Neural Auction: End-to-End Learning of Auction Mechanisms for E-Commerce Advertising

Xiangyu Liu[1*], Chuan Yu[1*], Zhilin Zhang[1], Zhenzhe Zheng[2†], Yu Rong[1], Hongtao Lv[2], Da Huo[2], Yiqing Wang[2], Dagui Chen[1], Jian Xu[1], Fan Wu[2], Guihai Chen[2] and Xiaoqiang Zhu[1]
[1]Alibaba Group, China, [2]Shanghai Jiao Tong University, China
[1]{qilin.lxy,yuchuan.yc,zhangzhilin.pt,homer.ry,dagui.cdg,xiyu.xj,xiaoqiang.zxq}@alibaba-inc.com
[2]{zhengzhenzhe,lvhongtao,sjtuhuoda,wangyiqing_2015}@sjtu.edu.cn, {fwu,gchen}@cs.sjtu.edu.cn

arXiv:2106.03593v2 [cs.GT] 14 Jul 2021

ABSTRACT

In e-commerce advertising, it is crucial to jointly consider various performance metrics, *e.g.*, user experience, advertiser utility, and platform revenue. Traditional auction mechanisms, such as GSP and VCG auctions, can be suboptimal due to their fixed allocation rules to optimize a single performance metric (*e.g.*, revenue or social welfare). Recently, data-driven auctions, learned directly from auction outcomes to optimize multiple performance metrics, have attracted increasing research interests. However, the procedure of auction mechanisms involves various discrete calculation operations, making it challenging to be compatible with continuous optimization pipelines in machine learning. In this paper, we design Deep Neural Auctions (DNAs) to enable end-to-end auction learning by proposing a differentiable model to relax the discrete sorting operation, a key component in auctions. We optimize the performance metrics by developing deep models to efficiently extract contexts from auctions, providing rich features for auction design. We further integrate the game theoretical conditions within the model design, to guarantee the stability of the auctions. DNAs have been successfully deployed in the e-commerce advertising system at Taobao. Experimental evaluation results on both large-scale data set as well as online A/B test demonstrated that DNAs significantly outperformed other mechanisms widely adopted in industry.

CCS CONCEPTS

• **Information systems** → **Computational advertising**; • **Theory of computation** → **Algorithmic mechanism design**; • **Computing methodologies** → **Neural networks**.

KEYWORDS

Learning-based Mechanism Design; Neural Auction; E-commerce Advertising

*Equal contribution. †Corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
KDD '21, August 14–18, 2021, Virtual Event, Singapore
© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-8332-5/21/08…$15.00
https://doi.org/10.1145/3447548.3467103

ACM Reference Format:
Xiangyu Liu, Chuan Yu, Zhilin Zhang, Zhenzhe Zheng, Yu Rong, Hongtao Lv, Da Huo, Yiqing Wang, Dagui Chen, Jian Xu, Fan Wu, Guihai Chen and Xiaoqiang Zhu. 2021. Neural Auction: End-to-End Learning of Auction Mechanisms for E-Commerce 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/10.1145/3447548.3467103

1 INTRODUCTION

In online e-commerce, the advertising platform is an intermediary to help advertisers deliver their products to interested users [18]. Auction mechanisms, such as Vickrey-Clarke–Groves (VCG) auction [34], Myerson auction [28] and generalized second-price auction (GSP) [14], have been used to enable efficient ad allocation in various advertising scenarios. On designing auction mechanisms for e-commerce advertising, we need to jointly consider and optimize multiple performance metrics from three major stakeholders, *i.e.*, users, advertisers, and the ad platform. Users look for good shopping experiences, advertisers want to accomplish their ad marketing objectives, and the ad platform would like to extract large revenue while also provide satisfying services to both users and advertisers [2, 39]. Furthermore, the ad platform may balance and adjust the importance of these metrics to satisfy the business's strategies for users and advertisers in different e-commerce scenarios. High-quality user experiences and advertising services would guarantee the long-term prosperity of e-commerce advertising.

The traditional auction mechanisms are suboptimal for the e-commerce advertising with multiple performance metrics in dynamic environments. VCG auction and Myerson auction focus on optimizing either social welfare or revenue, and the procedures of the auctions are also too complex to explain for advertisers. Although GSP auction has nice interpretation and is easy to deploy in industry, the fixed allocation rule limits its capability to optimize multiple performance metrics in dynamic environments. To overcome these limitations, we turn our attention to data-driven auction mechanisms, inspired by the recent increase of interest on leveraging modern machine learning, and in particular deep learning, for auction design [12, 16, 29, 30]. The data-driven auction mechanisms enable us to exploit rich information, such as the context of auction environment and the performance feedback from auction outcomes, to guide the design of flexible allocation rules for optimizing multiple performance metrics, which significantly enlarge the design space of auction mechanisms.

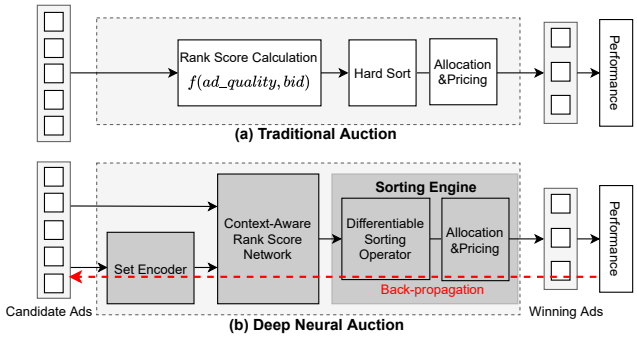


Figure 1: The comparison between traditional auctions and Deep Neural Auction. The set encoder and the context-aware rank score network are applied to extract auction features, which improves representation space and flexibility of rank scores, compared with the fixed rank score in traditional auctions. Furthermore, the differentiable sorting engine makes the auctions, including allocation and pricing, continuous and differentiable w.r.t the inputs, thereby supporting the end-to-end back-propagation training.

However, it remains open to both academia and industry on how to make full use of the powerful deep learning on designing data-driven auction mechanisms for the industrial e-commerce advertising. We consider two critical challenges in this direction. The first one comes from the contradiction between auction mechanism and deep learning in design principle. The auction mechanisms, including allocation and pricing, usually involve various discrete optimization operations, *e.g.*, the top-k ads selection in GSP auction [33], while the deep learning follows an end-to-end pipeline for continuous optimization. This contradiction prevents the performance feedback underlying the auction outcomes from integrating into the back-propagation model training as in deep learning. When designing learning models for data-driven auctions, we also need to take the game theoretical properties, such as *Incentive Compatibility* [34], into account, which further complicates the application of deep learning for auction design. Although some recent works have proposed deep neural network architectures for learning-based auction mechanisms [12, 16, 30], they focused on the theoretical auction setting, either the complex combinatorial auctions [12] or the simple single-bidder auctions [30], in lack of the insights from industrial deployment. Thus, it needs further efforts to integrate deep learning into the design and deployment of end-to-end auction mechanisms for practical industrial setting.

The second challenge is data efficiency. The current learning-based approaches [20, 39] usually require a large number of samples to learn the optimal auction due to an ambiguity issue we observed in the data from auctions. It is a common case that an advertiser with the same feature profile can result in different outcomes in distinct auctions, *e.g.*, wins in one auction but loses in another, due to the change of the auction context, *e.g.*, the competition from the other advertisers. From the machine learning perspective, this may cause an ambiguity issue [23], introducing the one-to-many relation in samples, *i.e.*, the same feature (advertiser feature profile) but contradictory labels (winning or losing). Naive neural network models, which do not incorporate much inductive bias [3], may not fully handle the ambiguity phenomenon on data samples from auctions, resulting in inefficient learning for the end-to-end auction design.

