Extending Myerson's Optimal Auctions to Correlated Bidders via Neural Network Interpolation

Mingyu Guo[1] **Jiayuan Liu**[2] **Vincent Conitzer**[2]
[1]Adelaide University, Adelaide, Australia
[2]Carnegie Mellon University, Pittsburgh, USA
mingyu.guo@adelaide.edu.au
jiayuan4@andrew.cmu.edu, conitzer@cs.cmu.edu

Abstract

We aim to design revenue-maximizing single-item auctions that are deterministic, strategy-proof, and ex post individually rational — the quintessential fundamental model for optimal mechanism design. While Myerson's seminal work solved this model for independent bidders, straightforward extensions to correlated settings often result in non-monotonic, and therefore non-strategy-proof allocations. Critically, optimal allocation design for correlated bidders is NP-hard to approximate, rendering theoretical optimality guarantees in this domain computationally intractable.

We establish a new standard of empirical rigor for this theoretically hard model by proposing an empirical pipeline. We synthesize neural interpolation of Myerson's greedy allocation based on marginal profits (the correlated variant of virtual valuation), formal neural verification to enforce exact strategy-proofness, and a constructive repair procedure.

Empirically, our method is highly effective, consistently achieves near-optimal revenue across a wide range of distributions, including synthetic, adversarial, and real-world (consumer, financial, and industrial) datasets. Compared to existing manual and neural baselines, our approach shows substantial improvement, often reducing the revenue gap by an order of magnitude. Our study is also the first to introduce (unattainable) greedy revenue as a rigorous upper bound for empirical benchmarking, providing a definitive quantitative performance measure where true optima remain theoretically elusive.

We demonstrate the generality of our approach by extending it to multi-unit auctions with unit demand and integrating our verification techniques into RegretNet to achieve exact strategy-proofness.

1 Introduction

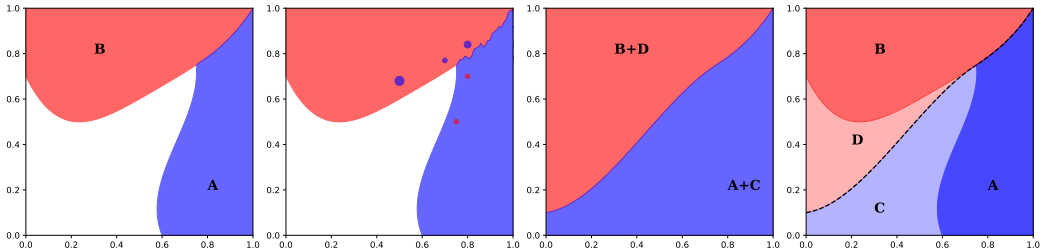
We address the design of revenue-maximizing single-item auctions – the canonical model that serves as the foundation of mechanism design. In his landmark work Myerson [1981], Myerson demonstrated that for independent bidders, an allocation rule is strategy-proof if and only if it is *monotone*. By greedily allocating based on *(ironed) virtual valuations*, one naturally satisfies monotonicity, yielding a deterministic and ex-post individually rational optimal auction.

However, this theoretical elegance is the primary bottleneck for practitioners: **correlation is prevalent in real-life auctions.** When one naturally extends Myerson's technique to correlated settings by greedily allocating based on *marginal profits* (the correlated variant of virtual valuations), the resulting allocations are frequently non-monotone. Theoretical attempts to find an exact characterization for this setting have hit a mathematical wall; **designing an optimal allocation for correlated bidders**

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

is proven NP-hard to approximate Papadimitriou and Pierrakos [2011], Caragiannis et al. [2016], rendering global optimality guarantees computationally intractable.

Recognizing this intractability, our work shifts the focus from fighting NP-hardness to empirical and computational rigor. We argue that for the correlated setting – the most natural extension of Myerson's foundational optimal auction model – the goal should be **computationally designed mechanisms that are verifiably strategy-proof and near-optimal**. We propose a new approach by synthesizing a pipeline of **neural interpolation** of Myerson's greedy allocation, **formal strategy-proofness verification**, and **repair**. This framework effectively provides a computational resolution to this foundational model – a cornerstone of mechanism design – for practical implementation in real-life auctions where theoretical optima are unavailable. Our results demonstrate that we can recover a perfectly strategy-proof and near-optimal mechanism from Myerson's greedy allocation even when the underlying target is mathematically "broken" (i.e., non-monotone). We summarize our contributions as follows:



(a) monotone allocation (b) greedy allocation (c) interpolation (d) revenue fix

Figure 1: Illustration of a 2-bidder setting where axes represent two bids in $[0, 1]$. Blue and red regions denote wins for bidder 1 and 2, respectively. (a) monotone allocation: A strategy-proof allocation where a winner necessarily remains the winner after a unilateral bid increase. (b) greedy allocation (based on marginal profits): Non-monotone allocation that gives the revenue upper bound — red and blue points enclosed in opposing regions represent monotonicity violations that preclude strategy-proofness. (c) (monotone) interpolation: A tiny neural network mimics the macro-trends of (b) while "glossing over" fine-grained violations due to its limited expressive capacity. This interpolation is formally verified to be monotone, thereby ensuring exact strategy-proofness. (d) revenue fix (leading to final allocation): The result after applying the revenue fix in Section 3, which determines the exact shape of the "no-sale" (white) region to maximize expected revenue without altering the strategy-proofness established in the previous stage.

Contribution 1: Neural network interpolation as a verification tool for strategy-proofness.

Given a specific correlated distribution, a natural first step is to check whether Myerson's greedy allocation remains monotone *despite the correlation*. Note that it is infeasible to enumerate all bid profiles to verify monotonicity. We train a neural network to interpolate Myerson's greedy allocation and then formally verify its monotonicity.[1]

We use a multilayer perceptron (MLP) with ReLU activation to model the allocation function. This architecture allows us to *exactly verify* the monotonicity of a trained allocation using mixed-integer programming (MIP). For training (i.e., the neural interpolation step), training samples are bid profiles drawn from the target distribution. We instruct the network to allocate to the highest marginal profit bidder.

Notably, we observe that for many correlated distributions, Myerson's greedy allocation is **not** monotone, but the supervision result is a verifiably monotone interpolation. While the greedy allocation is rarely monotone, the deviations are often minimal.[2] The fact that a non-monotone target

[1] Although there are neural network architectures that ensure "monotonicity", such as the *min-max networks* Sill [1997], it is important to note that the notion of "monotonicity" represented by min-max networks is different from the concept of "allocation monotonicity" in mechanism design, despite the name similarity. We defer the detailed discussion on this difference to Appendix H.

[2] For 2-bidder cases where it is scalable to run the classical MIP-based automated mechanism design (AMD) Conitzer and Sandholm [2002], we observe that greedy and AMD (strategy-proof therefore monotone) generally coincide, differing in only a small percentage of bid profiles, more in Appendix E.1.

yields a monotone interpolation is a consequence of our network architecture as a **"detail-filtering" regularizer**. Our MIP-based verification necessitates the use of relatively small networks. These "tiny" networks lack the expressive capacity to capture the fine-grained monotonicity violations of the greedy target; instead, they "gloss over" these violations to interpolate the underlying macro-trend. As illustrated in Figure 1, the interpolation (c) effectively filters the non-monotone noise of the greedy target (b) to recover a verifiably monotone allocation.

Finally, we propose that *neural network interpolation can serve as a general tool for verifying mechanism properties*, including but not limited to strategy-proofness. The mechanism may be expressed mathematically or in code. Our pipeline utilizes neural interpolation as a bridge to formal verification, providing a robust framework to audit mechanism properties that are otherwise computationally inaccessible.

Contribution 2: A suite of monotonicity-seeking techniques, including counterexample-guided training and post-processing monotonicity fix, both enabled by the MLP+ReLU architecture.

Besides verification, the MLP+ReLU architecture enables two monotonicity-seeking techniques. First, when verification fails, the "by-products" of the verification MIP are bid profiles that violate monotonicity, which can be used in *counterexample-guided training* Sivaraman et al. [2020]. The idea is to punish the violation in follow-up training until the violation disappears. Second, if counterexample-guided training still fails to reach monotonicity, then we can implement a post-processing *monotonicity fix* to ensure strategy-proofness.[3] We construct another MIP based on the network parameters, which decides how much to push up the winner's offer while fixing the other bids. The new offer ensures that the winner remains the winner regardless of any bid increment.

As mentioned earlier, limited by the MIP-based verification step, we are restricted to tiny networks. As described by the Lottery Ticket Hypothesis Frankle and Carbin [2019], for tiny networks, having a "lucky" initialization is important. Motivated by this, the last monotonicity-seeking technique is simply *repeated trials*, which turns out to be effective for most distributions that we experimented on. That is, when we encounter either a poorly performing allocation or when monotonicity verification fails, we simply train again from scratch with a fresh initialization, until a near-optimal and verifiably monotone interpolation appears. For many distributions, *repeated trials are all we need.* On the other hand, there are indeed distributions under which repeated trials do not work, and we do need the full suite of techniques, including counterexample-guided training and the post-processing monotonicity fix, to achieve near-optimal revenue. In Appendix E.2, we describe 2 adversarial distributions generated via a separate MIP-based heuristic. In Appendix F, we conducted 100 training trials for each of these adversarial distributions, corresponding to *hundreds of hours* of computation. While we do manage to get verifiably monotone allocations via (now a lot more) repeated trials, the achieved revenue is much worse than the revenue obtained using the full suite of techniques, which is near-optimal. That is, for the two adversarial distributions, even by trying for hundreds of hours, we still cannot reach a near-optimal auction. This demonstrates the need for counterexample-guided training and the monotonicity fix. Lastly, the monotonicity fix offers the peace of mind that regardless of the distribution, our approach always delivers strategy-proofness.

Contribution 3: Extensive experiments demonstrating near-optimal revenue across all distributions tested, with substantial improvements over baselines.

We performed an extensive suite of experiments on 65 correlated distributions, ranging from synthetic (randomly generated and hand-crafted) and real-world data spanning consumer, financial, and industrial domains (Appendix K), to adversarial distributions generated via two distinct methods to stress test our technique (Appendix E). Our performance is impressive across the board in terms of the revenue gap from the (unattainable) greedy upper bound. For instance, in Table 1, the best baseline—the fully strategy-proof variant of RegretNet Dütting et al. [2019], which is also based on our techniques—has an average revenue gap of 2.7%, while our neural network interpolation method achieves an average revenue gap of only 0.26%, representing a tenfold improvement!

Optimal auction design under correlation is a foundational challenge in mechanism design. We provide a robust empirical resolution to this problem through extensive evaluation. By introducing

[3]Our monotonicity fix post-processes the allocation network to ensure strategy-proofness. GemNet Wang et al. [2024] proposed a similar approach that post-processes the payment network to ensure menu compatibility (*i.e.*, preventing two bidders from winning the same item). Their method discretizes the bids and then extend to general bids via Lipschitz smoothness. In contrast, our technique operates directly on continuous values.

the (unattainable) greedy revenue as a rigorous upper bound – the first such benchmark for this setting – we demonstrate that our method consistently achieves near-optimal revenue. This evaluation is supported by a full suite of ablation studies and exhaustive comparisons against all established manual and neural baselines. We provide the complete breadth of these results and technical specifications in the Appendix, alongside our implementation code in the supplementary materials, to ensure total transparency and reproducibility.

Contribution 4: Examples demonstrating some generalisation capabilities of our techniques.

In Appendix I, we extend our techniques to *multi-unit auctions with unit demand*, showing similarly strong performance. In Appendix G, we show that it is convenient to integrate our techniques into RegretNet Dütting et al. [2019] to design fully strategy-proof auctions, while the original RegretNet is only approximately strategy-proof. RegretNet involves both the allocation and the payment networks. We require the allocation network be a tiny MLP with ReLU activation, so that we can apply our MIP-based techniques, including verification and post-processing monotonicity fix, to ensure that the final allocation is monotone. There is no restriction on the size or architecture of the payment network, as the payment network only serves as a surrogate, which will be thrown away when training ends. As long as the allocation is monotone, the "correct" payments can be reverse engineered from the allocation following Myerson's characterization.

2 Model Description

Following Myerson's setting, we aim to design revenue-maximizing single-item auctions that are *deterministic*, *strategy-proof* and *ex post individually rational*. There are n bidders. Bidder i's valuation for the single item is denoted as b_i. Without loss of generality, we assume $0 \leq b_i \leq 1$. Since we focus on strategy-proof auctions, we do not differentiate between reported bids and private valuations. We use $\vec{b} = (b_1, b_2, \ldots, b_n)$ to denote the bid profile, which is drawn from a correlated distribution with the joint probability density function $\phi(\vec{b})$. We assume ϕ is continuous, everywhere positive, and bounded above by a constant. Our goal is to design an auction that maximizes expected revenue under ϕ.

Following Myerson [1981], Papadimitriou and Pierrakos [2011], all deterministic, strategy-proof, and ex post individually rational mechanisms for our setting can be interpreted as *price-oriented rationing-free* mechanisms Yokoo [2003]. That is, every bidder faces a deterministic take-it-or-leave-it offer, which potentially depends on the other bids. The offers must be structured so that for each bid profile, at most one bidder is willing to accept her offer.[4]

An allocation function a maps each bid profile $\vec{b}$ to a binary vector $(a_1, a_2, \ldots, a_{n+1})$, where a_i being 1 means bidder i wins. The last dimension a_{n+1} represents an auxiliary bidder, whose win results in the item not allocated. We require $\sum_i a_i = 1$. We use $a(\vec{b})_i$ to denote the i-th dimension of the allocation vector. Allocation monotonicity requires that $\forall i, b_{-i}, b_i, b_i'$ with $b_i < b_i'$, $a(b_i, b_{-i})_i \leq a(b_i', b_{-i})_i$. Myerson's characterization says that the allocation rule uniquely determines the payment rule. Using the take-it-or-leave-it offer interpretation, if bidder i wins under profile (b_i, b_{-i}), then her payment must be exactly $\inf\{b_i' | a(b_i', b_{-i})_i = 1\}$. Losing bidders pay nothing.

Figure 1a illustrates an example monotone allocation for 2 bidders. Every bid profile $(b_1, b_2) = (x, y)$ corresponds to a point in the unit square. Region A, i.e., the blue points, is where bidder 1 wins. Region B, i.e., the red points, is where bidder 2 wins. The white region is where the item is not allocated. Allocation monotonicity can be interpreted as follows: the blue region must be *rightward-closed*: for every blue point, all points on its right must also be blue. Similarly, the red region must be *upward-closed*: for every red point, all points above it must be red. The region boundaries characterize the payments. If (x, y) is white, then the item is not allocated and no bidders pay. If (x, y) is blue, then bidder 1 wins and her payment is the minimum x' value so that (x', y) is on the blue boundary. If (x, y) is red, then bidder 2 wins and his payment is the minimum y' value so that (x, y') is on the red boundary.

[4] Considering that our objective is revenue in expectation and we assume a continuous probability density function bounded above by a constant, we can ignore all tie-breaking issues.

