ABSTRACT

Title of dissertation: LEARNING AND ROBUSTNESS WITH APPLICATIONS TO MECHANISM DESIGN

Michael J. Curry, Doctor of Philosophy, 2022

Dissertation directed by: Professor John P. Dickerson
Department of Computer Science

The design of economic mechanisms, especially auctions, is an increasingly important part of the modern economy. A particularly important property for a mechanism is strategyproofness – the mechanism must be robust to strategic manipulations so that the participants in the mechanism have no incentive to lie. Yet in the important case when the mechanism designer's goal is to maximize their own revenue, the design of optimal strategyproof mechanisms has proved immensely difficult, with very little progress after decades of research.

Recently, to escape this impasse, a number of works have parameterized auction mechanisms as deep neural networks, and used gradient descent to successfully learn approximately optimal and approximately strategyproof mechanisms. We present several improvements on these techniques.

When an auction mechanism is represented as a neural network mapping bids from outcomes, strategyproofness can be thought of as a type of adversarial robustness. Making this connection explicit, we design a modified architecture for learning auctions which is amenable to integer-programming-based certification techniques from the adversarial robustness literature. Existing baselines are empirically strategyproof, but with no way to be certain how strong that guarantee really is. By contrast, we are able to provide perfectly tight bounds on the degree to which strategyproofness is violated at any given point.

Existing neural networks for auctions learn to maximize revenue subject to strategyproofness. Yet in many auctions, fairness is also an important concern – in particular, fairness with respect to

the items in the auction, which may represent, for instance, ad impressions for different protected demographic groups. With our new architecture, ProportionNet, we impose fairness constraints in addition to the strategyproofness constraints, and find approximately fair, approximately optimal mechanisms which outperform baselines. With PreferenceNet, we extend this approach to notions of fairness that are learned from possibly vague human preferences.

Existing network architectures can represent additive and unit-demand auctions, but are unable to imposing more complex exactly-k constraints on the allocations made to the bidders. By using the Sinkhorn algorithm to add differentiable matching constraints, we produce a network which can represent valid allocations in such settings.

Finally, we present a new auction architecture which is a differentiable version of affine maximizer auctions, modified to offer lotteries in order to potentially increase revenue. This architecture is always perfectly strategyproof (avoiding the Lagrangian-based constrained optimization of RegretNet) – to achieve this goal, however, we need to accept that we cannot in general represent the optimal auction.

LEARNING AND ROBUSTNESS WITH APPLICATIONS TO MECHANISM DESIGN

by

Michael J. Curry

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2022

Acknowledgments

There are many people who helped me to complete this dissertation – I can't possibly name them all. I'd like to particularly thank my many collaborators and coauthors for their invaluable help and research discussion – particularly Ping-yeh Chiang for being such a great collaborator during the long months of COVID and after. And my officemates Sahil, Alex, and Phil for constantly distracting me from work with interesting questions. It of course would not have been possible without my advisors and mentors John and Tom. Outside of the academic life, I have to thank all the Buckley family – Anne, Gretta, Mary, Keith, Uncle Jerry and Aunt Debbie – for inviting me to so many wonderful dinners in DC and Baltimore. And most importantly, my parents, for their unwavering love and support.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

Parts of the introduction were adapted from work described in other chapters, so coauthors credited there also deserve some credit.

Auctions have been held for millennia, since at least Classical antiquity [103], and have played an important complementary role to set-price sales and bargaining to exchange goods. In recent decades, the advent of computation has resulted in a surge of large-scale fielded auctions in a variety of important industries, such as broadcasting [107], advertising [59], electricity markets [47], and many others [121, 140]. Auctions are not only of theoretical interest, but also of great practical importance.

Auction *design* is the problem faced by an auctioneer who, given uncertain knowledge of the demands of auction participants, wishes to set the rules of the auction to ensure, via proper incentive structure, a desirable outcome. Indeed, in both 2020 and 2007 the Nobel Prize for Economics was awarded for work in auction design. In the usual theoretical model of auctions [127], the bidders have some private valuation of the items, and the distribution from which these valuations are drawn is common knowledge. The auctioneer solicits bids from participants, awards the items up for sale to the winners, and charges some amount of money to each. Bidders may, however, choose to strategically lie about their valuation given their knowledge of the auction rules and anticipated behavior of other participants, resulting in a Bayes-Nash equilibrium which may be very hard for the auction designer to predict.

To avoid this problem, an auctioneer may simply wish to design a strategyproof (or truthful) mechanism in which participants are incentivized to truthfully reveal their valuations. The distribution of bids is then simply the valuation distribution itself, so it becomes easy to predict the results of the auction in expectation. While satisfying the strategyproofness constraint, the auctioneer can additionally optimize total utility, their own revenue, or other desirable properties.

If the auctioneer's goal is to maximize the welfare of all participants while maintaining strategyproofness, the Vickrey-Clarke-Groves (VCG) mechanism always gives a solution [41, 83, 160]. By contrast, if the auctioneer wishes to maximize revenue, strategyproof mechanisms are much harder to find. Some revenue-maximizing strategyproof mechanisms are known in limited cases: when selling a single item, the Myerson auction is strategyproof and maximizes revenue [123], and for selling multiple items to one agent, some results are known [54, 98, 115, 129]. But even for the simple case of selling *two* items to *two* agents, outside of the special case of binary distributions [167], the strategyproof revenue-maximizing mechanism is not known.

The theoretical difficulty of devising revenue-maximizing strategyproof auctions has resulted in a number of attempts to approximate them. Recently, [58] presented a method, RegretNet, for learning approximately incentive compatible mechanisms given samples from the bidder valuations. They parameterize the auction as a neural network, and learn to maximize revenue, and maintain strategyproofness, by gradient descent. This learning approach to auction design can replicate some known-optimal auctions, and can learn good auctions in settings where the optimal auction is not known. It has been extended in a variety of ways [50, 68, 77, 104, 132, 156].

1.1 The Private-Value Auction Setting

An auction involves a set of agents $N = \{1,\ldots,n\}$ bidding for items $M = \{1,\ldots,m\}$. Each agent $i \in N$ has a corresponding private valuation v_i, presumed to be drawn at random from a publicly known distribution V_i of valuations.

We denote a profile of the n valuations as $v = (v_1, \ldots v_n)$. In general these v_i may be functions over the power set 2^M, but in this work we focus on bidders with simpler valuations. With **additive** valuations, an agent's valuation for a subset of items is $v_i(S) = \sum_{j\in S} v_i(\{j\})$, the sum of the individual items' valuations. With **unit-demand** valuations, the value of a subset is $v_i(S) = \max_{j\in S} v_i(\{j\})$, the maximum individual valuation within that subset. Under some assumptions (that valuations would be weakly positive for receiving an item), unit-demand valuations can be thought of as additive valuations, where the allocation is constrained to offer at most 1 item to any given bidder. In 5 we also consider other valuation types, such as the case where each bidder must receive exactly k items. In all of these cases, the items are not complements or substitutes, so we can represent the valuation as a vector with one entry per item.

Given their private valuation v_i, each agent reports a bid vector b_i, which may differ from v_i, to the auctioneer.

Finally, based on the profile of bids $b = (b_1, \dots, b_n)$, the auction determines an outcome using the allocation and payment rules $g(b) : \mathbb{R}^{mn} \to [0,1]^{nm}$ and $p(b) : \mathbb{R}^{mn} \to \mathbb{R}^n$. We will refer to the matrix of allocation probabilities, whose rows must sum to 1, as $g(b) = z$, and the allocation probability $g(b)_{i,j}$ of item j to agent i as $z_{i,j}$. Likewise agent i's value of the jth item is $v_{i,j}$, and bidder i has payment function p_i. Moreover, for unit-demand auctions, we restrict the allocation to allow each bidder to win, in expectation, at most 1 item.

The utility for bidder i when bidding b_i with true valuation v_i, and opponent bid/valuation profile v_{-i} can be represented in linear form as $u_i(b_i, v_i, v_{-i}) = \sum_j v_{i,j} g_{i,j}(b_i, v_{-i}) - p_i(b_i, v_{-i})$.

1.1.1 Desirable Auction Properties

A mechanism is **individually rational** (IR) when an agent bidding truthfully is guaranteed non-negative utility: $u_i(v_i, v_i, v_{-i}) \geq 0 \ \forall i \in N, v \in \text{supp}(V)$. It is **dominant-strategy incentive-compatible** (DSIC) or **strategyproof** if every agent maximizes their own utility by bidding truthfully, regardless of the other agents' bids. To define this notion formally, we first define the notion of **regret**, the difference in utility between the bid player i actually made and the best possible strategic bid:

$$\text{rgt}_i(v) = \max_{b_i} u_i(b_i, v_i, v_{-i}) - u_i(v_i, v_i, v_{-i}) \tag{1.1}$$

An auction is DSIC when regret (for a truthful bid) is always zero under every valuation profile for every player – they have no incentive to do anything other than tell the truth.

In addition to satisfying the IR and DSIC constraints, the auctioneer seeks to **maximize their expected revenue**. If the auction is truly DSIC, we assume players will bid truthfully, and as a result revenue is simply $E_{v\sim V}\left[\sum_{i\in N} p_i(v)\right]$.

1.2 Deep Learning for Auctions

Because deriving analytic solutions to mechanism design problems has been so difficult, another trend within the research community has been to approximate mechanisms by formulating the mechanism design problem as a learning problem. Recently, Dütting et al. published work on RegretNet, a neural network architecture that models an auction mechanism [58] – this work has been extended and applied in other areas [68, 77, 152], and complements other work treating various aspects of mechanism design as a learning problem [24, 164, 169].

In this framework, the allocation and payment functions $(g(b), p(b))$ are represented as neural networks $(g^w(b), p^w(b))$ where w is the set of learned weights. These networks are standard feedforward networks. The allocation networks g end with a softmax layer, to ensure that allocations are valid categorical distributions (additionally in the unit-demand setting, that each player is allocated a single item). The payment network ends with a sigmoid layer, outputing a value $\tilde{p}_i$ in $[0,1]$ for each bidder; the final payment $p_i = \left(\sum_j g_{i,j} v_{i,j}\right)\tilde{p}_i$. This ensures individual rationality cannot be violated.

The training data for RegretNet is a dataset of L bid profiles sampled from the valuation distribution V; these are used for training by standard gradient descent to maximize mean payment. To enforce strategyproofness, the authors of RegretNet relax the notion of strict dominant-strategy incentive compatibility to a slightly weaker notion of expected regret: $\mathbb{E}_{v\sim V}\left[\sum_i \mathrm{rgt}_i(v)\right]$ – note if this is exactly zero, then the mechanism is truly DSIC.

To estimate the regret under a specific valuation, the authors of [58] perform gradient ascent on the network inputs to find a nontruthful bid that maximizes player utility – this is a quantity they call $\widehat{\mathrm{rgt}}_i$.

To enforce the regret constraint, RegretNet uses the augmented Lagrangian method and incorporates into their loss a set of Lagrange multipliers $\lambda = \{\lambda_1, ..., \lambda_n\}$ and a quadratic parameter ρ.

$$\mathscr{C}_\rho(w;\lambda) = -\frac{1}{L}\sum_{\ell=1}^{L}\sum_{i\in N} p_i^w(v^{(\ell)}) + \sum_{i\in N}\lambda_i \widehat{rgt}_i(w) + \frac{\rho}{2}\sum_{i\in N}\widehat{rgt}_i(w)^2, \tag{1.2}$$

$v^{(\ell)}, 1 \leq \ell \leq L$, denotes the ℓth training sample. The training procedure involves alternating gradient steps to solve $\min_w \max_\lambda C_\rho(w;\lambda)$, as well as gradient ascent steps at each iteration to approximate $\widehat{\mathrm{rgt}}_i$.

1.3 Limitations of RegretNet

RegretNet and its successors work very well for learning good mechanisms in challenging multi-agent multi-item settings, but they have important limitations.

One limitation is that the resulting mechanisms are extremely hard to interpret – they are just opaque neural networks. It is unclear how to satisfactorily explain the behavior of these learned mechanisms to human mechanism designers or to bidders, however good they might be.

The biggest limitation, though, is that the learned mechanisms, while empirically close to strategyproof, are not exactly strategyproof. Moreover, there are not even guarantees on the degree to which they violate strategyproofness. There may be points lurking in the support of the valuation distribution where regret is very high, but that were unseen during training or testing. Even worse, Dütting et al. [58] and its successors do not even provide a way to accurately measure regret at a single point – their gradient-based heuristic is in fact effective but might underestimate regret to an arbitrary extent. Dütting et al. [58] and its successors give experimental evidence that the learned mechanisms are in fact quite good (it is very hard to find high-regret points) – but these issues mean that one can never be completely confident.

1.4 Overview of Research

In this thesis, we present several extensions and improvements on RegretNet-style techniques. In some chapters, we aim to expand the scope of the differentiable economics approach to new mechanism design desiderata; in others, we aim to mitigate the aforementioned deficiencies.

In Chapter 2, we discuss a version of the RegretNet architecture which allows for exact computation of regret – not just the gradient-based approximation discussed above, which we find may underestimate the true extent to which incentive compatibility is violated.

