and *feature transfer* model, which have distinct characteristics. Feature sharing model treats two tasks fairly while feature transfer utilizes an auxiliary task to help the learning of the main task. To demonstrate the effectiveness of our MTL models and reveal how click and conversion labels influence each other, we conduct extensive ablation experiments on both commercial and synthetic datasets. The first preliminary experiment is based on a raw commercial data, and we conclude several practical lessons on making *homogenous data* which is more applicable for MTL models. The second experiment is based on the homogenous data and a synthetic data, and the consistent results demonstrate the superiority of our MTL models as well as the rationality of our modeling on users' impression-to-conversion action path. Finally, we present a thorough analysis to depict the inherent reasons behind and obtain several instructive conclusions, which may shed light on the application of MTL on CTR/CVR predictions in industry.

1 Introduction

Online advertising is a fast growing multi-billion business. It brings large amounts of revenue for the publisher platform by delivering targeted advertisements (ads) to users on mobile applications or web pages (Chen et al. 2016). The publisher's revenue is directly related to the ranking and selection of ad candidates to display, in which precisely predicting the click though rate (CTR) or conversion rate (CVR) are the key challenges.

In the common *cost per click* (CPC) model, the advertisers pay for an ad *impression* to the publisher only when a user has clicked this ad. Thus, the publisher's expected revenue for an impression is bid$\times$CTR where bid is the price that the advertiser agrees to pay and is negotiated in advance (Lee et al. 2012). In addition to the CPC model, *cost per action/conversion* (CPA) model is more popular for direct response advertisers. It reduces the risk of advertisers even further since they pay for an impression only when the desired conversion event(s) (such as downloading apps or registering accounts) have occurred, where the conversion event is defined by the advertisers. In CPA model, the publisher's expected revenue is bid$\times$CVR. Thus, the precision of predicted CTR/CVR is vital to the publisher since it is directly related to the revenue.

Figure 1 illustrates a typical user's impression-to-conversion action path. It is composed of 3 stages: impression, click, and conversion. The input features in the impression stage can be divided into three groups: ad features, user features, and context features (Rosales, Cheng, and Manavoglu 2012). The click label denoted by $y_1 \in \{0, 1\}$ represents whether a user has clicked the ad. The conversion label $y_2 \in \{0, 1\}$ indicates whether there is a conversion event (e.g., downloading apps) associated to the clicked impression. There may also exist more levels of conversions, which is dependent on the rules made by the advertisers.

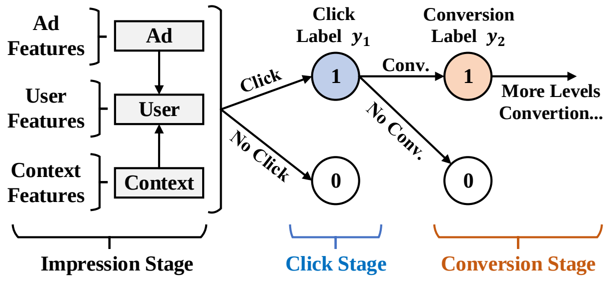

Figure 1: The impression-to-conversion path.

Previous literature mainly considers optimizing either CTR or CVR prediction by modeling various kinds of feature learning. Linear models such as logistic regression (LR) (Cheng and Cantú-Paz 2010; Cheng et al. 2012) and probit regression (Graepel et al. 2010) are widely applied since they are easy to implement and interpret. Hybrid models using the output of tree (He et al. 2014) or factorization machines (FM) (Rendle 2012) as the input of LR improve the prediction accuracy a lot. Recently, (Zhang, Du, and Wang 2016; Cheng et al. 2016; Guo et al. 2017) proposed to utilize deep neural networks (DNNs) to learn high-order feature interactions and achieved a good performance. However, current literature mainly treats predicting CTR and CVR as independent tasks. Intuitively, the click labels and conversion labels are highly relevant with each other and follow a hierarchical dependent relationship: the conversion may happen only when there is a click event. More specially, if there is no click event, no conversion happens[1]. It strongly motivates us to figure out the following things. 1) whether click labels are helpful when predicting CVR? 2) can conversion labels help the prediction of CTR? 3) how do click labels and conversion labels influence each other in practice? To find the answers, an intuitive way for us is to adopt multi-task learning (MTL) which solves multiple learning tasks simultaneously for CTR and CVR predictions.

To the best of our knowledge, there is very limited literature involved in predicting CTR/CVR at the same time. In this paper, we consider optimizing the predictions of both CTR and CVR through DNNs from the MTL perspective. Firstly, we propose two basic feature-based MTL models with distinct characteristics (Zhang and Yang 2017).

- **Feature Sharing Model.** Feature sharing is a 4-layer DNN model with two fairly treated targets. For CTR/CVR predictions problem, we design a novel loss function for it based on the hierarchical relationship between click and conversion labels.
- **Feature Transfer Model**. Feature transfer is a hybrid DNN model which adopts separate hidden layers for each task. It focuses on the performance improvement of the main task by transferring features from an auxiliary task.

To demonstrate the effectiveness of MTL models and reveal how click and conversion labels influence each other, we conduct extensive ablation experiments for our MTL models on both commercial and synthetic datasets.

