Transformer Deep Network Bid Landscape Forecasting

Daria Yakovleva,[1, 2] Sergei Telnov, [1] Andrey Filchenkov [1]
[1] ITMO University
[2] VK
dariyakovleva@gmail.com, sergeitelnow@gmail.com, aaafil@mail.ru

Abstract

In Real-Time Bidding, advertisers aim to optimally bid within a limited budget constraint. Effective dynamic bidding strategies require bid landscape forecasting to predict the probability distribution of market price for each ad auction. This distribution has a complicated form with many peaks and all bids probabilities depend on each other. Most existing solutions mainly focus on learning a parameterized model based on some heuristic assumptions of distribution forms. In this paper, we propose a Transformer model that considers dependencies between bids improving the bid landscape forecasting. We also increase the quality of model prediction on the ad cold start for cases, where data for training a predictive model is insufficient. Our experiments on two real-world datasets prove the model proposed significantly outperforms the state-of-the-art solutions both in terms of ANLP metrics by 8.75% and ROC-AUC by 1.1%.

Introduction

Real Time Auction (RTB) appeared in 2009 and became the main mechanism for selecting an ad to display in online advertising. The RTB mechanism opened new opportunities for research, automation and optimization of bidding strategies in auctions. The effectiveness of advertising directly depends on the quality of the strategy chosen.

The problem of calculating optimal bid in an auction is complicated. Low bid decreases the probability of winning an auction, while high bid increases probability to overspent. We can model bid distribution using historical data and calculate the winning probability an auction for every bid.

There is another challenge of the bid landscape foresting which is the censorship issue. Only the winner, who submits the highest bid price, will know the market price, i.e., the charged price, while others can only know that the market price is higher than their bids, which is called right-censored data (Wang et al. 2016). To handle this censorship, many works (Wang et al. 2016; Wu, Yeh, and Chen 2018) borrow the idea of survival analysis in medical data science and take the losing logs into consideration to better model the true market price distribution.

In this paper, we provide an approach to modeling the distribution of bids for an advertising campaign. The approach is based on an individual ad trading history. Every auction contains information about an bid, information whether the auction was won, as well as information about the user who could potentially see the ad.

Based on bids distribution we can calculate the expected amount of impressions for different bids. Impressions estimation helps to choose an optimal bid for an ad. Advertising systems are able to build their own bidding strategy to improve the performance of each ad, i.e. smoothly spend the ad budget throughout the day or maximize the number of targeted actions for a given budget.

The main goal of the work is to improve the quality of result bid distribution for an online auction. The problem is solved using the deep transformer network. This network was first introduced in 2017 in a machine translation problem as an alternative to the recurrent neural network.

An actual approach assumes that winning probability of bid b depends only on the winning probability of bid $b-1$. However, in real life all probabilities depend on each other.

We propose an approach that considers dependencies of winning probabilities. Moreover, we study the cold start problem of the ad. When the ad starts to participate in auctions and has no sufficient data for training a network for good quality prediction.

The novelty of this paper is as follows:

- First time applying the transformer network to consider dependencies of winning probabilities;
- Improving the predicted distribution on the ad cold start;
- Using additional neural network improvements to achieve faster training.

First, we describe the existing methods for solving the problem and make a formal problem definition. Then, we present the new method based on the Transformer model to solve the predicting bids distribution problem. Then, we describe the problem of the ad cold start and provide the approach to its solution. We compare results with start-of-the-art solutions on two real-world datasets and show that our model has a significant improvement under various evaluation metrics.

Related Works

Earliest approaches based on historical statistics make an assumption about the form of bids distribution.

ad id	win train	lose train	win test	lose test	% win
1458	997247	2085823	119397	495241	30,2
2259	296657	538901	99626	317572	31,63
2261	221454	469243	100477	246195	31,03
2821	142697	1179865	86136	575828	11,53
2997	43803	268634	26944	129120	15,1
3358	278637	1463468	36373	264555	15,42
3386	683319	2164487	136128	409295	24,15
3427	501868	2091909	153121	383676	20,92
3476	644951	1325553	78896	444969	29,02