In Chapter 3, we discuss an extension of RegretNet to learn auctions that satisfy certain fairness constraints. Crucially, these are fairness constraints not with respect to the mechanism participants, but rather with respect to the items. This is motivated by cases of ad auctions where the "items" up for sale may be ad impressions for certain demographic groups who must be treated fairly.

In Chapter 4, we discuss a further extension to this in which the fairness requirements are learned from non-explicit human preferences – for example, in surveys comparing preferred outcomes.

In Chapter 5, we discuss an altered RegretNet architecture which uses the Sinkhorn algorithm to impose differentiable bipartite matching constraints on the allocation outputs – in other words, each row or column of the matrix must sum to a specified value. This allows the imposition of "exactly-k" constraints for the agents.

In Chapter 6, we discuss a new differentiable auction architecture which is always perfectly strategyproof – a variant of affine maximizer auctions, modified to offer lotteries. These mechanisms also have the advantage of being somewhat easier to interpret and explain than RegretNet-style approaches. To get these nice properties, we have to give up on universal approximation – in general, the optimal auction is not necessarily an affine maximizer.

In Chapter 7, we discuss other possible future work in several areas.

Chapter 2: Certifying Strategyproof Auction Networks

Joint work with Michael Curry, Ping-Yeh Chiang, Tom Goldstein, John Dickerson. Appeared in NeurIPS 2020.

2.1 Introduction

Auctions are an important mechanism for allocating scarce goods, and drive billions of dollars of revenue annually in online advertising [59], sourcing [146], spectrum auctions [48, 107], and myriad other verticals [121, 140]. In the typical auction setting, agents who wish to bid on one or more items are presumed to have private valuations of the items, which are drawn at random from some prior valuation distribution. They then bid in the auction, possibly lying strategically about their valuation while attempting to anticipate the strategic behavior of others. The result is a potentially complicated Bayes-Nash equilibrium; even predicting these equilibria can be difficult, let alone designing auctions that have good equilibria.

This motivates the design of **strategyproof** auctions: these are auctions where players are incentivized to simply and truthfully reveal their private valuations to the auctioneer. Subject to the strategyproofness constraint, which makes player behavior predictable, it is then possible to optimize the mechanism to maximize revenue. A classic strategyproof auction design is the second-price auction—coinciding when selling a single item with the celebrated Vickrey-Clarke-Groves (VCG) mechanism [41, 83, 160]—in which participants bid only once and the highest bidder wins, but the price paid by the winner is only the amount of the second-highest bid.

In a groundbreaking work, Myerson [123] characterized the revenue-maximizing strategyproof auction for selling one item to many bidders. However, there has been very little progress in characterizing optimal strategyproof auctions in more general settings. Optimal mechanisms are known for some cases of auctioning two items to a single bidder [54, 86, 115, 129]. The optimal strategyproof auction even for 2 agents buying 2 items is still unknown.

A more recent set of approaches involves formulating the auction design problem as a learning problem. Dütting et al. [58] provide the general end-to-end approach which we build on in this paper. In brief, they design a neural network architecture to encode an auction mechanism, and train it on samples from the valuation distribution to maximize revenue. Their goal is to enforce, at least approximately, dominant-strategy incentive compatibility: a stronger notion than the Bayesian incentive compatibility of some other mechanism design approaches [35, 36, 37]. While they describe a number of network architectures which work in restricted settings, we focus on their RegretNet architecture, which can be used in settings with any number of agents or items.

The training procedure for RegretNet involves performing gradient ascent on the network inputs, to find a good strategic bid for each player; the network is then trained to minimize the difference in utility between strategic and truthful bids—this quantity is the eponymous "regret". The process is remarkably similar to adversarial training [113], which applies robust optimization schemes to neural networks, and the desired property of strategyproofness can be thought of as a kind of adversarial robustness. Motivated by this connection, we use techniques from the adversarial robustness literature to compute **certifiable** upper bounds on the amount by which a player with a specific valuation profile can improve their utility by strategically lying.

While the adversarial training approach seems effective in *approximating* strategyproof auction mechanisms, neural network training is fraught with local minima and suboptimal stationary points. One can discover strategic behaviors by using simple gradient methods on RegretNet auctions, but we note that it is known from the adversarial examples literature that such results are often sub-optimal [17] and depend strongly on the optimizer [162]. For this reason, it is unclear how strategyproof the results of RegretNet training are, and how much utility can be gained through strategic behavior in such auctions.

Our goal here is to learn auction mechanisms that are not only approximately strategyproof, but that come with rigorous bounds on how much they can be exploited by strategic agents, regardless of the strategy used. We achieve this by leveraging recent ideas developed for certifying adversarial robustness of neural classifiers [34, 112, 159], and adapting them to work within an auction framework.

Our contributions.

- We initiate the study of certifiably strategyproof learned auction mechanisms. We see this as a step toward achieving the best of both worlds in auction design—maintaining *provable properties* while expanding to more *general settings* than could be analyzed by traditional methods.

- We develop a method for formulating an integer-programming-based certifier for general learned auctions with additive valuations. This requires changes to the RegretNet architecture. We replace its softmax activation with the piecewise linear sparsemax [117], and we present two complementary techniques for dealing with the requirement of individual rationality: either formulating a nonconvex integer program, or removing this constraint from the network architecture and adding it as a learned penalty instead.

- We provide the first *certifiable* learned auctions for several settings, including a 2 agent, 2 item case where no known optimal auction exists; modulo computational scalability, our techniques for learning auctions and producing certificates for a given valuation profile work for settings with any number of items or (additive) bidders.

2.2 Background

Below, we introduce the general problem of automated mechanism design. We then describe the RegretNet approach for learning auction mechanisms, as well as the neural network verification techniques that we adapt to the auction setting. The RegretNet architecture originated the idea of parameterizing mechanisms as neural networks and training them using techniques from modern deep learning. This approach has been termed "differentiable economics", and several other papers have expanded on this approach in various settings beyond revenue-maximizing sealed-bid auctions [68, 77, 116, 153, 156].

2.2.1 Previous work

Automated mechanism design is the problem of finding good mechanisms for specific valuation distributions. In this area, one strand of work involves discretizing the space of types and solving a linear program to find the best auction in a family of mechanisms [44, 145]. For Bayesian incentive compatible revenue-maximizing auctions with additive bidders, [35, 36, 37] give techniques for finding the optimal mechanism, although Bayesian incentive compatibility is a weaker requirement than dominant-strategy incentive compatibility. Other work requires only access to samples from the valuation distribution over which revenue must be maximized [110, 147]. In this way, auction design becomes a learning problem, to which the tools of learning theory can be applied [22]. RegretNet falls into this latter family of techniques. In particular, it is an approach that learns from samples to approximate a DSIC mechanism.

2.2.2 RegretNet

In the standard auction setting, it is assumed that there are n agents (indexed by i) buying k items (indexed by j), and that the agents value the items according to values drawn from some distribution $P(\boldsymbol{v}_i)$. This distribution is assumed to be public, common knowledge (it is essentially the data-generating distribution). However, the specific sampled valuations are assumed to be private to each agent.

The auctioneer solicits a bid b_{ij} from all agents on all items. The auction mechanism $f(\boldsymbol{b}_1, \cdots, \boldsymbol{b}_n)$ aggregates bids and outputs the results of the auction. This consists of an allocation matrix g_{ij}, representing each player's probability of winning each item, and a payment vector p_i, representing the amount players are charged. Players receive some utility u_i based on the results; in this work, we focus on the case of additive utility, where $u_i = \sum_j g_{ij} v_{ij} - p_i$.

As previously mentioned, players are allowed to choose bids strategically to maximize their own utility, but it is often desirable to disincentivize this and enforce strategyproofness. The auctioneer also wants to maximize the amount of revenue paid. Dütting et al. [58] present the RegretNet approach: the idea is to represent the mechanism f as a neural network, with architectural constraints to ensure that it represents a valid auction, and a training process to encourage strategyproofness while maximizing revenue.

(We note that Dütting et al. [58] presents other architectures, RochetNet and MyersonNet, which are completely strategyproof by construction, but only work in specific settings: selling to one agent, or selling only one item. In our work, we focus only on certifying the general RegretNet architecture.)

Network architecture

The RegretNet architecture is essentially an ordinary feedforward network that accepts vectors of bids as input and has two output heads: one is the matrix of allocations and one is the vector of payments for each agent.

The network architecture is designed to make sure that the allocation and payments output are feasible. First, it must ensure that no item is overallocated: this amounts to ensuring that each column of the allocation matrix is a valid categorical distribution, which can be enforced using a softmax activation.

Second, it must ensure that no bidder (assuming they bid their true valuations) is charged more than the expected value of their allocation. It accomplishes this by using a sigmoid activation on the payment output head to make values lie between 0 and 1 – call these $\tilde{p}_i$. Then the final payment for each player is $\left(\sum_j v_{ij} g_{ij}\right) \tilde{p}_i$; this guarantees that utility can at worst be 0.

Both of these architectural features pose problems for certification, which we describe below.

Training procedure

The goal of the auctioneer is to design a mechanism that maximizes the expected sum of payments received $\mathbb{E}_{\boldsymbol{v} \sim P(\boldsymbol{v})} \left[\sum_i p_i(\boldsymbol{v})\right]$, while ensuring that each player has low regret, defined as the difference in utility between the truthful bid and their best strategic bid:

$$\mathrm{rgt}_i(\boldsymbol{v}) = \max_{\boldsymbol{b}_i} u_i(\boldsymbol{b}_i, v_i, \boldsymbol{v}_{-i}) - u_i(\boldsymbol{v}_i, v_i, \boldsymbol{v}_{-i}) \tag{2.1}$$

Note that this definition of regret allows only player i to change their bid. However, if $\mathbb{E}_{\boldsymbol{v}}[\mathrm{rgt}_i(\boldsymbol{v})]$ is low for all players then the mechanism must be approximately strategyproof; this is because every possible strategic bid by players other than i could also be observed as a truthful bid from the support of $P(\boldsymbol{v})$.

Dütting et al. [58] approximates regret using an approach very similar to adversarial training [113]. They define a quantity $\widehat{\mathrm{rgt}}_i$ by approximately solving the maximization problem using gradient ascent on the input – essentially finding an adversarial input for each player. Given this approximate quantity, they can then define an augmented Lagrangian loss function to maximize revenue while forcing $\widehat{rgt}$ to be close to 0:

$$\mathscr{L}(\boldsymbol{v}, \boldsymbol{\lambda}) = -\sum_i p_i + \sum_i \lambda_i \widehat{\mathrm{rgt}}_i(\boldsymbol{v}) + \frac{\rho}{2} \sum_i \left(\widehat{\mathrm{rgt}}_i(\boldsymbol{v})\right)^2 \tag{2.2}$$

They then perform stochastic gradient descent on this loss function, occasionally increasing the Lagrange multipliers λ, ρ and recomputing $\widehat{\mathrm{rgt}}$ at each iteration using gradient ascent. At test time, they compute revenue under the truthful valuation and regret against a stronger attack of 1000 steps. A number of high probability generalization bounds are provided for estimating regret and revenue from pointwise values on samples. With regards to the estimation of regret, we note that their generalization bound refers to the true regret at a single point (equation 2.1) – a quantity which a gradient-based method can only approximate but not compute exactly.

2.2.3 Mixed integer programming for certifiable robustness

Modern neural networks with ReLU activations are piecewise linear, allowing the use of integer programming techniques to verify properties of these networks. Bunel et al. [34] present a good overview of various techniques use to certify adversarial robustness, along with some new methods.

The general approach they describe is to define variables in the integer program representing neural network activations, and constrain them to be equal to each network layer's output:

$$\begin{aligned}\hat{\boldsymbol{x}}_{i+1} &= W_i\boldsymbol{x}_i + \boldsymbol{b}_i \\ \boldsymbol{x}_{i+1} &= \max(0, \hat{\boldsymbol{x}}_{i+1})\end{aligned} \tag{2.3}$$

With input constraints $x_0 \in S$ representing the set over which the adversary is allowed to search, solving the problem to maximize some quantity will compute the actual worst-case input. In most cases, this is some proxy for the classification error, and the input set is a ball around the true input; in our case, computing a certificate for player i involves maximizing $u_i(\boldsymbol{b}_i, v_i, \boldsymbol{v}_{-i})$ over all $\boldsymbol{b}_i \in \text{Supp}(P(\boldsymbol{v}_i))$, i.e. explicitly solving (2.1).