3 Virtual Valuation, Ironed Virtual Valuation, and Marginal Profit

Myerson [1981] introduced the *virtual valuation* and also a mathematical maneuver called *ironing*, leading to the *ironed virtual valuation*. Papadimitriou and Pierrakos [2011] introduced another variant called *marginal profit*, which comes with a technical change that is helpful for our neural interpolation approach, which we will discuss later in this section. While we can define the greedy allocation based on any of these three variants, the marginal profit version consistently comes out on top in experiments, as evidenced by Table 1. Due to space constraint, in this section, we only provide the bare minimum coverage of these concepts. Details are deferred to Appendix A. Exposition of the virtual valuations, including alternative interpretations, can also be found in Bulow and Roberts [1989], Fu [2017], Hartline [2013].

Virtual valuation: The virtual valuation is defined for independent bidders. Bidder i's virtual valuation only depends on her own distribution and her own bid. We use f_i and F_i to denote the probability density function and the cumulative distribution function of bidder i's valuation. We use o to denote the take-it-or-leave-it offer bidder i faces, with the understanding that o depends on the other bids b_{-i} (but this turns out to be irrelevant in the mathematical analysis). Since it is a take-it-or-leave-it auction from the perspective of bidder i, the expected revenue we can extract from i is $o(1-F_i(o))$. The derivative of $o(1-F_i(o))$ with respect to o equals $1-F_i(o)-of_i(o) = -(o-\frac{1-F_i(o)}{f_i(o)})f_i(o)$. Since $F_i(1)=1$, we can show

$$o(1-F_i(o)) = \int_o^1 \left(x - \frac{1-F_i(x)}{f_i(x)}\right) f_i(x)dx \tag{1}$$

The left side of the formula is the expected revenue we can extract from i. The right side can be interpreted as follows: Consider bidder i who bids x. When i wins, which is when $x \geq o$, her contribution to the revenue is $v_i(x) = x - \frac{1-F_i(x)}{f_i(x)}$, which is called i's virtual valuation. When i loses, which is when $x \leq o$, her contribution to the revenue is 0 as x is below the integration lower limit. Bidder i's expected contribution in terms of virtual valuation equals the expected revenue from i. This interpretation leads to Myerson's greedy allocation. That is, we simply convert the bidders' original bids to their virtual valuations according to the above function v_i, and allocate the item to whoever has the highest virtual valuation. If the highest virtual valuation is negative, then we do not allocate. Being greedy, this allocation goes for the maximum revenue. The caveat is that it may not be monotone. Myerson calls a distribution *regular* if under it, v_i is nondecreasing. If all bidders' distributions are regular and independent, then it is easy to show that the greedy allocation is monotone. If the winner increases her bid while the other bids stay the same, then due to distribution regularity, the winner's virtual valuation never decreases. The other bidders' virtual valuations do not change due to the independence assumption, so the winner still wins when she raises her bid while the other bids stay the same. The virtual valuation can be extended to correlated distributions by switching to conditional probability functions.

Ironed virtual valuation: Above we illustrated that monotonicity holds assuming both distribution regularity (i.e., nondecreasing virtual valuations) and independence. Myerson proposed an ironing technique that converts a not necessarily nondecreasing virtual valuation function to a nondecreasing version. Unfortunately, this ironing process is only mathematically sound with the independence assumption. Considering this limitation, we defer the details of the ironing process to Appendix A.

Marginal profit: The final variant of virtual valuation is marginal profit Papadimitriou and Pierrakos [2011]. Our presentation of the marginal profit is slightly modified from the original, mainly for simplifying presentation (refer to Appendix A for the original definition). We still use f_i and F_i to denote the probability density function and the cumulative distribution function of bidder i. The key difference between marginal profit and virtual valuation lies in the following *revenue fix* process.

Revenue fix: Given a monotone allocation, for any bidder i and any set of other bids b_{-i}, i faces a take-it-or-leave-it offer o, which depends on b_{-i}. We can raise the value of o arbitrarily and this would never break monotonicity or cause over-allocation. Therefore, suppose we have an allocation that offers o to bidder i when the other bids are b_{-i}, instead of offering o directly and achieve a revenue of $o(1-F_i(o))$, we should *optimally* raise the offer to $\arg\max_{x\geq o} x(1-F_i(x))$, leading to a revenue of $\max_{x\geq o} x(1-F_i(x))$.

The following equation defines the marginal profit, which is similar to Equation 1 for virtual valuation. In Equation 1, the left-hand side is the expected revenue extracted from bidder i. In Equation 2 below,

the left-hand side is the expected revenue extracted from bidder i, *after the revenue fix*.

$$\max_{x \geq o} x(1 - F_i(x)) = \int_o^1 \left(-\frac{\partial(\max_{x' \geq x} x'(1 - F_i(x')))}{\partial x} / f_i(x) \right) f_i(x) dx \tag{2}$$

We define $m_i(x) = -\frac{\partial(\max_{x' \geq x} x'(1-F_i(x')))}{\partial x} / f_i(x)$ to be the marginal profit of bidder i when her bid is x. The message of Equation 1 is that the expected revenue is equal to the expected virtual valuation. Equation 2 conveys a similar message, which is that the expected revenue *after the revenue fix* is equal to the expected marginal profit.

Virtual valuation versus marginal profit: Although the two concepts are minor variations of each other, interpolating based on marginal profit offers critical advantages, as confirmed by experiments in Table 1. A detailed technical comparison is provided in Appendix A. Here, we briefly discuss the high-level benefits of interpolating based on marginal profit. We recall that the greedy allocation based on the original virtual valuations works as follows: we allocate the item to the agent with the highest virtual valuation, *and if all agents' virtual valuations are negative, then we do not allocate*. On the other hand, marginal profits are always non-negative by construction. That is, the greedy allocation based on marginal profits simply allocates the item to the agent with the highest marginal profit, eliminating the need to explicitly consider the "no allocation" case. If all agents' marginal profits are zero, then *any* allocation is optimal. This simplification is possible because marginal profits are defined with the *revenue fix* step in mind. That is, the greedy allocation based on marginal profits can defer the "when not to allocate" decisions to the revenue fix step.

The following example illustrates the revenue-fix process and shows how interpolating based on marginal profits effectively allows more "slack" for the neural interpolation.

Revenue fix example: We refer to Figure 1b, which shows an example greedy allocation that is not monotone. Blue points are when bidder 1's marginal profits are strictly higher. Red points are when bidder 2's marginal profits are strictly higher. White points are when both bidders' marginal profits are zero. We train a neural network to fit the greedy allocation, resulting in a monotone allocation depicted in Figure 1c. Since our network size is tiny, it glosses over the fine details, thereby avoiding the minor monotonicity violations by the greedy allocation (i.e., the blue points enclosed by the red region and the red points enclosed by the blue region). We note that when a point's marginal profit is 0, it does not matter which bidder wins. In Figure 1c, the network simply allocates the white points arbitrarily. That is, the focus of interpolation is on regions that actually matter (i.e., the limited expressive power of our tiny network focuses on creating a separation between A and B). Figure 1d shows the aftermath of the revenue fix. The learned allocation allocates A+C to bidder 1 and B+D to bidder 2. After the fix, C and D become unallocated. The final allocation is Figure 1a.

4 Details of the Proposed Approach

We train a neural network to mimic the greedy allocation, which has three versions, depending on whether we greedily allocate according to the vanilla virtual valuation, the ironed virtual valuation, or the marginal profit. When there is no ambiguity, greedy allocation refers to the version based on marginal profit. Even though we are maximizing revenue, we do not need to reference any payment function in training. Training is solely carried out on the allocation function.

We represent the allocation rule as an MLP with ReLU activation. The inputs are the bids (n dimensions). The output dimension is $n + 1$. If the i-th ($i = 1, 2, \ldots, n$) output coordinate is the highest, then bidder i wins. If the $(n + 1)$-th coordinate is the highest, then the item is not allocated.

In Myerson's original approach, during the derivation of the optimal auction, randomization is allowed, which simplifies the mathematical analysis. Similarly, we allow randomization to facilitate the training. We apply softmax to the network's outputs, ensuring that the coordinate with the highest value corresponds to the highest proportion of the item won in the context of randomized auctions. During training, we sample a batch of bid profiles. For each profile $\vec{b}$, we calculate the marginal profits of each bidder. The marginal profit of the auxiliary bidder $(n + 1)$ is always 0. We use $m(\vec{b})$ to represent the vector of marginal profits. We use *NN* to represent the network. The training loss is simply the batched sum of $-m(\vec{b}) \cdot \text{softmax}(NN(\vec{b}))$ over all training samples in the batch. This mimics greedy allocation because in the case of perfect fit, if the marginal profit of bidder i is the highest, then the i-th coordinate of $\text{softmax}(NN(\vec{b}))$ should be 1 and the other coordinates should be

0's. During evaluation, we revert back to the deterministic version. That is, we do not apply softmax and simply pick the highest output coordinate as the winner.

Our techniques also include the use of two mixed-integer programs. The first is for verifying whether the trained allocation is monotone. For a trained network, its weights and biases are viewed as constants. Since our network is based on MLP+ReLU, for any node in the network, its value before activation can be written as a linear expression involving the activated versions of the node values from the previous layer. The ReLU activation step can be modelled using the big-M trick with the introduction of an auxiliary binary variable. When presenting the mixed-integer programs, we simply use $(a_1, a_2, \ldots, a_{n+1}) = NN(\vec{b})$ to represent that the a_i's are the outputs of the network when the input is $\vec{b}$, with the understanding that the a_i can be written as linear expressions of $\vec{b}$ with the help of auxiliary binary variables. The MIP for monotonicity verification finds the largest gap between b_i and b'_i so that i wins when bidding b_i, but after increasing her bid to b'_i while keeping the other bids fixed, a different bidder j wins (j can be $n+1$). The gap is the maximum over $\forall b_{-i}$. In order to verify monotonicity, we need to run n^2 MIPs (for every i from 1 to n and for every $j \neq i$ from 1 to $n+1$). All MIPs' objectives must be 0's (or infeasible) to conclude monotonicity.

MIP for monotonicity verification

Constants	$1 \leq i \leq n; j \neq i; 1 \leq j \leq n+1$
Variables	$0 \leq b_1, b_2, \ldots, b_n, b'_i \leq 1; b_i \leq b'_i$
Maximize	$b'_i - b_i$
subject to	$(a_1, a_2, \ldots, a_{n+1}) = NN(b_i, b_{-i})$
	$a_i = \max_t a_t$
	$(a'_1, a'_2, \ldots, a'_{n+1}) = NN(b'_i, b_{-i})$
	$a'_j = \max_t a'_t$

MIP for monotonicity fix

Constants	$1 \leq i \leq n; j \neq i; 1 \leq j \leq n+1$
	$0 \leq b_1, \ldots, b_{i-1}, b_{i+1}, \ldots, b_n \leq 1$
Variables	$0 \leq b'_i \leq 1$
Maximize	b'_i
subject to	$(a_1, a_2, \ldots, a_{n+1}) = NN(b'_i, b_{-i})$
	$a_j = \max_t a_t$

The by-products of the MIP for monotonicity verification are the counterexamples. Given a counterexample, we add a penalty term $\lambda \cdot \text{ReLU}(\text{softmax}\,(NN(b_i, b_{-i}))_i - \text{softmax}(NN(b'_i, b_{-i}))_i)$, which aims to ensure that by increasing her bid, bidder i's proportion of item won must never decrease.

With additional training on counterexamples, we sometimes, but not always, end up with verifiably monotone allocations. When we do not manage to achieve verifiable monotonicity, we can always apply a post-processing monotonicity fix using the second MIP. This fix is applied after collecting the bids. When calculating i's fix, the bids in b_{-i} are treated as constants. For bidder i, the second MIP finds the maximum bid for i where i loses (according to the trained network) to a different bidder j (j could be $n+1$). We need to solve n MIPs only for the (tentatively, i.e., according to the trained network) winning bidder, and the maximum over the n objective values (ignoring infeasible MIPs) is the monotonicity cutoff price for i. As i's bid goes from this cutoff price to 1, i always wins, which implies monotonicity. But if i's bid was below this, i does not win after all. The monotonicity fix is applied together with the revenue fix from Section 3. For example, let the original offer produced by the network be o. The revenue fix instructs to raise the offer to o_r and the monotonicity fix instructs to raise the offer to o_m. The final offer is then $\max\{o, o_r, o_m\}$.

5 Main Experiments

$$\begin{bmatrix} 11 & 24 & 7 & 20 & 3 \\ 4 & 12 & 25 & 8 & 16 \\ 17 & 5 & 13 & 21 & 9 \\ 10 & 18 & 1 & 14 & 22 \\ 23 & 6 & 19 & 2 & 15 \end{bmatrix}$$

To facilitate experiments, we introduce the *grid distributions*, where we use n-dimensional matrices to represent correlated distributions for n bidders. The example 2D matrix below represents a correlated distribution for 2 bidders. Here, the matrix size is 5×5. We divide the unit square $[0, 1] \times [0, 1]$ into 25 sub-squares of side length $\frac{1}{5}$. The bottom left element is 23, which means that when the bid profile (x, y) falls into sub-square $[0, \frac{1}{5}] \times [0, \frac{1}{5}]$, the joint probability density is *uniformly* 23. The element on the right of 23 is 6, which means that for the sub-square $[\frac{1}{5}, \frac{2}{5}] \times [0, \frac{1}{5}]$, the joint probability density is uniformly 6.[5]

[5]There is an additional normalization ratio, which will be multiplied with every density value to ensure that the total probability equals exactly 1. It is omitted to make the presentation cleaner.

There are multiple reasons for introducing the grid distributions. 1) They allow the *systematic generation* of correlated distributions. 2) All relevant numerical operations can be implemented *analytically*, including derivative, maximum, and integration. This is important for producing *numerically stable* experimental results as the marginal profit is the *partial derivative* of the *maximum* of a term involving the cumulative distribution function, which calls for an *integration*. 3) Theoretically speaking, the grid distributions can approximate *any* correlated distribution if we allow arbitrarily fine grids. 4) The grid distribution family makes it easy to generate *adversarial distributions* via tuning the matrix. For example, in Appendix E, we report on experiments using evolutionary computation to evolve an adversarial matrix, as well as generating adversarial matrices using mixed-integer-programming. 5) We can easily construct hand-crafted grid distributions. For example, we use *magic squares* – 2D matrices in which every row and every column sums to the same value – to create correlated distributions where each bidder's marginal distribution is uniform. The matrix shown above is a magic square. We use magic-square-based distributions to evaluate the effect of correlation on revenue. For such distributions, if we ignore correlation, then the distribution becomes i.i.d. uniform, where the second price auction with reserve is optimal. As a result, on correlated distributions, the gap between our revenue and the revenue from a second-price auction with a reserve price based on an i.i.d. distribution represents the additional revenue we can gain by considering correlation.