1. **Preliminary Experiments.** We first give a brief description of our commercial dataset and experimental settings, and conduct preliminary experiments on the raw dataset. Based on the diverse performance results, we conclude several practical lessons on making *homogenous data*, which is more applicable for MTL models.
2. **Ablation Study.** We conduct several ablation experiments for our MTL models on the homogenous data, and give an analysis on how our MTL models help on different targets.
3. **Synthetic Dataset.** To further make our ideas and models more convincing, we make a synthetic dataset modeling users' impression-to-conversion action path. The consistent results with that on commercial dataset demonstrate the effectiveness of our MTL models.
4. **Model Interpretation.** Based on the previous experimental results on various datasets, we present a thorough analysis to depict the inherent reasons behind and obtain several instructive conclusions.

[1]Under some special business scenarios, the conversion may happen even though there is no click event, but we do not consider such cases in this paper.

2 Related Work

2.1 CTR/CVR Predictions

Since online advertising has taken much share of the market, in order to maximize the publisher's revenue or reduce the advertisers' risk, many works related to CTR or CVR predictions have been published. Current works mainly focus on how to learn not only low-order but also high-order feature interactions.

Linear Models. In CTR/CVR prediction domain, the most commonly applied models are linear models, such as logistic regression (Cheng and Cantú-Paz 2010; Cheng et al. 2012; Richardson, Dominowska, and Ragno 2007; Rosales, Cheng, and Manavoglu 2012), probit regression (Graepel et al. 2010) and piece-wise linear model (Gai et al. 2017). In addition to the basic linear approaches, (Rendle 2012) proposed factorization machines (FM) to map the raw categorical features into latent embedding vectors. (He et al. 2014) proposed to use the output of boosted decision trees as the input of LR. Both of them are regarded as the most successful models in industry (Zhang, Du, and Wang 2016).

DNN Models. Recently, DNN based models are proposed to learn higher levels of feature interactions in CTR prediction. (Zhang, Du, and Wang 2016) proposed Factorization Machine supported Neural Network (FNN) in which DNN are used to learn non-trivial data patterns of embedding layer in FM. Similarly, (Qu et al. 2016) introduced Product-based Neural Networks (PNN), in which a product layer is adopted to capture interactive patterns between inter-field categories.

Hybrid Models. To utilize both low and high order feature interactions, hybrid models are also proposed recently. As a widely accepted model in industry, Wide & Deep (Cheng et al. 2016) proposed by Google uses a linear ("wide") part and a DNN part which are jointly trained to combine the benefits of memorization and generalization in recommender systems. DeepFM (Guo et al. 2017) is another hybrid model integrating an FM part and DNN part. It can be trained end-to-end without any feature engineering.

2.2 Multi-Task Learning

Multi-Task learning (MTL) is a learning paradigm which aims to leverage knowledge from multiple related tasks to improve the performance of each single task. It is viewed as a form of inductive transfer which helps improve a model by introducing information from others (Caruana 1997). Theoretically, MTL models can be classified to several categories: feature learning, feature selection, deep learning approach, and so on (Zhang and Yang 2017). Traditional MTL approaches, which adopt linear models for each single task, mainly focus on how to design the regularization framework (Argyriou, Evgeniou, and Pontil 2006), quantitate task similarity (Evgeniou, Micchelli, and Pontil 2005) and others. In the context of deep learning, (Ruder 2017) proposed that DNN-Based MTL can be typically classified into either *hard* or *soft parameter sharing* of hidden layers. In this paper, we mainly consider the hard parameter sharing approach, since it is more applicable to CTR/CVR predictions.

MTL has been successfully used in many areas such as speech recognition and computer vision. While in

CTR/CVR predictions domain, there is very limited work. (Ahmed, Das, and Smola 2014) first proposed a hierarchical MTL modeling and apply it in the conversion prediction problem with linear models. Recently, (Ma et al. 2018) proposed a multi-task model to estimate *post-click conversion rate* (i.e., CTR/CVR). The motivation of modeling over entire space is similar with ours, while in this paper our goal is to optimize CTR/CVR prediction simultaneously.

3 Methodology

We first present the notation of the dataset and formalize the prediction problem. Suppose there are N ad impressions $X = \{x^i, 1 \leq i \leq N\}$ and each impression x^i is a d-dimensional feature vector which consists of many categorical fields (e.g. `gender` or `city`). Given an x^i, there is a corresponding click label y_1 (short form of y_1^i) and a conversion label y_2 which are all binary, e.g., $y_1 = 1$ represents there exists a click event while $y_1 = 0$ if not. As mentioned before, y_1 and y_2 follow a hierarchical dependent relationship: 1) $y_2 = 1$ or 0 if $y_1 = 1$; 2) $y_2 = 0$ if $y_1 = 0$. In general, the whole problem can be formalized as a hierarchical binary classification problem with two targets, i.e., $X \to Y_1$ (here $Y_1 = \{y_1^i, 1 \leq i \leq N\}$), $X \to Y_2$, and our goal is to predict pCTR $\hat{y}_1 = \Pr(y_1 = 1)$ and pCVR $\hat{y}_2 = \Pr(y_2 = 1)$ for each possible impression.

Next, we are to present two basic DNN based MTL models: the *feature sharing* model and *feature transfer* model. Both of them include a CTR prediction target and a CVR prediction target. Feature sharing model treats the two tasks equivalently, while feature transfer only focuses on the target task. The two models are fundamental since we mainly focus on how MTL influence single task and want to avoid the performance improvement from the model itself.