The program is linear except for the ReLU term, but this can be represented by adding some auxiliary integer variables. In particular, Tjeng, Xiao, and Tedrake [159] present the following set of constraints (supposing a d-dimensional layer output), which they show are feasible iff $\boldsymbol{x}_i = \max(\hat{\boldsymbol{x}}_i, 0)$:

$$\begin{aligned}\boldsymbol{\delta}_i \in \{0,1\}^d, \quad \boldsymbol{x}_i \geq 0, \quad \boldsymbol{x}_i \leq \boldsymbol{u}_i\boldsymbol{\delta}_i \\ \boldsymbol{x}_i \geq \hat{\boldsymbol{x}}_i, \quad \boldsymbol{x}_i \leq \hat{\boldsymbol{x}}_i - \boldsymbol{l}_i(1-\boldsymbol{\delta}_i)\end{aligned} \tag{2.4}$$

The $\boldsymbol{u}_i, \boldsymbol{l}_i$ are upper and lower bounds on each layer output that are known a priori – these can be derived, for instance, by solving some linear relaxation of the program representing the network. In particular, an approach called Planet due to [61] involves resolving the relaxation to compute tighter bounds for each layer in turn. [34] provide a Gurobi-based [85] integer program formulation that uses the Planet relaxations, later updated for use by [112]; we modify that version of the code for our own approach. These methods output a solution which will at least be a lower bound on the true regret. Under the assumption that the chosen integer programming solver accurately reports when it has solved programs to global optimality, this will also be an upper bound – thus we will know the true maximum regret. Using bounds from Dütting et al. [58], by computing true expected regret at many sampled points, we can estimate the overall regret of the mechanism with high probability, and we can bound the probability of sampling a point with high regret.

2.3 Techniques

These neural network verification techniques cannot be immediately applied to the RegretNet architecture directly. We describe modifications to both the network architecture and the mathematical programs that allow for their use: a replacement for the softmax activation that can be exactly represented via a bilevel optimization approach, and two techniques for coping with the individual rationality requirement. We also use a regularizer from the literature to promote ReLU stability, which empirically makes solving the programs faster.

2.3.1 Sparsemax

The RegretNet architecture applies a softmax to the network output to produce an allocation distribution where no item is overallocated. In an integer linear program, there is no easy way to exactly represent the softmax. While a piecewise linear overapproximation might be possible, we elect instead to replace the softmax with the *sparsemax* [117]. Both softmax and sparsemax project vectors onto the simplex, but the sparsemax performs a Euclidean projection:

$$\text{sparsemax}(x) = \arg\min_{z} \frac{1}{2}\|x-z\|_2^2 \text{ s. t. } 1^T z - 1 = 0, 0 < z < 1 \tag{2.5}$$

(Martins and Astudillo [117] describes a cheap exact solution to this optimization problem and its gradient which are used during training. We use a PyTorch layer provided in [101, 102].)

In order to encode this activation function in our integer program, we can write down its KKT conditions and add them as constraints (a standard approach for bilevel optimization [43]), as shown in (2.6).

$$
\begin{aligned}
(z-x)+\mu_1-\mu_2+\lambda 1 &= 0 \\
z-1 \leq 0, -z \leq 0, 1^T z - 1 &= 0 \\
\mu_1 \geq 0, \mu_2 &\geq 0 \\
\mu_1(z-1) = 0, \mu_2(-z) &= 0
\end{aligned}
\tag{2.6}
$$

These constraints are all linear, except for the complementary slackness constraints – however, these can be represented as SOS1 constraints in Gurobi and other solvers.

The payment head also uses a sigmoid nonlinearity; we simply replace this with a piecewise linear function similar to a sigmoid.

2.3.2 Enforcing individual rationality

The RegretNet architecture imposes individual rationality – the requirement that no agent should pay more than they win – by multiplying with a fractional payment head, so that each player's payment is always some fraction of the value of their allocation distribution.

When trying to maximize utility (in order to maximize regret), this poses a problem. The utility for player i, with untruthful bids $\boldsymbol{b}_i$ and fixed truthful valuation $\boldsymbol{v}_i$, is $u_i(\boldsymbol{b}_i) = \sum_j g_{ij} v_{ij} - p_i$. The value of the allocation is a linear combination of variables with fixed coefficients. But $p_i = \left(\sum_j g_{ij} \boldsymbol{b}_{ij}\right) \tilde{p}_i$ – this involves products of variables, which cannot be easily represented in standard integer linear programs.

We propose two solutions: we can either formulate and solve a nonconvex integer program (with bilinear equality constraints), or remove the IR constraint from the architecture and attempt to enforce it via training instead.

Nonconvex integer programs The latest version of Gurobi can solve programs with bilinear optimality constraints to global optimality. By adding a dummy variable, we can chain together two such constraints to represent the final payment. We have variables in the integer program to represent the fractional payment head, the bidder surplus, and the final payment: $p_i = \tilde{p}_i y$, and $y_i = \sum_j g_{ij} b_{ij}$. It is desirable to enforce IR constraints at the architectural level, but as described in the experiments section, it can potentially be much slower.

Individual rationality penalty As opposed to constraining the model architecture to enforce individual rationality constraint, we also experiment with enforcing the constraint through an additional term in the Lagrangian (a similar approach was used in an earlier version of [58]). We can compute the extent to which individual rationality is violated:

$$\text{irv}_i = \max(p_i - \sum_j g_i b_i, 0) \tag{2.7}$$

We then allow the network to directly output a payment, but add another penalty term to encourage individual rationality:

$$\mathscr{L}(\boldsymbol{v}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = -\sum_i p_i + \sum_i \lambda_i \widehat{\text{rgt}}_i(\boldsymbol{v}) + \frac{\rho}{2} \sum_i \left(\widehat{\text{rgt}}_i(\boldsymbol{v})\right)^2 + \sum_i \mu_i \, \text{irv}_i^2 \tag{2.8}$$

With this approach, the final payment no longer involves a product between allocations, bids, and payment head, so the MIP formulation does not have any quadratic constraints.

Distillation loss Training becomes quite unstable after adding the individual rationality penalty; we stabilize the process using distillation [92]. Specifically, we train a teacher network using the original RegretNet architecture, and use a mean squared error loss between the student and the teacher's output to train our network. The teacher may have an approximately correct mechanism, but is difficult to certify; using distillation, we can train a similar student network with an architecture more favorable for certification. Additionally, combined with the individual rationality penalty in the Lagrangian, distilling from a teacher network which enforces IR by construction also allows us to learn a student network which is discouraged from violating IR.

We allow the payments to vary freely during distillation training, to avoid vanishing gradients, and simply clip the payments to the feasible range after the training is done. Through this method, empirically, we are able to train student networks that are comparable to the teachers in performance.

2.3.3 Regularization for fast certificates

Xiao et al. [166] point out that a major speed limitation in integer-programming based verification comes from the need to branch on integer variables to represent the ReLU nonlinearity (see Equation 2.4). However, if a ReLU unit is *stable*, meaning its input is always only positive or only negative, then there is no need for integer variables, as its output is either linear or constant respectively.

We adopt the approach in Xiao et al. [166], which at train time uses interval bound propagation [80] to compute loose upper and lower bounds on each activation, and adds a regularization term $-\tanh(1+ul)$ to encourage them to have the same sign. At verification time, variables where upper and lower bounds (computed using the tighter Planet relaxation) are both positive or both negative do not use the costly formulation of Equation 2.4.

2.4 Experiments

We experiment on two auction settings: 1 agent, 2 items, with valuations uniformly distributed on $[0,1]$ (the true optimal mechanism is derived analytically and presented by Manelli and Vincent [115]); and 2 agents, 2 items, with valuations uniformly distributed on $[0,1]$, which is unsolved analytically but shown to be empirically learnable in [58].

For each of these settings, we train 3 networks:

- A network with a sparsemax allocation head which enforces individual rationality using the fractional payment architecture, and uses the ReLU stability regularizer of [166]
- The same architecture, without ReLU regularization
- A network that does not enforce IR, trained via distillation on a network with the original RegretNet architecture

Additionally, to investigate how solve time scales for larger auctions, we consider settings with up to 3 agents and 3 items for the architecture without IR enforcement. All training code is implemented using the PyTorch framework [128].

2.4.1 Training procedure

We generate 600,000 valuation profiles as training set and 3,000 valuation profiles as the testing set. We use a batch size of 20,000 for training, and we train the network for a total of 1000 epochs. At train time, we generate misreports through 25 steps of gradient ascent on the truthful valuation profiles with learning rate of .02; at test time, we use 1000 steps. Architecturally, all our networks use a shared trunk followed by separate payment and allocation heads; we find the use of a shared embedding makes the network easier to certify. We generally use more layers for larger auctions, and the detailed architectures, along with hyperparameters of the augmented Lagrangian, are reported in Appendix 2.6.

Auction Setting	**IR**	**Relu Reg.**	**Solve time (s)**	**Revenue**	**Empirical Regret**	**Certified Regret**	**Emp./ Cert.**
1x2	Yes	No	25.6 (72.0)	0.593 (0.404)	0.014 (0.012)	0.019 (0.016)	0.731
1x2	Yes	Yes	7.2 (17.5)	0.569 (0.390)	0.003 (0.002)	0.004 (0.003)	0.700
1x2	No	Yes	0.034 (0.007)	0.568 (0.398)	0.009 (0.005)	0.011 (0.004)	0.839
2x2	Yes	No	13.9 (37.0)	0.876 (0.286)	0.009 (0.013)	0.014 (0.016)	0.637
2x2 (2nd)	Yes	No	17.4 (51.9)	—	0.007 (0.011)	0.011 (0.013)	0.676
2x2	Yes	Yes	5.8 (16.3)	0.874 (0.285)	0.008 (0.012)	0.013 (0.015)	0.626
2x2 (2nd)	Yes	Yes	7.520 (24.2)	—	0.008 (0.012)	0.012 (0.014)	0.680
2x2	No	Yes	5.480 (5.577)	0.882 (0.334)	0.006 (0.007)	0.011 (0.011)	0.533
2x2 (2nd)	No	Yes	2.495 (2.271)	—	0.011 (0.010)	0.017 (0.017)	0.666

Table 2.1: Summary of experimental results. Empirical regret is computed on 3000 random points and certified regret is computed on 1000 different points. (2nd) denotes the second agent in a multi-agent auction. Note that average empirical regret is only about 60-80% of the average true regret. The number in the parenthesis represents the standard deviation.

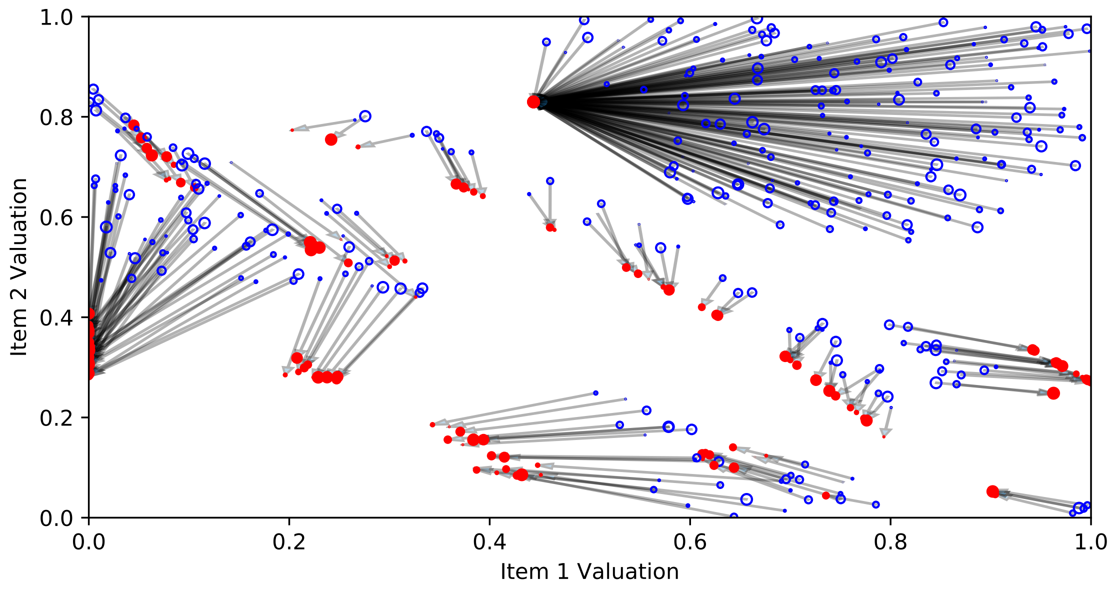


Figure 2.1: For the 1 agent, 2 item setting (regularized, IR enforced), this plot shows truthful bids (blue circle), with an arrow to the best strategic bid computed by the certifier (red filled). Only points with regret at least 0.005 are shown; the size of markers is proportional to the magnitude of regret. While the truthful and strategic bids are often far apart, this does not necessarily mean that violations of strategyproofness are large; in this plot, the highest regret of any point is still only 0.014.

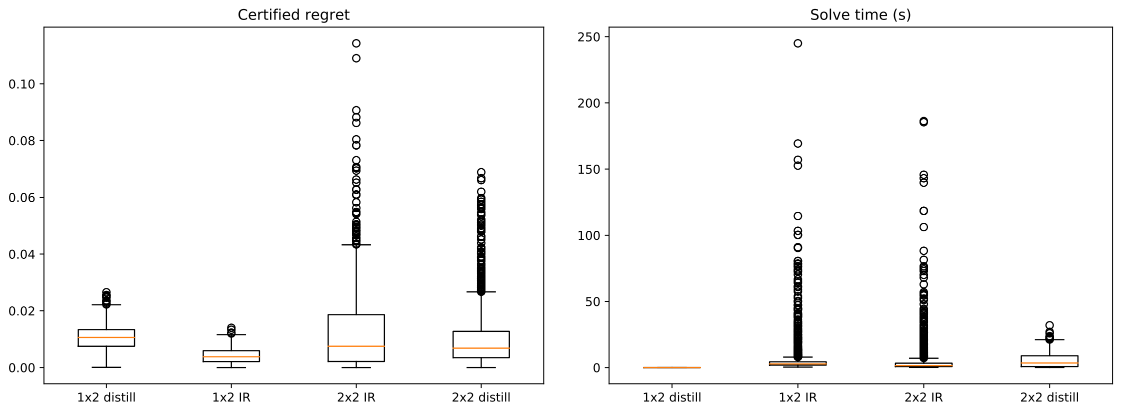


Figure 2.2: Certified regret and solve time for 1000 random points for IR and non-IR network architectures (regularized). Maximum utility in these settings is 2.0, so regrets are relatively small in most regions. At points with high regret, our certificates are able to detect this deficiency.