Table 1: 2 bidders: GAP(MP) $= 0.26\%$,GAP(RNET) $= 2.7\%$,GAP(2ND) $= 4.2\%$,GAP(MYER) $= 4.3\%$

DISTRIB.	GREEDY	AMD	MYER.	2ND	RNET	MP	VV	IVV
$G5_0$	0.4353	0.4368	0.4191	0.4197	0.4307	**0.4368**	0.4355	0.4329
$G5_1$	0.4047	0.4041	0.3939	0.3929	0.4021	**0.4037**	0.4032	0.4032
$G5_2$	0.3979	0.3967	0.3771	0.3823	0.3869	**0.3959**	0.3946	0.3955
$G5_3$	0.4629	0.4625	0.4533	0.4546	0.4522	**0.4628***	0.4612	0.4616
$G5_4$	0.4681	0.4665	0.4476	0.4472	0.4639	**0.4674***	0.4673	0.4671
$G5_5$	0.4204	0.4187	0.4075	0.4057	0.4062	**0.4151**	0.4148	0.4150
$G5_6$	0.3905	0.3912	0.3761	0.3741	0.3798	**0.3908**	0.3893	0.3886
$G5_7$	0.4865	0.4854	0.4494	0.4544	0.4771	**0.4850**	0.4828	0.4840
$G5_8$	0.4298	0.4273	0.3988	0.3981	0.4077	**0.4291***	0.4281	0.4279
$G5_9$	0.4531	0.4512	0.4306	0.4302	0.4379	**0.4498**	0.4482	0.4497
SATURN	0.4533	0.4545	0.4210	0.4226	0.4372	0.4510	0.4512	**0.4515**
JUPITER	0.4427	0.4404	0.4238	0.4219	0.4316	**0.4418***	0.4409	0.4410
MARS	0.4375	0.4377	0.4177	0.4167	0.4194	**0.4360**	0.4333	0.4359
SOL	0.4429	0.4414	0.4376	0.4364	0.4329	**0.4427***	0.4404	0.4419
VENUS	0.4381	0.4381	0.4175	0.4171	0.4233	**0.4365**	0.4319	0.4358
MERCURY	0.4273	0.4274	0.4179	0.4176	0.4150	**0.4281***	0.4271	0.4279
LUNA	0.4331	0.4335	0.4173	0.4172	0.4212	**0.4322**	0.4315	0.4318

Table 2: 3 and 5 bidders: M/2 represents the better between MYERSON and 2ND
$n = 3$: GAP(MP) $= 0.36\%$, GAP(RNET) $= 3.6\%$, GAP(2ND) $= 3.9\%$, GAP(MYER) $= 3.8\%$
$n = 5$: GAP(MP) $= 0.87\%$, GAP(RNET) $= 2.8\%$, GAP(2ND) $= 2.4\%$, GAP(MYER) $= 2.4\%$

	$n = 3$				$n = 5$			
DISTRIB.	GREEDY	M/2	RNET	MP	GREEDY	M/2	RNET	MP
$G5_0$	0.5512	0.5346	0.5346	**0.5517**	0.6862	0.6700	0.6670	**0.6799**
$G5_1$	0.5647	0.5475	0.5509	**0.5651**	0.6901	0.6738	0.6713	**0.6846**
$G5_2$	0.5316	0.5137	0.5153	**0.5302**	0.6838	0.6709	0.6681	**0.6783**
$G5_3$	0.5476	0.5283	0.5290	**0.5450**	0.6894	0.6724	0.6684	**0.6838**
$G5_4$	0.5485	0.5306	0.5310	**0.5471**	0.6853	0.6692	0.6664	**0.6787**
$G5_5$	0.5477	0.5237	0.5255	**0.5443**	0.6850	0.6720	0.6685	**0.6809**
$G5_6$	0.5371	0.5155	0.5158	**0.5332**	0.6902	0.6725	0.6686	**0.6839**
$G5_7$	0.5723	0.5528	0.5555	**0.5710**	0.6900	0.6719	0.6682	**0.6824**
$G5_8$	0.5473	0.5193	0.5185	**0.5429**	0.6868	0.6715	0.6659	**0.6807**
$G5_9$	0.5663	0.5410	0.5396	**0.5637**	0.6927	0.6768	0.6711	**0.6866**

We altogether experimented with 62 different grid distributions as follows. *In Appendix J, we provide additional experiments on* 3 *continuous distributions.*

• **40 randomly generated grid distributions:** For n from 2 to 5, we generate 10 random grid distributions of size $\underbrace{5 \times \ldots \times 5}_{n}$, where every matrix element is drawn from uniform 0 to 1. We use $G5_s$ to denote the distribution generated with seed s.

• **7 hand-crafted grid distributions:** These are based on magic squares described in Appendix D.

• **12 adversarially generated distributions:** 10 were generated using evolutionary computation and 2 were generated using mixed-integer-programming. We defer all discussion on adversarial distributions to Appendix E. *We use them to test the limit of our proposed approach.*

• **3 empirical distributions derived from real-world datasets:** These datasets come from diverse real-world market domains: (i) Consumer (eBay), involving layperson bidding for electronics; (ii) Financial (Ethereum), involving algorithmic bidding for block building rights; and (iii) Electricity Market (PJM), involving professional strategic bids from energy producers. Discussions on these distributions are deferred to Appendix K.

As mentioned in Section 3, there are three variants of the virtual valuation, leading to three greedy allocations that can be used as interpolation targets. For 2 bidders, we compare the revenue achieved using our approach based on MP (marginal profit), VV (vanilla virtual valuation) and IVV (ironed virtual valuation). MP consistently comes out on top. For 3 to 5 bidders, we only focus on MP. Our experiments also involve the following baseline auctions:

• **Greedy (based on marginal profit):** For distributions we experimented on, the greedy allocation is rarely monotone. GREEDY violates the key constraint and only serves as the revenue *upper bound*.

• **AMD:** We round the bids down to the nearest multiple of 0.01 and apply the classical approach to automated mechanism design Conitzer and Sandholm [2002, 2004], which is a mixed-integer-programming-based approach producing the optimal auction when we restrict to discrete bids. We use AMD as an alternative implementation to the FPTAS for 2 correlated bidders from Papadimitriou and Pierrakos [2011], which is also based on discrete bids. Both AMD and FPTAS are for 2 bidders only: AMD does not scale beyond 2 bidders for our setting and the FPTAS only applies to 2 bidders (optimal correlated auction design becomes NP-hard to approximate for 3 bidders).

• **Myerson:** Myerson's optimal auction ignoring correlation. The bidders' ironed virtual valuations are based on *marginal* distributions (instead of *conditional* distributions).

• **2nd:** Second price auction with the optimal reserve.

• **RNet:** Integration of verification to RegretNet Dütting et al. [2019], details in Appendix G.

In Tables 1 and 2, we present the results for $n \in \{2, 3, 5\}$. Full tables including standard errors, results for $n = 4$, and results for adversarial distributions are deferred to Appendix L. A value is bold if it is the best among the scalable approaches (AMD is not in the comparison). MP consistently achieves the best revenue among the scalable approaches. When MP also outperforms AMD, we mark the result with a '*'. Sometimes MP's value appears to be higher than the upper bound GREEDY. This is caused by Monte Carlo simulation (see evaluation details in Appendix B).

GREEDY's revenue is an unattainable upper bound. We use GAP(METHOD) to denote the average revenue gap between METHOD and GREEDY, formally defined as

$$\frac{\text{Average revenue of GREEDY} - \text{Average revenue of METHOD}}{\text{Average revenue of GREEDY}}$$

In Table 1, the best baseline RNET – the fully strategy-proof variant of RegretNet Dütting et al. [2019], which is also based on our techniques – has an average revenue gap of 2.7%, while our method MP achieves an average revenue gap of only 0.26%, representing a tenfold improvement!

In Appendix I, we extend to multi-unit auctions with unit demand, where we achieved similar strong performance. In Table 7, the baseline $(m + 1)$-th price auction with the optimal reserve has an average revenue gap of 5.1%, while our method MP has an average revenue gap of 0.67%.

6 Related Research

Papadimitriou and Pierrakos [2011] proposed the *marginal profit* (compared with virtual valuation in Appendix A), gave an FPTAS for two correlated bidders, and showed that the problem is NP-hard to approximate for three or more. Dobzinski et al. [2011] showed that under truthful-in-expectation, the optimal auction for correlated bidders can be found in time polynomial in the support size. Randomization matters here: determinism is without loss for independent bidders Myerson [1981] but costs revenue under correlation Caragiannis et al. [2016]. Roughgarden and Talgam-Cohen [2013] gave a sufficient condition for Myerson's greedy allocation to remain monotone under correlation, but it is unclear how to check the condition computationally for a given distribution, and for most distributions in this paper the greedy allocation is in fact not monotone. Ronen [2001] proposed an approximate auction that guarantees half of the revenue for correlated bidders. Crémer and McLean [1988] showed that it is possible to extract the full social surplus for correlated bidders, but the proposed auction is only individually rational in expectation. Bei et al. [2019] studied single-item auctions for correlated bidders from the lens of worst-case analysis. The authors evaluate an auction based on its revenue under the worst-case correlation.

This paper follows the line of research on neural network mechanism design initiated by Dütting et al. [2019]. Curry et al. [2020] first applied mixed-integer-programming to evaluate strategy-proofness of neural-network-based mechanisms. Guo [2024] proposed a suite of techniques on worst-case mechanism design via neural networks, including worst-case counterexample-guided training where the worst-case profiles are obtained via mixed-integer-programming and using the Lottery Ticket Hypothesis Frankle and Carbin [2019] to guide the search for tiny networks that are trainable for worst-case training objectives. Duan et al. [2024] studied *virtual valuation combinatorial auctions* Sandholm and Likhodedov [2015] via neural networks, where "virtual valuation" is a different concept from Myerson's. The monotonicity fix in our paper post-processes the allocation network to ensure strategy-proofness. GemNet Wang et al. [2024] proposed a similar approach that post-processes the payment network to ensure menu compatibility (*i.e.*, preventing two bidders from winning the same item). Their method discretizes the bids and then extend to general bids via Lipschitz smoothness. In contrast, our technique operates directly on continuous values.

Monotonic network structures such as min-max networks Sill [1997], Milani Fard et al. [2016] enforce monotonicity of the network outputs, which differs from allocation monotonicity despite the shared name. The work closest to ours is Huo et al. [2025], which extends MyersonNet Dütting et al. [2019] to correlated bidders. MyersonNet assigns every bidder an allocation priority value computed by a min-max network from her own bid only, so it is monotone by construction but cannot fully exploit bid correlation. Huo et al. [2025] instead make each priority value a multi-input min-max network that is nondecreasing in the bidder's own bid and nonincreasing in the others' (the *Lopomo assumptions* Roughgarden and Talgam-Cohen [2013]), so allocation monotonicity still holds by construction, and payments are derived analytically from the network structure. The price of guaranteeing strategy-proofness by construction is expressiveness. In Appendix H, we give an example where no allocation satisfying the Lopomo assumptions can express the optimal auction, and show experimentally that min-max variants, even when combined with our training techniques, fall below the second-price auction with an optimal reserve price, on the distributions that we experimented on. We instead use unconstrained MLPs and establish strategy-proofness by formal verification, and we map marginal profits directly to payments, keeping a direct link to Myerson's theory without additional architecture for computing payments. A second difference is in evaluation. Huo et al. [2025] treat RegretNet's revenue as the gold standard to approach, and use the second-price auction without reserve as baseline. Neither anchors the results to optimality: RegretNet is only approximately strategy-proof and comes with no optimality guarantee, and the second-price auction without reserve is far from optimal. We instead measure against the greedy revenue, a rigorous upper bound on the optimal revenue, and against stronger baselines: the second-price auction with an optimal reserve, Myerson's auction based on marginal distributions (Appendix C), and a fully strategy-proof RegretNet variant built with our verification technique. In Table 1, the strategy-proof RegretNet variant has an average revenue gap of 2.7% from the upper bound, while our method achieves 0.26%. The original RegretNet's revenue is inflated by its strategy-proofness violations and is therefore not comparable to that of strategy-proof auctions; once strategy-proofness is enforced, even the hand-crafted baselines sometimes outperform it (Tables 12 and 13).

Acknowledgments and Disclosure of Funding

This work was supported with supercomputing resources provided by the Phoenix HPC service at Adelaide University. Jiayuan Liu and Vincent Conitzer thank the Cooperative AI Foundation, Macroscopic Ventures (formerly Polaris Ventures / the Center for Emerging Risk Research) and Jaan Tallinn's donor-advised fund at Founders Pledge for financial support.

References

Xiaohui Bei, Nick Gravin, Pinyan Lu, and Zhihao Gavin Tang. Correlation-robust analysis of single item auction. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 193–208. SIAM, 2019.

Jeremy Bulow and John Roberts. The simple economics of optimal auctions. *Journal of political economy*, 97(5):1060–1090, 1989.

Ioannis Caragiannis, Christos Kaklamanis, and Maria Kyropoulou. Limitations of deterministic auction design for correlated bidders. *ACM Transactions on Computation Theory (TOCT)*, 8(4): 1–18, 2016.

Vincent Conitzer and Tuomas Sandholm. Complexity of mechanism design. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence*, UAI'02, page 103–110, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc. ISBN 1558608974.

Vincent Conitzer and Tuomas 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.

Jacques Crémer and Richard P McLean. Full extraction of the surplus in Bayesian and dominant strategy auctions. *Econometrica: Journal of the Econometric Society*, pages 1247–1257, 1988.

Michael Curry, Ping-Yeh Chiang, Tom Goldstein, and John Dickerson. Certifying strategyproof auction networks. *Advances in Neural Information Processing Systems*, 33:4987–4998, 2020.

Shahar Dobzinski, Hu Fu, and Robert D Kleinberg. Optimal auctions with correlated bidders are easy. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 129–138, 2011.

Zhijian Duan, Haoran Sun, Yichong Xia, Siqiang Wang, Zhilin Zhang, Chuan Yu, Jian Xu, Bo Zheng, and Xiaotie Deng. Scalable virtual valuations combinatorial auction design by combining zeroth-order and first-order optimization method. *arXiv preprint arXiv:2402.11904*, 2024.

Paul Dütting, Zhe Feng, Harikrishna Narasimhan, David Parkes, and Sai Srivatsa Ravindranath. Optimal auctions through deep learning. In *International Conference on Machine Learning*, pages 1706–1715. PMLR, 2019.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

Hu Fu. Notes on Myerson's revenue optimal mechanisms. Lecture notes, 2017.

Mingyu Guo. Worst-case VCG redistribution mechanism design based on the lottery ticket hypothesis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 9740–9748, 2024.

Jason D Hartline. Mechanism design and approximation. *Book draft. October*, 122(1), 2013.

Da Huo, Zhenzhe Zheng, and Fan Wu. Learning optimal auctions with correlated value distributions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 13944–13952, Apr. 2025.

Mahdi Milani Fard, Kevin Canini, Andrew Cotter, Jan Pfeifer, and Maya Gupta. Fast and flexible monotonic functions with ensembles of lattices. *Advances in neural information processing systems*, 29, 2016.

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

Christos H Papadimitriou and George Pierrakos. On optimal single-item auctions. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 119–128, 2011.

Amir Ronen. On approximating optimal auctions. In *Proceedings of the 3rd ACM conference on Electronic Commerce*, pages 11–17, 2001.

Tim Roughgarden and Inbal Talgam-Cohen. Optimal and near-optimal mechanism design with interdependent values. In *Proceedings of the fourteenth ACM conference on Electronic commerce*, pages 767–784, 2013.

Tuomas Sandholm and Anton Likhodedov. Automated design of revenue-maximizing combinatorial auctions. *Oper. Res.*, 63(5):1000–1025, oct 2015. ISSN 0030-364X.

