# Optimal-er Auctions through Attention

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

RegretNet is a recent breakthrough in the automated design of revenue-maximizing auctions. It combines the expressivity of deep learning with the regret-based approach to relax the Incentive Compatibility constraint (that participants benefit from bidding truthfully). We propose two independent modifications of RegretNet, namely a new neural architecture based on the attention mechanism, denoted as RegretFormer, and a new interpretable loss function that is significantly less sensitive to hyperparameters. We investigate both proposed modifications in an extensive experimental study and additionally test out-of-setting generalization of our network. In all experiments, we find that RegretFormer consistently outperforms existing architectures in revenue. Regarding our loss modification, we confirm its effectiveness in controlling the revenue-regret trade-off by varying a single interpretable hyperparameter.

## 1 Introduction

Several decades ago, Myerson [36] proposed a general solution for the problem of optimal (revenue-maximizing) design of single-item auctions. However, despite the collaborative effort of the community, the results of the same generality for the multi-item auctions have not been obtained. Automated auction design [6, 7] offers an alternative to the analytical solutions by viewing auction design as a constrained optimization problem. In particular, the objective is to maximize the revenue while enforcing Incentive Compatibility (IC), which is vital for eliciting truthful preferences. This approach can provide approximate solutions in the settings where the optimal mechanisms are unknown, as well as serve as a tool that guides the research towards new discoveries. While the early approaches utilized linear programming and classic machine learning, after the renaissance of deep learning coupling auction design with neural networks was only a question of time.

The pioneers of this approach are Dütting et al. [12] with their RegretNet architecture. RegretNet is trained via differential optimization on a mixture of two objectives: to maximize revenue and to minimize regret, which is a relaxation of the IC constraint proposed in [11]. RegretNet recovers near-optimal revenue in the analytically solved settings and outperforms the previous state-of-the-art in more complex settings while having vanishing regret guarantees. The success of RegretNet opened up avenues for the new studies trying to extend or improve the algorithm in different dimensions. In line with this research, we propose two independent modifications of RegretNet.

First, we propose a new neural architecture for the optimal auction design based on the attention mechanism. We name it as RegretFormer after the widely-known Transformers. The illustration is provided in Figure 1. RegretFormer augments RegretNet with the expressivity of the attention mechanism while also having several crucial properties. On the one hand, it is by default insensitive to the order of items and participants, which is highly desirable for learning symmetric auctions. On

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

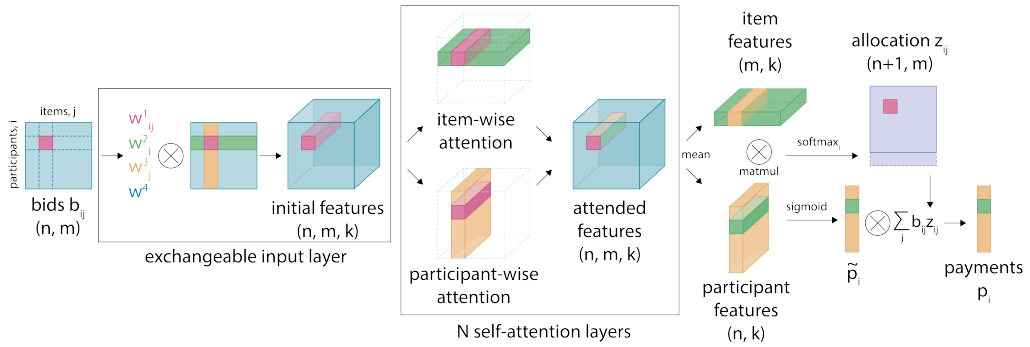


Figure 1: Our architecture RegretFormer. The detailed description is provided in Section 3.1.

the other hand, it does not assume a predetermined fixed input size, which makes it applicable to multiple settings at once, even beyond those provided during training.

Second, we propose an alternative objective for RegretNet. In contrast to the previous approaches based on optimizing a mixture of revenue and regret objectives, we maximize revenue given a fixed regret budget that is pre-specified by the designer. This is implemented by using method of Lagrange multipliers and dual gradient decent. As a result, we obtain an interpretable loss function that allows the designer to balance the revenue-regret trade-off by varying the acceptable regret budget. Furthermore, our approach requires less hyperparameters tuning to optimize the performance.

We conduct an extensive experimental study to verify the strengths of our modifications. Specifically, we compare our and existing architectures in the settings standard in the literature and additionally test out-of-setting generalization of our network. In all experiments, we find that RegretFormer consistently outperforms existing architectures in revenue given the same regret budget. This result holds for different input sizes and regret budgets. We additionally verify that this result is reliable with a novel validation procedure where we approximate the regret of RegretFormer given the misreports found by RegretNet. Regarding our loss modification, we confirm that specifying the regret budget results in the intended revenue-regret trades-off.

# 2 Background

## 2.1 Problem statement

An auction is defined as a set of $n$ bidders $N = \{1, ..., n\}$ and $m$ items $M = \{1, ..., m\}$. Each bidder $i$ selects a subset of items $S \subseteq M$ and evaluates it with valuation function $v_i : 2^M \to \mathbb{R}$. In our study, we consider the case of *additive* valuations, meaning that that for each item $j \in M$ a bidder has a valuation function $v_i(j)$, and a valuation of a subset of items $S \subseteq M$ equals to the sum of valuations of the items in the subset $v_i(S) = \Sigma_{j \in S} v_i(j)$.