In this paper, we aim to develop a data-efficient end-to-end Deep Neural Auction mechanism, namely DNA, to optimize multiple performance metrics for e-commerce advertising. Considering the nice properties of easy interpretation and deployment in industry, we stick to the rank score-based allocation and second-price payment procedures inherited from GSP auction. As shown in Figure 1, DNA contains a set encoder, a family of context-aware rank score functions and a differentiable sorting engine, with which the process of auction design can be integrated into an end-to-end learning pipeline. Specifically, the set encoder and a carefully designed neural network extract and compress the rich features of auction, such as auction context, bid, advertiser's profile, predicted auction outcomes, etc, into a context-aware rank score, tremendously increasing the representation space and flexibility of rank scores in GSP auction. We further propose a module to relax the sorting in auctions as a differentiable operation, which makes the whole process of GSP auction, including allocation and pricing, to be continuous and fully differentiable with respect to the inputs, and then supports the end-to-end back-propagation training. When designing the learning models for these three modules, we introduce several constraints over the network structures, such as the monotonicity of the neural network in terms of bid, to preserve the game theoretical properties of GSP auction for DNA. Our contributions in this paper can be summarized as follows:

- We make an in-depth study on leveraging the power of deep learning to design data-driven auctions for industrial e-commerce advertising. The proposed end-to-end Deep Neural Auction mechanisms, namely DNA, enable us to optimize multiple performance metrics using the real feedback from historical auction outcomes. The newly designed rank scores also largely enhance the flexibility of ad allocation, making it possible to adjust the auction mechanisms for various performance metrics in dynamic environments.
- We employ three deep learning models to facilitate the design of data efficient and end-to-end learning auction mechanisms with the guarantee of game theoretical property. A set encoder model and a monotone neural network model are proposed to encode various features of auction into the context-aware rank score. With the proposed differentiable sorting engine, we can formulate the design of data-driven auction as a continuous optimization problem, which can be integrated into an end-to-end learning pipeline.
- We have deployed the DNA mechanism in the advertising system at *Taobao*, one of the world's leading e-commerce advertising platforms. Experimental results on both large-scale industrial data set as well as the online A/B test showed that DNA mechanism significantly outperformed other widely used industrial auction mechanisms on optimizing multiple performance metrics, such as Utility-based GSP [2] and Deep GSP [39].

2 PRELIMINARIES

2.1 Ad Auction Model

We describe a typical ad platform architecture in e-commerce advertising. Formally, N advertisers compete for $K \leq N$ ad slots, which

are incurred by a PV (page view) request from the user. Each advertiser i submits bid b_i based on her private information, which could be the probability of the user's behaviors (*e.g.*, *pCTR*, etc.) over the ad, obtained by learning-based prediction module [6, 40]. We use vector $\mathbf{b} = (b_i, \mathbf{b}_{-i})$ to represent the bids of all advertisers, where $\mathbf{b}_{-i}$ are the bids from all advertisers except i. We represent the ad auction mechanism by $\mathcal{M}\langle\mathcal{R}, \mathcal{P}\rangle$, where $\mathcal{R}$ is the ad allocation scheme and $\mathcal{P}$ is the payment rule. The ad allocation scheme would jointly consider the bids and the quality (*e.g.*, *pCTR* and *pCVR*) of the ads in a principled manner. We use $\mathcal{R}_i(b_i, \mathbf{b}_{-i}) = k$ to denote the advertiser i wins the kth ad slot, while $\mathcal{R}_i(b_i, \mathbf{b}_{-i}) = 0$ represents the advertiser loses the auction. The K winning ads would be displayed to the user. The auction mechanism module further calculates the payments for the winning ads with a rule $\mathcal{P}$, which would be carefully designed to guarantee the economic properties and the revenue of the auction mechanism.

2.2 Problem Formulation

Follow the work [39], we formulate the problem as *multiple performance metrics optimization in the competitive advertising environments*. Given bid vector $\mathbf{b}$ from all the advertisers and L ad performance metric functions $\{f_1(\mathbf{b}; \mathcal{M}), .., f_L(\mathbf{b}; \mathcal{M})\}$ (such as Revenue, CTR, CVR, etc), we aim to design an auction mechanism $\mathcal{M}\langle\mathcal{R}, \mathcal{P}\rangle$, such that

$$\begin{aligned} \underset{\mathcal{M}}{\texttt{maximize}} \quad & \mathbb{E}_{\mathbf{b}\sim\mathcal{D}}[F(\mathbf{b}; \mathcal{M})] \\ \text{s.t.} \quad & \textit{Incentive Compatibility (IC) constraint,} \\ & \textit{Individual Rationality (IR) constraint,} \end{aligned} \tag{1}$$

where $\mathcal{D}$ is the advertisers' bid distribution based on which bidding vectors $\mathbf{b}$ are drawn. We define $F(\mathbf{b}; \mathcal{M}) = \lambda_1 \times f_1(\mathbf{b}; \mathcal{M}) + \cdots + \lambda_L \times f_L(\mathbf{b}; \mathcal{M})$, where the objective is to maximize a linear combination of the multiple performance metrics f_l's with preference parameters λ_l's. The parameters λ_l's are the inputs of our problem. The constraints of IC and IR guarantee that advertisers would truthfully report the bid, and would not be charged more than their maximum willing-to-pay for the allocation, which are important for the stability of the ad auction and would be discussed in details in Section 2.3.

In this work, we stick to the design rationale of classical GSP auction mechanism [2, 26], where the allocation scheme is to rank advertisers according to their *rank scores* with a non-increasing order. The pricing rule is to charge the winning advertisers with the minimum bid required to maintain the same ad slot. We study a learning-based GSP auction framework, leveraging the power of deep neural network to design a new rank score function and integrate it into the GSP. We use $r(b_i, \mathbf{x}_i)$ to denote this new rank score function, where $(b_i, \mathbf{x}_i)$ denotes all available information, including bid and other features related to the advertiser i's ad, in the auction. For ease of presentation, we also denote it as $r_i(b_i)$ if there is no ambiguity. The training of this non-linear model is under the guideline of optimization objective in (1). With this new rank score, the allocation scheme and payment rule can be summarized as follows:

• Allocation Scheme $\mathcal{R}$: Advertisers are sorted in a non-increasing order of new rank score $r_i(b_i)$. Without loss of generality, let

$$r_1(b_1) \geq r_2(b_2) \geq \cdots \geq r_N(b_N), \tag{2}$$

then the advertisers with top-K scores would win the corresponding ad slots, with ties broken randomly.

• Payment Rule $\mathcal{P}$: The payment for the winning advertiser i is calculated by the formula:

$$p_i = r_i^{-1}(r_{i+1}(b_{i+1})), \tag{3}$$

where $r_{i+1}(b_{i+1})$ is the rank score of the next highest advertiser, and $r_i^{-1}(\cdot)$ is the inversion function of $r_i(\cdot)$.

2.3 Economic Properties

In the auction mechanism design, one cannot just assume that an advertiser i would truthfully reveal her maximum willing-to-pay price m_i in the auction[2], since they have incentives to misreport these prices to manipulate their own interests [13]. This may seriously harm the stability of the advertising platform. Therefore, we need to guarantee the property of *incentive compatibility* (IC), from mechanism design. This property removes the computational burden of bidding strategy optimization from advertisers, and also, leads to reliable and predictable inputs for the auction mechanisms.

Definition 2.1 (Incentive Compatibility [34]). An auction mechanism is IC if it is in the best interest of each advertiser i to truthfully reveal her maximum willing-to-pay price, *i.e.*, $b_i = m_i$.