3.1 Feature Sharing Model

Feature sharing is an MTL model which utilizes shared hidden layers between DNNs of all tasks to learn a common feature representation (Ruder 2017). For CTR/CVR predictions problem, we propose a 4-layer DNN with two fairly treated output targets, in which the first two layers are shared and the last two are task-specific.

As shown in Figure 2, the first layer is a shared dense embedding layer:

$$e = \sigma(\text{lookup}(E, x)),$$

where x is the multi-field feature vector, E is a list of embedding vector, and σ is an activation function. To generate the corresponding embedding e for each feature vector, the model first looks up the embedding vectors from the embedding list E and concatenates them then. Note that E is updated with other hidden layers, and the back propagation only affects small parts of E since each raw feature in x only hits one embedding in the corresponding feature field.

The next hidden layers are all fully connected. $h_1 = \sigma(W_1 e + b_1)$ is a shared hidden layer (W_1, b_1 are the weight and bias parameters). $h_{2,i} = \sigma(W_{2,i} h_1 + b_{2,i}), i \in \{1, 2\}$ are two task-specific layers. The output layer gives a real number $\hat{y}_1 = \text{sigmoid}(W_{3,1} h_{2,1} + b_{3,1})$ as predicted CTR and $\hat{y}_2 = \text{sigmoid}(W_{3,2} h_{2,2} + b_{3,2})$ as predicted CVR.

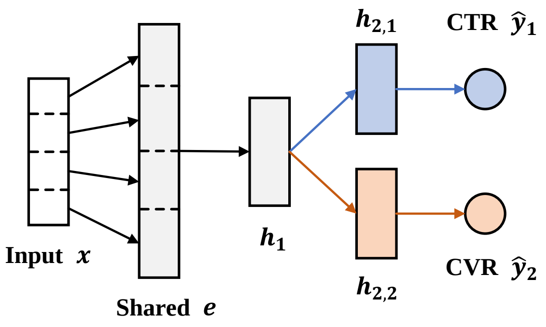

Figure 2: The feature sharing model.

Loss Function. For MTL with two classification tasks, a straightforward way is using the sum of Logarithmic Loss (LogLoss) of click labels Y_1 and conversion labels Y_2 as the whole loss function, i.e., $L = -\frac{1}{N}\sum_{i=1}^{N}\left[\ell_{y_1}(x^i) + \ell_{y_2}(x^i)\right]$ where

$$\ell_{y_k}(x^i) = (1 - y_k^i)\log(1 - \hat{y}_k^i) + y_k^i \log \hat{y}_k^i, \qquad (1)$$

$k \in 1, 2$ and $1 \leq i \leq N$. Here $\hat{y}_1^i$ means the predicted output (estimated value) for y_k^i. However, it treats the two targets equivalently and does not take the dependent relationship between Y_1 and Y_2 into account. Now we propose a new loss function from the perspective of maximizing the joint likelihood of the two labels. We first give our formula and present the derivation then.

Proposition 1 *For the hierarchical binary classification problem with two targets, the optimization goal for feature sharing model is*

$$L' = -\frac{1}{N}\sum_{i=1}^{N}\left[\ell_{y_1}(x^i) + y_1^i \cdot \ell_{y_2}(x^i)\right]. \qquad (2)$$

Under this loss function, the predicted output $\hat{y}_2$ for y_2 is a conditional probability $\Pr(y_2 = 1 | y_1 = 1)$.

Proof. Denote notation $p(\cdot)$ as the probability density function (PDF). For a random variable z with *Bernoulli* distribution $B(1, \theta)$, we have that $p(z) = \theta^z(1-\theta)^{1-z}$. Note that in our problem we have $p(y_1, y_2) = p(y_1) \cdot p(y_2|y_1)$, then log likelihood of all labels become

$$\begin{aligned} -\frac{1}{N}\log L(Y_1, Y_2) &= -\frac{1}{N}\log p(Y_1, Y_2) \\ &= -\frac{1}{N}\sum_{i=1}^{N}\left[\log p(y_1^i) + \log p(y_2^i|y_1^i)\right]. \end{aligned} \qquad (3)$$

Consider a single impression x^i, denote $\hat{y}_1^i = \Pr(y_1^i = 1)$ and $\hat{y}_{21}^i = \Pr(y_2^i = 1 | y_1^i = 1)$. By the dependent relationship between click labels and conversion labels, i.e., $\Pr(y_2^i = 0 | y_1^i = 0) = 1$, we have

$$\begin{aligned} p(y_2^i|y_1^i) &= \begin{cases} 1, & y_1^i = 0 \\ (\hat{y}_{21}^i)^{y_2^i}(1 - \hat{y}_{21}^i)^{1-y_2^i}, & y_1^i = 1 \end{cases} \\ &= p(y_2^i|y_1^i)^{y_1^i}. \end{aligned} \qquad (4)$$

Thus the log likelihood can be written more compactly as

$$\begin{aligned}
&-\frac{1}{N}\log p(Y_1, Y_2) \\
&= -\frac{1}{N}\sum_{i=1}^{N}\left[\log p(y_1^i) + y_1^i \cdot \log p(y_2^i|y_1^i)\right] \\
&= -\frac{1}{N}\sum_{i=1}^{N}\left[\ell_{y_1}(x^i) + y_1^i \cdot \ell_{y_2}(x^i)\right].
\end{aligned} \tag{5}$$

Note that when conducting the back propagation, the non-clicked instances ($y_1 = 0$) make no contribution to the parameters in target-specific layers for predicting $\hat{y}_2$. Thus, the predicted value for y_2 here is the post-click conversion probability $\hat{y}_{21} = \Pr(y_2^i = 1|y_1^i = 1)$. To obtain the CVR over all impression , the output should be calibrated to $\hat{y}_1 \cdot \hat{y}_{21}$.