The valuation function of a bidder $i$ is drawn independently from a distribution $F_i$ over all possible valuation functions $V_i$. A set of valuations of all bidders forms a valuation profile $v = (v_1, ..., v_n)$. An auctioneer does not know the reported bidder's valuation, however, he has a sample $S = v^1, ..., v^L$ of $L$ valuation profiles from the distribution $F = (F_1, ...F_n)$. The bidders report their valuation and the auction mechanism allocates the items among them charging a payment. An auction $(g, p)$ is denoted as a pair of allocation rules: $g_i : V \to [0, 1]^M$ and payment rules: $p_i : V \to \mathbb{R}_{\geq}$. The bidders propose bids $b = (b_1, ..., b_n) \in V$ and the mechanism computes allocations $g(b)$ and payments $p(b)$.

Based on bidder's valuation $v_i$ and given bids $b$ we compute the bidder's utility $u_i(v_i, b) = v_i(g_i(b)) - p_i(b)$. Define the valuation profile without the valuation $v_i$ as $v_{-i}$. Similarly, define the bids without $b_i$ as $b_{-i}$, and the valuation profile of all bidders except $i$ as $V_{-i} = \Pi_{k \neq i} V_k$. An auction is *dominant strategy incentive compatible* (DSIC) if the utility of each bidder is maximized by reporting their

true valuations regardless of the bids of other bidders: $\forall v, b : u_i(v_i, (v_i, b_{-i})) \geq u_i(v_i, (b_i, b_{-i}))$. Consequently, in DSIC auction the best strategy for each bidder is to report truthfully. An auction is *ex-post individually rational* (IR) if for all bidders $i \in N$, all valuations $v_i \in F$, and all bids $b_i \in V_i$, the individual utility is non-negative: $u_i(v_i, b_i) \geqslant 0$.

Define the revenue of the profile $v$ as $\Sigma_i p_i(v)$. The goal of the optimal auction design is to maximize the expected revenue, possibly subject to DSIC and IR constraints. The problem is analytically solved for the auctions with one item in the seminal work of Myerson [36], provided the valuation distributions are known. There are no general solutions for the auctions with $m > 1$.

The task of optimal auction design can be viewed as an optimization or a machine learning problem. In this case, the expected revenue takes the place of the objective. There is a class of auctions $(g(w), p(w)) \in M$ parameterized with $w \in \mathbb{R}^d$ for some $d \in \mathbb{N}$ and a set of bidders valuation profiles $S = v^1, ..., v^L$ drawn as independent and identically distributed variables from $F$. Then, the goal of the optimization procedure is to find an auction in $M$ that minimizes the negated expected revenue $-\mathbb{E}[\Sigma_{i \in N} p_i(v; w)]$ while satisfying DSIC and IR constraints.

### 2.2 RegretNet

RegretNet was proposed by Dütting et al. [12] as a deep learning solution for the optimal auction design. It can be used for multi-bidder and multi-item settings, the analytical solutions for which are unknown. The goal of RegretNet is to explicitly encode the allocation and payment rules of the auction mechanism. These rules are discovered as a result of an optimization procedure.

The RegretNet architecture consists of two parts: the allocation network and the payment network. Both networks take flattened fixed-sized bid matrix $B^{nm}$ as an input and process it through several fully-connected layers.

The allocation network maps the matrix of bids to the distribution of allocations between participants: $A_{net}(B^{nm}) = Z^{nm}$ ($A_{net} : \mathbb{R}^{nm} \to [0, 1]^{nm}$), where $z_{ij}$ is the probability of allocating the $j$-th item to the $i$-th bidder. Then, to obtain properly scaled allocation probabilities over participants for each item, the softmax activation is applied to the output of the final layer corresponding to each item. Specifically, $z_{ij} = e^{\tilde{z}_{ij}} / \sum_{i=1}^{n+1} e^{\tilde{z}_{ij}}$, where $\tilde{z}_{ij}$ denotes the unnormalized allocation probability.

The payment network maps the matrix of bids to the vector of pay values: $P_{net}(B^{nm}) = \hat{P}^n$ ($P_{net} : \mathbb{R}^{nm} \to [0, 1]^n$), where $\hat{p}_i$ is the fraction of the $i$-th bidder expected utility that the bidder transfers to the auctioneer. The payment is then computed as $p_i = \hat{p}_i \left( \sum_{j=1}^{m} z_{ij} b_{ij} \right)$ for $i = 1, \ldots, n$. To properly scale $\hat{p}_i \in [0, 1]$, a sigmoid activation is applied to the output of the final layer. Because the payment cannot exceed the bidder's expected utility, the mechanisms discovered by RegretNet always satisfy the IR constraint.