In traditional auction theory, the celebrated IC auction mechanisms, such as VCG auction [34] and Myerson auction [28], typically build upon the assumption that advertisers are *utility maximizers*, that is, the goal of each advertiser i is to optimize her quasi-linear utility, defined as the difference between her expected value v_i and the payment p_i, *i.e.*, $u_i = v_i - p_i$. However, we observe from the industrial e-commerce platform that this model could not fully capture the behavior pattern of advertisers. For example, in *Taobao* advertising platform, there are two representative types of advertisers: Optimized Cost Per Click (OCPC) advertisers with upper bounds of bids, and Multi-variable Constrained Bidding (MCB) advertisers with constraints over budgets and the average costs, such as pay-per-click (PPC) and pay-per-acquisition (PPA). The goal of both types of advertisers is to optimize the overall realized value of advertising, such as the quantity of conversions and clicks, under certain constraints over the payments. For these types of advertisers, they would calculate and report a maximum willing-to-pay price for each PV request based on the current status of the ad campaign, with the help of auto-bidding services [36, 41]. This behavior pattern of advertisers could be well captured by the model of *value maximizer* [35], which is defined as follows:

Definition 2.2 (Value maximizer [35]). A value maximizer i optimizes value v_i while keeping payment p_i below her maximum willing-to-pay m_i; when value is equal, a lower p_i is preferred.

In the auctions with multiple ad slots, a value maximizer prefers to the outcome with a higher slot when the payment is below the maximum willing-to-pay price, and then a smaller payment is preferred under the situation with equal value. The strategic behavior pattern of value maximizers would be quite different from the traditional utility maximizers. It has been proved that an auction

[2] m_i is not necessarily equal to the value v_i of the PV request. For example, there may be budget constraints.

mechanism is IC for value maximizers, as long as the following two conditions are satisfied [1, 35]:

• **Monotonicity**: An advertiser would win the same or a higher slot if she reports a higher bid;

• **Critical price**: The payment for the winning advertiser is the minimum bid that she needs to report to maintain the same slot.

We note that IR is also guaranteed under these two IC conditions. Obviously given the monotonicity constraint, the critical price is strictly lower than the bid, and hence is lower than the maximum willing-to-pay price, *i.e.*, $p_i < m_i$. It could be easily verified that GSP satisfies these conditions and hence is IC and IR for value maximizers. In this work, we would design learning-based auction mechanisms, following the above conditions.

3 DEEP NEURAL AUCTION

In this section, we present the details of Deep Neural Auction (DNA) mechanism for optimizing multiple performance metrics under the multi-slot setting for e-commerce advertising.

3.1 Overall Architecture

As illustrated in Fig. 2a, DNA consists of three modules: a set encoder, a context-aware rank score function, and a differentiable sorting engine. The set encoder learns a set embedding from the features of candidate ads, which encodes the context of the auction. This set embedding is attached as a complementary feature for each ad. Next, each advertiser employs a shared MIN-MAX neural network to generate context-aware rank scores from advertiser's features. This neural network is partially monotonic with respective to bids, which is critical to the guarantee of IC property. Another advantage of the designed neural network is a closed form expression for the inverse transform, which enables an easy payment calculation. Then, the differentiable sorting engine conducts a continuous relaxation of sorting operator in auctions, and outputs a row-stochastic permutation matrix. We can use this row-stochastic permutation matrix to express the expected revenue as well as other predicted performance metrics, from which training losses from real feedback underlying the auction outcome can be constructed. With these components, the whole DNA mechanism is fully differentiable with respect to its inputs, and can be integrated into the end-to-end learning pipeline as in deep learning. It is worth to note that both the set encoder and the context-aware rank score function are parameterized neural network models, while the differentiable sorting engine is a non-parameterized operator. We next introduce the details of these three modules.

3.2 Auction Context Encoding

As the final auction outcome is jointly determined by all the candidate ads, we design a set encoder to automatically extract the feature of the auction context from all the candidate ads, instead of the individual ad. We attach this auction context feature as an augmented feature for each ad to overcome the ambiguity issue discussed in Section 1.

The set encoder receives the whole set of ad features as input. As there is no inherent ordering among the advertisers in the set, the feature set is permutation-equivariant [37] to the auction outcome, *i.e.*, the auction outcome does not rely on the ordering of the features. Learning models that do not take this set structure into account (such as MLPs or RNNs) would cause the issue of discontinuities [38]. Inspired from the recent progress on learning on set [37, 38], we implement the set encoder by designing a new deep neural network, which uses DeepSet [37] network to aggregate individual ad features to form a representation for the auction context. The main idea is that, by setting equivariant layers and a final symmetric layer, the DeepSet can learn to aggregate all the features' information in a permutation-equivariant manner. The generated embedding vector can be trained to predict some interested statistical value about the whole set.

Concretely, as illustrated in Fig. 2b, the set encoder is composed of two groups of layers, ϕ_1 and ϕ_2. Given the features of candidate ads $\{\mathbf{x}_i\}_{i=1}^N$, each instance $\mathbf{x}_i$ is firstly mapped to a high-dimensional latent space through shared fully connected layers ϕ_1, resulting in a set of intermediate hidden states $\mathbf{h} = \{h_i\}_{i=1}^N$:

$$h_i = \sigma(\phi_1(\mathbf{x}_i)), \tag{4}$$

where σ is an Exponential Linear Unit (ELU) activation function [7]. Then, this hidden states set is processed with symmetric aggregation pooling (such as average pooling) to build the final set embedding h'_i for each ad i with another fully connected layer ϕ_2:

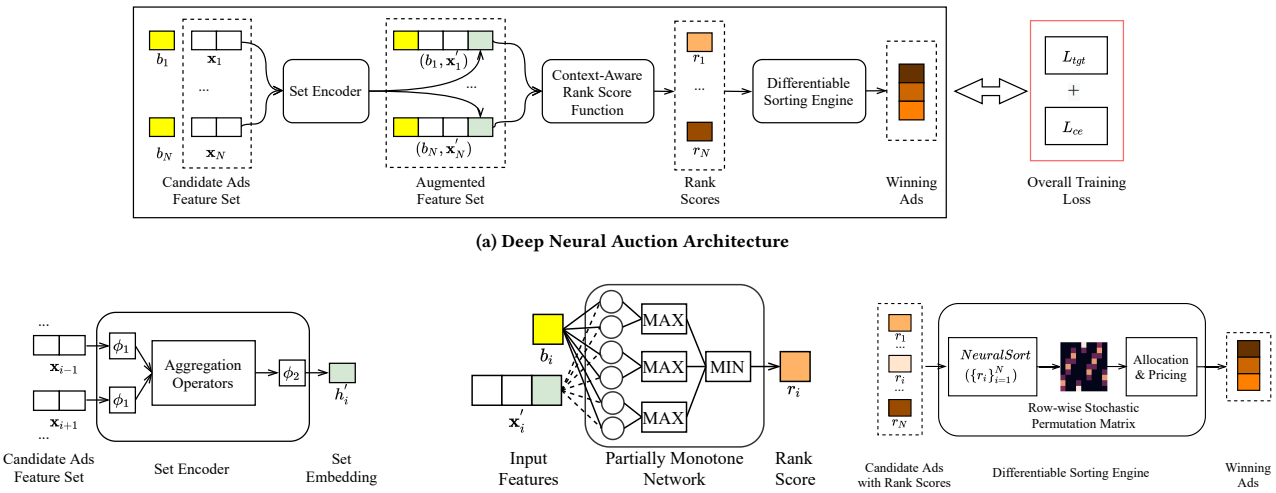
$$h'_i = \sigma(\phi_2(\texttt{avgpool}(\mathbf{h}_{-i}))), \tag{5}$$

where $\mathbf{h}_{-i}$ represent the hidden states from all advertisers except i. This set encoder is built by composing permutation-equivariant operations (shared nonlinear transformation) with symmetric aggregation operations (average pooling) at the end. Since the symmetric operations are commutative to the input items, the output is the same regardless of the order of the items. The set encoder learns to extract the context of auction on the whole set of candidate ads [15, 37], which is driven by the downstream training signals in an end-to-end manner. The output set embedding would be sent to the downstream rank score function as an augmented feature for each ad, which helps to infer each ad's rank score in the current candidate ads set. It should be noted that the set encoder does not include the bids from all candidate ads, as shown in Fig. 2b. This design is specified mainly for the guarantee of IC property, keeping the affection of each ad's rank score only through her bid, which will be elaborated in Section 3.3.