Table 1: iPinYou dataset

ad id	win train	lose train	win test	lose test	% win
1	82640	615846	9433	155596	10.66
2	71285	936251	8357	233155	6.38
3	57393	559516	4071	139835	8.08
4	82579	555062	9810	138917	11.75
5	59182	2248213	4396	562966	2.21
6	45302	2026978	3150	509520	1.87
7	25278	2399779	3309	592234	0.95
8	32729	1692800	2775	413617	1.66
9	35711	2627204	2961	651794	1.17
10	27365	2419557	2641	598133	0.98
11	28775	2762665	2862	683767	0.91
12	28700	2762665	2862	683767	0.91

Table 2: Social network dataset

One of approaches (Zhang et al. 2016) is a non-parametric method based on the Kaplen-Meier estimation (Kaplan and Meier 1958). The solution does not consider the user information, but only taking into account the advertiser's bid and the market price of the auction. The approach only deals with information of winning auctions.

Another approach (Wang et al. 2016) clusters data using decision trees. Each request is clustered using a tree. Based on the statistics of the bidding history of this cluster the distribution is calculated in the final nodes. The decision tree is constructed using the k-means method and the Kullback-Leibler distance.

In 2019, students from the China University of Shanghai published an article about the use of deep recurrent neural networks in Survival Analysis (Ren et al. 2019a), and they also made a separate publication (Ren et al. 2019b) using this decision in predicting the distribution of bids in auctions. The first article published the new solution application in different areas such as medicine, music industry and bidding predictions in auctions. They propose a deep neural network model without any assumptions about heuristic forms of market price distribution. Moreover, to handle the censorship they also adopt a comprehensive loss function over both winning logs and losing logs. According to the results of the experiment, this solution showed the best result, therefore we use it as main solution for comparison with our approach.

The transformer model was proposed in the machine translation problem as an alternative to the recurrent neural network, where text has to be translated from one language into another (Vaswani et al. 2017).

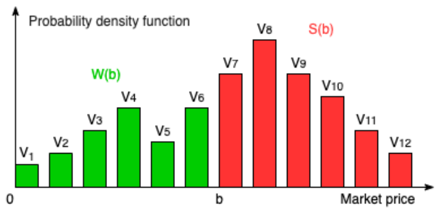

Figure 1: Probability density function in discrete space

Sentences for translation are considered as an ordered sequence of words with each next word in the sentence being translated based on previously translated words. This is an example of a time series problem (Box et al. 2015).

Problem Formulation

The problem is to predict the distribution of the auction market price based on historical data. Auction data contains the features of users (age, sex, ctr, etc.), who can potentially see an ad and information about the advertising platform: the site, the advertising banner information, domain hash, etc.

Formally each auction from a dataset contains the vector of user and platform information x, bid price in this auction b and observed market price z. The market price is known only if the ad wins the auction i.e. $b > z$, otherwise for losing auctions the market price is not provided because the auction is closed.

The goal of the model is to estimate the probability distribution of market price $p(z|x)$. For every bid this distribution provides the probability of winning auction.

The model takes user and platform information x from the auction as input and provides $p(z|x)$ as output.

Proposed Solution

In this section we present a new approach to improve the prediction quality of the market price distribution. We describe in detail a neural network based on the transformer model and additional neural network improvements. Additionally, we formulate the cold start problem for a new ad with a small amount of training data and provide the solution to improve this problem.

Network preparation A neural network cannot work with continuous space, so it is necessary to move from continuous to discrete space. We use an approach from article (Ren et al. 2019b).

They represent the set of bids as a sequence from smallest to largest $0 < b_1 < b_2 < ... < b_L$ and we split it into L equal blocks V_l such, that for $\forall l = 1, 2, \ldots, L$ $V_l = (b_{l-1}, b_l]$.

An example of market price probability density function in discrete space, the winning probability W and the losing probability S for the bid b is shown in Figure 1.

All bids belong to the set of natural numbers and for better prediction the difference between the blocks is $\forall l$ $b_{l+1} - b_l = 1$.