Given a sample of profiles $L$, RegretNet aims to optimize the following empirical objective:

$$\begin{aligned} \min_{w} \quad & -\frac{1}{|L|} \sum_{l \in L} \sum_{i \in N} p_i(v^l; w) \\ s.t. \quad & rgt_i(v^l; w) = 0, \ \ \forall i \in N, l \in L \end{aligned} \tag{1}$$

where $rgt_i$ denotes the $i$-th bidder's ex-post regret. Regret is a quantifiable relaxation of the DSIC constraint that was first introduced in [11]. It is defined as the difference between the $i$-th bidder's utilities given the optimal misreport (that provides the highest utility) and the truthful bid:

$$rgt_i(v^l; w) = \max_{v'_i} \left( u(v^l_i, (v'_i, v^l_{-i}); w) - u(v^l_i, v^l; w) \right) \tag{2}$$

To solve the constrained optimization problem 1 over the space of the network parameters $w$, the authors of RegretNet employ the augmented Lagrangian method. This yields a loss function that combines revenue optimization with a penalty for violation of DSIC constraints:

$$\mathcal{L}_{outer}(w) = \sum_{i \in N} \left[ -P_i + \lambda_i \widetilde{R}_i + \frac{\rho}{2} \widetilde{R}_i^2 \right] \tag{3}$$

where $B$ denotes mini-batch, $P_i = \frac{1}{|B|} \sum_{l \in B} p_i(v^l; w)$ is the average revenue from the $i$-th participant, $\widetilde{R}_i = \frac{1}{|B|} \sum_{l \in B} \widetilde{rgt}_i(v_i'^l, v^l; w)$ is the average approximate regret of the $i$-th participant, and $\widetilde{rgt}_i$ is an approximation of $rgt_i$ estimated via gradient descent in the inner optimization loop by maximizing the $i$-th participant ex-post utility as a function of their misreport:

$$\mathcal{L}_{inner}(v_i'^l) = -u(v_i^l, (v_i'^l, v_{-i}^l); w) \tag{4}$$

The Lagrange multipliers $\lambda_i$ and $\rho$ are periodically updated throughout the training according to the schedules: $\lambda_i \leftarrow \lambda_i + \rho \widetilde{R}_i$ and $\rho \leftarrow \rho + \rho_\Delta$. The starting values of $\lambda_i$ and $\rho$ and the learning rate $\rho_\Delta$ are hyperparameters that control the trade-off between revenue and regret of the learned mechanism.

### 2.3 Permutation-equivariant architecture

EquivariantNet was proposed by Rahme et al. [43] to effectively deal with multi-bidder symmetric auctions. The architecture relies on a theorem originally proven by [8] that there exists an optimal solution for a symmetric auction that is permutation-equivariant.

Like RegretNet [12], EquivariantNet has an allocation network and a payment network. Both networks consist of compositions of exchangeable layers that preserve permutation-equivariance [20]. The exchangeable layer adds to each element $i, j$ of a matrix an aggregated statistic (like sum) over the row $i$, the column $j$, and the whole matrix, weighted by learnable parameters that are shared among all matrix elements. The precise definition is provided in the Appendix. The EquiavariantNet is the first architecture that can generalize to unseen settings but its use is limited. The authors observe that it produces competitive results but does not outperform RegretNet.

### 2.4 Related work

Early work that explores the automated solutions to auction design formulates the problem as linear program [6, 46, 7] or searches within specific families of DSIC auctions [29, 47]. The former approach suffers from scalability issues due to the curse of dimensionality [19], while the latter approach searches within a limited family of auctions that might not contain the optimal mechanism. Classic machine learning has also been applied to the auction design [28, 11, 38], but these approaches are considered less flexible and general than the deep learning alternatives [12].

In recent years, multiple extensions of RegretNet have been proposed, including the extensions to the optimal auction design with budget constraints [13], fairness constraints [27], and human preferences over desirable allocations [42], as well as the problem of faculty allocation [14] and the matching problem [45]. Shen et al. [48] and Dütting et al. [12] propose revenue-maximizing alternatives to RegretNet that are exactly DSIC but that are only applicable to the auctions with one bidder. Rahme et al. [44] propose a simplified loss function for RegretNet that is easier to tune. Deep learning has been applied to other aspects of mechanism design, such as iterative combinatorial auctions [53], minimizing economic burden on bidders within the Groves family of auctions [50], optimal bidding strategies [40, 39], optimal redistribution mechanisms [32], and E-commerce advertising [30]. A plethora of research has also focused on the questions of sample complexity of designing revenue-maximizing auctions [5, 35, 10, 2, 34, 33, 37, 49, 15, 23, 21, 18, 16].

## 3 Our modifications of RegretNet

In this section, we describe our two independent modifications of RegretNet. The first modification is a novel neural architecture for the optimal auction design, which we denote as RegretFormer. The

second modification is an alternative objective formulation that yields an interpretable loss function which is less sensitive to hyperparameters.

### 3.1 RegretFormer: enhancing RegretNet with the power of attention

The neural architecture of RegretNet has several issues. One issue is the sensitivity of RegretNet to the order of items and participants in the input bid matrix. Such order should not affect the outcome of a symmetric auction, and even in non-symmetric auctions, symmetric solutions can achieve near-optimal revenue [22, 1, 24, 25, 26]. This makes the option to wire equivariance to input order into the architecture desirable. Another issue is that RegretNet assumes a fixed number of participants and items. This severely limits its practical applicability due to inability to generalize to unseen input sizes, as well as limited capability to learn from data with heterogenous input size. Finally, the fully-connected layers used in RegretNet have limited expressivity and may not have the right inductive biases for auction design. Taking these issues into account, we propose *RegretFormer*.

The architecture of RegretFormer is illustrated in Figure 1. The network takes the matrix of bids $B^{nm}$ as the input. First, we apply an exchangeable layer [20] to transform each bid into an initial vector of features that contains information about other bids. Then, we apply item-wise (i.e. row-wise) and participant-wise (i.e. column-wise) self-attention layers [51] to each feature vector corresponding to each bid. For a given bid, the outputs of the two self-attention layers are transformed into a single feature vector through a fully-connected layer. These self-attention transformations can be applied several times. Finally, by averaging over rows and columns we transform the output of self-attention layers into the allocation matrix $Z^{nm}$ and the payment vector $P^n$. Note that each layer shares the parameters across all bids to ensure that the network is insensitive to the order of bids and is applicable to different input sizes. The detailed definitions and order of all layers are provided in the Appendix. Ablations of our architecture are also investigated in the Appendix.