2.4.2 Results

Our results for regret, revenue and solve time are summarized in Table 2.1. We show the relationship between truthful and strategic bids for a learned 1 agent, 2 item mechanism in Figure 2.1.

Regret certificate quality We are able to train and certify networks with reasonably low regret – usually less than one percent of the maximum utility an agent can receive in these settings. Although mean regrets are small, the distributions are right skewed (particularly in the 2 agent, 2 item case) and there are a few points with high (approximately 0.1) regret. Crucially, we find that our certified regrets tend to be larger on average than PGD-based empirical regret, suggesting that our method reveals strategic behaviors that gradient-based methods miss.

Trained revenue As a baseline we consider the mean revenue from selling each item individually in a Myerson auction – in the 1 agent 2 item setting, this is 0.5; in the 2 agent 2 item setting, it is 0.8333. Our trained revenues exceed these baselines. For the 1 agent 2 item settings, the optimal mechanism [115] has a revenue of 0.55; our mechanisms can only exceed this because they are not perfectly strategyproof.

Individual rationality Empirically, the individual rationality penalty is very effective. On average, less than 5.53% of points give an allocation violating the individual rationality constraint, and even if it is violated, the magnitude of violation is on average less than .0002 (Table 2.4, Appendix 2.7). Filtering out IR-violating points after the fact results in lower revenue but by less than one percent.

Solve time The time required to solve the MIP is also quite important. In general, we find that ReLU stability regularization helps speed up solve time, and that solving the bilinear MIP (required for architectural enforcement of IR) is much harder than solving the mixed-integer linear program for the other architecture.

Auction setting	**Mean solve time (s)**	**Solve time std**	**Regret**	**Regret std**
2x3	160.66	142.86	0.0342	0.0169
3x2	5.039	3.40	0.0209	0.0152
3x3	71.81	54.24	0.0243	0.0204

Table 2.2: Solve times and regrets for non-IR architecture without clipped payments in larger settings on 250 random points. In general, increasing the number of items significantly slows down certification.

To investigate scalability, we also consider solve times and certified regrets for settings with larger numbers of agents and items; results are summarized in Table 2.2. Our experiments use the non-IR-enforcing architecture; additionally, for these experiments we do not apply hard clipping of payments. In general, increasing the number of items significantly increases the solve time – this is not too surprising, as increasing the number of items increases the dimensionality of the space that must be certified (while the same is not true for increasing the number of agents, because certificates are for one agent only). The larger solve time for 2 rather than 3 agents is harder to explain – it may simply be the result of different network properties or a more complex learned mechanism.

We note that both solve time and regret are heavily right-skewed, as shown in Figure 2.2. We also find that the difference between allocations, payments, and utilities computed by the actual network and those from the integer program is on the order of 10^{-6} – the constraints in the model correctly represent the neural network.

2.5 Conclusion and Future Work

Our MIP solution method is relatively naive. Using more advanced techniques for presolving, and specially-designed heuristics for branching, have resulted in significant improvements in speed and scalability for certifying classifiers [112, 159]. Our current work serves as strong proof-of-concept validation that integer-programming-based certifiability can be useful in the auction setting, and it is likely that these techniques could be applied in a straightforward way to yield a faster certifier.

The performance of our learned mechanisms is also not as strong as those presented in Dütting et al. [58], both in terms of regret and revenue. It is unclear to us whether this is due to differences in the architecture or to hyperparameter tuning. We observe that our architecture has the capacity to represent the same class of functions as RegretNet, so we are hopeful that improved training might close the gap. The recent paper [132] finds that RegretNet is indeed very sensitive to hyperparameters, and presents an improved algorithm for auction learning which is less sensitive. The neural architectures used with this new algorithm are essentially the same as RegretNet and can also be modified to allow for certificates.

In addition to generalization bounds provided in Dütting et al. [58], other work has dealt with the problem of estimating expected strategyproofness given only regret estimated on samples from the valuation distribution [24]. The methods presented in this work for solving the utility maximization problem have the potential to complement these bounds and techniques.

In this paper, we have described a modified version of the RegretNet architecture for which we can produce certifiable bounds on the maximum regret which a player suffers under a given valuation profile. Previously, this regret could only be estimated empirically using a gradient ascent approach which is not guaranteed to reach global optimality. We hope that these techniques can help both theorists and practitioners have greater confidence in the correctness of learned auction mechanisms.

Broader Impact

The immediate social impact of this work will likely be limited. Learned auction mechanisms are of interest to people who care about auction theory, and may eventually be used as part of the design of auctions that will be deployed in practice, but this has not yet happened to our knowledge. We note, however, that the design of strategyproof mechanisms is often desirable from a social good standpoint. Making the right move under a non-strategyproof mechanism may be difficult for real-world participants who are not theoretical agents with unbounded computational resources. The mechanism may impose a real burden on them: the cost of figuring out the correct move. By contrast, a strategyproof mechanism simply requires truthful reports—no burden at all.

Moreover, the knowledge and ability to behave strategically may not be evenly distributed, with the result that under non-strategyproof mechanisms, the most sophisticated participants may game the system to their own benefit. This has happened in practice: in Boston, some parents were able to game the school choice assignment system by misreporting their preferences, while others were observed not to do this; on grounds of fairness, the system was replaced with a redesigned strategyproof mechanism [2].

Thus, we believe that in general, the overall project of strategyproof mechanism design is likely to have a positive social impact, both in terms of making economic mechanisms easier to participate in and ensuring fair treatment of participants with different resources, and we hope we can make a small contribution to it.

2.6 Architectural and Training Details

We initialized the Lagrange multiplier of regret (λ_i) as 5, and update it every 6 batches, and we experiment with values for the constant ρ^{rgt} ranging between 0.5 to 2 (reporting the choice that gave the lowest regret). For the IR violation penalty, we initialize the Lagrange multiplier of IR violation (μ_i) as 20, and update the Lagrange multiplier every 6 iterations. μ is initialized as 5, and then incremented by 5 every 5 batches. For distillation, we take a mean squared error loss between the student and teacher's output, and use a multiplier of $\frac{1}{400}$. Specifically, the Lagrange multipliers are updated as follows.

$$\lambda_{i+1} = \lambda_i + \rho^{\text{rgt}}\widehat{\text{rgt}}_i \qquad \rho^{\text{rgt}}_{i+1} = \rho^{\text{rgt}}_i + \rho^{\text{rgt}}_{inc}$$

$$\mu_{i+1} = \mu_i + \rho^{\text{irv}}\,\text{irv}_i \qquad \rho^{\text{irv}}_{i+1} = \rho^{\text{irv}}_i + \rho^{\text{irv}}_{inc}$$

Auction Setting	**Inner Product**	**Relu Stability**	**Embedding Layer**
1 Agent x 2 Items	Yes	No	1 hidden layer x 128 units
1 Agent x 2 Items	Yes	Yes	1 hidden layer x 128 units
1 Agent x 2 Items	No	Yes	1 hidden layer x 128 units
2 Agents x 2 Items	Yes	No	2 hidden layer x 128 units
2 Agents x 2 Items	Yes	Yes	2 hidden layer x 128 units
2 Agents x 2 Items	No	Yes	2 hidden layer x 128 units

2.7 Appendix: Additional Experimental Information

Hardware All certification experiments were conducted on an AMD Ryzen 3600X CPU with 32GB RAM. Training of the network was conducted with a 2080 GPU on a university compute cluster.

Additional experiments Table 2.4 shows more detailed results for the non-IR-enforcing architecture. IR violations are relatively small, and filtering out these cases (sacrificing revenue) does not harm overall revenue too much.

Table 2.3 shows the results of scaling experiments for settings with more agents and items, in a setting where payment clipping is applied. Again, increasing the dimensionality of the input space by increasing the number of items seems to impose a greater cost than increasing the number of agents.

Auction setting	**Mean solve time (s)**	**Regret**
2x3	109.749 (159.212)	0.027 (0.016)
3x2	3.033 (2.377)	0.019 (0.016)
3x3	59.173 (53.431)	0.022 (0.020)

Table 2.3: Solve times and regrets for non-IR architecture with clipped payments in larger settings on 250 random points. In general, increasing the number of items significantly slows down certification. Standard deviations are in parentheses.

Auction Setting	**% of IR violation**	**Max IR viol.**	**Mean IR viol.**	**Revenue before IR**	**Revenue after IR**
1x2	5.53%	0.0053	0.0001 (0.0003)	0.5738	0.5681
2x2	4.60%	0.0083	0.0002 (0.0007)	0.8874	0.8824

Table 2.4: IR violation for the 1x2/2x2 auction settings. Note that the mean IR violation is small, and revenue after enforcing IR drops only slightly. The number in parenthesis represents the standard deviation.

Chapter 3: ProportionNet: Balancing Fairness and Revenue for Auction Design with Deep Learning

Joint work with Kevin Kuo, Anthony Ostuni, Elizabeth Horishny, Michael J Curry, Samuel Dooley, Ping-yeh Chiang, Tom Goldstein, John P Dickerson

3.1 Introduction

Auctions connect buyers and sellers to enable the exchange of money for goods and services. Auction theory has a rich history in economics and, more recently, computer science. Since 1994, the US Federal Communications Commission (FCC) has periodically run multi-billion dollar auctions to allocate electromagnetic spectrum broadcasting licenses requiring immense computational resources [107]. Technology giants such as Google, Facebook, and Baidu rely heavily on sophisticated auction-based advertising ecosystems to drive the majority of their revenue [59]. In aggregate, the contribution to the world economy of computational auctions is measured in the hundreds of billions, if not trillions, of dollars per year [7, 21, 64]. The design of auctions is thus quite important and is a major focus of the broader field of mechanism design [142].

In the typical theoretical model for auction mechanisms, players are presumed to have some private valuations of the items up for sale, which are drawn from some publicly-known distribution. To avoid trying to predict complex strategic behavior, it is common to focus on strategyproof, or incentive compatible, auctions. These are auctions where, even though players are free to lie about their private valuations, rational players will simply choose to tell the truth. Subject to this constraint, equilibrium play is simple, and the mechanism designer can focus on ensuring other

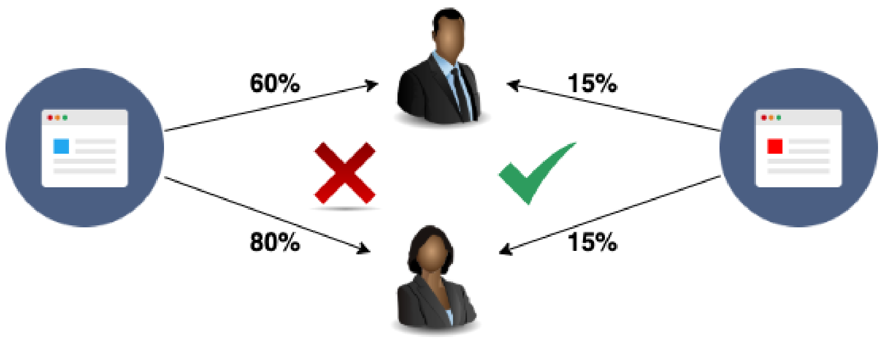


Figure 3.1: RegretNet determines revenue-maximizing allocations; yet, it is blind to the fairness of such allocations. Thus, there may be a high probability of unequal allocations, even for two items that are equivalent for all meaningful purposes. Concretely, it may allocate an online advertisement for a career opportunity to an equally qualified man and woman at notably different proportions (as on the left). Our network ProportionNet prevents such unfairness; (as on the right) it forces similar advertisement allocation proportions between the two similar individuals.

desirable properties. The classic strategyproof auction is known as the Vickrey-Clarke-Groves (VCG) auction [41, 83, 160], which has the additional desirable property of maximizing social welfare (i.e. the total utility enjoyed by all auction participants).

In many cases an auctioneer selling items may wish (or have an obligation, as in auctions of spectrum and other goods belonging to the public) to maximize their own revenue, subject to strategyproofness. The groundbreaking work due to Myerson [123] defined the optimal strategyproof auction for selling a single item, but further progress has been limited. While there are some results for selling multiple items to a single bidder [54, 76, 98, 115, 129], even for selling just two items to two bidders, no results are known, except in the case of binary valuations [167].

The persisting challenge of designing optimal auctions, and the fact that typical theoretical assumptions involve a probability distribution over valuations, have resulted in attempts to formulate the auction design problem as a machine learning problem. In particular, Dütting et al. [58] use neural networks to represent an auction mechanism (as a function from a vector of bids to outcomes), and define a learning objective to enforce strategyproofness while encouraging revenue maximization.