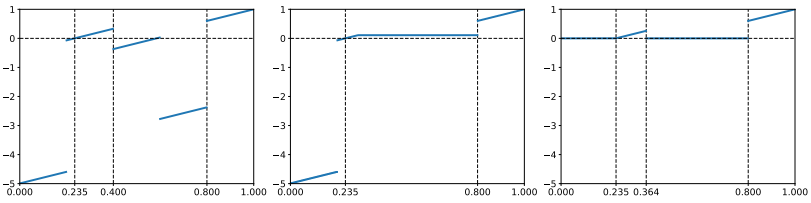
Joseph Sill. Monotonic networks. *Advances in neural information processing systems*, 10, 1997.

Aishwarya Sivaraman, Golnoosh Farnadi, Todd Millstein, and Guy Van den Broeck. Counterexample-guided learning of monotonic neural networks. *Advances in Neural Information Processing Systems*, 33:11936–11948, 2020.

Tonghan Wang, Yanchen Jiang, and David C. Parkes. Gemnet: Menu-based, strategy-proof multi-bidder auctions through deep learning. In *Proceedings of the Twenty-Fifth ACM Conference on Economics and Computation*, 2024.

Makoto Yokoo. Characterization of strategy/false-name proof combinatorial auction protocols: Price-oriented, rationing-free protocol. In *IJCAI*, pages 733–742, 2003.

A More on Virtual Valuation, Ironed Virtual Valuation, and Marginal Profit



(a) Virtual valuation (b) Ironed virtual valuation (c) Marginal profit

Figure 2: Plots for grid distribution $[0.2, 3.58, 0.32, 0.05, 0.85]$. The grid distribution family is formally defined in Section 5. The example grid distribution here describes a random variable with range $[0, 1]$, which is divided into 5 equally-sized sub-intervals. In $[0, \frac{1}{5}]$, the probability density value is uniformly 0.2. In $[\frac{1}{5}, \frac{2}{5}]$, the probability density value is uniformly 3.58.

In this section, we present the omitted details from Section 3. We start with Figure 2, which depicts the three variants of virtual valuations for the example grid distribution $[0.2, 3.58, 0.32, 0.05, 0.85]$.

Ironed virtual valuation: The greedy allocation based on the virtual valuation guarantees monotonicity when the distribution is both regular and independent. Actually, the regularity assumption can be relaxed as long as independence holds. Myerson proposed an ironing technique that converts a not necessarily nondecreasing virtual valuation function (i.e., Figure 2a) to a nondecreasing version (i.e., Figure 2b). The effect of ironing is that the decreasing regions (and some of their nearby regions) become "flat", and within a flat region, the original virtual valuation is replaced by the average (weighted according to the probability density). Unfortunately, this averaging process is only mathematically sound with the independence assumption. While we can still iron in the presence of correlation, and base the greedy allocation on the ironed virtual valuations, the resulting auctions are suboptimal, as confirmed by experiments in Table 1.

Considering that ironing does not align well with the correlated setting, we only very briefly describe its high-level idea and also why it causes "information loss" in training. We still use f_i and F_i to denote bidder i's probability density function and cumulative distribution function. Myerson's ironing approach requires the introduction of new notations. We use $q = 1 - F_i(o)$ to denote bidder i's *demand* when faced with offer o. We use $R_i(q)$ to denote the revenue extracted from bidder i when her demand is q. That is, $R_i(q) = qo = qF_i^{-1}(1-q)$. The derivative of this revenue with respect to the demand is then $\frac{dR_i(q)}{dq} = F_i^{-1}(1-q) - \frac{q}{f_i(F_i^{-1}(1-q))}$. If we rewrite this derivative in terms of o, noting that $o = F_i^{-1}(1-q)$, then the derivative becomes $o - \frac{1-F_i(o)}{f_i(o)}$, which is exactly $v_i(o)$, i.e., the virtual valuation at value o. If the virtual valuation is nondecreasing in o, then it is nonincreasing in q, as higher offers correspond to lower demands. This implies that for regular distributions, $R_i(q)$ is *concave* (i.e., its derivative does not increase in q). For non-regular distributions, $R_i(q)$ is not concave, which means that there exist two demand values q_1 and q_2, so that $\frac{R_i(q_1)+R_i(q_2)}{2} \geq R_i(\frac{q_1+q_2}{2})$. During Myerson's derivation of the optimal auction, randomized auctions are allowed. It is just that the final optimal auction is proven to be deterministic. When randomized auctions are allowed, the offer leading to demand $\frac{q_1+q_2}{2}$ is never a good idea, because more revenue can be achieved by replacing this offer by a half/half mixture of the offers leading to demand q_1 and q_2. In summary, Myerson showed that we can always resort to randomization to achieve a concave revenue function $\overline{R_i}(q)$ on the basis of the original $R_i(q)$, i.e., by pushing up $R_i(\frac{q_1+q_2}{2})$ to $\frac{R_i(q_1)+R_i(q_2)}{2}$ for the non-concave regions. In terms of implementation, one option is the Graham scan algorithm, a computational geometry algorithm for generating the convex hull given a set of points. Recall that the derivative of the original revenue function $R_i(q)$ is the original virtual valuation. Myerson uses the derivative of $\overline{R_i}(q)$ to serve as the ironed virtual valuation, which is guaranteed to be nondecreasing (in the bid) due to the concavity of $\overline{R_i}(q)$. Note that R_i and $\overline{R_i}$ would differ in one or several regions. For example, let $[q_1, q_2]$ be one region where R_i and $\overline{R_i}$ differ. The curve of $\overline{R_i}$ on $[q_1, q_2]$ is exactly the straight line connecting $(q_1, R_i(q_1))$ and $(q_2, R_i(q_2))$, which means that the derivative of $\overline{R_i}$, i.e., the ironed virtual valuation, must be a constant in $[q_1, q_2]$. This is how the flat region in Figure 2b is calculated.

Next, we discuss why the above ironing process is no longer valid for correlated distributions and why it causes "information loss" in training. In Figure 2b, there is a long flat region $[0.289, 0.8]$. The virtual valuation in $[0.289, 0.8]$ is replaced by the average 0.109. Suppose that the curve shown in Figure 2b is for bidder i. For independent bidders, when we apply the greedy allocation, if the other bidders' ironed virtual valuations do not exceed 0.109, then bidder i would win if her bid is at least 0.289, which makes it correct to use the average to replace the actual virtual valuations, because bidder i "wins" for the whole interval $[0.289, 0.8]$. If the maximum of the other bidders' ironed virtual valuations exceeds 0.109, then bidder i does not win with any bid below 0.8, so the virtual valuation changes in $[0.289, 0.8]$ (due the averaging process) are irrelevant. When we extend to correlated distributions (by defining ironed virtual valuations in terms of conditional distributions), the above analysis no longer holds. When bidder i changes her bid within $[0.289, 0.8]$, the other bidders' ironed virtual valuations may change due to correlation. For example, it is possible that when bidder i bids below 0.4, there is another bidder whose virtual valuation is higher than 0.109, and when bidder i bids above 0.4, all other bidders' virtual valuations are below 0.109. In this situation, within the flat region, bidder i would only win in the region $[0.4, 0.8]$. The ironed virtual valuation suggests that getting allocated in $[0.4, 0.8]$ is beneficial to the revenue, as this flat region is positive in Figure 2b. But according to the actual virtual valuation in Figure 2a, getting allocated in $[0.4, 0.8]$ actually hurts the revenue.

Our presentation of marginal profit vs the original: Our presentation of marginal profit is slightly modified from the original definition from Papadimitriou and Pierrakos [2011]. We use $m_i(b_i|b_{-i})$ to denote the marginal profit of bidder i when her bid is b_i and the other bids are b_{-i}. Recall that ϕ is the joint probability density function. We use $\varphi(b_{-i})$ to denote the joint probability density function of the bids except for i's bid. The original definition in Papadimitriou and Pierrakos [2011] is

$$m_i(b_i|b_{-i}) = -\frac{\partial\left(\max_{x' \geq b_i} x' \int_{x'}^{1} \phi(t, b_{-i})dt\right)}{\partial b_i}$$

$$= -\phi(\vec{b})\frac{\partial\left(\max_{x' \geq b_i} x'(1 - F_i(x'|b_{-i}))\right)}{\partial b_i}\Big/ f_i(b_i|b_{-i})$$

Our presentation (after switching to conditional distributions) is essentially

$$m_i(b_i|b_{-i}) = -\frac{\partial\left(\max_{x'\geq b_i} x'(1-F_i(x'|b_{-i}))\right)}{\partial b_i}\Big/ f_i(b_i|b_{-i}).$$

The only difference is the multiplier $\phi(\vec{b})$. Since what we truly care about is the relative order of the marginal profits, i.e., which bidder has the highest marginal profit, the two presentations are equivalent for this purpose. Our presentation enabled the direct comparison between the marginal profit and the (ironed) virtual valuation in Figure 2.

Detailed comparison between virtual valuation versus marginal profit: If the distribution is regular, then the derivative of $x(1-F_i(x))$ with respect to x, i.e., the original virtual valuation, is nondecreasing, which means that there is a cutoff value x^0, so that $x(1-F_i(x))$ is nonincreasing when $x \leq x^0$ and it is nondecreasing when $x \geq x^0$. This implies $m_i(x) = 0$ if $x \leq x^0$ and $m_i(x) = v_i(x)$ if $x \geq x^0$. That is, for regular distributions, $m_i(x) = \max\{v_i(x), 0\}$. The only difference between the virtual valuation function v_i and the marginal profit function m_i is the additional "max". For general distributions, the difference is more notable, but it remains based on the same underlying principle.

Although the two concepts are minor variations of each other, interpolating based on marginal profit offers critical advantages, as confirmed by experiments in Table 1. Below, we provide an example to illustrate the benefits of marginal profit over virtual valuation. We take the uniform distribution over $[0, 1]$ as an example, which is regular. Suppose we only have one bidder. The optimal offer should be 0.5, which can be interpreted as an optimal reserve price. Myerson's virtual valuation function is $v_i(x) = x - \frac{1-F_i(x)}{f_i(x)} = 2x-1$. The marginal profit function is $m_i(x) = \max\{2x-1, 0\}$. According to the virtual valuation, when the bid x is below 0.5, the virtual valuation is negative, which means we should allocate the item only if the bid is at least 0.5, therefore explaining the optimal reserve at 0.5. On the other hand, the marginal profit is 0 when the bid x is below 0.5. Therefore, when the bid is lower than 0.5, allocating is neither helpful nor detrimental by this measure. When the bid is higher than 0.5, the marginal profit is positive, so allocating is helpful. So according to marginal profit, every offer in $[0, 0.5]$ is optimal, including offering 0. *Actually* offering $o < 0.5$ is certainly not optimal, but such a suboptimal offer can be easily fixed by raising it to the best revenue point, i.e., to $\arg\max_{x\geq o} x(1-F_i(x)) = 0.5$. Another way to put it is that the suboptimal offer can be fixed by pushing it up until the marginal profit becomes positive, i.e., to $\inf\{x|m_i(x) > 0, x \geq o\} = 0.5$. The above example illustrates an advantage of marginal profit – it cuts more slack to our learning procedure. If we learn the allocation for this example by mimicking the greedy allocation based on either the virtual valuation or the marginal profit, then if we go with the virtual valuation, the learned reserve must be *exactly* 0.5. Any more or less is not optimal. On the other hand, according to the marginal profit, 0.43 is optimal, and so is 0.37. The exact optimal reserve can always be recovered by pushing up the offer via the revenue fix process. It should be noted that we *must* go through the trouble of performing the revenue fix process if the learning target is based on the marginal profit. Otherwise, we run the risk of using a reserve of 0, which is far from optimal.

B Training Parameters, Evaluation Details, and Hardware

The description below is for our main approach. For integration with RegretNet, please refer to Appendix G.

Training: We use an MLP with 2 hidden layers to represent the allocation function. Each layer contains 20 nodes. We use the Adam optimizer with learning rate 0.001. Unless otherwise specified, we allow the optimizer to step $20,000$ times. The batch size is 16. For counterexample-guided training, we use all counterexamples and the penalty term is multiplied by $1,000$.

The following description applies to all distributions except for WORST10 and WORST100 (Appendix E.2). For 2 bidders, we only need to train once to achieve a monotone allocation. No counterexample-guided training or monotonicity fix is needed. This is true even for the 10 adversarial distributions generated using evolutionary computation (Appendix E.1). For 3 to 5 bidders, we train 10 times as some trials failed to achieve monotonicity. 10 trials are more than enough. Even in the worst situation, at least 6 of the 10 trials ended up monotone without using counterexample-guided

training or monotonicity fix. For WORST10 and WORST100, we do need counterexample-guided training and monotonicity fix. Otherwise, near-optimal revenue cannot be achieved. The experiments on these two distributions are presented in Appendix F and E.2.

Evaluation: We evaluate an auction's revenue via Monte Carlo average with sample size $100,000$. When monotonicity fix is needed, the sample size is reduced to $10,000$, as for each profile, monotonicity fix involves n mixed-integer programs.

Hardware: The only large-scale experiments are the experiments described in Figure 3, Figure 4 and Figure 5, where for each of the three selected distributions ($G5_6$, WORST10, WORST100), we performed 100 training trails. This is carried out on Adelaide University's high-performance cluster. Our performance bottleneck is the MIPs for monotonicity fix, which needs to be repeated $n \times 10,000$ times in evaluation, so our jobs are CPU intensive. (Running the auction once is always instant.) For each of the 300 trials, we allocate one CPU core, 128 GB memory, and no GPU. The CPU type is Intel(R) Xeon(R) Platinum 8360Y. Each trial takes from a few minutes (when monotonicity fix is not needed) to up to six hours. We have to point out that the wall clock time is highly inaccurate for the cluster we use (i.e., CPUs are shared; jobs may hang or slow down without notice). We estimate that the total computation time is on the order of 500 hours.

C Details of Baseline Auctions

GREEDY: The greedy allocation simply allocates the item to the bidder with the highest marginal profit. Based directly on the definition of marginal profit Papadimitriou and Pierrakos [2011], the greedy allocation's revenue can serve as the revenue *upper bound*. To calculate this upper bound, we use Monte Carlo simulation to calculate the expectation of the highest marginal profit. For the vast majority of the distributions we use, the greedy allocation is **not** monotone, and therefore **not** strategy-proof, which is proved via counterexamples. When counterexamples cannot be found, monotonicity/strategy-proofness is inconclusive.

AMD: Papadimitriou and Pierrakos [2011] proposed an FPTAS for deriving the optimal auction for 2 correlated bidders. The proposed algorithm first rounds down the bids to grid points, for example, to multiples of 0.01. The rest of the algorithm converts the instance to a maximum independent set instance on a bipartite graph. We use AMD (Automated Mechanism Design Conitzer and Sandholm [2002]) as an alternative implementation. We round the bids down to multiples of 0.01. After this step, there are 100^2 possible bid profiles. For each bid profile, we create 2 continuous variables to represent the payments and 2 binary variables to represent the allocation. We follow the standard AMD process. The expected revenue can be easily expressed as a linear expression of the variables. Strategy-proofness and individual rationality can also easily be expressed as linear inequalities involving the variables. The optimal auction for discrete bids is derived via a mixed-integer program. For 2 bidders, the AMD solution takes only seconds. (This is completely expected. When the underlying problem has a polynomial-time solution, state-of-the-art MIP solvers often manage to solve it fast.) AMD does not scale to 3 or more bidders. The FPTAS is also restricted to 2 bidders. As mentioned in Papadimitriou and Pierrakos [2011], optimal auction design for correlated bidders becomes NP-hard to approximate when $n \geq 3$. For the above reasoning, AMD suffices as an alternative implementation.