Our architecture has several advantages. On the one hand, it leverages the expressivity of attention layers that help to achieve the state-of-the-art performance in many diverse and complex problems, e.g. [51, 9, 54]. On the other hand, it maintains the equivariance and the invariance to the order of items and participants. The former forces the resulting mechanisms to be symmetric, which drastically reduces the solution search space. The latter allows RegretFormer to learn from data with varied input sizes, as well as to generalize to unseen settings.

### 3.2 Interpretable objective through specifying regret budget

Both RegretNet and EquivariantNet optimize a mixture of two conflicting objectives (3), namely revenue maximization and regret minimization, and control their trade-off with hyperparameters like initial values and schedules of the Lagrange multipliers. There are two issues with this approach. The first issue is sensitivity. These hyperparameters have to be precisely tuned for each experiment, and as Rahme et al. [44] show, may massively degrade the performance if selected improperly. The second issue is uninterpretability. While increasing any of $\lambda_i, \rho, \rho_\Delta$ tightens the regret budget of the learned mechanism, the exact effect of such changes on resulting revenue and regret is unpredictable. Furthermore, the recipe for hyperparameters selection in new settings is unclear.

We propose an alternative perspective that mitigates both issues. Instead of balancing the two objectives, we maximize the revenue given a maximal regret budget $R_{max}$, which is pre-specified by the designer. This essentially corresponds to a relaxed version of the constrained objective (1):

$$\begin{aligned} \min_{w} \quad & -\frac{1}{|L|}\sum_{l\in L}\sum_{i\in N} p_i(v^l;w) \\ s.t. \quad & \frac{1}{|L|}\sum_{l\in L}\sum_{i\in N} rgt_i(v^l;w) \leq R_{max} \end{aligned} \tag{5}$$

To optimize the constrained objective (5), we introduce the Lagrange multiplier $\gamma$:

$$\mathcal{L}_{outer}(w) = -\sum_{i \in N} P_i + \gamma \left( \sum_{i \in N} \widetilde{R}_i - R_{max} \right) \tag{6}$$

Critically, unlike the original RegretNet we do not hand-select the Lagrange multiplier. Instead, we adaptively update $\gamma$ to enforce the constraint satisfaction while exhausting all available regret budget $R_{max}$. To this end, we employ dual gradient descent [3]. Specifically, we iterate between one gradient update of the network parameters $w$ to minimize (6) and one update of $\gamma$ according to:

$$\gamma \leftarrow \max \left( 0, \gamma + \gamma_{\Delta} \left( \log(\sum_{i \in N} \widetilde{R}_i / \sum_{i \in N} p_i) - \log(R_{max}) \right) \right) \tag{7}$$

where $\gamma_{\Delta}$ denotes the learning rate for the dual variable. Notice that in the dual update (7) we normalize regret by revenue. This way, $R_{max} \in [0, 1]$ specifies the regret budget in percentage from revenue. Also notice that we apply logarithms to the last two terms for faster convergence. We also apply a decreasing schedule to $R_{max}$, the details of which are reported in the Appendix. Another example of applying dual gradient descent to enforce a constraint can be found in [41].

The proposed approach has several advantages. On the one hand, it resolves the dichotomy of two conflicting objectives. While the regret budget needs to be explicitly set based on the designer's preferences, all other hyperparameters are then straightforward to tune by maximizing the revenue given the specified regret budget. This is unlike the existing approach (3) where the regret budget is chosen implicitly through specifying a combination of uninterpretable hyperparameters like $\lambda_i$, $\rho$, $\rho_{\Delta}$, while the same hyperparameters are also crucial for the performance. On the other hand, our approach is also significantly less sensitive to the hyperparameters compared to the original objective of RegretNet (3) and we use the same hyperparameters in all experiments.

# 4 Experiments

In this section, we empirically investigate our modifications of RegretNet. We compare RegretFormer with existing architectures in several settings with fixed number of participants and items. These settings were used in the RegretNet paper and are standard in the subsequent the literature. In each setting, we vary regret budgets and further investigate the differences between architectures using a novel validation procedure. The training details and the hyperparameters are reported in the Appendix. We also report additional experiments in the Appendix where we apply RegretFormer to settings with varied input size. In all settings, the valuations of all participants are additive and are independently drawn for each item from the Uniform distribution: $v_i(j) \in \mathbb{U}[0, 1]$. The settings only differ in the number of participants $n$ and items $m$, so we denote the settings as nxm. We conduct experiments in five settings: 1x2, 2x2, 2x3, 2x5, and 3x10. The 1x2 setting is the celebrated Manelli-Vincent auction, the analytical solution for which is provided in [31]. The optimal revenue for this auction equals $0.55$. For the rest of the settings, the analytical solutions are unknown.

## 4.1 Comparing architectures and varying regret budgets

We compare our and existing neural architectures in the five settings given three different regret budgets $R_{max} \in \{10^{-3}, 10^{-4}, 10^{-5}\}$. We additionally include the classic VCG [52, 4, 17] and Myerson [36] mechanisms for comparison, both of which are DSIC and ex-post IR. The Myerson auctions are optimal for $m = 1$. We report its two extensions to multi-good settings: the 'item-wise' when each item is sold separately and the 'bundled' when all items are sold in a single bundle.