Table 3: ROC-AUC metric comparison for iPinYou dataset

ad id	KM	STM	DLF	Transformer	Impr (%)
1458	0.698	0.764	**0.911**	0.909	-0.14
2259	0.685	0.768	0.847	**0.870**	+2.67
2261	0.666	0.812	0.926	**0.938**	+1.33
2821	0.677	0.790	0.875	**0.895**	+2.25
2997	0.734	0.835	0.916	**0.920**	+0.46
3358	0.704	0.811	0.940	**0.948**	+0.9
3386	0.716	0.849	0.919	**0.923**	+0.44
3427	0.724	0.798	0.870	**0.884**	+1.66
3476	0.692	0.830	0.923	**0.928**	+0.5
avg	0.7	0.807	0.903	**0.913**	+1.1

Therefore, the probability formulas have the following form:

$$\begin{aligned} W(b_l) &= Pr(z < b_l) = \sum_{j<l} Pr(z \in V_j) \\ S(b_l) &= Pr(z \geq b_l) = \sum_{j \geq l} Pr(z \in V_j) \\ p(b_l) &= Pr(z \in V_l) = W(b_{l+1}) - W(b_l) \\ &= S(b_l) - S(b_{l+1}) \end{aligned} \tag{1}$$

The conditional winning probability in an auction for the bid from the V_l block is an event that the market price z belongs to the block V_l, provided that the value of the market price z is greater than all the values of the rates in the previous blocks.

$$\begin{aligned} h_l &= Pr(z \in V_l \mid z \geq b_{l-1}) \\ &= \frac{Pr(z \in V_l)}{Pr(z \geq b_{l-1})} = \frac{p_l}{S(b_{l-1})} \end{aligned} \tag{2}$$

These conditional probabilities is predicted by the neural network.

The auction winning probability for the sample x^i, with the bid b_l is:

$$W(b_l \mid x^i; \Theta) = 1 - \prod_{l_i : l_i \leq l} (1 - h_l^i) \tag{3}$$

Transformer network The first approach we attempted to improve the model was the attention mechanism. This mechanism improves distribution prediction by remembering the most important information for every bid from predicting previous bids. The attention mechanism is described in detail in the article, where the problem of document classification is solved (Yang et al. 2016). This approach does not give a significant improvement, but provides ideas for following research.

The main idea of the transformer model is to use the mechanism of internal attention. The attention mechanism can improve the performance indicators of the neural network, which is actively used in modern deep learning models. In addition to improving the model results, another benefit is an high learning rate for parallelism (Alammar 2018).

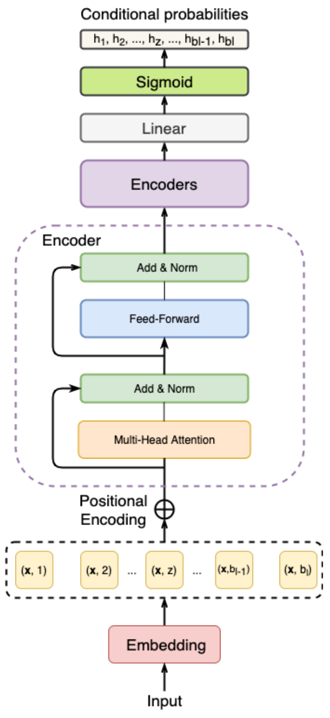

Figure 2: Transformer model

The internal attention mechanism in the transformer network allowed to take into account the winning predictions of previous and next states and find dependencies between them.

The original transformer model consists of an encoder and a decoder.

Information about the user and the platform is transmitted to the model's input, which passes through the Embedding layer. Then, the output is duplicated L times to create blocks. Transformer network has no convolutional layer, therefore it is necessary to add mechanics to give the model information about the relative position of each block in the sequence. For this, a Positional Encoder vector is added to the embedding vector, which creates the distance between the blocks (ten 2019).

Then the sequence of blocks is transferred to the coding component, which consists of encoder blocks.

In an encoder each block goes through a layer of inner attention. Internal attention allows the model to look at other positions and find clues to better code each block.

The output of the inner attention layer is sent to the feed-forward neural network. After leaving the encoding component, the vector needs to be transformed to obtain conditional probabilities, so that all values are from 0 to 1. For this, a fully connected layer is added to obtain a sequence of L blocks. In the end a monotonic activation sigmoid function is applied to obtain probabilities.

The Transformer model described is shown in Figure 2.

To get the final bid winning probability prediction we have to calculate the product of conditional probabilities on

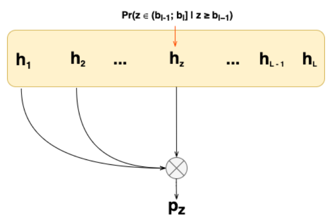

Figure 3: Winning probability calculation

the prefix from the first block to the block which the bid belongs to. An example of obtaining the distribution density is given in Figure 3.