3.3 Context-Aware Rank Score

We design a deep neural network to transform each advertiser's augmented feature to a context-aware rank score. We use $r(b_i, \mathbf{x}'_i)$ to denote this rank score, where $\mathbf{x}'_i$ represents the augmented features except bid, *i.e.*, $\mathbf{x}'_i = (\mathbf{x}_i, h'_i)$. From Section 2.3, we need to satisfy two conditions to guarantee the IC property for value maximizers: monotone allocation and critical price. Thus, we aim to design a strictly monotone neural network with respect to bid, and supports efficient inverse transform given the next highest rank score.

We incorporate the aforementioned constraints within the network architecture design, and restrict the search space to a family of partially monotone parameterized neural network. We model the rank score function as a two-layer feed-forward network with `min` and `max` operations over linear functions [9] as shown Fig. 2c. For Q groups of Z linear functions, we associate strictly positive weights $e^{w_{qz}}$ with b_i and other unconstrained weights w'_{qz} with $\mathbf{x}'_i$, as well as intercepts α_{qz}, where $q = 1, ..., Q, z = 1, ..., Z$. For



(a) Deep Neural Auction Architecture

(b) Set Encoder: Deep Set Architecture

(c) Context-Aware Rank Score Function: Partially Monotone MIN-MAX Network

(d) Differentiable Sorting Engine: NeuralSort Module

Figure 2: (a) The overall Deep Neural Auction architecture. (b) The Deep Set based set encoder receives the whole set of ad features and outputs a set embedding. (c) Partially monotone MIN-MAX neural network based context-aware rank score function. The straight lines represent connections with non-negative weights, whereas the dashed lines represent unconstrained connections. (d) The differentiable sorting engine takes in the generated rank scores and outputs a row-stochastic permutation matrix of `argsort` as well as its corresponding allocation and payments, by using NeuralSort.

simplicity, we denote $r(b_i, \mathbf{x}_i')$ as r_i, and assume $r_1 \geq r_2 \geq ... \geq r_N$ without loss of generality. We can define:

$$r_i = \min_{q\in[Q]} \max_{z\in[Z]} (e^{w_{qz}} \times b_i + w'_{qz} \times \mathbf{x}'_i + \alpha_{qz}). \tag{6}$$

Since each of the above linear function in Eq. (6) is strictly non-decreasing on b_i, so is r_i. This partially monotone MIN-MAX neural network has been proved with the capability to approximate any function [9]. Another particular advantage of this representation is that the inverse transform can be directly obtained from the parameters for the forward transform in a closed form expression. For example, given the next highest rank score r_{i+1}, the payment for advertiser i can be formulated as follows:

$$p_i = \max_{z\in[Z]} \min_{q\in[Q]} e^{-w_{qz}}(r_{i+1} - \alpha_{qz} - w'_{qz} \times \mathbf{x}'_i). \tag{7}$$

With the above designed MIN-MAX neural network, the two conditions for IC property can be satisfied, given the assumption that the bids affect the rank scores only through the input b_i in $r(b_i, \mathbf{x}_i')$. However in the industrial advertising environment, there are some engineered features in $\mathbf{x}_i$ from domain knowledge that may have complex dependence relation with the bids. Therefore the bids may affect the rank scores and then the allocation in a complex way, which may violate this assumption. In a large-scale advertising platform, this effect of a change of one's bid on the rank scores via this route would be quite small from our observations on the industrial data sets. To investigate the influence of this issue on IC property, we conduct comprehensive experiments in Section 4.2.3 to calculate the data-driven IC metric of DNA for value maximizers. We reserve the discussion about the strictly IC DNA mechanism design as an interesting open problem in our future work.

3.4 Differentiable Sorting Engine

After calculating the rank scores of all ads, the mechanism determines the allocation and payment, following Eq. (2) and (3). However, treating allocation and payment outside the model learning (*i.e.*, as an agnostic environment) is in some sense poorly suited for deep learning. That is the processes of allocation and payment (actually the sorting operation) are not natively differentiable, and the gradients must all be evaluated via finite difference or likelihood ratio methods (such as the policy search used in Deep GSP [39]), with some additional issues of convergence stability and data-efficiency. Also, in another line of related work, the model-based reinforcement learning (RL) has achieved some notable successes [27]. Some recent works used a general neural network to learn a differentiable dynamic model [24] and argued that the model-based approaches are often more superior and data-efficient than model-free RL methods for many tasks [10, 22]. These insights give us a motivation to model the whole process of allocation and payment inside the Deep Neural Auction framework.

In various types of auction mechanisms, both the allocation and payment are built on a basic sorting operation. Sorting operation outputs two vectors, neither of which is differentiable. On the one hand, the vector of sorted values is piecewise linear. On the other hand, the sorting permutation (more specifically, the vector of ranks via `argsort` operator) also has no differentiable properties as it is integer-valued.

To overcome this issue, we propose a differentiable sorting engine that caters to the top-K selection in the multi-slot auctions. We present a novel use of differentiable sorting operator, *i.e.*, NeuralSort [21], to derive a differentiable top-K permutation matrix, which

can be used to generate the various expected outcomes of the auctions. Given a set of unsorted rank scores $\mathbf{r} = [r_1, r_2, \cdots, r_N]^T$, we are concerned with the `argsort` operator, where `argsort`$(\mathbf{r})$ returns the permutation that sorts $\mathbf{r}$ in a decreasing order. Formally, we define the `argsort` operator as the mapping from N-dimensional real vectors $\mathbf{r} \in \mathbb{R}^N$ to the permutations over N elements, where the permutation matrix $M_r \in \{0,1\}^{N\times N}$ is expressed as

$$M_r[k,i] = \begin{cases} 1 & \text{if } i = \texttt{argsort}(\mathbf{r})[k], \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

Here $M_r[k,i]$ indicates if r_i is the kth largest rank score in $\mathbf{r}$. The results from [21] showed the identity:

$$M_r[k,i] = \begin{cases} 1 & \text{if } i = \texttt{argmax}(c_k), \\ 0 & \text{otherwise,} \end{cases} \tag{9}$$

where $c_k = (N+1-2k)\mathbf{r} - A_r\mathbb{1}$, A_r denotes the absolute pairwise differences of elements in $\mathbf{r}$ such that $A_r[i,j] = |r_i - r_j|$, and $\mathbb{1}$ denotes the column vector of all ones. Then, by relaxing the operator `argmax` in Eq. (9) by a row-wise `softmax`, we can arrive at the following continuous relaxation for the `argsort` operator M_r, which is called NeuralSort in [21]:

$$\hat{M}_r[k,:] = \texttt{softmax}(\frac{c_k}{\tau}), \tag{10}$$

where $\tau > 0$ is a temperature parameter that controls the degree of the approximation, and as $\tau \to 0$, $\hat{M}_r \to M_r$. Intuitively, the kth row of $\hat{M}_r$ can be interpreted as the 'choice' probabilities on all elements in $\mathbf{r}$, for getting the kth highest item.

This row-stochastic permutation matrix $\hat{M}_r$, can be used as a basic operator to construct task-specific sorting procedures according to the order of generated rank scores in a differentiable manner. For instance, if we let $\mathbf{p} = [p_1, p_2, \cdots, p_N]^T$ denotes the payments calculated by Eq. (7) for N advertisers in a PV request, then the top-K payments, sorted by their corresponding rank scores, can be recovered by a simple matrix multiplication:

$$f_{pay} = \hat{M}_r[1{:}K,:] \cdot \mathbf{p}. \tag{11}$$

This row-stochastic permutation matrix $\hat{M}_r$ acts as a differentiable sorting engine that makes the discrete sorting procedure compatible with differentiability.