The resulting revenue and regret values are reported in Table 1. Additionally, we report the learning curves for all settings in the Appendix. Below we summarize our findings.

In all settings and given all regret budgets, RegretFormer outperforms both RegretNet and EquivariantNet by revenue while achieving similar regret values. The performance gap but becomes especially

Table 1: Architecture comparison. The highest revenue in a setting is highlighted with bold.

| $R_{max}$ | setting | RegretNet | | EquivariantNet | | RegretFormer | |
|---|---|---|---|---|---|---|---|
| | | revenue | regret | revenue | regret | revenue | regret |
| $10^{-3}$ | 1x2 | 0.579 | 0.00123 | 0.576 | 0.0009 | **0.583** | 0.00099 |
| | 2x2 | 0.888 | 0.00066 | 0.883 | 0.00069 | **0.914** | 0.00065 |
| | 2x3 | 1.352 | 0.00102 | 1.404 | 0.00135 | **1.411** | 0.00091 |
| | 2x5 | 2.355 | 0.00205 | 2.428 | 0.002 | **2.471** | 0.00181 |
| | 3x10 | 5.746 | 0.00292 | 6.057 | 0.00305 | **6.157** | 0.00332 |
| $10^{-4}$ | 1x2 | 0.55 | 0.00017 | **0.555** | 0.00018 | **0.555** | 0.00046 |
| | 2x2 | 0.849 | 0.00009 | 0.832 | 0.0001 | **0.871** | 0.0001 |
| | 2x3 | 1.265 | 0.00012 | 1.269 | 0.00023 | **1.32** | 0.00013 |
| | 2x5 | 2.235 | 0.00034 | 2.285 | 0.00035 | **2.346** | 0.00034 |
| | 3x10 | 5.016 | 0.00062 | 5.624 | 0.00089 | **5.89** | 0.00068 |
| $10^{-5}$ | 1x2 | 0.531 | 0.00001 | 0.493 | 0.00005 | **0.537** | 0.00003 |
| | 2x2 | 0.775 | 0.00004 | 0.713 | 0.00002 | **0.803** | 0.00001 |
| | 2x3 | 1.165 | 0.00006 | 1.076 | 0.00004 | **1.2** | 0.00002 |
| | 2x5 | 1.856 | 0.00014 | 1.895 | 0.0001 | **2.105** | 0.00006 |
| | 3x10 | 4.047 | 0.00026 | **5.071** | 0.0003 | 5.019 | 0.00013 |

| $R_{max}$ | setting | VCG | | Myerson item-wise | | Myerson bundled | |
|---|---|---|---|---|---|---|---|
| | | revenue | regret | revenue | regret | revenue | regret |
| 0 | 1x2 | 0 | 0 | 0.5 | 0 | 0.544 | 0 |
| | 2x2 | 0.667 | 0 | 0.833 | 0 | 0.839 | 0 |
| | 2x3 | 1 | 0 | 1.25 | 0 | 1.278 | 0 |
| | 2x5 | 1.667 | 0 | 2.083 | 0 | 2.188 | 0 |
| | 3x10 | 5 | 0 | 5.312 | 0 | 5.003 | 0 |

prominent in the larger settings. While the permutation-equivariance property of RegretFormer likely plays a role in these results, it cannot fully explain the results since EquivarintNet also has this property. We therefore attribute the success of RegretFormer to the expressivity of attention layers.

In the bigger settings, EquivariantNet can also outperform RegretNet. This result is somewhat surprising since it was not observed by the authors of EquivariantNet, but it can simply be explained by the fact that they do not test the architecture in the settings that are complex enough. Another interesting result is that given low regret budgets EquivariantNet can underperform RegretNet.

Varying the regret budget within the same setting affects the revenue-regret trade-off as expected: decreasing $R_{max}$ from $10^{-3}$ to $10^{-4}$ and $10^{-5}$ respectively lowers the resulting regret by approximately one and two magnitudes. Furthermore, for each specific setting and $R_{max}$ we expect by design the revenue to be higher than regret by a factor of $R_{max}$ (see Section 3.2). Often this approximately holds, but sometimes the ratio can be several times higher or lower than expected. The reason is the difference of regret computation during training and validation. Following the procedure of Dütting et al. [12], we compute 25 inner optimization steps per each outer update during training and a much higher value of 1000 steps during validation. The latter values are reported in the paper and they can both underestimate and overestimate regret compared to training due to overfitting during validation or underfitting during training, respectively. For a tighter regret approximation and a better match of the revenue/regret ratio to $R_{max}$, one could tune the number of inner optimization steps used during training, provided that a longer training time is acceptable.