Two error functions are used for training (Ren et al. 2019b). The first is based on density distribution and its main goal is to minimize ANLP for all winning cases D_{win}.

$$\begin{aligned} L_1 &= -\log \prod_{(x^i,z^i)\in D_{win}} p_l^i \\ &= -\log \prod_{(x^i,z^i)\in D_{win}} h_{l_i}^i \prod_{l:l<l^i} (1-h_l^i) \\ &= -\sum_{(x^i,z^i)\in D_{win}} \left[\log h_{l_i}^i + \sum_{l:l<l^i} \log(1-h_l^i) \right], \end{aligned} \tag{4}$$

where l^i is the number of the interval, which includes the market price $z^i \in V_{l^i}$ for the i-th example.

L_1 is the loss function that maximizes the value of the distribution density for the bid equal to the market price z. This formula is used to train the model. It also provides the conditional winning probability h_l^i and its opposite value $\hat{h_l^i} = 1 - h_l^i$. Then, all the described formulas from the introduction are also used with the modified conditional probability.

The second error function is based on the distribution function and contains two parts: for a winning and losing auctions.

For the winning auction the market price z is observed, and the bid has to be equal to the market price or higher to win the impression. Therefore, the values of winning probability W_{win} in interval $[0, z]$ has to be as less as possible.

On the contrary, the probability of losing L_{win} should be as higher as possible because all bids in this interval are losing. The value of the distribution density in the market price z tends to one.

In the case of a losing auction the market price is not observed, therefore it is not possible to select a bid, for which the distribution density values tend to somewhere. However, it is impossible to win the auction with the bid b. It is also impossible to win the auction with bids less or equal to this bid. Therefore, it is necessary to make lower the probability of winning W_{lose} in interval $[0, b]$ or, on the contrary, make higher the probability of losing L_{lose}.

To use both errors together, we use a simple w formula that depends only on the bid and market price in the auction.

Table 4: ANLP metric comparison for iPinYou dataset

ad id	KM	STM	DLF	Transformer	Impr (%)
1458	10.532	4.761	3.218	**3.150**	-2.12
2259	14.671	5.471	3.731	**3.378**	-9.44
2261	14.665	4.818	2.748	**2.020**	-26.5
2821	19.582	5.572	2.864	**2.653**	-7.36
2997	16.203	5.083	1.557	**1.200**	-22.95
3358	19.253	5.539	3.317	**2.893**	-12.77
3386	15.973	5.228	3.046	**2.921**	-4.12
3427	16.902	5.321	3.785	**3.743**	-1.12
3476	10.507	4.537	3.231	**3.133**	-3.02
avg	15.366	5.148	3.055	**2.788**	-8.75

Table 5: ANLP metric comparison for Social network dataset

ad id	KM	DLF	Transformer	Impr (%)
1	6.389	5.329	**5.327**	-0.04
2	6.136	**4.513**	4.639	+2.81
3	6.506	**5.064**	5.135	+1.4
4	5.658	4.301	**4.175**	-2.92
5	6.609	**4.227**	4.332	+2.46
6	8.442	5.198	**5.196**	-0.04
7	9.093	**4.776**	4.803	+0.57
8	7.218	5.495	**5.477**	-0.34
9	6.989	**4.322**	4.367	+1.03
10	7.575	4.646	**4.509**	-2.94
11	7.556	4.622	**4.425**	-4.26
12	7.556	4.613	**4.448**	-3.58
avg	7.144	4.759	**4.736**	-0.48

$$w = \begin{cases} 1, & \text{if } b > z \\ 0, & \text{if } b \leq z \end{cases} \tag{5}$$

The second error function L_2 is the combination of L_{win} and L_{lose}.

$$\begin{aligned} L_2 &= L_{win} + L_{lose} \\ &= -log \prod_{(x^i,b^i)\in D_{win}} Pr(b^i > z \mid x^i; \Theta) \\ &\quad -log \prod_{(x^i,b^i)\in D_{lose}} Pr(z \geq b^i \mid x^i; \Theta) \\ &= -\log \prod_{(x^i,z^i)\in D} \left[W(b^i \mid x^i; \Theta)\right]^{w^i} \\ &\quad \cdot \left[1 - W(b^i \mid x^i; \Theta)\right]^{1-w^i} \\ &= -\sum_{(x^i,b^i)\in D} \left[w^i \log W(b^i \mid x^i; \Theta)\right. \\ &\quad \left. +(1-w^i) log(1 - W(b^i \mid x^i; \Theta)\right) \end{aligned} \tag{6}$$

The result is the usual cross-entropy formula.