MYERSON: We calculate every bidder's ironed virtual valuations based on their *marginal* distributions, instead of their *conditional* distributions. We then allocate the item to the bidder with the highest ironed virtual valuation. The item is not allocated if the highest ironed virtual valuation is below 0. This auction guarantees monotonicity and therefore strategy-proofess, as all it does is to pretend that correlation does not exist and runs the optimal auction for independent bidders. The revenue of this auction is expected to be sub-optimal as it ignores correlation completely.

2ND: Second price auction with the optimal reserve price. The optimal reserve is calculated by simply trying all multiples of 0.001.

D Hand-Crafted Grid Distributions Based on Magic Squares

$$\text{SATURN} = \begin{bmatrix} 4 & 9 & 2 \\ 3 & 5 & 7 \\ 8 & 1 & 6 \end{bmatrix}$$

$$\text{JUPITER} = \begin{bmatrix} 4 & 14 & 15 & 1 \\ 9 & 7 & 6 & 12 \\ 5 & 11 & 10 & 8 \\ 16 & 2 & 3 & 13 \end{bmatrix}$$

$$\text{MARS} = \begin{bmatrix} 11 & 24 & 7 & 20 & 3 \\ 4 & 12 & 25 & 8 & 16 \\ 17 & 5 & 13 & 21 & 9 \\ 10 & 18 & 1 & 14 & 22 \\ 23 & 6 & 19 & 2 & 15 \end{bmatrix}$$

$$\text{SOL} = \begin{bmatrix} 6 & 32 & 3 & 34 & 35 & 1 \\ 7 & 11 & 27 & 28 & 8 & 30 \\ 19 & 14 & 16 & 15 & 23 & 24 \\ 18 & 20 & 22 & 21 & 17 & 13 \\ 25 & 29 & 10 & 9 & 26 & 12 \\ 36 & 5 & 33 & 4 & 2 & 31 \end{bmatrix}$$

$$\text{VENUS} = \begin{bmatrix} 22 & 47 & 16 & 41 & 10 & 35 & 4 \\ 5 & 23 & 48 & 17 & 42 & 11 & 29 \\ 30 & 6 & 24 & 49 & 18 & 36 & 12 \\ 13 & 31 & 7 & 25 & 43 & 19 & 37 \\ 38 & 14 & 32 & 1 & 26 & 44 & 20 \\ 21 & 39 & 8 & 33 & 2 & 27 & 45 \\ 46 & 15 & 40 & 9 & 34 & 3 & 28 \end{bmatrix}$$

$$\text{MERCURY} = \begin{bmatrix} 8 & 58 & 59 & 5 & 4 & 62 & 63 & 1 \\ 49 & 15 & 14 & 52 & 53 & 11 & 10 & 56 \\ 41 & 23 & 22 & 44 & 45 & 19 & 18 & 48 \\ 32 & 34 & 35 & 29 & 28 & 38 & 39 & 25 \\ 40 & 26 & 27 & 37 & 36 & 30 & 31 & 33 \\ 17 & 47 & 46 & 20 & 21 & 43 & 42 & 24 \\ 9 & 55 & 54 & 12 & 13 & 51 & 50 & 16 \\ 64 & 2 & 3 & 61 & 60 & 6 & 7 & 57 \end{bmatrix}$$

$$\text{LUNA} = \begin{bmatrix} 37 & 78 & 29 & 70 & 21 & 62 & 13 & 54 & 5 \\ 6 & 38 & 79 & 30 & 71 & 22 & 63 & 14 & 46 \\ 47 & 7 & 39 & 80 & 31 & 72 & 23 & 55 & 15 \\ 16 & 48 & 8 & 40 & 81 & 32 & 64 & 24 & 56 \\ 57 & 17 & 49 & 9 & 41 & 73 & 33 & 65 & 25 \\ 26 & 58 & 18 & 50 & 1 & 42 & 74 & 34 & 66 \\ 67 & 27 & 59 & 10 & 51 & 2 & 43 & 75 & 35 \\ 36 & 68 & 19 & 60 & 11 & 52 & 3 & 44 & 76 \\ 77 & 28 & 69 & 20 & 61 & 12 & 53 & 4 & 45 \end{bmatrix}$$

The baseline auction MYERSON refers to Myerson's original optimal auction extended to correlated bidders, simply ignoring correlation altogether. This is achieved by pretending that the distribution is independent and using the marginal distribution of a bidder to calculate her ironed virtual valuation. To assess the amount of revenue lost by ignoring correlation, we resort to the *magic squares*. These are 2D matrices whose every row and every column sums up to the same value. Under grid distributions based on magic squares, every bidder's marginal distribution is always $U(0, 1)$. The magic squares listed in this section are from a 1531 book titled *De occulta philosophia*. We experimented on these magic-square-based grid distributions. The revenue gap between MYERSON (optimal if correlation does not exist) and our auction (proven near-optimal based on the upper bound) is the amount of revenue lost by ignoring correlation. For i.i.d. distributions, MYERSON is just the second price auction with a reserve based on the i.i.d. distributions.

E Adversarial Distributions

After experimenting on 47 distributions, including 10 randomly generated grid distributions for n from 2 to 5, and 7 hand-crafted grid distributions based on the magic squares in Appendix D, an unexpected observation is that for all these distributions, a near-optimal verifiably monotone allocation can be found without incorporating counterexample-guided training. The post-processing monotonicity fix step is therefore also not needed. For 2 bidders, we always find a verifiably monotone allocation with one trial. For more bidders, we sometimes do end up with non-monotone allocations, but this can be resolved by repeated trials. We do not need an excessively large number of trials. We used 10 trials in experiments. In the worst situation (5 bidders, distribution $G5_6$), we ended up with a verifiably monotone allocation in 6 out of 10 trials. That is, in expectation, we only need less than 2 trials to obtain a monotone allocation. Furthermore, for all these distributions, we achieve near-optimal revenue.

While the above observation shows the effectiveness of our approach, a natural question to ask is whether these results are artifacts caused by our selection of distributions. To test the limit of our approach, we experimented with two methods for generating adversarial distributions. The first method is based on evolutionary computation. We managed to generate distributions under which the greedy allocation is (probably) "wrong" for more bid profiles, in the sense that its decision is inconsistent with decisions obtained via Automated Mechanism Design. Nevertheless, our approach still delivers near-optimal results with one trial, without any counterexample-guided training. The evolutionary computation based adversarial distributions did increase the revenue gap between the greedy upper bound and the achieved revenue, from maximally 1.3% for non-adversarial distributions to 2.1%. The second method relies on a mixed-integer program to generate adversarial distributions under which the greedy allocation maximally "overestimates" the revenue. This method successfully leads to two adversarial distributions for which we do need counterexample-guided training and monotonicity fix. That is, simply relying on luck (i.e., train again until we reach a verifiably monotone allocation with near-optimal revenue) no longer works. We do need the full suite of techniques (verification, counterexample-guided training and post-processing monotonicity fix) to achieve near-optimal revenue.

E.1 Adversarial Distributions via Evolutionary Computation

Our approach aims to find a monotone interpolation of the greedy allocation. One adversarial situation is when the learning target, i.e., the greedy allocation, makes allocation mistakes for many bid profiles. In this section, we apply evolutionary computation to search for a distribution under which the greedy allocation differs the most from the "correct" allocation (i.e., the optimal allocation).

However, the correct allocation is unknown. We settle for the AMD allocation instead, which we expect to be mostly correct. The main gist of AMD is that we can round the bids down to multiples of $1/H$. Afterwards, the optimal auction for these discrete bids can be derived via mixed-integer programming. Since AMD is only scalable for 2 bidders, we focus on 2 bidders. Going through the H^2 bid profiles, we count the number of times greedy and AMD are inconsistent. This counter is used as the fitness function for evolutionary computation – we want its value to go up. In our experiment, we set H to 20 to speed up the fitness evaluation.

We apply the simple (1+1)EA algorithm. We start with the grid distribution $G5_s$ for s from 0 to 9. We use d to denote the current distribution. In every evolutionary round, for every density value in the 2D matrix representing d, we replace the original density value x by $0.9 \cdot x + u$, where u is drawn from $U[0, 1]$. After mutation, the density values are normalized to ensure that the total probability equals 1. We evaluate the fitness of the mutated distribution d'. If it is better (i.e., more adversarial), then we keep it by replacing d by d'. Otherwise, we throw d' away and go back to d. We do the above for 100 rounds. The resulting distribution is recorded as $EA(G5_s)$.

Despite the effort, under the most adversarial distribution we found via evolutionary computation, greedy and AMD only differ under 3.5% of the bid profiles. This empirically suggests that the greedy allocation is a high quality learning target even for (somewhat) adversarial distributions, justifying our approach. For the evolved adversarial distributions, our neural network interpolation approach still produces near-optimal revenue as shown in Table 3. Among all synthetic, hand-crafted and adversarial distributions we experimented on, $EA(G5_2)$ leads to the worst revenue gap between the

greedy upper bound and the achieved revenue, which is 2.1%. (For non-adversarial distributions, the maximum revenue gap is 1.3%.)

Table 3: 2 bidders, evolutionary computation generated adversarial distributions

DISTRIB.	GREEDY	AMD	MYER.	2ND	RNET	MP
$EA(G5_0)$	0.3935	0.3918	0.3857	0.3869	0.3875	**0.3905**
	± 0.001		± 0.0007	± 0.0008	± 0.0008	± 0.0008
$EA(G5_1)$	0.4143	0.4111	0.4030	0.4010	0.4028	**0.4106**
	± 0.001		± 0.0008	± 0.0008	± 0.0008	± 0.0008
$EA(G5_2)$	0.4311	0.4279	0.4137	0.4116	0.3836	**0.4220**
	± 0.001		± 0.0007	± 0.0007	± 0.001	± 0.0008
$EA(G5_3)$	0.4165	0.4140	0.3999	0.3986	0.4033	**0.4117**
	± 0.001		± 0.0008	± 0.0008	± 0.0008	± 0.0008
$EA(G5_4)$	0.4271	0.4228	0.4152	0.4156	0.4175	**0.4260***
	± 0.001		± 0.0007	± 0.0008	± 0.0008	± 0.0008
$EA(G5_5)$	0.4374	0.4320	0.4277	0.4246	0.4258	**0.4320***
	± 0.001		± 0.0007	± 0.0008	± 0.0008	± 0.0007
$EA(G5_6)$	0.4381	0.4362	0.4256	0.4279	0.4307	**0.4336**
	± 0.001		± 0.0008	± 0.0008	± 0.0008	± 0.0008
$EA(G5_7)$	0.4025	0.3998	0.3912	0.3934	0.3958	**0.4002***
	± 0.001		± 0.0007	± 0.0008	± 0.0008	± 0.0008
$EA(G5_8)$	0.4355	0.4320	0.4211	0.4224	0.4199	**0.4324***
	± 0.001		± 0.0008	± 0.0008	± 0.0008	± 0.0008
$EA(G5_9)$	0.4141	0.4107	0.4068	0.4076	0.4051	**0.4097**
	± 0.001		± 0.0008	± 0.0008	± 0.0008	± 0.0009

E.2 Adversarial Distributions via Mixed-Integer-Programming

We explore another heuristic direction for finding adversarial distributions. We use R^G to denote the greedy revenue, which is generally not attainable as greedy is often not strategy-proof. R^G serves as the revenue upper bound. We use R^* to denote the optimal revenue. We aim to find the distribution that maximizes the ratio $\frac{R^G}{R^*}$. That is, we are searching for distributions under which the greedy allocation significantly overestimates the revenue, which would make greedy an unsuitable learning target.

We still work within the structure of the grid distributions, focusing on 5×5 matrices as illustrated below. Our goal is to search for a set of 25 values for the $p_{i,j}$, so that the corresponding grid distribution is adversarial based on the metric mentioned above.

$$\text{WORST}(\lambda) = \begin{bmatrix} p_{0,4} & p_{1,4} & p_{2,4} & p_{3,4} & p_{4,4} \\ p_{0,3} & p_{1,3} & p_{2,3} & p_{3,3} & p_{4,3} \\ p_{0,2} & p_{1,2} & p_{2,2} & p_{3,2} & p_{4,2} \\ p_{0,1} & p_{1,1} & p_{2,1} & p_{3,1} & p_{4,1} \\ p_{0,0} & p_{1,0} & p_{2,0} & p_{3,0} & p_{4,0} \end{bmatrix}$$

We construct a mixed-integer program to find these 25 values. There are 25 continuous variables, i.e., the $p_{i,j}$, ranging from 1 to λ, where λ is a distribution parameter. We derived two distributions called WORST10 (setting $\lambda = 10$) and WORST100 (setting $\lambda = 100$).

We face two challenges when constructing the mixed-integer program:

The first challenge is that the marginal profit is not linear in the probability densities. To make it linear, we re-interpret the above matrix as a discrete distribution. Every bidder's bid can only be from the following 5 values: $\{0, 0.2, 0.4, 0.6, 0.8\}$. $p_{0,0}$ refers to the probability that both bidders bid 0s. $p_{1,3}$ refers to the probability that bidder 1 bids 0.2 and bidder 2 bids 0.6. Essentially, for the original 2D grid distribution, there are 25 uniform sub-squares, we assign all probability mass to the bottom left corner of the sub-square to make the distribution discrete. By pretending that the distribution is discrete, the marginal profits become linear in the $p_{i,j}$ with the help of axillary binary variables. We use $r^k_{i,j}$ with $k \in \{1, 2\}$ and $0 \leq i, j \leq 4$ to represent the maximum revenue we can extract from

bidder k when bidder 1 bids $i/5$ and bidder 2 bids $j/5$. We have

$$r^1_{i,j} = \max_{i' \geq i} \left(\frac{i'}{5} \sum_{i'' \geq i'} p_{i'',j} \right)$$

$$r^2_{i,j} = \max_{j' \geq j} \left(\frac{j'}{5} \sum_{j'' \geq j'} p_{i,j''} \right)$$

The marginal profit of bidder k when bidder 1 bids $i/5$ and bidder 2 bids $j/5$ is denoted as $m^k_{i,j}$, which equals the following. (This is not the "per density" version. Please refer to Appendix A.)

$$m^1_{i,j} = \max\{r^1_{i,j} - r^1_{i+1,j}, 0\}$$

$$m^2_{i,j} = \max\{r^2_{i,j} - r^2_{i,j+1}, 0\}$$

The greedy revenue R^G is then

$$\sum_{0 \leq i,j \leq 4} \max\{m^1_{i,j}, m^2_{i,j}\}$$

The second challenge is that the optimal revenue is unknown. We settle for an alternative revenue. We define two auctions for discrete bids. Auction 1 allocates the item to bidder 1 if and only if $b_1 \geq b_2$. Auction 2 allocates the item to bidder 1 if and only if $b_1 > b_2$. The only difference between these two auctions is the way they perform tie-breaking. Note that tie-breaking is consequential for discrete distributions. We evaluate the revenue of these two auctions using marginal profit. The revenue under auction 1 is

$$\sum_{0 \leq j \leq i \leq 4} m^1_{i,j} + \sum_{0 \leq i < j \leq 4} m^2_{i,j}$$

The revenue under auction 2 is

$$\sum_{0 \leq j < i \leq 4} m^1_{i,j} + \sum_{0 \leq i \leq j \leq 4} m^2_{i,j}$$

The maximum revenue between these two auctions is denoted as R^S.