3.5 End-to-End Model Training

3.5.1 Data for Training. All data sets we used were generated under GSP auction, which is IC for e-commerce value maximizing advertisers [35]. The data contains all advertisers' bids, the estimated values (*e.g.*, *pCTR*, *pCVR*), ads information (*e.g.*, category, price of product), user features (*e.g.*, genders, age, income level) as well as the context information (*e.g.*, the source of traffic). These information consists of the input features of the DNA architecture. The data also contains the real feedback information (*e.g.*, click, conversion or transaction) from users.

3.5.2 Training Loss. As the training data contains the user feedback for each ad exposure, we can directly use the row-stochastic permutation matrix $\hat{M}_r$ to compute the K-slots expected performance metrics via: $\hat{M}_r[1{:}K,:] \cdot F_{all}$, where F_{all} represents the vector of aggregated performance metrics for all ads from real feedback:

$$F_{all} = [\sum_{l=1}^{L} \lambda_l \times f_l^1, \cdots, \sum_{l=1}^{L} \lambda_l \times f_l^N]^T, \tag{12}$$

with f_l^i standing for the lth performance metric for ith ad from a PV request. Therefore, we can formulate the learning problem as minimizing the sum of top-K expected negated performance metrics for each PV request:

$$\mathcal{L}_{tgt} = -\sum_{i=1}^{K} \hat{M}_r[i,:] \cdot F_{all}. \tag{13}$$

One exception is the calculation of revenue. Due to the change of allocation order, the payment for each ad is distinct from what has happened. Thus we use the generated payments, defined in Eq. (11) to replace the ones appeared in the training data.

We set another auxiliary task to help train the DNA mechanism. With the benefit of hindsight from real feedback, we can access the optimal allocation to maximize the performance metrics in each PV request. Thus we set another multiclass prediction task, whose loss is the row-wise cross-entropy (CE) between the ground-truth and the predicted row-stochastic $N \times N$ permutation matrix:

$$\mathcal{L}_{ce} = -\frac{1}{N}\sum_{k=1}^{N}\sum_{i=1}^{N} \mathbb{1}(M_y[k,i] = 1)\texttt{log}\hat{M}_r[k,i], \tag{14}$$

where M_y is the ground-truth permutation matrix, calculated by sorting their real feedback. We found that this auxiliary task was beneficial to yield a stable training process in our experiments. We use a hyper-parameter to balance the target loss $\mathcal{L}_{tgt}$ and the cross-entropy term $\mathcal{L}_{ce}$.

However, the user feedback, especially with respect to the conversion behaviors, is scarce in industrial e-commerce advertising, *e.g.*, users typically decide to purchase a product after seeing dozens of ads. To alleviate this problem, we replace the sparse user behaviors (typically one-hot) in data with the dense values from the prediction model (such as *pCTR* and *pCVR*), and debias these predicted values with real user behaviors by the *calibration* techniques [4, 11]. We also eliminate the deviation of CTR between different slots by debiasing with the posterior inherent CTR of different slots.

4 EXPERIMENTAL EVALUATIONS

4.1 Experiment Setup

4.1.1 Evaluation Metrics. We consider the following metrics in our offline and online experiments, which reflect the platform revenue, user experience, as well as advertisers' utility in the e-commerce advertising. For all experiments in this paper, metrics are normalized to a same scale.

1) Revenue Per Mille (RPM). $RPM = \frac{\sum click \times PPC}{\sum impression} \times 1000$.

2) Click-Through Rate (CTR). $CTR = \frac{\sum click}{\sum impression}$.

3) Conversion Rate (CVR). $CVR = \frac{\sum order}{\sum impression}$.

4) GMV Per Mille (GPM). $GPM = \frac{\sum merchandise\ volume}{\sum impression} \times 1000$.

Apart from the advertising performance indicators, we also evaluate the effectiveness of our designed learning-based auction mechanisms on the property of IC.

5) IC Metric (Ψ). We propose a new data-driven metric of IC, Ψ, to represent the ex-post regret of value maximizers, similar to the data-driven IC for utility maximizers [17]. The metric of Ψ consists of the regret on value Ψ_v and the regret on payment Ψ_p, which denotes, through bid perturbation, the maximum percentage of value increase under the payment constraint, and the maximum percentage of payment decrease with the identical allocation, respectively. Concretely, we formulate $\Psi = (\Psi_v, \Psi_p)$ as[3]:

$$\Psi_v = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{\beta_{k_i}}\max_{b'_i}((\beta_{k'_i}-\beta_{k_i})\times\mathbb{1}(p_i(b'_i)<m_i)), \tag{15}$$

$$\Psi_p = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{\beta_{k_i}p_i(b_i)}\max_{b'_i}((\beta_{k_i}p_i(b_i)-\beta_{k'_i}p_i(b'_i))\times\mathbb{1}(k'_i=k_i)), \tag{16}$$

where we denote k_i and k'_i as the allocated slot indexes of advertiser i when bidding truthfully and bidding a perturbed b'_i, respectively. We use β_k as the click-through rate of slot k, and $p_i(b_i)$ as the payment of advertiser i when bidding b_i. It should be noted that the value of a click v_i for advertiser i is reduced from the fraction in Eq. (15). Intuitively, Ψ measures to what extent a value-maximizing advertiser could get better off via manipulating her bid. A larger value of Ψ_v indicates that an advertiser could obtain more extra value under the constraint of maximum willing-to-pay price. Similarly, a larger value of Ψ_p indicates that an advertiser could be undercharged more while obtaining the same ad slot. For example, as GSP is IC for value maximizers, its values of both Ψ_v and Ψ_p are 0.

4.1.2 Baselines Methods. We compare DNA with the widely used mechanisms in the industrial ad platform.

1) Generalized Second Price auction (GSP). The rank score in the classical GSP is simply the bids multiplying $pCTR$, namely effective Cost Per Milles (eCPM). The payment rule is the value of the minimum bid required to retain the same slot. The work [26] suggested incorporating a squashing exponent σ into the rank score function, *i.e.*, $bid \times pCTR^{\sigma}$ could improve the performance, where σ can be adjusted to weight the performance of revenue and CTR. We refer to this exponential form extension as GSP in the experiments.

2) Utility-based Generalized Second Price auction (uGSP). uGSP extends the conventional GSP by taking the rank score as a linear combination of multiple performance metrics using estimated values: $r_i(b_i) = \lambda_1 \times b_i \times pCTR_i + o_i$, where o_i represents other utilities, such as CTR and CVR: $o_i = \lambda_2 \times pCTR_i + \lambda_3 \times pCVR_i$(where $\lambda_l \geq 0$). The payment of uGSP follows the principle from GSP: $p_i = \frac{\lambda_1 \times b_{i+1} \times pCTR_{i+1} + o_{i+1} - o_i}{\lambda_1 \times pCTR_i}$. uGSP is widely used in industry to optimize multiple performance metrics [2].

3) Deep GSP [39] Deep GSP uses a deep neural network to map ad's related features to a new rank score within the GSP auction. This new rank score function is optimized using model-free reinforcement learning to maximize the interested performance metrics.

[3]Since our training data comes from a vanilla GSP mechanism, which is IC for value maximizers, we directly take the bid b_i as the maximum willing-to-pay price m_i of advertiser i.

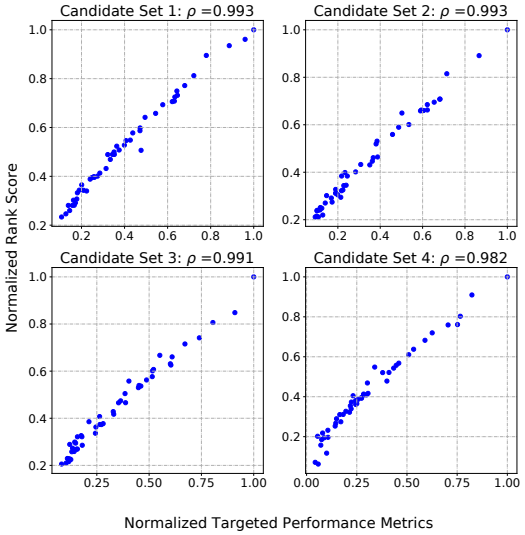


Figure 3: The positive correlations between learned rank scores and the targeted performance metrics. Each blue dot represents an ad in the candidate set.