A combination of two formulas will be used to train the model.

$$\underset{\Theta}{\operatorname{argmin}}\, \alpha L_1 + \beta L_2, \tag{7}$$

where α and β are hyperparameters that control the magnitude of the gradient to stabilize the model training.

Additional network improvement One of the main problems in machine learning models is overfitting. Also the transformer model also has a lot of learning vectors, which increases the training time.

To overcome the overfitting and speed up learning in deep neural networks, we use Dropout mechanism. The main idea is to "exclude" some neurons from the network with probability. The excluded neurons do not contribute to the learning process at any stage of the backpropagation algorithm (wun 2017).

Droupout forces the neural network to train on more robust parameters, which are useful in combination with many different random subsets of other neurons. Due to this fact, the number of iterations required for convergence is roughly doubled, but the training time for each epoch is reduced (Budhiraja 2016).

Another way to protect overfitting is Early stopping. It uses a very simple idea — to read model performance on validation data that is not used for training. Despite the improvement of the metrics at each step of training, at some point the model is likely to reach the optimal value on the test data and then to degrade with further training iterations. Therefore, it is reasonable to stop training at the moment when the model achieves the minimum error (ear 2017).

Also deep neural networks training is time consuming and many are spent for this. Pre-training reduces the time of model convergence to an optimal value. For example, in a machine translation task, a trained neural network in another language is used to teach a new language. The article (Qi et al. 2018) describes the application of pre-training approach in a machine translation task.

Table 6: ROC-AUC metric comparison for Social network dataset

ad id	KM	DLF	Transformer	Impr (%)
1	0.799	0.838	**0.841**	+0.44
2	0.687	0.707	**0.714**	+1.01
3	0.765	0.783	**0.804**	+2.63
4	0.602	0.680	**0.683**	+0.41
5	0.737	**0.794**	0.790	-0.57
6	0.897	**0.930**	0.929	-0.12
7	0.758	0.792	**0.800**	+0.95
8	0.851	0.883	**0.893**	+1.12
9	0.757	0.877	**0.883**	+0.68
10	0.811	0.843	**0.858**	+1.78
11	0.793	0.829	**0.846**	+2.07
12	0.793	0.834	**0.854**	+2.36
avg	0.771	0.816	**0.825**	+1.06

Cold Start problem The problem with cold start requires data to train neural networks. To collect this data an ad must participate in a large number of auctions without bids distribution knowledge.

A similar problem arises in many areas where deep learning is used e.g. in recommendation systems (Wei et al. 2017).

To correctly model the distribution of bids, all advertisements need to accumulate data for training the neural network to find dependencies and only then predict the probabilities. In the case of cold start, when a new ad has just appeared, this data is unavailable. The new ad has to to participate in the auction without correct knowledge of the market price distribution.

The idea is that the function of distributing bids for ads is similar, but the targeting of users is different, so we can take a neural network model trained on another ad as a basis and train it on the existing ones.

Experiments

In this section we provide the analysis of our solutions. Firstly, we describe datasets, which we use for training and testing of all the models. The approaches are compared, the confidence interval is calculated, and the statistical significance of the results is checked. We also show the improvement of the models in the case of cold start.

The analysis has been performed for the following solutions:

- KM — a decision based on statistics;
- STM — a solution based on the clustering problem, which uses decision trees;
- DLF — the state of the art solution and our main benchmark for performance;

- TLF — Transformer network, our proposed solution.

For comparison, we use STM source code from the article itself [1]

For comparison, the KM, DLF and Transformer models were implemented independently [2].

Datasets Two datasets were used for training and testing: an open dataset of iPinYou (Liao et al. 2014) and an anonymous dataset of the social network Vkontakte [3].

iPinYou dataset

iPinYou dataset is public, allows to compare the results of our algorithm with other articles.

The dataset contains auction information for 9 different advertisements. The data is divided into two parts: for training (train) and for testing (test).

The dataset description is given in Table 1. This table contains the number of winning and losing auctions for each part (train and test) and percentage of winning auctions.

Social network dataset

The dataset of the social network is composed of anonymous bidding history for 12 advertisements. The dataset description is given in Table 2, which contains the information similar to iPinYou dataset. The dataset from the social network involves less winning auctions.