In general, feature sharing adopts the shared layers E and h_1 to share the representations learned from CTR and CVR prediction tasks. Theoretically, it can help each task learn hidden features matter to the other one and implicitly transfer features between each task, which helps the model perform better (Ruder 2017). While in our experiments, it shows diverse performance under different kinds of input data and we will present more details in Section 4.

3.2 Feature Transfer Model

Different from treating the CTR and CVR prediction fairly, in this section, we propose the feature transfer model which focuses on the performance improvement of the main task.

Modeling. As shown in Figure 3, the feature transfer model is composed of an auxiliary task (predict $\hat{y}_a$) at the top part and a main task (predict $\hat{y}_m$) at the bottom. The two tasks are connected by a transfer layer between h_{a1} and h_{m2}. Each of the two parts is a 4-layer DNN with similar network setting in feature sharing model, i.e., the first layer is a dense embedding, the two hidden layers in the middle are fully connected, and the output layer gives a predicted CTR or CVR. The difference is that the two tasks holds separate dense embeddings e_a and e_m here and there is no shared fully connected layer as well.

In the transfer layer, the hidden layer h_{a1} as well as h_{m1} are fully connected to h_{m2}, i.e.,

$$h_{m2} = \sigma(W_{a1}h_{a1} + W_{m1}h_{m1} + b_{m2})$$

where W_{a1} and W_{m1} are weight matrices and b_{m2} is a bias. It conducts an explicit feature transfer from the auxiliary task to the main task.

This kind of transfer approach is straightforward, since we mainly focus on the influence from auxiliary labels and would like to avoid the performance improvement from other aspects. One can also adopt more effective approaches involving features crossing such as adopting a product operation of different features like PNN (Qu et al. 2016) or apply a deep crossing like DCN (Wang et al. 2017), which is beyond the scope of this paper.

In each batch training, the two tasks are trained separately by minimizing the LogLoss for each target (denoted as ℓ_{y_a} and ℓ_{y_m} here). Back propagation for minimizing ℓ_a only affects layers in the auxiliary task. As to the main task, back

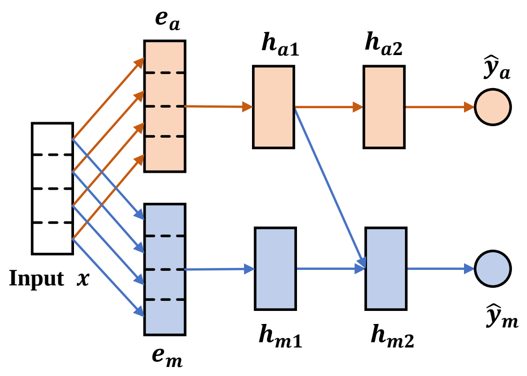

Figure 3: The feature transfer model.

propagation for minimizing ℓ_m stops after meeting h_{a1}, i.e., the training for main task does not influence the hidden layers in the auxiliary task.

In the feature transfer model, the auxiliary task essentially plays a role of feature extraction which is utilized for the learning of the main task. In our experiments, we use either CTR or CVR prediction as the main task while the remaining one as the auxiliary task.

4 Experiments

In this section, we conduct extensive comparative experiments on both CTR and CVR predictions among the proposed MTL models. In Section 4.1, we first give a brief description on our commercial dataset and experimental settings, and then present a preliminary experiment on the raw commercial dataset. In Section 4.3 and Section 4.4, we evaluate our MTL models on the homogenous data as well as the synthetic data. Finally, in section 4.5 we present a thorough analysis based on the diverse results and make several instructive conclusions.

4.1 Datasets and Experimental Settings

Commercial Dataset. We collected 8 consecutive days of impression records from the advertising platform. With a simple sampling and data cleaning on the impression records, we generate a raw commercial dataset. The first 7 days are used as training data and the last day is for testing. Table 1 shows a detailed description of the dataset.

Experimental Settings. In order to demonstrate the effectiveness of MTL models and reveal how click and conversion labels influence each other, we conduct several ablation experiments. We implement 4 models in TensorFlow (Abadi et al. 2016):

- **Baseline.** We use a single-task learning model, i.e., a 4-layer DNN, as the baseline. The network setting is the same as *each single task* in all MTL models, e.g., the setting for each target-specific layer in feature sharing is the same as the corresponding layer in baseline.
- **Feature Sharing with Loss L.** In this model, the loss to minimize is the the sum of LogLoss of Y_1 and Y_2, i.e., $L = -\frac{1}{N}\sum_{i=1}^{N}\left[\ell_{y_1}(x^i) + \ell_{y_2}(x^i)\right]$.
- **Feature Sharing with Loss L'.** In this model, the loss to minimize is the new loss deduced

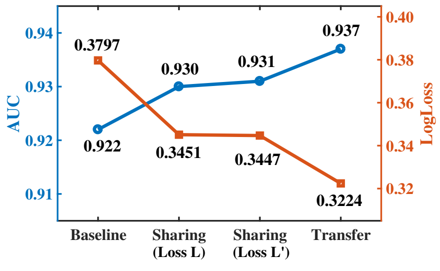

Figure 4: Results of y_1 on synthetic dataset.