Given the smallest regret budget, RegretNet and EquivariantNet considerably underperform both item-wise and bundled Myerson auctions. While RegretFormer performs on par with Myerson in such conditions, its regret is still non-zero. This result highlights a potential problem of the regret-based approach to the auction design: when any violations of DSIC are highly undesirable, a better revenue might be achieved with classic mechanisms that are also guaranteed to be DSIC. It should be noted

Table 2: Cross-misreport regret estimates. The highest regret for a network is highlighted with bold.

| setting | misreports of | regret of RegretNet | regret of EquivariantNet | regret of RegretFormer |
|---|---|---|---|---|
| 1x2 | RegretNet | 0.00112 | 0.00063 | 0.00109 |
| | EquivariantNet | **0.00126** | **0.00105** | **0.00160** |
| | RegretFormer | 0.00104 | 0.00073 | 0.00136 |
| 2x2 | RegretNet | **0.00072** | 0.00049 | 0.00042 |
| | EquivariantNet | 0.00034 | **0.00073** | 0.00032 |
| | RegretFormer | 0.00064 | 0.00064 | **0.00074** |
| 2x3 | RegretNet | **0.00102** | 0.00042 | 0.00040 |
| | EquivariantNet | 0.00093 | **0.00146** | 0.00089 |
| | RegretFormer | 0.00096 | 0.00100 | **0.00111** |
| 2x5 | RegretNet | **0.00201** | 0.00070 | 0.00060 |
| | EquivariantNet | 0.00133 | **0.00204** | 0.00117 |
| | RegretFormer | 0.00160 | 0.00163 | **0.00210** |
| 3x10 | RegretNet | **0.00282** | 0.00039 | 0.00025 |
| | EquivariantNet | 0.00158 | **0.00381** | 0.00113 |
| | RegretFormer | 0.00266 | 0.00342 | **0.00473** |

that the Myerson mechanism relies on the access to the true valuation distribution and might not be applicable to arbitrary nonparametric distributions. Furthermore, RegretFormer still reliably outperforms VCG in revenue, which is more frequently used in practice than Myerson.

### 4.2 Is the performance gap genuine?

In the previous subsection, we observed that RegretFormer outperforms both RegretNet and EquivariantNet and attribute its success to the permutation-equivariance and the expressivity of attention layers. However, there is an alternative explanation.

Recall that the regret is approximated in the inner optimization (4) by repeatedly back-propagating through the whole neural network $w$. Therefore, the reported regret values explicitly depend on the neural architecture, including the number, the types, the size, and the order of layers. Due to the absence of a regret approximation procedure that is identical for all networks, there is no guarantee that similar regret estimates between different neural architectures correspond to actual similar regret values, i.e. for different architectures $w_1, w_2 : \tilde{R}(w_1) \approx \tilde{R}(w_2) \not\Rightarrow R(w_1) \approx R(w_2)$, where $R(w)$ is the expected regret and $\tilde{R}(w)$ is its approximation. In particular, we are concerned whether RegretFormer achieves higher revenues due to approximating the regret worse, rather than due to maximizing the revenue better. We have designed two procedures to test this hypothesis.

The first procedure denoted as ‘cross-misreports‘ is based on estimating the regret of one network on the optimal misreports approximated by another network: $\widetilde{R}_i^{cross}(w_1, w_2) = \frac{1}{|B|}\sum_{l\in B} \widetilde{rgt}_i(v_i'^l(w_2), v^l; w_1)$. We apply this procedure to the trained networks in all five settings given $R_{max} = 10^{-3}$. If the networks approximate regret equally well, we expect the regret estimated on cross-misreports to not exceed the normally computed regret. However, if the regret estimates of RegretFormer are higher on the misreports estimated by RegretNet or EquivariantNet, this would point towards poor regret approximation by RegretFormer.

We report the results in Table 2. We do not find any evidence that RegretFormer underestimates the regret. In all settings, it finds misreports that produce either higher or similar regret compared to the misreports of the other two architectures. This is also the case for RegretNet and EquivariantNet.

The second procedure denoted as ‘distillation‘ is based on training a ‘student‘ network to approximate the predictions of a trained ‘teacher‘ network. The results are similar to the cross-misreport validation and are reported in the Appendix.

Table 3: Out-of-setting generalization

| Training setting | Validation setting | EquivariantNet revenue | EquivariantNet regret | RegretFormer revenue | RegretFormer regret |
|---|---|---|---|---|---|
| 1x2 | 2x2 | 0.690 | 0.04403 | 0.669 | 0.03776 |
| | 2x3 | 1.084 | 0.07074 | 1.056 | 0.07167 |
| | 2x5 | 1.917 | 0.12465 | 1.863 | 0.13151 |
| | 3x10 | 4.308 | 0.29291 | 4.197 | 0.29106 |
| 2x2 | 1x2 | 0.695 | 0.13343 | 0.768 | 0.21671 |
| | 2x3 | 1.350 | 0.00071 | 1.412 | 0.01221 |
| | 2x5 | 2.307 | 0.03169 | 2.402 | 0.03250 |
| | 3x10 | 5.156 | 0.55869 | 4.943 | 0.32002 |
| 2x3 | 1x2 | 0.686 | 0.14900 | 0.775 | 0.20051 |
| | 2x2 | 0.875 | 0.00116 | 0.904 | 0.00137 |
| | 2x5 | 2.318 | 0.00615 | 2.432 | 0.00938 |
| | 3x10 | 5.271 | 0.37967 | 4.929 | 0.27183 |
| 2x5 | 1x2 | 0.743 | 0.19830 | 0.816 | 0.23945 |
| | 2x2 | 0.900 | 0.00066 | 0.903 | 0.00072 |
| | 2x3 | 1.401 | 0.00103 | 1.415 | 0.00103 |
| | 3x10 | 5.517 | 0.24757 | 4.884 | 0.28009 |
| 3x10 | 1x2 | 0.552 | 0.53700 | 0.801 | 0.18780 |
| | 2x2 | 0.693 | 0.19767 | 1.053 | 0.05893 |
| | 2x3 | 1.099 | 0.33665 | 1.611 | 0.07595 |
| | 2x5 | 1.936 | 0.59959 | 2.754 | 0.10181 |

### 4.3 Out-of-setting generalization

In these experiments, we investigate how well the architectures generalize to unseen settings. It's clear that RegretNet cannot be applied out-of-domain since its layers rely on fixed input size, so we compare our network with EquivariantNet. The networks are trained in five settings 1x2, 2x2, 2x3, 2x5, 3x10 and then tested in all settings but the setting used for training.