In addition to their economic importance, the design of auctions can also have serious social impact. With this in mind, what properties, in addition to strategyproofness, might the designer of a revenue-maximizing auction have reason to enforce? **A major concern must be fairness with respect to protected characteristics.**

Consider the case of online advertising—one of the most important real-world applications for the theory of mechanism design. When placing advertisements in certain categories (job ads, ads for certain financial services, ads for housing, etc.), companies have a legal obligation to avoid discrimination on the basis of protected characteristics such as race, gender, and national origin. Yet a 2015 study [55] showed a difference between employment advertisements received by male and female users: male users were shown advertisements promising higher salaries than female users. Furthermore, Ali et al. [6] observe that Facebook's preemptive categorization of ad-user relevancy skews ads toward certain genders and racial groups.

A number of papers have considered the mechanism design problem when fairness with respect to protected characteristics is required [39, 95]. These notions of fairness do not consider the bidders in the auction, but rather the individuals whose impressions are the "items" for sale – thus imposing fairness constraints means imposing constraints on the allocations made by the auction mechanism.

Defining a strategyproof, fair auction that maximizes revenue remains theoretically challenging. Recent work in this area includes Celis, Mehrotra, and Vishnoi [38], which provides theory and algorithms for finding optimal itemwise Myerson auctions under fairness constraints. Nasr and Tschantz [124] computes fair strategies from the bidder's point of view.

As with other auctions, we see the use of machine learning as a way out of this impasse – we aim to extend the techniques of Dütting et al. [58] and others to allow the imposition of fairness constraints on learned mechanisms. Doing so allows insights into the cost to revenue of imposing fairness, and the structure of fair mechanisms in some settings.

Our contributions.

- We provide a deep-learning-based method for designing approximately fair, strategyproof, revenue-maximizing auctions given access to samples from the valuation distribution. Our approach extends the RegretNet approach [58] with fairness constraints (and preserves its generalization guarantees): a melding of ideas from the fair ML and economics & computation communities.

- This represents a step towards the larger problem of designing revenue-maximizing multi-item auctions under not only strategyproofness but also fairness constraints, potentially motivating and informing future theoretical work.

3.2 Background

3.2.1 Fairness

There are real-world examples of unfairness in advertising auctions. Here, the unfairness is suffered by the individuals whose ad impressions are the "items" up for auction. Several studies have found unfair treatment of various protected demographic categories in real-world ad auctions [55, 65, 105, 120, 155]. We discuss such instances in more detail in Section 3.5.1. We now describe one attempt from the literature to formalize fairness, which we will adopt as an additional constraint in the RegretNet approach to auction learning.

In all these cases, as mentioned above, fairness is with respect to the ads served to the users, corresponding to "items" in the typical model of auctions. (We emphasize this to distinguish our case from the more typical problem of fair mechanism design, where one is concerned with a fair allocation for the mechanism participants.) To mathematically formalize a notion of fairness in this context, we utilize the definition of **total variation fairness** from Ilvento, Jagadeesan, and Chawla [95] due to its generality.

Formalizing Unfairness

Let $C = \{C_1, ..., C_c\}$, denote a partition of the set $[n]$ of advertisers, or agents, into c categories. For $1 \leq k \leq c$, let $d^k : M \times M \to [0,1]$ define a distance metric between all pairs of users, or items. The auction mechanism satisfies **total variation fairness** if the ℓ_1-distance between allocations (summed over a subset of advertisers C_k) for any two users is at most the distance between those users. That is, total variation fairness is satisfied when

$$\forall k \in \{1,...,c\}, \forall j, j' \in M, \sum_{i \in C_k} |z_{i,j} - z_{i,j'}| \leq d^k(j, j'). \tag{3.1}$$

In other words, similar users cannot be treated too differently, although the degree of permissible different treatment might be tighter (for instance, for job or housing advertisements) or looser. For instance, certain advertisers who need not worry about fairness could be put in a different category C_k; or items where unfairness is not a concern could have a greater distance.

3.2.2 Our Work

Our research is an amalgamation of deep learning techniques and fairness concerns; we propose a machine learning solution to find an auction that is DSIC, IR, revenue maximizing, and fair. Our work extends the RegretNet architecture to satisfy the total variation fairness constraint between all pairs of auction items.

3.3 Methodology

3.3.1 Fairness Constraint

To adapt definition 3.1 for use as a neural loss function, we define **unfairness** as a measure of how much the total variation constraint is violated by an auction allocation. The unfairness experienced by a user j is:

$$\text{unf}_j = \sum_{j' \in M} \sum_{C_k \in C} \max(0, (\sum_{i \in C_k} \max(0, z_{i,j} - z_{i,j'})) - d^k(j, j')) \tag{3.2}$$

A sum of unfairness over all users $\overline{\text{unf}} = \sum_{j \in M} \text{unf}_j$ allows us to quantify how unfair an auction outcome is for all users involved.

3.3.2 Network Architecture

We use the same additive and unit-valuation network architectures as RegretNet for arbitrary numbers of agents and items. We enforce our fairness constraint using the augmented Lagrangian approach in RegretNet by incorporating an additional set of multipliers λ_f. Our modified loss function $\mathscr{C}_\rho(w;\lambda)$ is defined as:

$$\begin{aligned}
\mathscr{L}_{\text{rgt}} &= \sum_{i\in N} \lambda_{(r,i)} \operatorname{rgt}_i(w) + \frac{\rho_r}{2}(\sum_{i\in N} \operatorname{rgt}_i(w))^2 \\
\mathscr{L}_{\text{unf}} &= \sum_{j\in M} \lambda_{(f,j)} \operatorname{unf}_j(w) + \frac{\rho_f}{2}(\sum_{j\in M} \operatorname{unf}_j(w))^2 \\
\mathscr{C}_\rho(w;\lambda) &= -\frac{1}{L}\sum_{l=1}^{L}\sum_{i\in N} p_i^w(v^{(l)}) + \mathscr{L}_{\text{rgt}} + \mathscr{L}_{\text{unf}}
\end{aligned} \tag{3.3}$$

3.3.3 Training Procedure

The procedure closely matches that of RegretNet, with the addition of updating λ_f and an additional quadratic parameter ρ_f for our fairness penalty. Here, $\widetilde{rgt}_i(w)$ and $\widetilde{unf}_i(w)$ denote the empirical regret in (1.1) and unfairness in (3.2), respectfully, computed on minibatch S_t.

3.3.4 Generalization Bound

When measuring expected unfairness, we cannot directly compute the expected value—instead, we must estimate it from samples of individual valuation profiles. Similarly to Dütting et al. [58], we wish to bound the generalization error when estimating auction unfairness from samples—hewing closely to techniques and definitions presented there, we do this in terms of the covering number of the class of auctions, showing that with high probability, our sample estimate is a good upper bound of true expected unfairness.

Theorem 3.1. *Let $\mathscr{M}$ be a class of auctions that satisfy individual rationality and have $\ell_{\infty,1}$ covering number $\mathscr{N}_\infty(\mathscr{M},\cdot)$. Fix $\delta \in (0,1)$. With probability at least $1-\delta$ over a draw of L valuation profiles, for any $(g^w, p^w) \in \mathscr{M}$,*

Algorithm 1 ProportionNet Training

```
1:  Input: Minibatches $S_1, ..., S_T$ of size $B$
2:  Parameters: $\rho_r^t, \rho_f^t, \gamma, \eta \in \mathbb{R}_{\geq 0}$ $G \in \mathbb{N}$
3:  Initialize: $w^0 \in \mathbb{R}^d, \lambda_r^0 \in \mathbb{R}^n, \lambda_f^0 \in \mathbb{R}^m$
4:  for $t = 0$ to $T$ do
5:      Receive minibatch $S_t = \{v^{(1)}, ..., v^{(B)}\}$
6:      Initialize misreports $v_i'^{(\ell)} \in V_i, \forall \ell \in [B], i \in N$
7:      for $\gamma = 0$ to $G$ do
8:          for $\ell \in [B], i \in N$ do
9:              $v'^{(\ell)}_i \leftarrow v'^{(\ell)}_i + \gamma \nabla_{v'_i} u_i^w(v_i^{(\ell)}; (v'^{(\ell)}_i, v_{-i}^{(\ell)}))$
10:     Compute Lagrangian gradient and update $w^t$:
11:     $w^{t+1} \leftarrow w^t - \eta \nabla_w \mathscr{C}_{\rho_t}(w^t, \lambda_r^t, \lambda_f^t)$
12:     Update Lagrange multipliers every $Q_r$ (regret) and $Q_f$ (fairness) iterations:
13:     if $t$ is a multiple of $Q_r$ then
14:         $\lambda_{(r,i)}^{t+1} \leftarrow \lambda_{(r,i)}^t + \rho_r^t \widetilde{rgt}_i(w^{t+1}), \forall i \in N$
15:     else
16:         $\lambda_{(r,i)}^{t+1} \leftarrow \lambda_{(r,i)}^t$
17:     if $t$ is a multiple of $Q_f$ then
18:         $\lambda_{(f,i)}^{t+1} \leftarrow \lambda_{(f,i)}^t + \rho_f^t \widetilde{unf}_i(w^{t+1}), \forall i \in M$
19:     else
20:         $\lambda_{(f,i)}^{t+1} \leftarrow \lambda_{(f,i)}^t$
```

$$\mathbb{E}_v\left[\sum_{j=1}^{m}\mathrm{unf}_j\circ g^w(v)\right]\leq\frac{1}{L}\sum_{\ell=1}^{L}\sum_{j=1}^{m}\mathrm{unf}_j\circ g^w(v^\ell)+2D_L+4C\sqrt{\frac{2\log(4/\delta)}{L}}$$

where C is a constant and

$$D_L=\inf_{\varepsilon>0}\left((nm^2+\varepsilon)\sqrt{\frac{2\log\mathcal{N}_\infty(\mathcal{M},\frac{\varepsilon}{2m^3})}{L}}+\varepsilon\right).$$

3.4 Experiments

We train ProportionNet in a variety of auction settings. Following Dütting et al. [58], we consider settings involving selling to one agent where revenue-maximizing solutions are known, and additionally add fairness constraints. We then consider settings with more agents, items, and fairness constraints beyond the reach of theory. Finally, we compare ProportionNet's performance to that of existing fair auction algorithms.

3.4.1 Experimental Parameters

For each configuration of n agents and m items, we trained ProportionNet for a maximum of 120 epochs using 640,000 training samples. We used two hidden layers for settings A, B, G and three for D, E, F. The hidden layers for setting C are shown in table 3.1 and table 3.2 in the appendix. All networks used the Adam optimizer and 100 hidden nodes per layer. We incremented both ρ_r and ρ_f every two epochs and λ_r and λ_f every 100 iterations. Finally, we report means and standard deviations of various statistics by simulating the allocations of 10,000 testing samples.

3.4.2 The Manelli-Vincent and Pavlov Auctions

Dütting et al. [58] successfully reproduced the analytic solutions where they were known using the RegretNet framework. These settings are as follows:

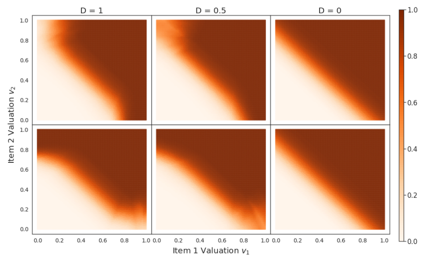


Figure 3.2: Setting A. The top and bottom rows respectively show the allocation rules for items 1 and 2.

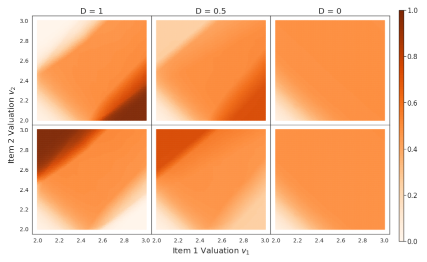


Figure 3.3: Allocation rule for Setting B.

A. Single-bidder with additive valuations over two items. Item values are independent draws from $U[0,1]$ [115].

B. Single-bidder, unit-demand valuations over two items. Item values are independent draws from $U[2,3]$ [129].

We train on these settings and additionally apply a uniform fairness constraint to both of these settings. The entries of the total variation fairness distance matrix D are all set to a constant $d(j,j') = d$ for all pairs of users j and j'.

Figures 3.2 and 3.3 show allocation mechanisms for these settings with varying levels of fairness. Training using $d = 1$ approximates the revenue-maximizing auction (as this is just standard RegretNet). Note that in both cases the unconstrained revenue-maximizing auction has regions of "maximum unfairness", where only one item is allocated with probability 1. Training with $d = 0$ results in an auction where both items are always allocated with equal probability.