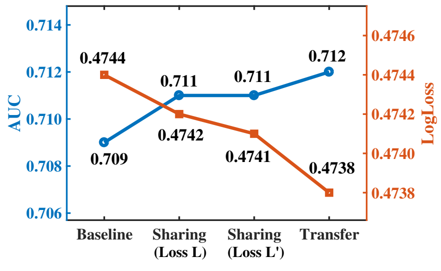

Figure 6: Results of y_1 on homogenous data.

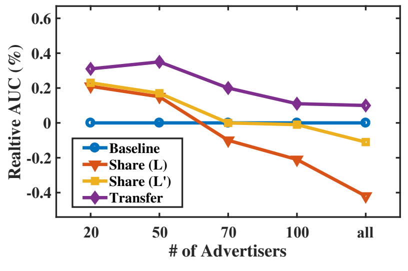

Figure 8: AUC of y_1 on commercial dataset.

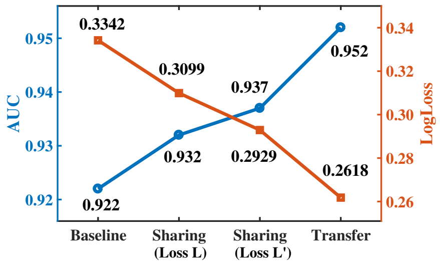

Figure 5: Results of y_2 on synthetic dataset.

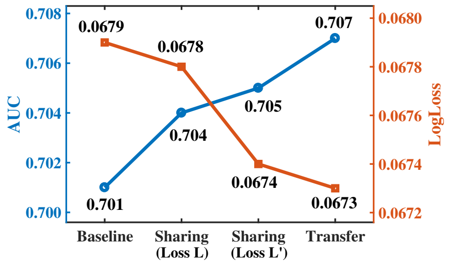

Figure 7: Results of y_2 on homogenous data.

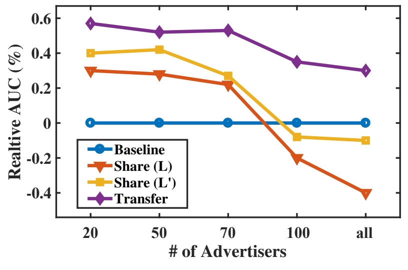

Figure 9: AUC of y_2 on commercial dataset.

from maximizing the joint likelihood, i.e., $L' = -\frac{1}{N}\sum_{i=1}^{N}\left[\ell_{y_1}(x^i) + y_1^i \cdot \ell_{y_2}(x^i)\right]$. [2]

- **Feature Transfer.** In this model, the two tasks are trained separately by minimizing the LogLoss for each target, i.e., ℓ_{y_a} and ℓ_{y_m}. To achieve a fair comparison, we adjust the number of neurons in h_{a1} and h_{m1} to ensure that it is the same with the corresponding layer in other models.

Table 1: The raw commercial dataset

Property	Description
Dataset size	About 20 million
# of Raw categorical field	18
# of one-hot feature	about 0.5 million
# of advertisers	more than 180
# clicked instances / # all instances	15.2%
# conv. instances / # all instances	1.7%
Example fields for user features	`gender`, `age`, `education`, etc.
Example fields for ad features	`ad category`, `advertiser`, etc.
Example fields for context features	`device os`, `network status`, etc.

For each model, the shape of dense embedding E is $(\cdot, 10)$. The number of neurons in the fully connected hidden layers are all set to 160 and the activation function is ReLU. We use Stochastic Gradient Descent (SGD) with a mini-batch size of 256 and the learning rate starts from 0.1.

Evaluation Metrics. We use the area under receiver operator curve (AUC) and LogLoss (cross entropy) to measure the performance of each label. The AUC and LogLoss are widely used measures in CTR/CVR predictions domain. As to different labels, we compare the performance of Y_1 and Y_2 separately.

4.2 Results on the Raw Commercial Dataset

We first make several preliminary experiments on the raw commercial dataset, i.e., we use impression instances from all advertisers (more than 180, see Table 1) for training. Table 2 shows the performance of AUC and LogLoss of all MTL models. We can observe that: 1) feature sharing model surprisingly degrades the performance on both y_1 and y_2, while the degradation is relatively more significant with loss L. 2) feature transfer model is slightly better than baseline but the improvement is not significant enough.