4.2 Offline Experiments

4.2.1 Datasets. The data sets we used for experiments come from *Taobao*, a leading e-commerce advertising system in China. We randomly select 5 million records logged data under GSP auctions from *July 4, 2020* as training data, and 870k records logged data from *July 5, 2020* as test data. Unless stated otherwise, all experiments are conducted under the setting of top-3 ads displayed (*i.e.*, 3-slot auctions) in each PV request. Other details about the model configurations and training procedure are in Appendix A.

4.2.2 Performance in Offline Simulations. We conduct experiments to compare the performance of DNA and other baseline mechanisms. In order to facilitate intuitive comparisons, we set only two performance metrics with the form $\lambda \times RPM + (1-\lambda) \times X$, where X is one of the metric selected from $\{CTR, CVR, GPM\}$. For uGSP, we set the rank score function with $\lambda \times pCTR \times bid + (1-\lambda) \times X$. For GSP, we tune the variable σ in the interval $[0.5, 2.0]$. For both Deep GSP and DNA, we directly set the objective by selecting the values of λ uniformly from the interval $[0, 1]$.

We show the relation between the learned rank scores and the targeted performance metrics, and illustrate some results in Fig. 3. We calculate the Pearson's Correlation Coefficient (ρ), which is 0.989 on the test data set, together with the p-value less than 1e−7. This result indicates the strong positive correlation between the learned rank scores and the performance metrics, implying that the ads with higher targeted objectives also have higher rank scores. This would encourage advertisers to optimize their ads' quality (such as CTR) to enhance their competitiveness in the auctions.

As some performance metrics may be conflicting, we next plot the Pareto Front for different baselines in Fig. 4. We observe that all the learning-based methods (both Deep GSP and DNA) are above the curves of other baselines. The flexible learning-based rank score models have the ability to perform automatic feature extraction

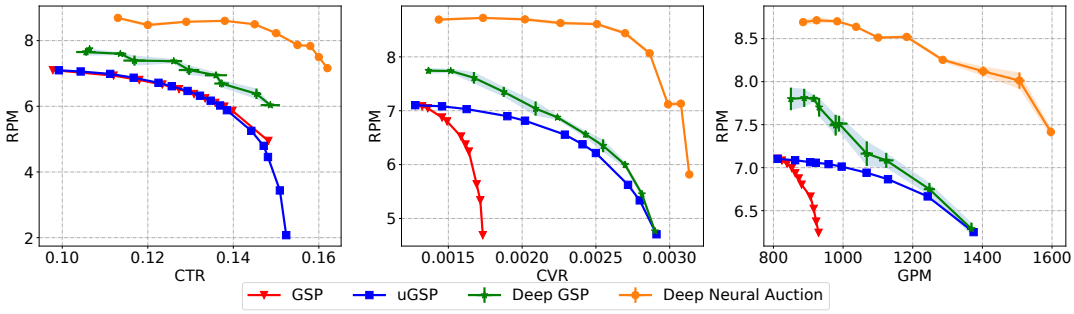


Figure 4: The performance of DNA and other baseline mechanisms in the offline experiments.

from raw data. Learning-based methods can alleviate the problem of inaccurate predicted values (such as *pCTR*) used in GSP and uGSP to some extend, and learn ad auctions directly from the real feedback. We also note that the performance of GSP is poor when considering CVR and GPM, as GSP does not model the effect of these indicators explicitly in its rank score formulation.

DNA outperforms Deep GSP baseline by a clear margin. The rank score function of Deep GSP is only conditioned on each ad's private information. While in DNA, it also contains a set embedding which models the context of auction from the candidate ads set explicitly, making it more competitive than the rank scores in the Deep GSP auctions. We analyze this set embedding in more details in Appendix B. The upgrade of training method also contributes to this superior performance improvement. Deep GSP treats the whole process of allocation as the environment and uses exploration based algorithms (*i.e.*, policy search) to optimize the rank score functions. In comparison, DNA directly differentiates the sorting procedure in allocation, which is more data-efficient.

4.2.3 Evaluation on IC Property. We present the IC property of DNA, *i.e.*, the regret metric Ψ, in Table 1. We compare with only the regret of truthful bidding in the Utility-based Generalized First Price (uGFP) auction, as the regret of other mechanisms (GSP and uGSP) are all 0. uGFP mechanism allocates the ad slots in the same way as uGSP while the payment of a winning advertiser is simply her bid. One can observe from Table 1 that Ψ_v of both DNA and uGFP are 0, which indicates that advertisers could not win higher slots under the payment constraints. For Ψ_p, DNA outperforms uGFP significantly under all experiment settings. For example of the first row (1.0×RPM), an advertiser could be undercharged at most 0.042% payment in DNA with the same allocation result, which is 13.312% in uGFP.

4.3 Online Experiments

We present the online experiments by deploying the proposed DNA in *Taobao* advertising system. As we only use the trained model to make inference in online system, we set the temperature parameter $\tau = 0$ in the differentiable sorting engine to output the winning ads as well as the corresponding payments to the online advertising platform. Other deployment details are described in Appendix C.

To demonstrate the performance of DNA, we conduct online A/B tests with 1% of whole production traffic from Jan 25, 2021 to Feb 8, 2021 (about one billion auctions). We also consider RPM, CTR, CVR and GPM metrics and conduct online experiments as in the offline experiments, *i.e.*, setting only two performance metrics at a time. In order to make fair and efficient comparisons between different baselines in production traffic, we set the λ in uGSP with 0.8, and tune the λ's of both Deep GSP and DNA until the observed RPM performance reaches the same level with the one in uGSP. Then we record the relative improvements of the other metrics of Deep GSP and DNA compared with uGSP, which is shown in table 2. From the results, we can find that the DNA mechanism achieves the highest promotion for CTR, CVR and GPM. To verify the performance stability of the DNA, we conduct a relatively long-term experiment (nearly two months) to compare with the GSP auction. Table 3 shows that all the performance metrics related to users, advertisers, and advertising platform are promoted. Considering the massive advertisers and users, the promotion of marketing performance verifies the effectiveness of our proposed DNA framework.

Table 1: IC Metric (Ψ) experiments under four tasks with 1000 PV requests randomly selected from the test data. For each bidder, we randomly generate 100 perturbations (ranging from 0.0 to 2.0 times) to her value.

	DNA		uGFP	
	Ψ_v	Ψ_p	Ψ_v	Ψ_p
1.0×RPM	0	**0.042%**	0	13.312%
0.5×RPM+0.5×CTR	0	**0.059%**	0	21.616%
0.5×RPM+0.5×CVR	0	**0.118%**	0	19.280%
0.5×RPM+0.5×GPM	0	**0.028%**	0	16.400%

Table 2: Online A/B test compared with uGSP on promoting different performance metrics, keeping the same RPM level.

% Improved	Deep GSP	DNA
CTR	+6.43%	**+11.58%**
CVR	+6.38%	**+31.26%**
GPM	+2.77%	**+16.17%**

Table 3: Online A/B test (nearly two months) compared with GSP on promoting all performance metrics.

	RPM	CTR	CVR	GPM
% Improved	+5.68%	+18.93%	+14.68%	+14.53%

5 RELATED WORK

A plethora of works have used learning-based techniques for revenue-maximizing mechanism design in theoretical auction settings. Conitzer and Sandholm [8] proposed the paradigm of automated mechanism design (AMD) and laid the groundwork for this direction. Since then, many works [20, 25] have adopted learning approaches for mechanism design problems. Recently, Dütting et al. [12] first leveraged deep neural networks for the automated design of optimal auctions. Several other works extended this study for various scenarios [16, 19, 29].