Sweep Revenue, Mean (StDev)							
n x m	ℓ	Myr	D				
-	-	-	1.00	0.75	0.50	0.25	0.00
1 x 4	2	1.00	1.199 (0.602)	1.191 (0.579)	1.182 (0.591)	1.189 (0.591)	1.180 (0.589)
2 x 4	2	1.67	1.720 (0.462)	1.709 (0.452)	1.683 (0.444)	1.680 (0.438)	1.681 (0.438)
3 x 4	3	2.12	2.081 (0.404)	2.040 (0.386)	2.011 (0.374)	2.003 (0.373)	2.001 (0.371)
4 x 4	3	2.45	2.247 (0.302)	2.204 (0.307)	2.156 (0.334)	2.159 (0.334)	2.158 (0.336)
5 x 4	4	2.69	2.389 (0.326)	2.381 (0.349)	2.271 (0.305)	2.291 (0.314)	2.318 (0.314)

Table 3.1: Setting C Revenue – auctions with $U[0,1]$ valuations for n bidders and m items.

3.4.3 Scaling Up

Next, we experiment with larger auctions where there may be no viable analytical solution, even without considering fairness constraints. We define setting C:

C. n bidders with additive valuations over m items. All values (regardless of bidder or item) are independent draws from $U[0,1]$.

We tested all combinations of $n = 1,...,5$ bidders, $m = 2,...,6$ items, and fairness constraints $d = 0, 0.25, ..., 1.00$.

Table 3.1 shows a cross-section of the results with $m = 4$. Myr denotes expected revenue of the itemwise Myerson auction – selling each item independently in a strategyproof auction – while ℓ denotes the number of hidden layers used for training. Revenue, regret, and unfairness values for all testing configurations can be found in the appendix.

Note that as the number of agents and items increases, it becomes increasingly difficult to both maximize revenue and obey the regret and unfairness constraints. Thus, some of our results (which were primarily selected on the criterion of low mean and standard deviation of regret and unfairness) do not exceed the baseline itemwise Myerson revenue. Additionally, results for the 4 x 6 and 5 x 5 auctions significantly exceed the itemwise Myerson, but their regret and unfairness values are also high. However, prior success in applying the augmented Lagrangian to higher-complexity auctions (3 x 10, 5 x 10) in addition to RegretNet's sensitivity to hyperparameter search [132] suggest that these problems can be resolved with greater computational resources.

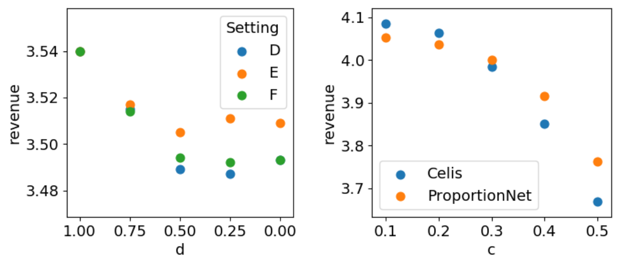


Figure 3.4: Left: ProportionNet revenue for Settings D, E, F. Right: Revenue for Setting G subject to the uniform coverage fairness parameter c, for Celis et al. and ProportionNet.

3.4.4 Non-uniform Fairness

In addition to the uniform fairness experiments above, we define three simple settings to investigate enforcement of fairness in cases where preferences may be more complex. Here, we consider auctions with three bidders and four items denoted $u_{1...4}$. Each item has binary features f_1 and f_2 which are used to construct the distance values used in our fairness constraint – in an ad auction setting, these could be characteristics of different groups of users. We consider three settings which have identical valuations but are subject to different fairness constraints.

D. All bidders are constrained by a distance metric on f_2.

E. Setting D, and the f_2 feature of user u_1 is changed from 0 to 1.

F. Setting D, but bidder 3 is constrained by a distance metric on f_1 rather than f_2.

We vary the parameter d of our distance function d^*:

$$d^*(j, j', f_i) = d + (1-d)|f_i(u_j) - f_i(u_{j'})|$$

Figure 3.4, left, shows the relative performance in terms of revenue of different settings for the case where valuations are uniform on $[0,1]$ when $f_1 = 0$, and otherwise on $[0.75, 1.75]$. The fairness constraints are easier to satisfy in settings E and F than in setting D, and as expected, when d is set to require fairness, there is therefore a gap in revenue. Tables with complete data for revenue, unfairness and regret over a range of valuations are in Section 3.5.5 along with further discussion.

3.4.5 Comparing to Online Algorithms

To evaluate the performance of ProportionNet, we adapt the optimization and constraint objectives of Ilvento, Jagadeesan, and Chawla [95] and Celis, Mehrotra, and Vishnoi [38] to train networks which satisfy the same fairness constraints. These two algorithms consider the online auction setting, where ad impressions are sold in a single-item auction. As such, we can consider these algorithms as a baseline for applying fairness in the generalized multi-item setting, just as the itemwise Myerson is used as a baseline for the revenue-maximizing auction. We run experiments on Setting G:

G. 2 bidders with additive valuations over 2 items. Bidder $i's$ valuation for item j is drawn from $U[1,2]$ if $i = j$, and is otherwise drawn from $U[2,3]$.

Celis, Mehrotra, and Vishnoi [38] perform gradient descent to find the optimal α-shifted mechanism whose coverage q_{ij} across all advertisers i and users j satisfies their (ℓ,u)-fairness constraints: $\ell_{ij} \leq q_{ij}/\sum_{j\in M} q_{ij} \leq u_{ij}$.

They reduce the constraint to a single parameter $c \in [0.1, 0.5]$, which sets $\ell_{ij} = 1 - u_{ij} = c \; \forall i, j$ in the two item setting. Note that the optimal Myerson auction for a single item from Setting G (i.e. selling a single item to two bidders with valuations $v_{11} \sim U[1,2]$ and $v_{12} \sim U[2,3]$ has coverage $(q_{11}, q_{12}) = (\frac{1}{8}, \frac{7}{8})$. Therefore, for $c \leq \frac{1}{8}$, the optimal Myerson is also fair.

We approximate coverage in the multi-item setting as $q_{ij} = \sum_{b\sim S_2} g(b)$, where b is a sample from a secondary training dataset S_2 of valuations. To apply a soft (ℓ,u)-fairness constraint to ProportionNet, we use a new unfairness function for user j:

$$unf_j = \sum_{i\in N} relu(\ell_{ij} - q_{ij}) + relu(q_{ij} - u_{ij})$$

To train ProportionNet on this coverage constraint, we slightly modify the training procedure. Each iteration, the loss terms for revenue and regret are calculated using a minibatch B ($|B| = 128$) drawn from the main training dataset S ($|S| = 640{,}000$), while the fairness term is calculated using all samples in a secondary training dataset S_2 ($|S_2| = 40{,}000$). Figure 3.4 (right) shows

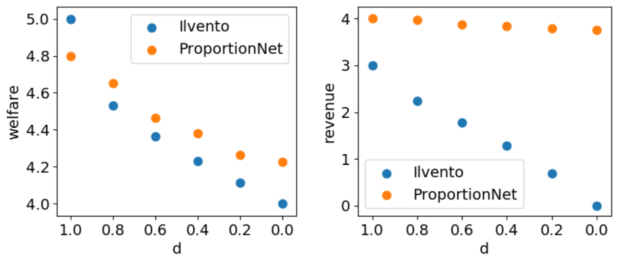


Figure 3.5: Welfare and revenue for Setting G subject to the uniform distance fairness parameter d.

ProportionNet's performance in revenue, which improves over [38]'s algorithm as the fairness constraint is tightened. On the testing samples, we achieve mean regret < 0.002 and zero unfairness for all configurations.

Ilvento, Jagadeesan, and Chawla [95] present a class of allocation functions which satisfy total-variation fairness while maintaining high welfare. As they do not consider revenue maximization or incentives, we compute an accompanying strategyproof payment rule using Myerson's lemma. Figure 3.5 shows ProportionNet's performance in both welfare and revenue while varying fairness parameter d and using the original unfairness function based on pairwise user distances. ProportionNet achieves mean unfairness and regret < 0.003 for all configurations. Improvement is to be expected, as the algorithm given by Ilvento, Jagadeesan, and Chawla [95] is agnostic to bidder valuations; it only guarantees a fair allocation which achieves a high ratio of the optimal welfare.

3.5 Conclusions & Future Work

Future work might include incorporating improvements to the training procedure as in Rahme, Jelassi, and Weinberg [132], or making use of techniques that can exactly evaluate the degree to which strategyproofness is violated, as in Curry et al. [50].

Additionally, the theoretical question of characterizing which fair, strategyproof mechanisms maximize revenue is an interesting one. Celis, Mehrotra, and Vishnoi [38] has provided some useful work in this direction already, for a specific class of auctions and notion of fairness; Kash and Frongillo [98] provides strong duality results which could be applied under fair allocation restrictions. Perhaps the use of deep-learning-based techniques to approximate fully general multi-item mechanisms can provide a starting point for theory as has happened in Dütting et al. [58]. Our experiments provide some hints about the fairness-revenue tradeoffs for specific offline auction settings, and a straightforward progression would be to formalize these tradeoffs for various classes of offline auctions.

We have chosen one definition of fairness to consider, but the question of which formalization of fairness is desirable in a given setting is a subtle one. Different formalizations of fairness may be valuable in different situations, and simply applying our techniques without careful thought and engagement with experts in law and ethics might do more harm than good. We do not claim to have the correct technique for designing fair auctions and discuss these issues further in Section 3.5.2.

Finally, while our work is motivated by the problem of unfairness in advertising auctions, our models are still quite stylized. Enhancing the realism of the model with real-world data, valuations, or fairness constraints derived from real settings would be quite interesting.

3.5.1 Unfairness in Real-World Ad Auctions

The ad auctions currently in place throughout the internet have been shown to produce discriminatory ad allocations. A core feature of online ad platforms is the ability to target users with certain properties. Thus, in an online ad allocation, platforms will typically consider additional factors in tandem with advertiser bids. For instance, key components include the demographics of their users and the target audience of the advertiser. Google's and Facebook's platforms take in user attributes (such as location, device type, and search query) as well as advertisement relevance in each auction [63, 78]. The practice of targeting advertisements has in the past lead to discriminatory allocations. For example, an experiment in 2013 [155] publicized the disproportionate likelihood of receiving online ads related to arrest records with a search query of a black-sounding name in contrast with a white-sounding name, even when the advertiser's preferred search queries and bids represented white and black sounding names equally. Additionally, [55] showed that between female and male users, male users with the same Google search queries tended to receive advertisements for higher-paying job offers than female users.

Facebook's auctions are similar to Google's, with an extended focus on user targeting, with over 2,000 differentiating user categories, including location, age, and income. A study in 2018 showcased the immensity of Facebook's resources, with the proven ability to target users by the single-person and single-household level. As highlighted in the study, this not only violates user privacy, but could put users in vulnerable locations, such as cancer treatment facilities, Planned Parenthood, and rehab centers, at risk [65]. Furthermore, Ali et al. [6] describes an automated advertisement classification system which determines the ideal demographic for an ad regardless of an advertiser's preferences. This feature has lead to discriminatory allocations based on users' race and gender for ads such as jobs and housing.

Finally, research exploring the disparity of advertisements of STEM job opportunities between male and female users has concluded that Facebook's determination of user prices could lead to discriminatory allocations. In other words, a low-bidding job advertisement that intends to advertise to all users regardless of gender may win more allocations with male users, because Facebook rates female users as more expensive, as women have been noted to interact with advertisements more [105]. Ongoing lawsuits regarding Facebook's discriminatory advertisement mechanism confirm unfairness within online ad auctions is a real concern [120].

3.5.2 Ethical Impact

Work in the field of automated mechanism design, including recent work like the RegretNet approach of Dütting et al. [58], show that tools and techniques from machine learning can help address persistent challenges in the theory of auctions. Our work builds on this body of research by positing the tools of machine learning can address another problem in auctions: fairness considerations on the item side. Above, we have shown compelling empirical evidence that support this claim based on the addition of fairness constraints to RegretNet's augmented Lagrangian technique.

One of the major social problems associated with online advertising is its use in the job market. The value proposition of online advertising often involves targeting ads to specific demographic groups, and this is a serious problem when those groups may represent, even indirectly, protected classes. At least in the United States, antidiscrimination law is codified in Title VII of the Civil Rights Act of 1964 which limits the type of behaviors employers can engage in. The US Supreme Court decided in Griggs v. Duke Power Co. [81], that certain behaviors which might cause discriminatory results, even if they are performed unintentionally, are illegal – the doctrine of "disparate impact". The Court has shied away from rigidly defining disparate impact (in quantitative terms) [135], but the Equal Employment Opportunity Commission (EEOC) makes determinations about disparate impact based on the 80% rule [60]. This rule generally states that a group of individuals in a protected class cannot have a selection rate less than 80% of the highest rate for another class. Since this generally can be mathematically formalized, it has been studied in the fair ML literature from a practical and technical perspective [26, 67]. Our definition of fairness does not map directly onto the 80% rule, but it shares some similarities; when the distance metric is defined in terms of protected classes, it arguably constrains allocations from having "disparate impact".

While in some sense, we thus provide a way to learn a mechanism that will satisfy widely-held definitions of fairness, to view our proposed approach as a cure-all would be misguided. There is a growing body of work that shows there is significant daylight between how a computer scientist thinks about fairness and how others do. Holstein et al. [93] conducted interviews with developers about their desired fairness outcomes and showed them to sometimes be at odds with each other. Saha et al. [143] demonstrated that laypeople often don't comprehend computer science notions of fairness as well. Then there are also more sociological critiques of the general fair ML approaches writ large [87, 150]. As Selbst et al. [150] and Hutchinson and Mitchell [94] point out, the reification of fairness concepts into mathematical formulae has inherent problems. Fair ML has achieved prominence through the translation of nebulous and debatable definitions into concrete mathematics, often providing a veneer of objectivity over highly contested notions of equality and justice.