The results are reported in Table 3. Both networks look promising when the number of objects varies and the number of bidders remains constant. However, generalization to the settings where the number of bidders varies is poor for both networks due to complex interactions between the participants. Similar results were observed by Rahme et al. [43].

## 5 Conclusion

In this study, we reach the new state-of-the-art performance in the application of deep learning to optimal auction design. Our RegretFormer leverages the recent advances in deep learning to unlock the full potential of the regret-based optimization while enforcing the equivariance and the invariance to the order of participants and items. We extensively test the effectiveness of RegretFormer in multiple experimental settings and find that our network consistently outperforms the existing analogues in revenue given the same regret budgets. In addition, we rethink the objective formulation of RegretNet. The resulting loss function allows us to disentangle balancing the revenue-regret trade-off and optimizing the performance. At the same time, it reduces the burden of hyperparameter tuning and simplifies the application of the networks to new auctions. Finally, we suggest a novel validation procedure based on cross-misreport estimation that may find its use in future studies. As future work, our architecture could be applied to asymmetric settings by using positional encoding.

# References


[1] S. Alaei, J. Hartline, R. Niazadeh, E. Pountourakis, and Y. Yuan. Optimal auctions vs. anonymous pricing. *Games and Economic Behavior*, 118:494–510, 2019.

[2] M.-F. F. Balcan, T. Sandholm, and E. Vitercik. Sample complexity of automated mechanism design. *Advances in Neural Information Processing Systems*, 29, 2016.

[3] S. Boyd, S. P. Boyd, and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[4] E. H. Clarke. Multipart pricing of public goods. *Public choice*, pages 17–33, 1971.

[5] R. Cole and T. Roughgarden. The sample complexity of revenue maximization. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 243–252, 2014.

[6] V. Conitzer and T. Sandholm. Complexity of mechanism design. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, pages 103–110, 2002.

[7] V. Conitzer and T. Sandholm. Self-interested automated mechanism design and implications for optimal combinatorial auctions. In *Proceedings of the 5th ACM Conference on Electronic Commerce*, pages 132–141, 2004.

[8] C. Daskalakis and S. M. Weinberg. Symmetries and optimal multi-dimensional mechanism design. In *Proceedings of the 13th ACM conference on Electronic commerce*, pages 370–387, 2012.

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, jun 2019.

[10] P. Dhangwatnotai, T. Roughgarden, and Q. Yan. Revenue maximization with a single sample. *Games and Economic Behavior*, 91:318–333, 2015.

[11] P. Dütting, F. Fischer, P. Jirapinyo, J. K. Lai, B. Lubin, and D. C. Parkes. Payment rules through discriminant-based classifiers. *ACM Transactions on Economics and Computation*, 3(1), 2015.

[12] P. Dütting, Z. Feng, H. Narasimhan, D. Parkes, and S. S. Ravindranath. Optimal auctions through deep learning. In *International Conference on Machine Learning*, pages 1706–1715. PMLR, 2019.

[13] Z. Feng, H. Narasimhan, and D. C. Parkes. Deep learning for revenue-optimal auctions with budgets. In *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems*, pages 354–362, 2018.

[14] N. Golowich, H. Narasimhan, and D. C. Parkes. Deep learning for multi-facility location mechanism design. In *IJCAI*, pages 261–267, 2018.

[15] Y. A. Gonczarowski and N. Nisan. Efficient empirical revenue maximization in single-parameter auction environments. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 856–868, 2017.

[16] Y. A. Gonczarowski and S. M. Weinberg. The sample complexity of up-to-$\varepsilon$ multi-dimensional revenue maximization. *Journal of the ACM (JACM)*, 68(3):1–28, 2021.

[17] T. Groves. Incentives in teams. *Econometrica: Journal of the Econometric Society*, pages 617–631, 1973.

[18] C. Guo, Z. Huang, and X. Zhang. Settling the sample complexity of single-parameter revenue maximization. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 662–673, 2019.

[19] M. Guo and V. Conitzer. Computationally feasible automated mechanism design: General approach and case studies. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 24, pages 1676–1679, 2010.

[20] J. Hartford, D. Graham, K. Leyton-Brown, and S. Ravanbakhsh. Deep models of interactions across sets. In *International Conference on Machine Learning*, pages 1909–1918. PMLR, 2018.

[21] J. Hartline and S. Taggart. Sample complexity for non-truthful mechanisms. In *Proceedings of the 2019 ACM Conference on Economics and Computation*, pages 399–416, 2019.

[22] J. D. Hartline and T. Roughgarden. Simple versus optimal mechanisms. In *Proceedings of the 10th ACM conference on Electronic commerce*, pages 225–234, 2009.

[23] Z. Huang, Y. Mansour, and T. Roughgarden. Making the most of your samples. *SIAM Journal on Computing*, 47(3):651–674, 2018.

[24] Y. Jin, P. Lu, Q. Qi, Z. G. Tang, and T. Xiao. Tight approximation ratio of anonymous pricing. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 674–685, 2019.