A particular stream of research have focused on the mechanism design in online advertising. The GSP and VCG auction have been widely adopted and investigated in various advertising system [14, 26]. Building on the success of deep reinforcement learning, Tang *et al.* [5, 31] proposed reinforcement mechanism design for the optimization of reserve prices in online advertising. A recently proposed learning-based ad auction mechanism, called Deep GSP [39], leveraged the deep learning technique to optimize multiple performance metrics in e-commerce advertising.

6 CONCLUSION

In this paper, we have proposed a Deep Neural Auction mechanism, towards learning data efficient and end-to-end auction mechanisms for e-commerce advertising. We have deployed the DNA mechanism on one of the leading e-commerce advertising platforms, *Taobao*. The offline experiments as well as the online A/B test showed that DNA mechanism significantly outperformed other existing auction mechanism baselines on optimizing multiple performance metrics.

ACKNOWLEDGMENTS

This work was supported in part by Science and Technology Innovation 2030 –"New Generation Artificial Intelligence" Major Project No. 2018AAA0100905, in part by China NSF grant No. 62025204, 62072303, 61902248, and 61972254, and in part by Alibaba Group through Alibaba Innovation Research Program, and in part by Shanghai Science and Technology fund 20PJ1407900. The authors would like to thank Rui Du, Haiping Huang, Haiyang He and Guan Wang who did the really hard work for online system implementation. The opinions, findings, conclusions, and recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies or the government.

REFERENCES

[1] Gagan Aggarwal, S Muthukrishnan, Dávid Pál, and Martin Pál. 2009. General auction mechanism for search advertising. In *WWW*. 241–250.
[2] Yoram Bachrach, Sofia Ceppi, Ian A Kash, Peter Key, and David Kurokawa. 2014. Optimising trade-offs among stakeholders in ad auctions. In *EC*. 75–92.
[3] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. 2018. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261* (2018).
[4] Alexey Borisov, Julia Kiseleva, Ilya Markov, and Maarten de Rijke. 2018. Calibration: A simple way to improve click models. In *CIKM*. 1503–1506.
[5] Qingpeng Cai, Aris Filos-Ratsikas, Pingzhong Tang, and Yiwei Zhang. 2018. Reinforcement Mechanism Design for e-commerce. In *WWW*. 1339–1348.
[6] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*. 7–10.
[7] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. 2016. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). In *ICLR*.
[8] Vincent Conitzer and Tuomas Sandholm. 2002. Complexity of mechanism design. In *UAI*. 103–110.
[9] Hennie Daniels and Marina Velikova. 2010. Monotone and partially monotone neural networks. *IEEE Transactions on Neural Networks* 21, 6 (2010), 906–917.
[10] Filipe de Avila Belbute-Peres, Kevin Smith, Kelsey Allen, Josh Tenenbaum, and J Zico Kolter. 2018. End-to-end differentiable physics for learning and control. *NeurIPS* 31 (2018), 7178–7189.
[11] Chao Deng, Hao Wang, Qing Tan, Jian Xu, and Kun Gai. 2021. Calibrating User Response Predictions in Online Advertising. In *ECML PKDD 2020*. 208–223.
[12] Paul Dütting, Zhe Feng, Harikrishna Narasimhan, David Parkes, and Sai Srivatsa Ravindranath. 2019. Optimal Auctions through Deep Learning. In *ICML*. 1706–1715.
[13] Benjamin Edelman and Michael Ostrovsky. 2007. Strategic bidder behavior in sponsored search auctions. *Decision support systems* 43, 1 (2007), 192–198.
[14] Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz. 2007. Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. *American economic review* 97, 1 (2007), 242–259.
[15] Harrison Edwards and Amos J. Storkey. 2017. Towards a Neural Statistician. In *ICLR*.
[16] Zhe Feng, Harikrishna Narasimhan, and David C Parkes. 2018. Deep learning for revenue-optimal auctions with budgets. In *AAMAS*. 354–362.
[17] Zhe Feng, Okke Schrijvers, and Eric Sodomka. 2019. Online learning for measuring incentive compatibility in ad auctions. In *WWW*. 2729–2735.
[18] Avi Goldfarb and Catherine Tucker. 2011. Online display advertising: Targeting and obtrusiveness. *Marketing Science* 30, 3 (2011), 389–404.
[19] Noah Golowich, Harikrishna Narasimhan, and David C Parkes. 2018. Deep Learning for Multi-Facility Location Mechanism Design.. In *IJCAI*. 261–267.
[20] Negin Golrezaei, Max Lin, Vahab Mirrokni, and Hamid Nazerzadeh. 2017. Boosted second price auctions: Revenue optimization for heterogeneous bidders. *Available at SSRN 3016465* (2017).
[21] Aditya Grover, Eric Wang, Aaron Zweig, and Stefano Ermon. 2019. Stochastic Optimization of Sorting Networks via Continuous Relaxations. In *ICLR*.
[22] David Ha and Jürgen Schmidhuber. 2018. World models. *arXiv preprint arXiv:1803.10122* (2018).
[23] Eyke Hüllermeier and Jürgen Beringer. 2006. Learning from ambiguously labeled examples. *Intelligent Data Analysis* 10, 5 (2006), 419–439.
[24] Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and Pieter Abbeel. 2018. Model-Ensemble Trust-Region Policy Optimization. In *ICLR*.
[25] Sébastien Lahaie. 2011. A kernel-based iterative combinatorial auction. In *AAAI*, Vol. 25.
[26] Sébastien Lahaie and David M Pennock. 2007. Revenue analysis of a family of ranking rules for keyword auctions. In *EC*. 50–56.
[27] Thomas M Moerland, Joost Broekens, and Catholijn M Jonker. 2020. Model-based reinforcement learning: A survey. *arXiv preprint arXiv:2006.16712* (2020).
[28] Roger B Myerson. 1981. Optimal auction design. *Mathematics of operations research* 6, 1 (1981), 58–73.
[29] Jad Rahme, Samy Jelassi, and S. Matthew Weinberg. 2021. Auction Learning as a Two-Player Game. In *ICLR*.
[30] Weiran Shen, Pingzhong Tang, and Song Zuo. 2019. Automated mechanism design via neural networks. In *AAMAS*. 215–223.
[31] Pingzhong Tang. 2017. Reinforcement mechanism design.. In *IJCAI*. 5146–5150.
[32] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, 11 (2008).
[33] Hal R Varian. 2007. Position auctions. *international Journal of industrial Organization* 25, 6 (2007), 1163–1178.
[34] William Vickrey. 1961. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of finance* 16, 1 (1961), 8–37.
[35] Christopher A Wilkens, Ruggiero Cavallo, and Rad Niazadeh. 2017. GSP: the cinderella of mechanism design. In *WWW*. 25–32.
[36] Xun Yang, Yasong Li, Hao Wang, Di Wu, Qing Tan, Jian Xu, and Kun Gai. 2019. Bid optimization by multivariable control in display advertising. In *KDD*. 1966–1974.
[37] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. 2017. Deep sets. In *NIPS*. 3391–3401.
[38] Yan Zhang, Jonathon Hare, and Adam Prügel-Bennett. 2019. FSPool: Learning Set Representations with Featurewise Sort Pooling. In *ICLR*.
[39] Zhilin Zhang, Xiangyu Liu, Zhenzhe Zheng, Chenrui Zhang, Miao Xu, Junwei Pan, Chuan Yu, Fan Wu, Jian Xu, and Kun Gai. 2021. Optimizing Multiple Performance Metrics with Deep GSP Auctions for E-commerce Advertising. In *WSDM*. 993–1001.
[40] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *KDD*. 1059–1068.
[41] Han Zhu, Junqi Jin, Chang Tan, Fei Pan, Yifan Zeng, Han Li, and Kun Gai. 2017. Optimized cost per click in taobao display advertising. In *KDD*. 2191–2200.