We acknowledge that our work at its present stage is a technical intervention, rather than an analysis or critique of a sociotechnical system. We do not aim to make prescriptive statements on our own about the ultimately correct way to design auctions that must be fair, as that is best done in an interdisciplinary group with multiple stakeholders. However, we hope that our work can be one useful contribution to the challenging problem of fair mechanism design.

3.5.3 Generalization Bound for Unfairness

We restate the theorem below:

Theorem 3.1. *Let $\mathscr{M}$ be a class of auctions that satisfy individual rationality and have $\ell_{\infty,1}$ covering number $\mathscr{N}_\infty(\mathscr{M},\cdot)$. Fix $\delta \in (0,1)$. With probability at least $1-\delta$ over a draw of L valuation profiles, for any $(g^w, p^w) \in \mathscr{M}$,*

$$\mathbb{E}_v\left[\sum_{j=1}^{m} \mathrm{unf}_j \circ g^w(v)\right] \leq \frac{1}{L}\sum_{\ell=1}^{L}\sum_{j=1}^{m} \mathrm{unf}_j \circ g^w(v^\ell) + 2D_L + 4C\sqrt{\frac{2\log(4/\delta)}{L}}$$

where C is a constant and

$$D_L = \inf_{\varepsilon>0}\left((nm^2+\varepsilon)\sqrt{\frac{2\log \mathscr{N}_\infty(\mathscr{M},\frac{\varepsilon}{2m^3})}{L}} + \varepsilon\right).$$

Proof. Let $\mathscr{G}_j$ be the class of **item-wise** allocation functions for item j defined on a class of auctions $\mathscr{M}$.

Let $\mathrm{unf}_j \circ G$ be the class of **unfairness functions** for item j. A function $f_j \in \mathrm{unf}_j \circ G$ maps $f_j : V \to \mathbb{R}$. Extended to all items, $\mathrm{unf} \circ G$ is the class of tuples $(f_1, ..., f_m)$. Such a vector-valued function $f \in \mathrm{unf} \circ G$ maps $f : V \to \mathbb{R}^m$. Finally, we also define the class of **sum unfairness functions**:

$$\overline{\mathrm{unf}} \circ G = \{f : V \to \mathbb{R} \mid f(v) = \sum_{j=1}^{m} f_j(v) \text{ for some } (f_1, ...f_m) \in \mathrm{unf} \circ G_j\}$$

We prove bounds for the simple case of a uniform distance constraint d between all users, with all bidders in one advertising category. In this case, given a mechanism (g, p), the quantity for item

j's unfairness is:

$$\mathrm{unf}_j(v) = \sum_{j' \in M} \max(0, \sum_{i \in N} \max(0, g_{i,j}(v) - g_{i,j'}(v)) - d)$$

Our proof hews very closely to the generalization bound for regret in D.2.4 of [58]. We use the same notion of $\ell_{\infty,1}$ distance between functions and, under this distance, relate covering numbers of the function classes defined above:

$$\mathcal{N}_\infty(\overline{\mathrm{unf}} \circ \mathcal{G}, \varepsilon) \leq \mathcal{N}_\infty(\mathrm{unf} \circ \mathcal{G}, \frac{\varepsilon}{m}) \leq \mathcal{N}_\infty(\mathcal{G}, \frac{\varepsilon}{2m^3}) \leq \mathcal{N}_\infty(\mathcal{M}, \frac{\varepsilon}{2m^3})$$

These covering numbers in turn bound empirical Rademacher complexity, which allows us to apply the same lemma of [151] used in [58].

Step 1. Bounding $\mathcal{N}_\infty(\mathcal{G}, \varepsilon) \leq \mathcal{N}_\infty(\mathcal{M}, \varepsilon)$

By the definition of $\mathcal{N}_\infty(\mathcal{M}, \varepsilon)$, there exists some cover $\hat{\mathcal{M}}$ (where $|\hat{\mathcal{M}}| \leq \mathcal{N}_\infty(\mathcal{M}, \varepsilon)$) such that $\forall (g, p) \in \mathcal{M}, \exists (\hat{g}, \hat{p}) \in \hat{\mathcal{M}}$ where

$$\sup_{v \in V} \sum_{i,j} |g_{i,j}(v) - \hat{g}_{i,j}(v)| + \|p(v) - \hat{p}(v)\|_1 \leq \varepsilon.$$

It is trivial to bound the distance between any $g \in \mathcal{G}$ and its covering $\hat{g} \in \hat{\mathcal{G}}$, $\forall v \in V$:

$$\sum_{j \in M} \|g_{\cdot,j} - \hat{g}_{\cdot,j}\|_1 = \sum_{i,j} |g_{i,j}(v) - \hat{g}_{i,j}(v)| \leq \sum_{i,j} |g_{i,j}(v) - \hat{g}_{i,j}(v)| + \|p(v) - \hat{p}(v)\|_1 \leq \varepsilon.$$

Therefore, $\mathcal{N}_\infty(\mathcal{G}, \varepsilon) \leq \mathcal{N}_\infty(\mathcal{M}, \varepsilon)$

Step 2. Bounding $\mathcal{N}_\infty(\mathrm{unf} \circ \mathcal{G}, \varepsilon) \leq \mathcal{N}_\infty(\mathcal{G}, \frac{\varepsilon}{2m^2})$

We first bound $\mathcal{N}_\infty(\mathrm{unf}_j \circ \mathcal{G}, \varepsilon) \leq \mathcal{N}_\infty(\mathcal{G}, \frac{\varepsilon}{2m})$ for a single j.

Taking $g, \hat{g}$ satisfying the definition of $\mathcal{N}_\infty(\mathcal{G}, \frac{\varepsilon}{2m})$ and fixing a fairness parameter $d \in [0,1]$, we bound the $\ell_{\infty,1}$ distance between $\mathrm{unf}_j \circ g$ and $\mathrm{unf}_j \circ \hat{g}$. Note that $g_j(v)$ has been shortened to g_j, $g_{i,j}(v)$ to $g_{i,j}$, etc. for convenience. We use the fact that $|\max(0,a) - \max(0,b)| \le |a-b|$.

$$\sup_{v \in V} \left| \mathrm{unf}_j \circ g - \mathrm{unf}_j \circ \hat{g} \right|$$

$$= \sup_{v \in V} \left| \sum_{j' \in M} \max(0, \left(\sum_{i \in N} \max(0, g_{i,j} - g_{i,j'}) \right) - d) - \sum_{j' \in M} \max(0, \left(\sum_{i \in N} \max(0, \hat{g}_{i,j} - \hat{g}_{i,j'}) \right) - d) \right|$$

$$\le \sup_{v \in V} \sum_{j' \in M} \left| \max(0, \left(\sum_{i \in N} \max(0, g_{i,j} - g_{i,j'}) \right) - d) - \max(0, \left(\sum_{i \in N} \max(0, \hat{g}_{i,j} - \hat{g}_{i,j'}) \right) - d) \right|$$

$$\le \sup_{v \in V} \sum_{j' \in M} \left| \sum_{i \in N} \max(0, g_{i,j} - g_{i,j'}) - \sum_{i \in N} \max(0, \hat{g}_{i,j} - \hat{g}_{i,j'}) \right|$$

$$\le \sup_{v \in V} \sum_{j' \in M} \sum_{i \in N} \left| \max(0, g_{i,j} - g_{i,j'}) - \max(0, \hat{g}_{i,j} - \hat{g}_{i,j'}) \right|$$

$$\le \sup_{v \in V} \sum_{j' \in M} \sum_{i \in N} \left| g_{i,j} - g_{i,j'} - \hat{g}_{i,j} + \hat{g}_{i,j'} \right|$$

$$= \sup_{v \in V} \sum_{j' \in M} \sum_{i \in N} \left| (g_{i,j} - \hat{g}_{i,j}) - (g_{i,j'} - \hat{g}_{i,j'}) \right|$$

$$\le \sup_{v \in V} \sum_{j' \in M} \sum_{i \in N} \left| g_{i,j} - \hat{g}_{i,j} \right| + \sup_{v \in V} \sum_{j' \in M} \sum_{i \in N} \left| g_{i,j'} - \hat{g}_{i,j'} \right|$$

$$= \sup_{v \in V} m \cdot \sum_{i \in N} \left| g_{i,j} - \hat{g}_{i,j} \right| + \sup_{v \in V} \sum_{j' \in M} \sum_{i \in N} \left| g_{i,j'} - \hat{g}_{i,j'} \right|$$

$$\le \sup_{v \in V} m \cdot \sum_{j \in M} \sum_{i \in N} \left| g_{i,j} - \hat{g}_{i,j} \right| + \sup_{v \in V} \sum_{j' \in M} \sum_{i \in N} \left| g_{i,j'} - \hat{g}_{i,j'} \right| \le \frac{(m+1)\varepsilon}{2m} \le \varepsilon$$

where the second-to-last inequality follows due to the definition of the cover to which $\hat{g}_{i,j}, \hat{g}_{i,j'}$ belong.

Therefore, $\mathcal{N}_\infty(\mathrm{unf}_j \circ \mathcal{G}, \varepsilon) \le \mathcal{N}_\infty(\mathcal{G}, \frac{\varepsilon}{2m})$, which implies $\mathcal{N}_\infty(\mathrm{unf} \circ \mathcal{G}, \varepsilon) \le \mathcal{N}_\infty(\mathcal{G}, \frac{\varepsilon}{2m^2})$

Step 3. Bounding $\mathscr{N}(\overline{\text{unf}} \circ \mathscr{G}, \varepsilon) \leq \mathscr{N}(\text{unf} \circ \mathscr{G}, \frac{\varepsilon}{m})$

We take $g_j, \hat{g}_j$ from a cover of $\text{unf} \circ \mathscr{G}$. The ℓ_∞ distance between $\overline{\text{unf}} \circ g$ and $\overline{\text{unf}} \circ \hat{g}$ is:

$$\sup_{v \in V} \left| \overline{\text{unf}} \circ g - \overline{\text{unf}} \circ \hat{g} \right| \leq \sum_{j \in M} \sup_{v \in V} \left| \text{unf}_j \circ g - \text{unf}_j \circ \hat{g} \right| \leq \sum_{j \in M} \frac{\varepsilon}{m} = \varepsilon,$$

Therefore, $\mathscr{N}_\infty(\overline{\text{unf}} \circ \mathscr{G}, \varepsilon) \leq \mathscr{N}_\infty(\text{unf} \circ \mathscr{G}, \frac{\varepsilon}{m})$.

Combining these inequalities, we get $\mathscr{N}_\infty(\overline{\text{unf}} \circ \mathscr{G}, \varepsilon) \leq \mathscr{N}_\infty(\mathscr{M}, \frac{\varepsilon}{2m^3})$ as desired.

Applying bounds

For convenience denote $\mathscr{F} = \overline{\text{unf}} \circ \mathscr{G}$, with $\hat{\mathscr{F}}$ as its cover. Denote by $\hat{f}_f \in \hat{\mathscr{F}}$ the closest covering point to some $f \in \mathscr{F}$.