[25] Y. Jin, P. Lu, Z. G. Tang, and T. Xiao. Tight revenue gaps among simple mechanisms. *SIAM Journal on Computing*, 49(5):927–958, 2020.

[26] P. Kothari, S. Singla, D. Mohan, A. Schvartzman, and S. M. Weinberg. Approximation schemes for a unit-demand buyer with independent items via symmetries. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 220–232. IEEE, 2019.

[27] K. Kuo, A. Ostuni, E. Horishny, M. J. Curry, S. Dooley, P.-y. Chiang, T. Goldstein, and J. P. Dickerson. Proportionnet: Balancing fairness and revenue for auction design with deep learning. *arXiv preprint arXiv:2010.06398*, 2020.

[28] S. Lahaie. A kernel-based iterative combinatorial auction. In *Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.

[29] A. Likhodedov, T. Sandholm, et al. Approximating revenue-maximizing combinatorial auctions. In *AAAI*, volume 5, pages 267–274, 2005.

[30] X. Liu, C. Yu, Z. Zhang, Z. Zheng, Y. Rong, H. Lv, D. Huo, Y. Wang, D. Chen, J. Xu, et al. Neural auction: End-to-end learning of auction mechanisms for e-commerce advertising. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 3354–3364, 2021.

[31] A. M. Manelli and D. R. Vincent. Bundling as an optimal selling mechanism for a multiple-good monopolist. *Journal of Economic Theory*, 127(1):1–35, 2006.

[32] P. Manisha, C. Jawahar, and S. Gujar. Learning optimal redistribution mechanisms through neural networks. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 345–353, 2018.

[33] M. Mohri and A. M. Medina. Learning algorithms for second-price auctions with reserve. *The Journal of Machine Learning Research*, 17(1):2632–2656, 2016.

[34] J. Morgenstern and T. Roughgarden. Learning simple auctions. In *Conference on Learning Theory*, pages 1298–1318. PMLR, 2016.

[35] J. H. Morgenstern and T. Roughgarden. On the pseudo-dimension of nearly optimal auctions. *Advances in Neural Information Processing Systems*, 28, 2015.

[36] R. B. Myerson. Optimal auction design. *Mathematics of operations research*, 6(1):58–73, 1981.

[37] H. Narasimhan and D. C. Parkes. A general statistical framework for designing strategy-proof assignment mechanisms. In *UAI'16 Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, 2016.

[38] H. Narasimhan, S. B. Agarwal, and D. C. Parkes. Automated mechanism design without money via machine learning. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, 2016.

[39] T. Nedelec, N. El Karoui, and V. Perchet. Learning to bid in revenue-maximizing auctions. In *International Conference on Machine Learning*, pages 4781–4789. PMLR, 2019.

[40] T. Nedelec, J. Baudet, V. Perchet, and N. El Karoui. Adversarial learning for revenue-maximizing auctions. In *20th International Conference on Autonomous Agents and Multiagent Systems*, 2021.

[41] X. B. Peng, A. Kanazawa, S. Toyer, P. Abbeel, and S. Levine. Variational discriminator bottleneck: Improving imitation learning, inverse rl, and gans by constraining information flow. In *International Conference on Learning Representations*, 2018.

[42] N. Peri, M. Curry, S. Dooley, and J. Dickerson. Preferencenet: Encoding human preferences in auction design with deep learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[43] J. Rahme, S. Jelassi, J. Bruna, and S. M. Weinberg. A permutation-equivariant neural network architecture for auction design. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(6):5664–5672, May 2021. URL `https://ojs.aaai.org/index.php/AAAI/article/view/16711`.

[44] J. Rahme, S. Jelassi, and S. M. Weinberg. Auction learning as a two-player game. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=YHdeAO61l6T`.

[45] S. S. Ravindranath, Z. Feng, S. Li, J. Ma, S. D. Kominers, and D. C. Parkes. Deep learning for two-sided matching. *arXiv preprint arXiv:2107.03427*, 2021.

[46] T. Sandholm. Automated mechanism design: A new application area for search algorithms. In *International Conference on Principles and Practice of Constraint Programming*, pages 19–36. Springer, 2003.

[47] T. Sandholm and A. Likhodedov. Automated design of revenue-maximizing combinatorial auctions. *Operations Research*, 63(5):1000–1025, 2015.

[48] W. Shen, P. Tang, and S. Zuo. Automated mechanism design via neural networks. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 215–223, 2019.

[49] V. Syrgkanis. A sample complexity measure with applications to learning optimal auctions. *Advances in Neural Information Processing Systems*, 30, 2017.

[50] A. Tacchetti, D. Strouse, M. Garnelo, T. Graepel, and Y. Bachrach. A neural architecture for designing truthful and efficient auctions. *CoRR*, abs/1907.05181, 2019. URL `http://arxiv.org/abs/1907.05181`.

[51] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[52] W. Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of finance*, 16(1):8–37, 1961.

[53] J. Weissteiner and S. Seuken. Deep learning—powered iterative combinatorial auctions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2284–2293, 2020.

[54] L. Yuan, Y. Chen, T. Wang, W. Yu, Y. Shi, Z.-H. Jiang, F. E. Tay, J. Feng, and S. Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 558–567, 2021.

## Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] **See the end of Section 4.1.**
   (c) Did you discuss any potential negative societal impacts of your work? [No] **We do not foresee such impact.**
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] **We include the code in the supplemental material.**
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] **See the Appendix.**
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] **In the Appendix, we report the learning curves with the error bars.**
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] **See the Appendix.**