Metrics Two metrics are used to check the quality of the predicted bids distribution in the auction.

The first metric is averaged negative log probability (ANLP).

$$ANLP(D_{win}) = -\frac{1}{|D_{win}|} \sum_{(x^i, z^i) \in D_{win}} \log p_{z^i}(z^i \mid x^i) \tag{8}$$

The ANLP metric is calculated on winning auctions D_{win}, because only in these cases the market price z is observed. The metric checks the value of the density distribution at the bid b equal to the market price z.

The ideal algorithm for every auction should predict the winning probability equal to one for a bid, which is greater than, or equal to the market price and zero for the opposite cases. Therefore, the metric value for an ideal algorithm is zero.

The second metric ROC-AUC is ranking metric, it considers algorithm as a classifier.

This metric is calculated on winning and losing auctions. The ROC-AUC metric ranks the predicted probabilities of winning an auction for each bid. In an ideal case the probability for low bids should be close to zero. With bid increasing, the winning probability should rise as well. The winning probability for the bid equal to, or larger than the market price should be close to one.

The ANLP metric is more indicative, since it is responsible for the quality of the prediction.

[1]https://github.com/zeromike/bid-lands

[2]https://github.com/SerTelnov/landscape-forecasting

[3]https://github.com/SerTelnov/landscape-forecasting/blob/master/stuff/useful_links.md

Results

Before starting the analysis of the results, we describe the process of training all the models and resources involved.

All models were trained on virtual machines with the same characteristics. The amazon aws ec2 T2 image was chosen for training. These are scalable instances that provide a baseline and upgradeable CPU performance (Kim and Fischman 2015).

All models were trained until the change in metrics became insignificant or early stopping was detected.

For comparison, the state of the model was saved at the moment of the best result for the entire training time using the ANLP metric (Formula 8). It was the state and the metrics of this model saved that were used to analyze the results.

We use the Student's t-test to check statistical significance of the results.

Here, we provide the initialization of the hyperparameters for the models. For DLF model we take hyperparameters from the article (Ren et al. 2019b).

- $\alpha = 0.25$
- $\beta = 0.75$
- embedding size = 32
- learning rate = 0.0001
- batch size = 128.

Transformer network Table 4 shows the performance of the models on the ANLP metric. The column "Impr" (Improvement) shows the percentage difference between the results of the DLF model and the transformer model.

Based on the comparison results, we prove that the transformer model improved the ANLP metric results by 8.75% on average.

Comparisons for the ROC-AUC metric are shown in Table 3.

The transformer model also improved on this metric when compared to the DLF model. The metric value improve by 1.09%.

Figure 4 show the training DLF and Transformer models on the same iPinYou dataset (3476), here the iteration is the processing of one data batch.

Similar comparisons of the models were made on a social network dataset. ANLP metrics are shown in Table 5. The indicator of this metric improved on average by 0.4854%.

The ROC-AUC metrics are shown in Figure 6. The results of this metric improved on average by 1.06%.

Figure 5 shows predicting bids distribution in an auction. As an example, we take the winning auction from the test iPinYou dataset, where the models are not trained. The figure illustrates the distribution density function and the probability of winning the auction for each bid from the lowest to the highest. There are also two straight lines on the figure: the bid value and the market price of the auction.

Figure 6 gives a similar prediction of probabilities, but for a losing auction. In the losing auction the market price is not observed, so only the losing bid is presented.

Table 7: ANLP metric on cold start

ad id	DLF	pt DLF	%	TLF	pt TLF	%
1458	3.673	**3.659**	-0.39	3.247	**3.197**	-1.53
2259	4.352	**4.139**	-4.9	3.742	**3.697**	-1.2
2261	4.692	**3.741**	-20.26	2.684	**2.550**	-4.98
2821	**3.608**	3.679	+1.97	3.004	**2.926**	-2.62
2997	4.341	**2.351**	-45.85	1.816	**1.813**	-0.14
3358	4.124	**3.944**	-4.36	3.738	**3.143**	-15.91
3386	3.504	**3.297**	-5.92	3.046	**2.908**	-4.52
3427	4.101	**3.619**	-11.74	3.774	**3.608**	-4.39
avg	4.049	**3.554**	-12.24	3.131	**2.980**	-4.82