Since the revenue of R^S is evaluated in terms of marginal profit, the revenue fix is already automatically included. The revenue fix basically adds a reserve for each bidder and the reserve can depend on the other bid. The optimal reserve is already reflected in the marginal profits.

So far, all values mentioned above can be expressed as linear expressions of the $p_{i,j}$ (with the help of many auxiliary binary variables to represent "max"). We finally add a constraint $R^G \geq \alpha \cdot R^S$, where α is a constant. We search for the largest constant α that still makes the above inequality holds, which can be solved via a mixed-integer program (i.e., via feasibility check). The $p_{i,j}$ values corresponding to the largest α characterize the adversarial distribution. Below are the solutions for $\lambda \in \{10, 100\}$:

$$\text{WORST10} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 10 & 9 & 1 & 10 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 2.59 & 1 \\ 1 & 1 & 1 & 7.77 & 1 \end{bmatrix}$$

$$\text{WORST100} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 100 & 1 & 1 & 1 \\ 1 & 1 & 48.5 & 24.25 & 72.75 \\ 1 & 14.25 & 1 & 1 & 3 \\ 1 & 42.75 & 1 & 1 & 1 \end{bmatrix}$$

For WORST10 and WORST100, our approach still manages to produce near-optimal revenue, as shown in Table 4. However, these two distributions indeed are significantly more challenging to handle. Unlike other distributions where we can completely skip counterexample-guided training and monotonicity fix (by resorting to repeated trials), for WORST10 and WORST100, we do need the full suite of techniques. More detailed case studies on these two distributions are presented in Appendix F.

Table 4: 2 bidders, mixed-integer-programming generated adversarial distributions

DISTRIB.	GREEDY	AMD	MYER.	2ND	RNET	MP
WORST10	0.4401	0.4379	0.3733	0.3919	0.4130	**0.4441***
	$\pm$0.001		$\pm$0.0009	$\pm$0.0006	$\pm$0.0007	$\pm$0.003
WORST100	0.4678	0.4565	0.3885	0.3805	0.4126	**0.4632***
	$\pm$0.001		$\pm$0.0007	$\pm$0.0004	$\pm$0.0006	$\pm$0.003

F Case Study on the Benefit of Counterexample-Guided Training and Monotonicity Fix

To test the necessity and effectiveness of counterexample-guided training and monotonicity fix, we run the following experiment. For each selected distribution, we run 100 training trials to reach statistically meaningful conclusions. Each training trial proceeds as follows:

- Train $20,000$ (optimizer) steps without counterexample-guided training. We call the result at the end of $20,000$ steps the *stage-one* result.
- We verify whether the stage-one allocation is monotone and evaluate its revenue. If the stage-one allocation is already monotone, then we do not need to apply monotonicity fix when evaluating its revenue. Otherwise, the monotonicity fix is used.
- We continue to train another $10,000$ steps. If the stage-one allocation is not monotone, then we switch to counterexample-guided training and use the monotonicity violations from stage-one's evaluation as counterexamples. If the stage-one allocation is already monotone, then we do not need to switch to counterexample-guided training (and we do not have counterexamples anyway). We call the end result the *stage-two* result.
- We verify whether the stage-two allocation is monotone and evaluate its revenue. Similar to the situation for stage-one, we apply monotonicity fix when necessary.

The trials were divided into four categories, which are represented using different colours in Figure 3, Figure 4 and Figure 5:

- Blue: Stage-one and two are both monotone. That is, blue results do not involve counterexample-guided training or monotonicity fix.
- Red: Stage-one and two are both not monotone. Counterexample-guided training and monotonicity fix are both used.
- Green: Stage-one is not monotone and stage-two is monotone. That is, counterexample-guided training managed to guide the allocation to monotonicity. Both counterexample-guided training and monotonicity fix are used.
- Black: Stage-one is monotone but stage-two is not monotone. That is, with additional training, the network learned more details and turned not monotone. Monotonicity fix is used but not counterexample-guided training.

We perform a detailed analysis of the following 3 distributions. As mentioned earlier, in this experiment, we run 100 trails for each distribution. For 3 distributions, we estimate that the total computation time is on the order of 500 hours, so we cannot afford to test every distribution.

- $G5_6$ for 5 bidders: This distribution is selected because it is more difficult to reach monotonicity under $G5_6$ compared to the other randomly generated distributions. When constructing Table 11, Table 12 and Table 13, for every distribution we tried 10 times and recorded how many times the result is monotone. For $G5_6$, 6 out of 10 times are monotone. $6/10$ appears to be a fairly high success rate, but is actually the worst among the randomly generated distributions.
- WORST10 and WORST100 for 2 bidders: These are adversarial distributions designed to test the limit of our approach. They are derived in Appendix E.2.

The experimental results are summarized in Table 5. The individual distribution's reports are given in Figure 3, Figure 4 and Figure 5.

Table 5: Summary of best revenue per category, where 100 trials are divided into 4 categories based on the monotonicity status at the end of stage-one and two: M=Monotonicity; $\overline{M}$=non-monotonicity; $M \to \overline{M}$ means the stage-one result is monotone and the stage-two result is not monotone

DISTRIB.	BLUE $M \to M$	RED $\overline{M} \to \overline{M}$	GREEN $\overline{M} \to M$	BLACK $M \to \overline{M}$
$G5_6$ ($n = 5$)	0.6841 ± 0.0006	0.6551 ± 0.002	0.6847 ± 0.002	0.6844 ± 0.002
WORST10 ($n = 2$)	0.4347 ± 0.0009	0.4441 ± 0.003	0.4410 ± 0.003	0.4370 ± 0.003
WORST100 ($n = 2$)	0.4517 ± 0.0008	0.4632 ± 0.003	0.4516 ± 0.002	0.4544 ± 0.0008

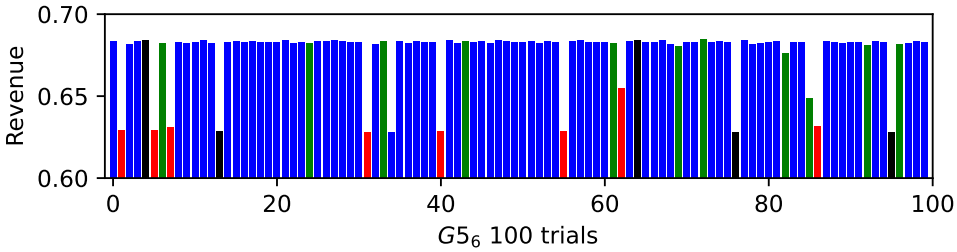


Figure 3: While green and black reached slightly higher highs, the gap between blue and green/black is statistically insignificant considering the standard errors. In conclusion, counterexample-guided training and monotonicity fix are not needed for this distribution.

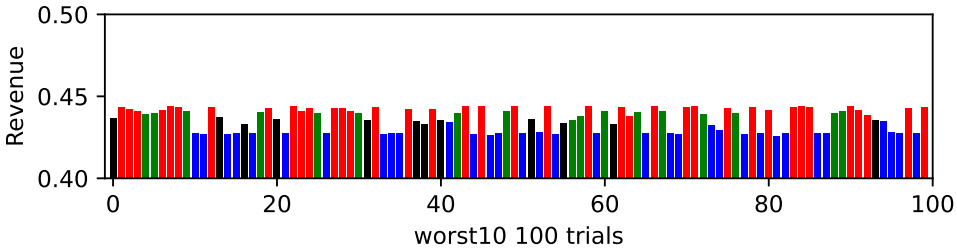


Figure 4: Red and green reached higher, so counterexample-guided training and monotonicity fix do help. We do need the whole suite of techniques to reach near-optimal revenue.

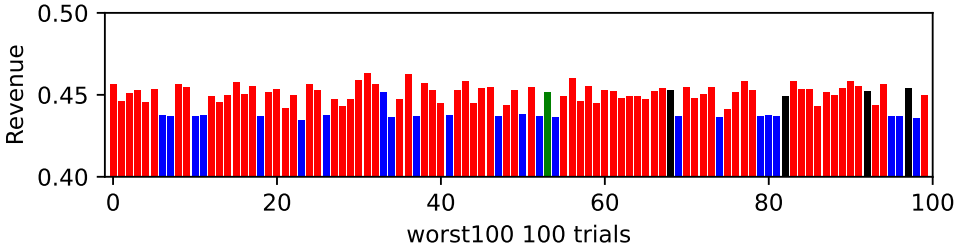


Figure 5: Red reached higher, so counterexample-guided training and monotonicity fix do help. We do need the whole suite of techniques to reach near-optimal revenue.

G Details of RegretNet Implementation

RegretNet Dütting et al. [2019] implements two neural networks, one for allocation and one for payments. In our RegretNet implementation, the two networks have the following properties:

- Each network takes the n bids as inputs.
- Each network is a multilayer perceptron (MLP) with ReLU activation. In our implementation, both networks have 2 intermediate layers each with width 50. This is larger than the networks used in the main approach, which is 2 layers with width 20.

- The allocation network has $n+1$ outputs where the first n outputs correspond to the n bidders and the last output corresponds to no allocation. We take the argmax among the $n+1$ outputs to decide which bidder wins. In training, we take softmax on the outputs.
- The payment network has n outputs where the i-th output decides bidder i's payment if the item is allocated to bidder i.

RegretNet is unsupervised. The training samples are bid profiles drawn according to the given correlated distribution. Unlike the main approach, where we need the distribution, as otherwise we cannot calculate the marginal profits, RegretNet does not need the access to the distribution. Samples are all it needs.

Following RegretNet's notation, we define the set of bidders as N, denote the model parameters as w, denote v_i as the true valuation of bidder i. Bidder i's utility is $u_i^w(v_i;(v_i',v_{-i}))$ when she has valuation v_i and bids v_i'. Bidder i's payment is denoted as $p_i^w(v)$.

The aim of RegretNet is to maximize the expected total payment $\mathbb{E}[\sum_{i\in N} p_i^w(v)]$ while ensuring strategy-proofness as well as ex post individual rationality. During training, we use a batch of L bid profiles to get an unbiased estimation $\frac{1}{L}\sum_{l=1}^{L}\sum_{i\in N} p_i^{w,l}(v)$.

The strategy-proofness constraint is given by minimizing the "Regret" defined as the sum of how profitable the optimal deviation is for every bidder. Formally,

$$rgt_i(w) = \mathbb{E}\left[\max_{v_i'\in V_i} u_i^w(v_i;(v_i',v_{-i})) - u_i^w(v_i;(v_i,v_{-i}))\right].$$

During training, we use a batch of bid profiles to give an unbiased estimation $\widehat{rgt}_i(w)$.

Individual rationality is ensured by taking the sigmoid of the payment network's outputs, so that the range becomes $[0,1]$, then times the bid. This method can ensure that the payment from each bidder is nonnegative and no more than the bid itself, thus satisfying individual rationality.

The loss function used in unsupervised training is as follows:

$$Loss(w;\lambda) = -\frac{1}{L}\sum_{l=1}^{L}\sum_{i\in N} p_i^{w,l}(v) + \sum_{i\in N}\lambda_i \widehat{rgt}_i(w).$$

In the original implementation of RegretNet Dütting et al. [2019], there is also a second order term, which we dropped. The factor λ_i's are initialized to 50 and updated periodically. We set the update interval to be 20 backward steps. The original RegretNet uses a gradient descent method to find the optimal update direction and amount, while in our implementation, we set it higher if $\widehat{rgt}_i(w)$ is larger than 0.0001, and smaller if it is less than it. 0.0001 is the largest regret that we tolerate.

Since the allocation network is using the same network architecture as the main approach, i.e., MLP+ReLU, the existing suite of MIP-based techniques still apply, including verification, counterexample-guided training, and monotonicity fix. After training, the payment network is thrown away and we only evaluate the allocation network. The existing evaluation process from the main approach still applies. For all results on RegretNet listed in this paper, the auction is verified 100% strategy-proof (no counterexample-guided training was needed). For the distributions in Table 10, our RegretNet implementation is able to outperform MYERSON and 2ND, and achieve revenue that is reasonably close to our main approach, noting again that our main approach requires more than just the samples, but also the exact distribution. RegretNet also outperforms MYERSON and 2ND for 8 out of 12 adversarial distributions (Table 3 and Table 4). On the other hand, for the distributions in Table 11, Table 12 and Table 13 (i.e., $n=3,4,5$), the revenue achieved is less ideal. We set the same training budget for our main approach and RegretNet, which is $20,000$ optimizer steps. For 3 or more bidders, perhaps RegretNet requires a lot more training resources. After all, it trains two networks instead of one and the networks are significantly larger. We did not perform excessive hyper-parameter tuning for the RegretNet approach as RegretNet is not the main focus of this paper (mostly used as a baseline). Furthermore, given its larger size, it is a lot more time consuming to work with. For example, for the 10 auctions in Table 13, MIP-based verification takes 21 to 100 minutes.

H Alternative Network Architectures

The concept of monotonicity represented by the min-max networks Sill [1997] is different from the concept of allocation monotonicity in our mechanism design context. For example, in our model, the i-th input of the allocation network is agent i's bid and the i-th output of the allocation network represents the priority value of agent i. Whoever has the highest priority value wins. Allocation monotonicity basically requires the following: if the i-th output is currently the highest, then when we increase the i-th input, we want the i-th output to still be the highest. The i-th output does not have to be monotone with respect to the i-th input – it is fine that an agent's priority value drops when she increases her bid, as long as she still wins (for example, the other agents' priority values may drop more, or the winner's priority value drop is not big enough to go below another agent). It is not clear how to reconcile the difference between monotonicity in terms of values between inputs and outputs and allocation monotonicity. In other words, it is not clear how to use the min-max networks to represent the whole space of strategy-proof allocations.

There are two ways to use min-max networks to represent allocation functions in our setting. The first is MyersonNet Dütting et al. [2019]. Under MyersonNet, agent i's bid b_i is mapped to i's allocation priority value $f_i(b_i)$, where f_i is nondecreasing function represented by a min-max network. Whoever has the highest priority value wins. This representation is strategy-proof, but the following example shows that MyersonNet cannot represent the full space of strategy-proof mechanisms and sometimes it leads to significant revenue loss. In our experiments, we used the fully strategy-proof variant of RegretNet Dütting et al. [2019] as a baseline, which is more expressive than MyersonNet.