A MODEL CONFIGURATIONS AND OFFLINE TRAINING PROCEDURE

In the set encoder module (Fig. 2b), ϕ_1 consists of two fully connected layers with 128, 32 neurons respectively. ϕ_2 is a single fully connected layer with 16 neurons, thus the final size of the set embeddings h'_i is 16. The ELU non-linearity is applied to the output of every layer. In the context-aware rank score module (Fig. 2c), we use 5 groups of 20 linear functions in the partially monotone MIN-MAX neural network, *i.e.*, $Q = 5, Z = 20$.

We use the Adam optimizer with $\beta_1 = 0.9, \beta_2 = 0.99, \epsilon = 1e{-}8$. Each batch contains 128 PV requests in total. We also leverage some temperature annealing schedules for adjusting τ in the differentiable sorting engine during the training process, such as polynomial decay and exponential decay. But we did not observe significant performance differences between these schedules. The offline training procedure of DNA is as follows:

Algorithm 1 Offline Training Procedure of DNA

Require: Online log data with user behaviors, temperature annealing schedule in the differentiable sorting engine

1: Data preprocessing: feature construction, ground-truth label generation
2: Initialize the neural network parameters of the set encoder and the context-aware rank score function, initialize the temperature τ in the differentiable sorting engine
3: **while** not converged **do**
4: Sample a random minibatch from training data
5: Compute the target loss $\mathcal{L}_{tgt}$ and the cross-entropy loss $\mathcal{L}_{ce}$ with Eq. (13)(14), given the generated rank scores and payments with Eq. (4)(5)(6)(7)
6: Update the network parameters using stochastic gradient descent optimizer (*i.e.*, Adam)
7: Decrease τ by one step
8: **end while**

B ANALYSIS OF THE SET EMBEDDINGS

We next provide empirical evidence suggesting the meaningfulness of set embedding h'_i learned by the set encoder. We conducted experiments on training DNA mechanism without the set encoder. The learning curves on four different tasks are plotted in Fig. 5. We find that the learning performance degrades when disabling the set encoder, indicating that the context-aware rank scores are beneficial to promote the performance. To qualitatively study the latent set embeddings from the set encoder, we randomly select some *top-10* and *last-10* ads from the test data set and generate their corresponding set embeddings h'_i using the trained set encoder. The t-SNE [32] plots can be seen in Fig. 6, where each point represents an ad. It is interesting to find that the top-ranked ads are clustered together, and the "weak" ads are separated from the cluster. This indicates that the set embedding h'_i may carry the "competitiveness" information from the other candidate ads, assisting the subsequent rank score module to learn rank scores towards optimizing the overall performance.

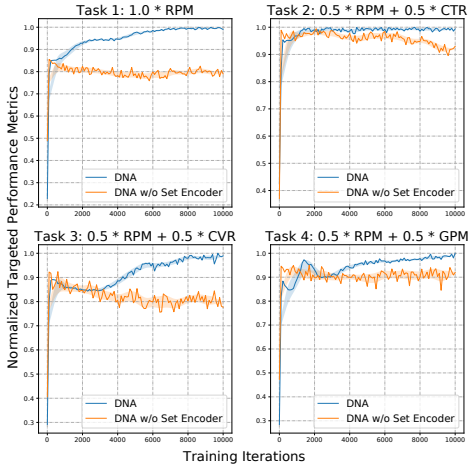


Figure 5: Learning curves on DNA and DNA w/o set encoder in four training tasks.

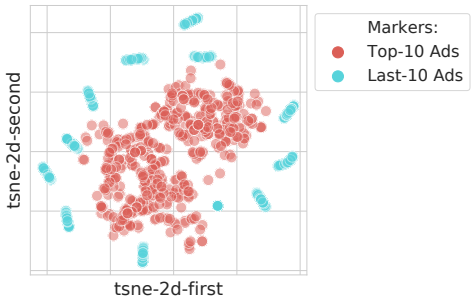


Figure 6: Distribution of the set embedding for sampled tok-10 and end-10 ads in latent space using t-SNE.

C DEPLOYMENT IN E-COMMERCE ADVERTISING SYSTEM

The Deep Neural Auction mechanism is deployed under "Advertisement Intelligent Decision-mAking system" (AIDA) in Taobao display advertising system. The online inference procedure can be formulated as follows:

Algorithm 2 Online Auction Service of DNA

Require: Online auction data (all candidate ads in a PV request), the trained DNA

1: Data preprocessing: feature construction
2: Generate rank scores of DNA with Eq. (4)(5)(6)(7)
3: Obtain the top-K winning ads and their corresponding payments with differentiable sorting engine by setting $\tau = 0$

Fig. 7 shows the overall architectures of the training system pipeline, including online ad platform and offline trainer. Firstly, the online ad platform receives a page view (PV) request from a user. The relevant candidate ads are selected, together with the generated user response predictions (*e.g.*, *pCTR*, *pCVR*). Then, the auction

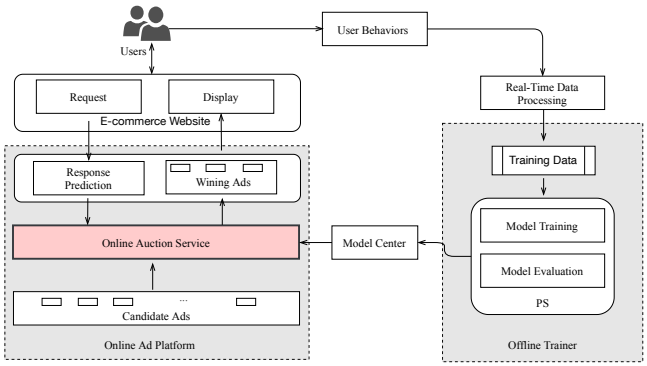


Figure 7: The Pipeline of Training System.

mechanism is conducted and the top-K winning ads will be selected and displayed to the user. Once the user finishes interacting with the ad, *e.g.*, click, or place an order, these behaviors are recorded as log data and sent to the real-time data processing module, where hundreds of thousands of log data are processed per second. After that, the training procedure is performed via parameter-server (PS) framework and the model evaluation will be periodically carried on to monitor the training process. When the convergence condition is satisfied, the model checkpoint will be pushed to the model center module. The model checkpoint can be delivered in several minutes from offline to online. We also design a model version management tool inside the model center, endowing the training system with rollback ability in response to training crash or service breakdown. The whole pipeline from real-time data processing to model checkpoint transmission can be completed in less than 20 minutes.

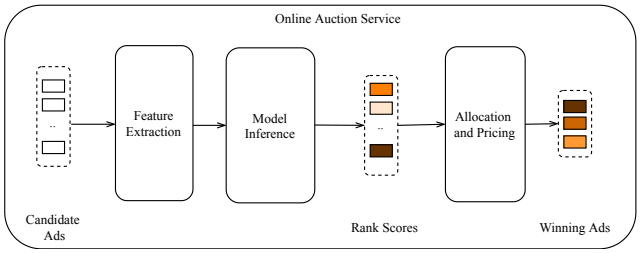


Figure 8: Online Auction Service.

The online auction service, as illustrated in Fig. 8, consists of three main components: a feature extraction module, a model inference module, and an allocation & pricing module. Given the candidate ads set with the available information, the feature extraction module conducts feature engineering, which extracts useful information from the raw user and ads. The trained model checkpoint is then loaded and executed to generate rank scores for all ads in the model inference module. Finally, the allocation and payments are determined using the differentiable sorting engine in DNA (by setting $\tau = 0$). We mainly focus on response time (RT) for the online auction service. The online auction service processes tens of thousands of ad auctions per second, and the RT is 6ms average with dozens of 32-CPU-cores servers. In the last Double Eleven shopping festival of *Taobao*, the online auction service accommodated a tremendous nearly 1 million ad auctions per second during the peak, keeping all services in working order.