Table 8: ROC-AUC metric on cold start

ad id	DLF	pt DLF	%	TLF	pt TLF	%
1458	**0.891**	0.881	-1.1	**0.905**	0.903	-0.21
2259	0.759	**0.811**	+6.97	**0.841**	0.832	-1.07
2261	0.821	**0.879**	+7.00	0.911	**0.916**	+0.5
2821	0.802	**0.811**	+1.17	0.844	**0.855**	+1.4
2997	0.727	**0.885**	+21.82	**0.911**	0.907	-0.41
3358	0.879	**0.896**	+1.98	0.927	**0.937**	+1.07
3386	**0.906**	0.906	-0.05	0.920	**0.921**	+0.16
3427	**0.860**	0.860	-0.08	**0.886**	0.886	-0.01
avg	0.831	**0.866**	+4.29	0.893	**0.895**	+0.18

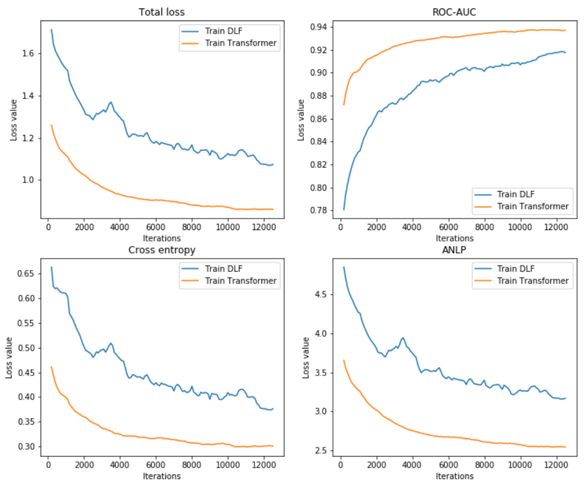

Figure 4: Learning curves over Campaign 3476 of iPinYou dataset

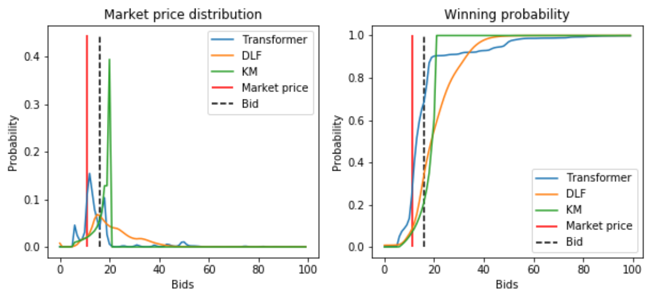

Figure 5: Winning probability prediction on winning auction

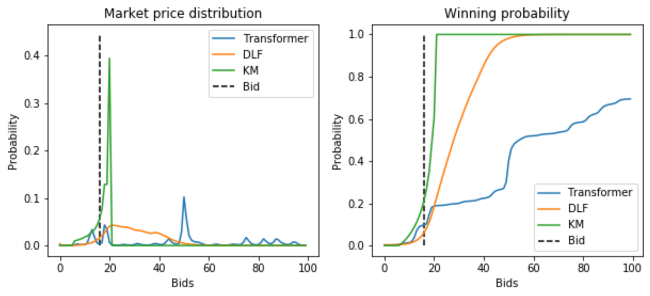

Figure 6: Winning probability prediction on losing auction

Cold start results To reproduce the cold start problem on iPinYou dataset we take the first 5% data from the training data (train). The test dataset did not changed.

DLF and transformer models were trained for each campaign from Table 1 from scratch and using the pre-training model. We compared these two methods to test our hypothesis.

The results of ANLP metric for testing DLF and transformer (TLF) models without pre-training and with pre-training are presented in Table 7. The % column indicates the percentage improvement of the pre-trained model and the regular one. For the DLF model this metric improved by 12.24% on average and for the transformer model (TLF) by 4.82%.

Similar results for the ROC-AUC metric are given in Table 8. For the DLF model the performance improved by 4.28% on average and for the transformer model (TLF) by 0.18%.

Pre-training on another ad dataset improved the performance metrics for both models (DLF and TLF), with the exception of the ROC-AUC metric for the TLF model, where the metrics did not changed significantly. Therefore using the pre-trained model for training a new ad with small amount of data significantly improves performance metrics.

Conclusion and Future Works

We developed an algorithm for modeling the distribution of bids in the auction for all advertisements has been developed, which has significantly improved the quality of predicting the winning probability in comparison with the state-of-the-art solution on two metrics: average negative log probability (ANLP) and the ranking metric ROC-AUC (Ren et al. 2019b).