Example: We have two bidders. Bidder 2's value is either 0 or ϵ (an infinitesimal value), *i.e.*, almost all revenue will come from bidder 1. When bidder 2's value is 0, bidder 1's conditional distribution is D_0. When bidder 2's value is ϵ, bidder 1's conditional distribution is D_1. That is, bidder 2 serves the purpose of sending a signal that tells us whether bidder 1's distribution is D_0 or D_1. Since almost all revenue comes from bidder 1, a near-optimal auction has the following form: When bidder 2's value is 0, the auction becomes a take-it-or-leave-it auction for bidder 1 with an optimal reserve price derived based on D_0, which we call $\textsc{Reserve}(D_0)$. When bidder 2's value is ϵ, the auction becomes a take-it-or-leave-it auction for bidder 1 with an optimal reserve price derived on D_1, which we call $\textsc{Reserve}(D_1)$. We further assume that $\textsc{Reserve}(D_0) > \textsc{Reserve}(D_1)$. MyersonNet cannot express the above auction. Under MyersonNet, when bidder 2's bid increases, bidder 1's winning price never drops. Basically, under MyersonNet, bidder 1 must face sub-optimal reserve under either D_0 or D_1 (or both), and it is easy to construct distributions where sub-optimal reserves significantly impact the revenue.

Roughgarden and Talgam-Cohen [2013] mentioned the *Lopomo assumption*. Using our terminology, if a bidder's allocation priority value is increasing in her own bid and decreasing in others' bids, then allocation monotonicity is easily satisfied. We can certainly use the min-max networks to model such kind of allocations, but the auction in the above example cannot be expressed. In the above example near-optimal auction, bidder 1, without changing her own bid, can go from losing to winning when bidder 2 increases her bid, which goes against the Lopomo assumption.

We further present experimental results on four additional approaches:

- MYERNET Dütting et al. [2019]: Agent i's bid b_i is mapped to $f_i(b_i)$, where f_i is a nondecreasing function. $f_i(b_i)$ represents agent i's allocation priority value. The agent with the highest priority value wins the item. If all agents' priority values are below 0, then the item is thrown away. The monotonicity of the f_i guarantees strategy-proofness. The f_i are represented using the monotone min-max networks Sill [1997]. We train the f_i using the standard RegretNet Dütting et al. [2019] approach. We simultaneously train $n+1$ networks. Each f_i is a min-max network with 5 "max" groups and each group contains 5 nodes. There is also a shared payment network, which is a MLP with 2 hidden layers and 20 nodes for each layer. (Despite the tiny networks, in practise, it is already fairly expensive to train, with more details given below.) The payment network is only used during training. After training, the payment network is thrown away and the "correct" payments are derived from the (architecturally guaranteed) monotonic allocation function. All other implementation details follow our RegretNet implementation described in Appendix G.
- MINMAX: This is a generalised version of MYERNET. Agent i's priority value $f_i(b_i, b_{-i})$ now also depends on the others' bids. f_i is nondecreasing in b_i and nonincreasing in every

dimension in b_{-i}. This representation also guarantees strategy-proofness. We also use the min-max networks to represent the f_i.[6] Same as MYERNET, each f_i is a min-max network with 5 "max" groups and each group contains 5 nodes. The only difference is that the input dimension of f_i is now n instead of 1. All other settings are the same as MYERNET.

On training difficulty: Both MYERNET and MINMAX are fairly expensive to train. For experiments reported in this subsection, every instance takes about 12 hours on average to train. We conducted our experiments (20 instances) in parallel on a high-performance cluster with Intel 8360Y CPUs. As mentioned earlier, in our experiments, every min-max network contains only 5 groups and each group contains 5 nodes. If we enlarge the network size to 10 groups and 10 nodes each group, then training takes substantially longer. Based on our estimation, *each* instance takes 30+ hours with an Nvidia A100 GPU.

Below we present two modified approaches that are significantly more scalable, by adopting the neural network interpolation idea from our paper.

- MYERNET+: We apply our paper's main approach to improve MYERNET. That is, we apply supervised training to train only the allocation function (i.e., the f_i). The supervision goal is to train the f_i to replicate exactly Myerson's greedy allocation – the agent with the highest (conditional) virtual valuation should win the item and the item is thrown away if all virtual valuations are below 0.

  Each f_i is represented using a min-max network with 10 "max" groups and each group contains 10 nodes. All other implementation details follow our main approach described in Appendix B. Each training instance takes around 35 minutes with an Intel 8360Y CPU.

  There are several reasons why MYERNET+ is significantly faster: 1) there is no need to calculate regret, which is the most time consuming step; 2) we switch from unsupervised training to supervised; 3) we have one less network to train as the payment network is no longer needed.

- MINMAX+: Same as MYERNET+, we apply our paper's main approach to improve MINMAX. Same as the case for MYERNET+, the supervision goal is to train the f_i to replicate exactly Myerson's greedy allocation. Each f_i is also represented using a min-max network with 10 "max" groups and each group contains 10 nodes. Each training instance also takes around 35 minutes with an Intel 8360Y CPU.

Before diving into experimental results, we revisit the earlier example showing that MYERNET/MINMAX style allocations may lead to significant revenue loss. Here we further elaborate on that example using specific numbers. We assume that the bid profile is either $(1, 0)$ or $(0.5, \epsilon)$, each with 50% chance. That is, with 50% chance, bidder 1's value is 1 and bidder 2's value is 0, and with 50% chance, bidder 1's value is 0.5 and bidder 2's value is ϵ, where ϵ is infinitesimal. A near optimal mechanism works as follows: if bidder 2's value is 0, then bidder 1 faces a take-it-or-leave-it offer of 1, and if bidder 2's value is ϵ, then bidder 1 faces a take-it-or-leave-it offer of 0.5. The expected revenue is 0.75. Suppose our allocation follows the style of either MYERNET or MINMAX. We use p_0 to represent bidder 1's critical price for winning when bidder 2 bids 0 and we use p_ϵ to represent bidder 1's critical price for winning when bidder 2 bids ϵ. Since we assume our allocation follows the style of either MYERNET or MINMAX, we must have $p_0 \leq p_\epsilon$. For example, suppose the allocation follows the style of MINMAX, then when bidder 2's bid increases from 0 to ϵ, bidder 1's priority value either stays the same or drops and bidder 2's priority value either stays the same or increases. In order for bidder 1 to still beat bidder 2, bidder 1's minimum winning bid must either stay the same or increase. If $p_\epsilon \leq 0.5$, then the maximum revenue extracted from bidder 1 is then at most $0.5p_0 + 0.5p_\epsilon \leq 0.5$. If $p_\epsilon > 0.5$, then the maximum revenue extracted from bidder 1 is then at most $0.5p_0 \leq 0.5$. Therefore, the maximum revenue extracted from both bidders is at most $0.5 + \epsilon$. Earlier, we showed that the optimal revenue is at least 0.75. That is, by adopting MYERNET or MINMAX, for this example, we lose one third of the revenue.

In Table 6, we present experimental results on randomly generated grid distributions ($G5_0$ to $G5_9$) for 5 bidders. We still use the revenue gap (to the unattainable greedy upper bound) as the performance indicator. Our method MP's revenue gap is 0.87%. The revenue gaps of MYERNET+ and MINMAX+,

[6] By flipping the sign of input dimension i, we can change the output from being nondecreasing to nonincreasing in dimension i.

Table 6: Additional experiments for 5 bidders
GAP(MP) = 0.87%,GAP(MYER.) = 2.4% (note: GAP(2ND) is also 2.4%)
GAP(MYERNET+) = 3.3%,GAP(MYERNET) = 19%
GAP(MINMAX+) = 3.4%,GAP(MINMAX) = 18%

DISTRIB.	GREEDY	MYERNET	MINMAX	MYERNET+	MINMAX+	MYER.	MP
$G5_0$	0.6862	0.5754	0.5135	0.6630	0.6625	0.6695	**0.6799**
	±0.0008	±0.0005	±0.0008	±0.0006	±0.0006	±0.0006	±0.0006
$G5_1$	0.6901	0.5746	0.4503	0.6681	0.6670	0.6737	**0.6846**
	±0.0008	±0.0005	±0.0007	±0.0006	±0.0006	±0.0006	±0.0006
$G5_2$	0.6838	0.5216	0.5688	0.6642	0.6635	0.6709	**0.6783**
	±0.0008	±0.0006	±0.0006	±0.0006	±0.0006	±0.0006	±0.0006
$G5_3$	0.6894	0.6549	0.4469	0.6665	0.6659	0.6724	**0.6838**
	±0.0008	±0.0006	±0.0009	±0.0006	±0.0006	±0.0006	±0.0006
$G5_4$	0.6853	0.5809	0.6270	0.6632	0.6627	0.6692	**0.6787**
	±0.0008	±0.0006	±0.0005	±0.0006	±0.0006	±0.0006	±0.0006
$G5_5$	0.6850	0.6051	0.5904	0.6650	0.6642	0.6717	**0.6809**
	±0.0008	±0.0005	±0.0006	±0.0006	±0.0006	±0.0006	±0.0006
$G5_6$	0.6902	0.6139	0.6369	0.6661	0.6658	0.6710	**0.6839**
	±0.0008	±0.0006	±0.0005	±0.0006	±0.0006	±0.0006	±0.0006
$G5_7$	0.6900	0.6041	0.6161	0.6659	0.6655	0.6719	**0.6824**
	±0.0008	±0.0006	±0.0006	±0.0006	±0.0006	±0.0006	±0.0006
$G5_8$	0.6868	0.2196	0.6210	0.6644	0.6638	0.6707	**0.6807**
	±0.0008	±0.0008	±0.0006	±0.0006	±0.0006	±0.0006	±0.0006
$G5_9$	0.6927	0.6312	0.5969	0.6688	0.6680	0.6752	**0.6866**
	±0.0008	±0.0005	±0.0006	±0.0006	±0.0006	±0.0006	±0.0006

both are based on techniques proposed in our paper, are 3.3% and 3.4%, respectively. For reference, we have also included the revenues of the manual baseline MYER., which is Myerson's greedy allocation where virtual valuations are calculated based on the *marginal* distributions (essentially ignoring correlation altogether). Both MYERNET+ and MINMAX+ do not outperform MYER. Lastly, MYERNET and MINMAX have revenue gaps of 19% and 18%.

I Multi-Unit Auctions with Unit Demand

We extend our techniques to multi-unit auctions with unit demand. We use m to denote the number of items, where $m < n$. Myerson's greedy allocation is extended by assigning one item to each bidder with the m highest marginal profit.

In the single-item setting, our neural network's output dimension is $n+1$. If the i-th ($i = 1, 2, \ldots, n$) output coordinate is the highest, then bidder i wins. If the $(n+1)$-th coordinate is the highest, then the item is not allocated. For the multi-unit setting, the network's output dimension remains $n+1$. Bidder i wins if and only if the i-th output coordinate is among the m highest output coordinates and the i-th output coordinate is higher than the $(n+1)$-th output coordinate. This allows us to represent all possible numbers of items allocated (*i.e.*, from 0 to m).

The extended monotonicity condition requires that any winner i, who is in the original *winner set*, must remains in the winner set when i increases her bid while the other bids stay the same. This more complex version requires us to enumerate all combinations of i, all original winner sets that contain i and all new winner sets that exclude i. The monotonicity verification MIP for the single-item setting can be trivially adapted, as any given winner set can be described by a set of inequalities.

In Table 7, we present our experimental results. The baseline is the $(m+1)$-th price auction with the optimal reserve price. For each distribution, we run 10 trials, and in every trial, we stop after $5,000$, $10,000$, $15,000$ and $20,000$ optimizer steps to perform monotonicity verification. We record the best-performing verifiably monotone allocations.

J Continuous Distributions

We present a few additional experiments on continuous distributions in Table 8. We experimented with three continuous distributions for 2 bidders. We use $f(x, y)$ to denote the joint probability

Table 7: Multi-unit auctions with unit demand with 2 items (i.e., $m = 2$)
M+1 refers to the (m+1)-th price auction with the optimal reserve price, GAP(MP) $= 0.67\%$, GAP(M+1) $= 5.1\%$

	$n = 3$			$n = 4$			$n = 5$		
DISTRIB.	GREEDY	M+1	MP	GREEDY	M+1	MP	GREEDY	M+1	MP
$G5_0$	0.7629	0.7251	**0.7624**	0.9387	0.8923	**0.9360**	1.0856	1.0428	**1.0757**
	±0.002	±0.001	±0.001	±0.002	±0.001	±0.001	±0.002	±0.001	±0.001
$G5_1$	0.7896	0.7413	**0.7901**	0.9646	0.9172	**0.9608**	1.0973	1.0477	**1.0844**
	±0.002	±0.001	±0.001	±0.002	±0.001	±0.001	±0.002	±0.001	±0.001
$G5_2$	0.7311	0.6900	**0.7311**	0.9380	0.8932	**0.9319**	1.0868	1.0394	**1.0755**
	±0.002	±0.001	±0.001	±0.002	±0.001	±0.001	±0.002	±0.001	±0.001
$G5_3$	0.7497	0.7048	**0.7500**	0.9344	0.8853	**0.9293**	1.0930	1.0462	**1.0799**
	±0.002	±0.001	±0.001	±0.002	±0.001	±0.001	±0.002	±0.001	±0.001
$G5_4$	0.7724	0.7280	**0.7733**	0.9457	0.8959	**0.9397**	1.0900	1.0456	**1.0737**
	±0.002	±0.001	±0.001	±0.002	±0.001	±0.001	±0.002	±0.001	±0.001
$G5_5$	0.7511	0.6956	**0.7473**	0.9329	0.8841	**0.9266**	1.0938	1.0463	**1.0748**
	±0.002	±0.001	±0.001	±0.002	±0.001	±0.001	±0.002	±0.001	±0.001
$G5_6$	0.7400	0.6911	**0.7401**	0.9507	0.8964	**0.9449**	1.0965	1.0474	**1.0852**
	±0.002	±0.001	±0.001	±0.002	±0.001	±0.001	±0.002	±0.001	±0.001
$G5_7$	0.8122	0.7734	**0.8096**	0.9663	0.9201	**0.9643**	1.0945	1.0498	**1.0825**
	±0.002	±0.001	±0.001	±0.002	±0.001	±0.001	±0.002	±0.001	±0.001
$G5_8$	0.7531	0.6924	**0.7525**	0.9390	0.8936	**0.9349**	1.0925	1.0462	**1.0785**
	±0.002	±0.001	±0.001	±0.002	±0.001	±0.001	±0.002	±0.001	±0.001
$G5_9$	0.7949	0.7329	**0.7934**	0.9682	0.9130	**0.9604**	1.1016	1.0523	**1.0894**
	±0.002	±0.001	±0.001	±0.002	±0.001	±0.001	±0.002	±0.001	±0.001

density function. (We need to apply a normalisation factor α to ensure that $\int_0^1 \int_0^1 \alpha f(x,y)dxdy = 1$. We ignore α for presentation purpose.) Our method MP is near optimal. *Its gap to greedy is only on average* 0.02%*, also noting that the greedy revenue is not attainable.* On the other hand, 2ND (second price auction with optimal reserve) also performs quite well for all three cases.