Table 2: AUC and LogLoss on the raw commercial dataset

Model	AUC	LogLoss
Baseline	0.703	0.5712
Feature sharing (L)	0.699 (0.4% ↓)	0.5718 (0.06% ↑)
Feature sharing (L')	0.702 (0.1% ↓)	0.5714 (0.02% ↑)
Feature transfer	0.704 (0.1% ↑)	0.5709 (0.03% ↓)

The results of different MTL models on the raw data is somewhat uncommon. To find out the reason for the degradation, we perform several case studies on the raw dataset then. Specially, we evaluate our MTL models on several sampled data in an advertiser-level granularity. We check the pCTR and pCVR and find that keeping the homogeneity of data from different advertisers is very important in training MTL models. For example, the post-click CVR (pCVR/pCTR) should be close enough.

Combined with the case study results and our understanding on online advertising, we conclude the following practi-

[2]Note that L' is only the optimization goal for training, in the following experiments, we still use standard LogLoss when evaluating Y_1 or Y_2.

cal lessons on keeping the homogeneity of data for training MTL models:

- **Same Advertising Pipeline.** Since click/conversion definitions vary among different advertising pipelines, we choose data from a typical one, e.g., all conversion events are downloading apps.
- **Homogenous Click/Conversion Definitions.** We take a preview on the post-click CVR among all advertisers and make sure they are close enough. This helps to remove outliers (e.g., low post-click CVR means the advertisers might probably lure the users to click) and ensure the homogeneity of the data.
- **Reliable Conversion Data.** In our dataset, there are actually multi-levels of conversion (in the experiments we only consider the level 1 conversion). For data under the same advertiser, the correlation of different levels of post-click CVRs should be high enough to ensure the conversion labels reported by advertisers are reliable.

Based on such practical lessons, we generate several homogenous data for the ablation experiments next. As to the largest one, the size is about 5 million. The ratio of instances with click and conversion labels over all instances are 19.2% and 1.3% respectively. Next, we are to compare the models on the homogenous data and the results are based on this largest one by default.

4.3 Results on the Homogenous Data

The homogenous data. Figure 6 and Figure 7 show the results with respect to y_1 and y_2 respectively on the homogenous data. The dotted blue line is the AUC performance while the squared orange line is the LogLoss. Compared with the baseline, we observe that

- **On Feature Sharing Model.** Overall, the feature sharing model outperforms baseline on varying degree. Figure 7 shows that the proposed loss L' is slightly better than loss L as to y_2 (About 0.1% AUC improvement). While as to y_1 in Figure 6, the difference between loss function L and L' seems not significant.
- **On Feature Rransfer Model.** The feature transfer model always performs best on both y_1 and y_2. The AUC improvement on y_1 and y_2 are 0.3% and 0.6% respectively, which are significant enough in practice.
- **On y_1 and y_2.** For all MTL models, the improvement on y_2 from all MTL models is always more significant than that on y_1.

Ablation Analysis. The results on the homogenous data show that our MTL models are superior to the STL models on CTR/CVR predictions, while the effectiveness is more significant when predicting CVR. However, compared with the results in Table 2, it seems that the performance of MTL models varies and are highly dependent on the dataset.

As shown in Table 1, in practice we mainly consider advertisers as the granularity. Thus, to evaluate how MTL performs on data of different levels of homogeneity, we further generate 5 pieces of data which contains 20, 50, 70, 100, and "all" advertisers respectively. Here, for data including from 20 to "all" advertisers, the homogeneity of advertisers decreases monotonically. For each of the generated data, we train the 4 models mentioned in Section 4.1 and compare the test AUC of them.

Figure 8 and 9 show the diverse performance of all MTL models on these data. The x-axis *# of advertisers* represents the number of advertisers we choose for training/testing. The y-axis is the relative AUC with respect to the baseline model, i.e., in each data we plot the AUC offset of all MTL models compared with baseline. Note that the "all" column is same as the results in Table 2.

From Figure 8 and 9, we observe that the results of different MTL models with different advertisers involved is closely related to the homogeneity of the advertisers, i.e., both feature sharing and feature transfer model tend to perform worse when adding more advertisers, especially from the data with all advertisers included. As to the two MTL models, feature sharing model with loss L always performs the worst (about -0.4% degradation at most) while the performance under loss L' seems more stable; Feature transfer is the least affected and it still improves significantly on y_2 when all advertisers are included.

4.4 Experiments on Synthetic dataset

Before we give the analysis on the previous experimental results, we would like to propose our design and experiments on a synthetic dataset which models users' impression-to-conversion action path. The motivation of generating such a synthetic dataset is due to two reasons. 1) the diversity and high uncertainty of the commercial dataset may reduce the confidence on our MTL models. Thus, we need a more clear data to achieve a more convincing result; 2) we need a reasonable modeling on the impression-to-conversion path to help demonstrate that MTL models do work on such kind of data, rather than only fitting well on some specific dataset.

Synthetic Dataset. The synthetic dataset is based on a user-interest hypothesis. The generation process includes 3 parts: generating user/ad pools, generating instances, and assigning click/conversion labels.