According to the test results, the new solution on the iPinYou public dataset improved ANLP metrics by 8.75% (6.53% taking into account the additional interval) and ROC-AUC by 1.0953% (0.82% taking into account additional interval). On the dataset of the social network, the ROC-AUC metric improved by 1.06%(0.76% taking into account the additional interval). The new approach also reduces the training time of the model using parallelism.

Moreover, we have described the cold start problem, and our approach using the pre-trained model improves the results by 12.24% on average (10.61% taking into account the additional interval) by the ANLP metric and by 4.28%(3.97% taking into account the additional interval) by the ROC-AUC metric for the DLF model and 4.82%(2.61% taking into account the additional interval) interval) according to the ANLP metric for the transformer model. Based on these results, we conclude that using the pre-trained model for another ad significantly improves the prediction.

For the future work, we are going to use the proposed Transformer model in bid optimization problems.

References

2017. Dropout - solution of overfitting in neural networks. URL https://habr.com/ru/company/wunderfund/blog/330814/.

2017. Four Years Remaining. URL http://fouryears.eu/2017/12/06/the-mystery-of-early-stopping/comment-page-1/.

2019. Transformer model for language understanding : TensorFlow Core. URL https://www.tensorflow.org/tutorials/text/transformer?hl=ru#positional_encoding.

Alammar, J. 2018. The Illustrated Transformer. URL http://jalammar.github.io/illustrated-transformer/.

Box, G. E.; Jenkins, G. M.; Reinsel, G. C.; and Ljung, G. M. 2015. *Time series analysis: forecasting and control*. John Wiley & Sons.

Budhiraja, A. 2016. Learning Less to Learn Better - Dropout in (Deep) Machine learning. URL https://medium.com/@amarbudhiraja/https-medium-com-amarbudhiraja-learning-less-to-learn-better-dropout-in-deep-machine-learning-74334da4bfc5.

Kaplan, E. L.; and Meier, P. 1958. Nonparametric estimation from incomplete observations. *Journal of the American statistical association* 53(282): 457–481.

Kim, T.; and Fischman, S. 2015. Amazon EC2 instance types. URL https://aws.amazon.com/ru/ec2/instance-types/.

Liao, H.; Peng, L.; Liu, Z.; and Shen, X. 2014. iPinYou global rtb bidding algorithm competition dataset 1–6.

Qi, Y.; Sachan, D. S.; Felix, M.; Padmanabhan, S. J.; and Neubig, G. 2018. When and why are pre-trained word embeddings useful for neural machine translation? *arXiv preprint arXiv:1804.06323* .

Ren, K.; Qin, J.; Zheng, L.; Yang, Z.; Zhang, W.; Qiu, L.; and Yu, Y. 2019a. Deep recurrent survival analysis 33: 4798–4805.

Ren, K.; Qin, J.; Zheng, L.; Yang, Z.; Zhang, W.; and Yu, Y. 2019b. Deep Landscape Forecasting for Real-time Bidding Advertising 363–372.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need 5998–6008.

Wang, Y.; Ren, K.; Zhang, W.; Wang, J.; and Yu, Y. 2016. Functional bid landscape forecasting for display advertising 115–131.

Wei, J.; He, J.; Chen, K.; Zhou, Y.; and Tang, Z. 2017. Collaborative filtering and deep learning based recommendation system for cold start items. *Expert Systems with Applications* 69: 29–39.

Wu, W.; Yeh, M.-Y.; and Chen, M.-S. 2018. Deep censored learning of the winning price in the real time bidding. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2526–2535.

Yang, Z.; Yang, D.; Dyer, C.; He, X.; Smola, A.; and Hovy, E. 2016. Hierarchical attention networks for document classification 1480–1489.

Zhang, W.; Zhou, T.; Wang, J.; and Xu, J. 2016. Bid-aware gradient descent for unbiased learning with censored data in display advertising 665–674.

Hardware

All models were trained on virtual machines with the same characteristics. The amazon aws ec2 T2 image is chosen for training. These are scalable performance instances, that provide a baseline and upgradeable CPU performance (Kim and Fischman 2015).

Virtual machine characteristics:

- Intel Xeon E processors
- 2 vCPUs
- 8 GB of RAM
- EBS storage