Table 8: Experiments on three continuous distributions
GAP(MP) $= 0.02\%$,GAP(2ND) $= 0.5\%$

DISTRIB.	GREEDY	2ND	MP
$f(x,y) = 2x + y$	0.5095 ± 0.001	0.5079 ± 0.0008	**0.5094** ± 0.0008
$f(x,y) = 2x - y + 1$	0.4518 ± 0.001	0.4469 ± 0.0008	**0.4517** ± 0.0008
$f(x,y) = sin(x) + cos(y) + 2$	0.4216 ± 0.001	0.4211 ± 0.0008	**0.4215** ± 0.0008

K Distributions Based on Real-World Datasets

In Table 9, we evaluate our mechanism using empirical bid distributions derived from three real-world datasets. These datasets come from diverse real-world market domains: (i) **Consumer (eBay)**, involving **layperson bidding** for electronics; (ii) **Financial (Ethereum)**, involving **algorithmic bidding** for block building rights; and (iii) **Electricity Market (PJM)**, involving **professional strategic bids** from energy producers.

- **Consumer (eBay):** This dataset consists of historical eBay auction data for second-hand Palm PDAs, a market typically characterized by high valuation correlation, representing a *layperson bidding* environment. From a total of 194 auctions sourced via `modelingonlineauctions.com`, we identify a frequently co-appearing bidder pair (IDs CHIMAM and DIDOJAN) who participated in the same 13 auctions. These 13 joint appearances provide 13 bid profile samples, using their respective maximum bids in each instance. The bids are normalized to the $[0, 1]$ interval (relative to a \$200 maximum) and scattered into a 100×100 grid distribution.
- **Financial (Ethereum):** This dataset represents high-frequency auctions for block-building rights, featuring intense competition between sophisticated *algorithmic bidders*. On May

1st, 2026, a total of 7,117 auctions were recorded, with the top two winners securing 1,200 and 976 auctions, respectively (their public keys end with 48E681 and 8EAB08). Notably, these two bidders co-appeared in 5,711 auctions. While this environment is highly contested, the shared nature of the financial rewards leads to a setting where second-price mechanisms already perform quite well, which is also confirmed in our experiments. We collect joint bid profiles from these two frequent winners, using their maximum bids in co-appearing auctions as samples. Bids exceeding 0.05 ETH are discarded. The remaining samples are normalized to $[0, 1]$ and scattered into the 100×100 grid distribution. Data is sourced from `relayscan.io`.

- **Electricity Market (PJM):** Hourly wholesale auctions managed by the PJM Interconnection (a Regional Transmission Organization coordinating the largest competitive electricity market in the US) between Dec 1st 2025 and Jan 1st 2026. These auctions determine the *generation rights of MWh for specific delivery hours*, featuring strategic bids from professionals on behalf of power producers. We extract joint bid profiles from the most frequently co-appearing bidder pair, filtering out "static" bidders who submit constant bids. The identified bidder pair (anonymized ID ending with YMJEWNA and AYMJEW) co-appear in 744 auctions, one bids between \$25.64 to \$72.45 and the other bids between \$77.75 to \$495.17. We take the bidders' maximum hourly supply offers as bid samples. We normalize the bids to $[0, 1]$ (maximum bid is \$500) and scatter them into the 100×100 grid distribution. Data is retrieved from the PJM Data Miner 2 repository `dataminer2.pjm.com`.

Table 9: Experiments on Distributions Based on Real-World Datasets:
$\text{GAP}(\text{MP}) = 2.0\%$,$\text{GAP}(\text{2ND}) = 22\%$,$\text{GAP}(\text{MYER.}) = 30\%$

DISTRIB.	GREEDY	2ND	MYER.	MP
Consumer (eBay)	0.5445 ± 0.0006	0.4211 ± 0.0001	0.3568 ± 0.0002	**0.5373** ± 0.0006
Financial (Ethereum)	0.3040 ± 0.0007	0.2805 ± 0.0007	0.2598 ± 0.0008	**0.3022** ± 0.0007
Electric Market (PJM)	0.2508 ± 0.0005	0.1578 ± 0.0001	0.1575 ± 0.0001	**0.2375** ± 0.0004

The empirical results in Table 9 demonstrate that our technique significantly outperforms manual baselines. The following analysis explains why these baselines fail.

- Failure of the second-price auction with the optimal reserve (2ND): For eBay and PJM, there is a substantial distance between the highest and second-highest bids. Because the second bid is frequently much lower than the first (with gap of 35% and 68%), the second price auction essentially functions as a single-bidder auction with a reserve, failing to extract the full value from the winning bidder.
- Failure of Myerson's technique based on marginal distributions (MYER.): For eBay, there is a strong negative correlation (Pearson correlation coefficient -0.87) between the two bidders, while for PJM, there is a strong positive correlation (coefficient 0.85) between the two bidders. MYER. fails to adapt to the correlation patterns.

In contrast, our technique achieves near-optimal revenue for all three datasets.

L Complete Tables Including Standard Errors

Lastly, in Table 10 ($n = 2$), 11 ($n = 3$), 12 ($n = 4$), 13 ($n = 5$), we present the complete experimental results including standard errors.

Table 10: 2 bidders (with standard errors)
GAP(MP) $= 0.26\%$,GAP(RNET) $= 2.7\%$,GAP(2ND) $= 4.2\%$,GAP(MYER) $= 4.3\%$

DISTRIB.	GREEDY	AMD	MYER.	2ND	RNET	MP	VV	IVV
$G5_0$	0.4353	0.4368	0.4191	0.4197	0.4307	**0.4368**	0.4355	0.4329
	±0.001		±0.0008	±0.0009	±0.0009	±0.0009	±0.0009	±0.0009
$G5_1$	0.4047	0.4041	0.3939	0.3929	0.4021	**0.4037**	0.4032	0.4032
	±0.001		±0.0007	±0.0006	±0.0007	±0.0007	±0.0007	±0.0007
$G5_2$	0.3979	0.3967	0.3771	0.3823	0.3869	**0.3959**	0.3946	0.3955
	±0.001		±0.0009	±0.001	±0.0009	±0.001	±0.001	±0.001
$G5_3$	0.4629	0.4625	0.4533	0.4546	0.4522	**0.4628***	0.4612	0.4616
	±0.001		±0.0006	±0.0006	±0.0007	±0.0006	±0.0007	±0.0007
$G5_4$	0.4681	0.4665	0.4476	0.4472	0.4639	**0.4674***	0.4673	0.4671
	±0.001		±0.0009	±0.0009	±0.0008	±0.0008	±0.0008	±0.0008
$G5_5$	0.4204	0.4187	0.4075	0.4057	0.4062	**0.4151**	0.4148	0.4150
	±0.001		±0.0008	±0.0008	±0.0009	±0.0009	±0.001	±0.001
$G5_6$	0.3905	0.3912	0.3761	0.3741	0.3798	**0.3908**	0.3893	0.3886
	±0.001		±0.0007	±0.0008	±0.0008	±0.0007	±0.0008	±0.0008
$G5_7$	0.4865	0.4854	0.4494	0.4544	0.4771	**0.4850**	0.4828	0.4840
	±0.001		±0.0006	±0.0008	±0.0007	±0.0008	±0.0008	±0.0008
$G5_8$	0.4298	0.4273	0.3988	0.3981	0.4077	**0.4291***	0.4281	0.4279
	±0.001		±0.001	±0.0009	±0.0009	±0.0008	±0.0009	±0.0009
$G5_9$	0.4531	0.4512	0.4306	0.4302	0.4379	**0.4498**	0.4482	0.4497
	±0.001		±0.0009	±0.0009	±0.0009	±0.0009	±0.001	±0.0009
SATURN	0.4533	0.4545	0.4210	0.4226	0.4372	0.4510	0.4512	**0.4515**
	±0.001		±0.0008	±0.0008	±0.0008	±0.0008	±0.0008	±0.0008
JUPITER	0.4427	0.4404	0.4238	0.4219	0.4316	**0.4418***	0.4409	0.4410
	±0.001		±0.0008	±0.0009	±0.0008	±0.0009	±0.0009	±0.0009
MARS	0.4375	0.4377	0.4177	0.4167	0.4194	**0.4360**	0.4333	0.4359
	±0.001		±0.0008	±0.0008	±0.0008	±0.0009	±0.0009	±0.0009
SOL	0.4429	0.4414	0.4376	0.4364	0.4329	**0.4427***	0.4404	0.4419
	±0.001		±0.0008	±0.0008	±0.0009	±0.0008	±0.0008	±0.0008
VENUS	0.4381	0.4381	0.4175	0.4171	0.4233	**0.4365**	0.4319	0.4358
	±0.001		±0.0008	±0.0008	±0.0008	±0.0008	±0.0009	±0.0008
MERCURY	0.4273	0.4274	0.4179	0.4176	0.4150	**0.4281***	0.4271	0.4279
	±0.001		±0.0008	±0.0008	±0.0008	±0.0008	±0.0008	±0.0008
LUNA	0.4331	0.4335	0.4173	0.4172	0.4212	**0.4322**	0.4315	0.4318
	±0.001		±0.0008	±0.0008	±0.0008	±0.0008	±0.0009	±0.0009

Table 11: 3 bidders (with standard errors)
GAP(MP) $= 0.36\%$, GAP(RNET) $= 3.6\%$, GAP(2ND) $= 3.9\%$, GAP(MYER) $= 3.8\%$

DISTRIB.	GREEDY	MYER.	2ND	RNET	MP
$G5_0$	0.5512±0.001	0.5346±0.0007	0.5338±0.0008	0.5346±0.0007	**0.5517**±0.0007
$G5_1$	0.5647±0.0009	0.5467±0.0007	0.5475±0.0007	0.5509±0.0007	**0.5651**±0.0007
$G5_2$	0.5316±0.001	0.5137±0.0007	0.5118±0.0008	0.5153±0.0008	**0.5302**±0.0008
$G5_3$	0.5476±0.001	0.5283±0.0007	0.5282±0.0007	0.5290±0.0007	**0.5450**±0.0007
$G5_4$	0.5485±0.0009	0.5297±0.0007	0.5306±0.0007	0.5310±0.0007	**0.5471**±0.0007
$G5_5$	0.5477±0.001	0.5237±0.0007	0.5231±0.0007	0.5255±0.0007	**0.5443**±0.0008
$G5_6$	0.5371±0.001	0.5155±0.0008	0.5138±0.0008	0.5158±0.0007	**0.5332**±0.0007
$G5_7$	0.5723±0.0009	0.5527±0.0007	0.5528±0.0007	0.5555±0.0006	**0.5710**±0.0007
$G5_8$	0.5473±0.001	0.5186±0.0008	0.5193±0.0007	0.5185±0.0007	**0.5429**±0.0008
$G5_9$	0.5663±0.001	0.5409±0.0007	0.5410±0.0008	0.5396±0.0007	**0.5637**±0.0008

Table 12: 4 bidders (with standard errors)
GAP(MP) = 0.78%, GAP(RNET) = 3.5%, GAP(2ND) = 3.2%, GAP(MYER) = 3.2%

DISTRIB.	GREEDY	MYER.	2ND	RNET	MP
$G5_0$	0.6270±0.0009	0.6086±0.0007	0.6082±0.0007	0.6072±0.0006	**0.6242**±0.0007
$G5_1$	0.6417±0.0009	0.6215±0.0006	0.6205±0.0007	0.6192±0.0006	**0.6379**±0.0007
$G5_2$	0.6250±0.0009	0.6096±0.0007	0.6101±0.0007	0.6072±0.0006	**0.6205**±0.0007
$G5_3$	0.6266±0.0009	0.6054±0.0006	0.6055±0.0006	0.6034±0.0006	**0.6220**±0.0007
$G5_4$	0.6296±0.0009	0.6084±0.0006	0.6077±0.0007	0.6076±0.0006	**0.6233**±0.0007
$G5_5$	0.6213±0.0009	0.6029±0.0007	0.6028±0.0007	0.6020±0.0006	**0.6158**±0.0007
$G5_6$	0.6347±0.0009	0.6099±0.0007	0.6085±0.0007	0.6051±0.0006	**0.6287**±0.0007
$G5_7$	0.6427±0.0009	0.6210±0.0006	0.6234±0.0006	0.6193±0.0006	**0.6369**±0.0006
$G5_8$	0.6286±0.0009	0.6100±0.0007	0.6091±0.0007	0.6085±0.0007	**0.6241**±0.0007
$G5_9$	0.6411±0.0009	0.6199±0.0006	0.6201±0.0006	0.6190±0.0006	**0.6357**±0.0007

Table 13: 5 bidders (with standard errors)
GAP(MP) = 0.87%, GAP(RNET) = 2.8%, GAP(2ND) = 2.4%, GAP(MYER) = 2.4%

DISTRIB.	GREEDY	MYER.	2ND	RNET	MP
$G5_0$	0.6862±0.0008	0.6695±0.0006	0.6700±0.0006	0.6670±0.0006	**0.6799**±0.0006
$G5_1$	0.6901±0.0008	0.6737±0.0006	0.6738±0.0006	0.6713±0.0005	**0.6846**±0.0006
$G5_2$	0.6838±0.0008	0.6709±0.0006	0.6691±0.0006	0.6681±0.0005	**0.6783**±0.0006
$G5_3$	0.6894±0.0008	0.6724±0.0006	0.6721±0.0006	0.6684±0.0005	**0.6838**±0.0006
$G5_4$	0.6853±0.0008	0.6692±0.0006	0.6682±0.0006	0.6664±0.0006	**0.6787**±0.0006
$G5_5$	0.6850±0.0008	0.6717±0.0006	0.6720±0.0006	0.6685±0.0006	**0.6809**±0.0006
$G5_6$	0.6902±0.0008	0.6710±0.0006	0.6725±0.0006	0.6686±0.0006	**0.6839**±0.0006
$G5_7$	0.6900±0.0008	0.6719±0.0006	0.6715±0.0006	0.6682±0.0005	**0.6824**±0.0006
$G5_8$	0.6868±0.0008	0.6707±0.0006	0.6715±0.0006	0.6659±0.0006	**0.6807**±0.0006
$G5_9$	0.6927±0.0008	0.6752±0.0006	0.6768±0.0006	0.6711±0.0005	**0.6866**±0.0006

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: I confirm that the abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Throughout the paper, we clearly state that the empirical results are regarding the distributions that we experimented on. We are capable of achieving near-optimal revenue for all of these distributions. We described two techniques for generating adversarial distributions. In one approach, we use evolutionary computation to find distributions under which the greedy allocation and the allocation derived via Automated Mechanism Design (which is optimal if we restrict to discrete bids) differ the most. In another approach, we use a mixed integer programming heuristic to generate adversarial distributions under which greedy's estimation of revenue is maximally off. For all adversarial distributions, we still manage to achieve near-optimal revenue. The limitation of our work is that the effectiveness of our approach depends on the distribution. We try very hard to find adversarial cases, but to no avail. Of course, this does not mean that our approach is capable of achieving near-optimal revenue for every distribution (and we are not making this claim).

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.

- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: NA

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Code is provided, as well as training parameters, evaluation details, and hardware.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code is provided.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Training and evaluation details are provided. Code is also provided.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The full tables in the appendix include the standard errors.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Hardware and time of execution are described in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: I have carefully reviewed the NeurIPS Code of Ethics and confirm the paper conforms to it in every respect. This paper is studying a theoretical model.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

Justification: This paper solves a theoretical auction design model via neural networks

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The real-world datasets used in Appendix K are publicly available, and their sources are credited there.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.

- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLM is only used for very occasional sentence restructuring and word selection.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.