1. **Generating User/Ad Pools.** Assume that there are K categories of ads in total, e.g., `sports`, `tech`, etc. For each ad category, assume both the interest values of users and relevance of ads are normally distributed. Under such a hypothesis, we randomly generate a users pool and an ads pool. As shown in Figure 10, each user u^i is implicitly represented by a k-dimension embedding vector $[u^i_1, u^i_2, \ldots, u^i_K]$ where each component $u^i_j, 1 \le j \le K$ is sampled from a normal distribution $N(\mu_j, \sigma_j^2)$, which can be interpreted as one's interest on the ad category. Each ad a^i also corresponds to an embedding vector $[a^i_1, a^i_2, \ldots, a^i_k]$ representing its relevance on all categories and $a^i_j, 1 \le j \le K$ are sampled from $N(\mu'_j, \sigma'^2_j)$.
2. **Generating Instances.** With the users/ads pools, we now simulate the impression events. For each impression, we randomly choose a user and an ad from the pools and assign the dot product of their embedding vector as the *interest value* v of this impression. The interest value represents how much interest the user has on this ad. Besides,

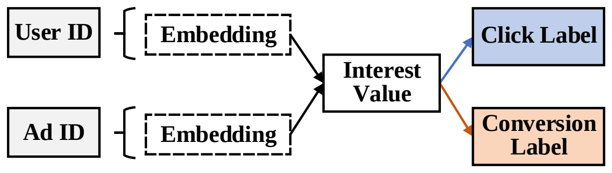

Figure 10: Generating process of synthetic dataset.

we also introduce some noise by randomly dropping the interest value out (set to 0) with a small probability.

3. **Assigning Click/Conversion Labels.** Given a set of impression instances, it is believed that click event can happen only if the v is large enough (He et al. 2014) while a conversion event may even require a larger v. Specifically, we choose instances with interest value larger than the median to assign with a click label $y_1 = 1$, while instances with interest value larger than 70% of all to assign with a conversion label $y_2 = 1$. The rest are marked as non-clicked and non-conversion.

In our experiments, we generate 1 million users, 1 million ads and two million impressions. We use *user IDs* and *ad IDs* as the input features and see how the model learns without the knowledge from interest values. The embedding size is 10 and the drop out rate is 0.1.

Experiment Results. The results on synthetic dataset of y_1 and y_2 are shown in Figure 4 and Figure 5. Compared with baseline, feature sharing with loss L or L' significantly improves the performance on both y_1 and y_2 (about 0.8% AUC on y_1 and 1% AUC on y_2). Figure 5 shows that minimizing loss L' is better that loss L since it significantly improves the AUC by 0.5% on y_2. To feature transfer model, it always performs the best on both y_1 and y_2. As to different labels, we notice that the all MTL models perform much more significantly on y_2 than on y_1, e.g., parameter transfer model improve 3.0% AUC and 7.3% LogLoss on y_2 while they are only 1.5% and 5.7% on y_1.

The consistent results on synthetic dataset and homogenous data generally demonstrate the rationality of our synthetic modeling on the impression-to-conversion path. Varying degrees of improvements of feature sharing/transfer model show that MTL can indeed help optimize both CTR and CVR predictions. We also find that 1) optimizing the likelihood (minimizing loss L') is more appropriate for CTR/CVR prediction patterns; 2) the click labels are more helpful for predict CVR while in the opposite way the improvement is less significant, which is also consistent with that MTL will tend to help harder problems (Caruana 1997).

4.5 Results Analysis and Model Interpretation

In this section, we first present our observation and thoughts on the characteristics of the commercial dataset. Then, we give a thorough analysis and conclusion based on the results on the raw commercial dataset, homogenous commercial data, and synthetic dataset.

Conversion Delay. In CPA model, a final positive conversion example is empirically generated by attributing a conversion event to the correct click event. However, a conversion event may happen minutes, hours or even days after a click, which is called a *conversion delay* (Rosales, Cheng, and Manavoglu 2012). Thus, a critical point we have observed is that in addition to the basic features (users, ads and context features) mentioned before, conversion labels are further affected by features during conversion delay. Unfortunately, those features are usually hard to capture and their forms vary among different advertisers.

Inconsistent Conversion Definitions. As is mentioned in Section 1, the conversion events are usually defined distinctively by different advertisers to maximize their revenue (Becker et al. 2009), which badly influence the so-called "homogeneity" of the data. From our preliminary experiments, we observe that under different advertisers both the distribution of CTR and conditional CVR of different ads varies a lot.

Combined with the experiment results and our observations, we obtain the following conclusion:

- Sharing features between CTR and CVR prediction models are helpful for each single task when the advertising scenarios are similar enough (see Figure 6 and Figure 7), i.e., the conversion delay is small or the conversion definitions are of little difference. Otherwise, conversion labels may bring a negative effect on CTR prediction (see feature sharing with loss L in Figure 8). Besides, all results show that minimizing L' is more appropriate for feature sharing approach.
- Transferring features from CTR prediction models are usually helpful for predicting CVR (see Figure 7 and Figure 8) since it may capture a richer representation of the data. While in the opposite case, the conversion labels are of relatively less help for CTR prediction (see Figure 6) since it may bring much redundant information from conversion delay or heterogeneous conversion definitions (see Figure 8).
- Our modeling on synthetic dataset is reasonable due to the consistent results in Section 4.3. However, it is limited to only depicting single advertiser scenario. A possible explanation is that the interest value in real scenarios should be represented by a high dimensional vector rather than a single value. Furthermore, it is believed that in addition to following a logically dependent relationship, the click and conversion labels depict different relationship in depth between the users and ads.

5 Conclusion

In this paper, we consider optimizing the predictions of both CTR and CVR through DNNs from the MTL perspective. We propose two basic feature-based MTL models with distinct characteristics: feature sharing model and feature transfer model. To demonstrate the effectiveness of MTL models and reveal how click and conversion labels influence each other, we conduct extensive ablation experiments based on MTL models on both commercial and synthetic datasets. Based on the experiment results, we present thorough analysis and practical lessons on the application of MTL models on CTR/CVR predictions, which may shed light on the application of MTL on CTR/CVR predictions in industry.

References

Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; Kudlur, M.; Levenberg, J.; Monga, R.; Moore, S.; Murray, D. G.; Steiner, B.; Tucker, P. A.; Vasudevan, V.; Warden, P.; Wicke, M.; Yu, Y.; and Zheng, X. 2016. Tensorflow: A system for large-scale machine learning. In *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 265–283.

Ahmed, A.; Das, A.; and Smola, A. J. 2014. Scalable hierarchical multitask learning algorithms for conversion optimization in display advertising. In *International Conference on Web Search and Data Mining (WSDM)*, 153–162.

Argyriou, A.; Evgeniou, T.; and Pontil, M. 2006. Multi-task feature learning. In *Advances in Neural Information Processing Systems (NIPS)*, 41–48.

Becker, H.; Broder, A. Z.; Gabrilovich, E.; Josifovski, V.; and Pang, B. 2009. What happens after an ad click?: Quantifying the impact of landing pages in web advertising. In *ACM Conference on Information and Knowledge Management (CIKM)*, 57–66.

Caruana, R. 1997. Multitask learning. *Machine Learning* 28(1):41–75.

Chen, J.; Sun, B.; Li, H.; Lu, H.; and Hua, X. 2016. Deep CTR prediction in display advertising. In *ACM Conference on Multimedia Conference (MM)*, 811–820.

Cheng, H., and Cantú-Paz, E. 2010. Personalized click prediction in sponsored search. In *International Conference on Web Search and Web Data Mining (WSDM)*, 351–360.

Cheng, H.; van Zwol, R.; Azimi, J.; Manavoglu, E.; Zhang, R.; Zhou, Y.; and Navalpakkam, V. 2012. Multimedia features for click prediction of new ads in display advertising. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, 777–785.

Cheng, H.-T.; Koc, L.; Harmsen, J.; Shaked, T.; Chandra, T.; Aradhye, H.; Anderson, G.; Corrado, G.; Chai, W.; and Ispir, M. 2016. Wide & deep learning for recommender systems. In *Workshop on Deep Learning for Recommender Systems (DLRS)*, 7–10. ACM.

Evgeniou, T.; Micchelli, C. A.; and Pontil, M. 2005. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research* 6:615–637.

Gai, K.; Zhu, X.; Li, H.; Liu, K.; and Wang, Z. 2017. Learning piece-wise linear models from large scale data for ad click prediction. *CoRR* abs/1704.05194.

Graepel, T.; Candela, J. Q.; Borchert, T.; and Herbrich, R. 2010. Web-scale bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine. In *International Conference on Machine Learning (ICML)*, 13–20.

Guo, H.; Tang, R.; Ye, Y.; Li, Z.; and He, X. 2017. Deepfm: A factorization-machine based neural network for CTR prediction. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 1725–1731.

He, X.; Pan, J.; Jin, O.; Xu, T.; Liu, B.; Xu, T.; Shi, Y.; Atallah, A.; Herbrich, R.; Bowers, S.; and Candela, J. Q. 2014. Practical lessons from predicting clicks on ads at facebook. In *International Workshop on Data Mining for Online Advertising (ADKDD)*, 5:1–5:9.

Lee, K.; Orten, B.; Dasdan, A.; and Li, W. 2012. Estimating conversion rate in display advertising from past performance data. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 768–776. ACM.

Ma, X.; Zhao, L.; Huang, G.; Wang, Z.; Hu, Z.; Zhu, X.; and Gai, K. 2018. Entire space multi-task model: An effective approach for estimating post-click conversion rate. In *International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR)*, 1137–1140.

Qu, Y.; Cai, H.; Ren, K.; Zhang, W.; Yu, Y.; Wen, Y.; and Wang, J. 2016. Product-based neural networks for user response prediction. In *International Conference on Data Mining (ICDM)*, 1149–1154.

Rendle, S. 2012. Factorization machines with libfm. *ACM TIST* 3(3):57:1–57:22.

Richardson, M.; Dominowska, E.; and Ragno, R. 2007. Predicting clicks: Estimating the click-through rate for new ads. In *International Conference on World Wide Web (WWW)*, 521–530.

Rosales, R.; Cheng, H.; and Manavoglu, E. 2012. Post-click conversion modeling and analysis for non-guaranteed delivery display advertising. In *International Conference on Web Search and Web Data Mining (WSDM)*, 293–302.

Ruder, S. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098.*

Wang, R.; Fu, B.; Fu, G.; and Wang, M. 2017. Deep & cross network for ad click predictions. In *International Workshop on Data Mining for Online Advertising (ADKDD)*, 12:1–12:7.

Zhang, Y., and Yang, Q. 2017. A survey on multi-task learning. *CoRR* abs/1707.08114.

Zhang, W.; Du, T.; and Wang, J. 2016. Deep learning over multi-field categorical data. In *European Conference on Information Retrieval (ECIR)*, 45–57. Springer.