We wish to bound the empirical Rademacher complexity $\hat{\mathscr{R}}_L(\mathscr{F})$, which is

$$\begin{aligned}
&\frac{1}{L}\mathbb{E}_\sigma \left[\sup_{f \in \mathscr{F}} \sum_{\ell=1}^{L} \sigma_\ell f(v^\ell) \right] \\
&= \frac{1}{L}\mathbb{E}_\sigma \left[\sup_{f \in \mathscr{F}} \sum_{\ell=1}^{L} \sigma_\ell \left(\hat{f}_f(v^\ell) + f(v^\ell) - \hat{f}_f(v^\ell) \right) \right] \\
&\leq \frac{1}{L}\mathbb{E}_\sigma \left[\sup_{f \in \mathscr{F}} \sum_{\ell=1}^{L} \sigma_\ell \hat{f}_f(v^\ell) \right] + \frac{1}{L}\mathbb{E}_\sigma \left[\sup_{f \in \mathscr{F}} \sum_{\ell=1}^{L} \left(f(v^\ell) - \hat{f}_f(v^\ell) \right) \right] \\
&\leq \frac{1}{L}\mathbb{E}_\sigma \left[\sup_{f \in \mathscr{F}} \sum_{\ell=1}^{L} \sigma_\ell \hat{f}_f(v^\ell) \right] + \frac{1}{L}\mathbb{E}_\sigma \left[\sum_{\ell=1}^{L} \sup_{f \in \mathscr{F}} \left(f(v^\ell) - \hat{f}_f(v^\ell) \right) \right] \\
&\leq \frac{1}{L}\mathbb{E}_\sigma \left[\sup_{f \in \mathscr{F}} \sum_{\ell=1}^{L} \sigma_\ell \hat{f}_f(v^\ell) \right] + \varepsilon \\
&= \frac{1}{L}\mathbb{E}_\sigma \left[\sup_{\hat{f} \in \hat{\{}} \sum_{\ell=1}^{L} \sigma_\ell \hat{f}(v^\ell) \right] + \varepsilon \\
&\leq \frac{1}{L} \max_{\hat{f} \in \hat{\mathscr{F}}} \sqrt{\sum_{\ell=1}^{L} \hat{f}(v^\ell)^2} \sqrt{2 \log \mathscr{N}_\infty(\mathscr{F}, \varepsilon)} + \varepsilon \text{ (by Massart's lemma)} \\
&\leq \frac{1}{L} \max_{f \in \mathscr{F}} \sqrt{\sum_{\ell=1}^{L} \left(f(v^\ell) + \varepsilon \right)^2} \sqrt{2 \log \mathscr{N}_\infty(\mathscr{F}, \varepsilon)} + \varepsilon \\
&\leq \frac{1}{L} \sqrt{\sum_{\ell=1}^{L} \left(nm^2 + \varepsilon \right)^2} \sqrt{2 \log \mathscr{N}_\infty(\mathscr{F}, \varepsilon)} + \varepsilon
\end{aligned}$$

$$\leq \frac{1}{L}\sqrt{\sum_{\ell=1}^{L}(nm^2+\varepsilon)^2}\sqrt{2\log\mathcal{N}_\infty(\mathscr{F},\varepsilon)}+\varepsilon$$
$$=\left(nm^2+\varepsilon\right)\sqrt{\frac{2\log\mathcal{N}_\infty(\mathscr{F},\varepsilon)}{L}}+\varepsilon$$

Given $\mathcal{N}_\infty(\mathscr{F},\varepsilon) \leq \mathcal{N}_\infty(\mathscr{M},\frac{\varepsilon}{2m})$, we can then apply the lemma of [151] as in [58] to say that with probability $1-\delta$, for a distribution-independent constant C:

$$\begin{aligned}\mathbb{E}_v[f(v)] &\leq \frac{1}{L}\sum_{\ell=1}^{L}f(v^\ell)+2\hat{\mathscr{R}}_L(\mathscr{F})+4C\sqrt{\frac{2\log(4/\delta)}{L}}\\ &\leq \frac{1}{L}\sum_{\ell=1}^{L}f(v^\ell)+\\ &\quad 2\inf_{\varepsilon>0}\left((nm^2+\varepsilon)\sqrt{\frac{2\log\mathcal{N}_\infty(\mathscr{M},\frac{\varepsilon}{2m^3})}{L}}+\varepsilon\right)\\ &\quad +4C\sqrt{\frac{2\log(4/\delta)}{L}}\end{aligned}$$

□

3.5.4 Additional Results

The following tables show the mean and standard deviation of revenue, regret, and unfairness for all configurations of Settings C, D, E, and F. We consistently achieve low values for regret and unfairness. A non-vanishing value as the network is trained implies that the auction is empirically non-strategyproof (if regret cannot be minimized) or unfair (if unfairness cannot be minimized).

3.5.5 Non-uniform Fairness

We further describe the settings involving non-uniform fairness. Each auction has three bidders and four items denoted $u_{1...4}$. Each item has binary features f_1 and f_2 which are used to construct the distance values used in our fairness constraint – in an ad auction setting, these could be characteristics of different groups of users. Values of these features are in table 3.14. We restate

Sweep Revenue, Mean (StDev)							
n x m	ℓ	Myr	D				
-	-	-	1.00	0.75	0.50	0.25	0.00
1 x 2	2	0.50	0.546 (0.369)	0.545 (0.363)	0.544 (0.367)	0.541 (0.353)	0.538 (0.374)
1 x 3	2	0.75	0.858 (0.504)	0.856 (0.491)	0.844 (0.482)	0.845 (0.474)	0.845 (0.484)
1 x 4	2	1.00	1.199 (0.602)	1.191 (0.579)	1.182 (0.591)	1.189 (0.591)	1.180 (0.589)
1 x 5	2	1.25	1.540 (0.720)	1.532 (0.714)	1.533 (0.711)	1.529 (0.698)	1.534 (0.720)
1 x 6	2	1.50	1.877 (0.832)	1.895 (0.801)	1.892 (0.805)	1.892 (0.811)	1.886 (0.819)
2 x 2	2	0.83	0.865 (0.349)	0.858 (0.343)	0.852 (0.334)	0.838 (0.327)	0.830 (0.323)
2 x 3	2	1.25	1.234 (0.399)	1.221 (0.388)	1.215 (0.382)	1.209 (0.383)	1.206 (0.382)
2 x 4	2	1.67	1.720 (0.462)	1.709 (0.452)	1.683 (0.444)	1.680 (0.438)	1.681 (0.438)
2 x 5	3	2.08	2.194 (0.525)	2.168 (0.509)	2.138 (0.496)	2.135 (0.492)	2.133 (0.489)
2 x 6	3	2.50	2.665 (0.570)	2.631 (0.552)	2.594 (0.537)	2.594 (0.533)	2.591 (0.531)
3 x 2	2	1.06	1.056 (0.287)	1.036 (0.280)	1.022 (0.274)	1.010 (0.269)	1.004 (0.268)
3 x 3	2	1.59	1.546 (0.344)	1.533 (0.336)	1.502 (0.326)	1.499 (0.323)	1.500 (0.324)
3 x 4	3	2.12	2.081 (0.404)	2.040 (0.386)	2.011 (0.374)	2.003 (0.373)	2.001 (0.371)
3 x 5	3	2.66	2.577 (0.443)	2.540 (0.426)	2.510 (0.416)	2.498 (0.413)	2.495 (0.412)
3 x 6	3	3.19	3.073 (0.481)	3.048 (0.471)	3.011 (0.462)	3.013 (0.458)	3.007 (0.456)
4 x 2	2	1.23	1.209 (0.302)	1.177 (0.276)	1.151 (0.256)	1.135 (0.242)	1.129 (0.244)
4 x 3	3	1.84	1.769 (0.286)	1.720 (0.284)	1.678 (0.286)	1.645 (0.293)	1.642 (0.294)
4 x 4	3	2.45	2.247 (0.302)	2.204 (0.307)	2.156 (0.334)	2.159 (0.334)	2.158 (0.336)
4 x 5	4	3.06	2.819 (0.426)	2.765 (0.394)	2.707 (0.375)	2.692 (0.377)	2.693 (0.379)
4 x 6	4	3.68	4.284 (0.491)	3.279 (0.421)	3.246 (0.413)	3.306 (0.399)	3.007 (0.447)
5 x 2	3	1.34	1.307 (0.295)	1.305 (0.242)	1.268 (0.228)	1.239 (0.221)	1.230 (0.224)
5 x 3	3	2.02	1.906 (0.360)	1.894 (0.301)	1.824 (0.273)	1.777 (0.263)	1.767 (0.269)
5 x 4	4	2.69	2.389 (0.326)	2.381 (0.349)	2.271 (0.305)	2.291 (0.314)	2.318 (0.314)
5 x 5	4	3.36	3.670 (0.402)	2.906 (0.358)	2.876 (0.353)	2.851 (0.348)	2.836 (0.352)
5 x 6	5	4.03	3.489 (0.618)	3.467 (0.417)	3.491 (0.420)	3.445 (0.493)	3.385 (0.370)

Table 3.2: Revenue for Setting C – auctions with $U[0,1]$ valuations for n bidders and m items. Note that for the 5 x 6 setting, the network converged to a suboptimal solution with both high payment and high regret.

the three settings three settings which have identical valuations but are subject to different fairness constraints.

D. All bidders are constrained by a distance metric on f_2.

E. Setting D, and the f_2 feature of user u_1 is changed from 0 to 1.

F. Setting D, but bidder 3 is constrained by a distance metric on f_1 rather than f_2.

Sweep Regret, Mean (StDev)						
n x m	ℓ	D				
-	-	1.00	0.75	0.50	0.25	0.00
1 x 2	2	0.000 (0.000)	0.000 (0.001)	0.000 (0.000)	0.000 (0.000)	0.001 (0.001)
1 x 3	2	0.001 (0.001)	0.001 (0.001)	0.000 (0.000)	0.001 (0.001)	0.001 (0.000)
1 x 4	2	0.001 (0.001)	0.000 (0.001)	0.001 (0.001)	0.001 (0.001)	0.001 (0.001)
1 x 5	2	0.001 (0.002)	0.001 (0.001)	0.001 (0.001)	0.001 (0.001)	0.001 (0.000)
1 x 6	2	0.001 (0.001)	0.001 (0.001)	0.001 (0.001)	0.001 (0.001)	0.001 (0.000)
2 x 2	2	0.001 (0.001)	0.001 (0.001)	0.001 (0.001)	0.001 (0.001)	0.002 (0.002)
2 x 3	2	0.002 (0.003)	0.001 (0.002)	0.002 (0.003)	0.001 (0.001)	0.001 (0.001)
2 x 4	2	0.002 (0.003)	0.002 (0.003)	0.001 (0.002)	0.001 (0.002)	0.002 (0.002)
2 x 5	3	0.003 (0.004)	0.002 (0.003)	0.004 (0.006)	0.001 (0.002)	0.001 (0.001)
2 x 6	3	0.003 (0.004)	0.003 (0.004)	0.002 (0.003)	0.003 (0.005)	0.002 (0.003)
3 x 2	2	0.002 (0.002)	0.002 (0.002)	0.001 (0.001)	0.001 (0.002)	0.001 (0.001)
3 x 3	2	0.003 (0.003)	0.003 (0.003)	0.001 (0.002)	0.002 (0.002)	0.001 (0.001)
3 x 4	3	0.006 (0.006)	0.004 (0.005)	0.002 (0.003)	0.002 (0.002)	0.002 (0.002)
3 x 5	3	0.006 (0.007)	0.005 (0.006)	0.002 (0.005)	0.002 (0.003)	0.003 (0.003)
3 x 6	3	0.009 (0.011)	0.006 (0.007)	0.004 (0.005)	0.004 (0.005)	0.002 (0.002)
4 x 2	2	0.003 (0.002)	0.003 (0.002)	0.003 (0.002)	0.002 (0.002)	0.001 (0.001)
4 x 3	3	0.004 (0.003)	0.004 (0.002)	0.003 (0.003)	0.002 (0.002)	0.001 (0.001)
4 x 4	3	0.006 (0.004)	0.006 (0.004)	0.002 (0.002)	0.003 (0.003)	0.002 (0.003)
4 x 5	4	0.007 (0.005)	0.007 (0.004)	0.004 (0.002)	0.003 (0.004)	0.003 (0.003)
4 x 6	4	0.010 (0.076)	0.007 (0.006)	0.006 (0.006)	0.006 (0.007)	0.003 (0.003)
5 x 2	3	0.003 (0.003)	0.005 (0.004)	0.005 (0.003)	0.004 (0.003)	0.005 (0.006)
5 x 3	3	0.006 (0.003)	0.010 (0.005)	0.008 (0.004)	0.006 (0.004)	0.003 (0.003)
5 x 4	4	0.007 (0.004)	0.011 (0.005)	0.006 (0.008)	0.003 (0.004)	0.003 (0.004)
5 x 5	4	0.003 (0.045)	0.008 (0.007)	0.007 (0.008)	0.005 (0.010)	0.002 (0.003)
5 x 6	5	0.190 (0.377)	0.005 (0.008)	0.005 (0.015)	0.005 (0.010)	0.003 (0.004)

Table 3.3: Regret for Setting C.

Sweep Unfairness, Mean (StDev)						
n x m	ℓ	D				
-	-	1.00	0.75	0.50	0.25	0.00
1 x 2	2	0.000 (0.000)	0.000 (0.000)	0.000 (0.000)	0.000 (0.000)	0.002 (0.002)
1 x 3	2	0.000 (0.000)	0.000 (0.001)	0.000 (0.004)	0.000 (0.002)	0.005 (0.010)
1 x 4	2	0.000 (0.000)	0.000 (0.000)	0.000 (0.004)	0.000 (0.005)	0.009 (0.018)
1 x 5	2	0.000 (0.000)	0.000 (0.009)	0.000 (0.008)	0.000 (0.007)	0.006 (0.015)
1 x 6	2	0.000 (0.000)	0.000 (0.006)	0.000 (0.003)	0.000 (0.009)	0.009 (0.021)
2 x 2	2	0.000 (0.000)	0.000 (0.003)	0.000 (0.003)	0.000 (0.001)	0.006 (0.010)
2 x 3	2	0.000 (0.000)	0.000 (0.003)	0.000 (0.003)	0.000 (0.000)	0.005 (0.009)
2 x 4	2	0.000 (0.000)	0.000 (0.006)	0.000 (0.001)	0.000 (0.000)	0.015 (0.023)
2 x 5	3	0.000 (0.000)	0.000 (0.007)	0.000 (0.000)	0.000 (0.000)	0.008 (0.011)
2 x 6	3	0.000 (0.000)	0.003 (0.043)	0.001 (0.017)	0.000 (0.000)	0.031 (0.047)
3 x 2	2	0.000 (0.000)	0.001 (0.006)	0.001 (0.007)	0.000 (0.006)	0.004 (0.008)
3 x 3	2	0.000 (0.000)	0.000 (0.007)	0.000 (0.001)	0.000 (0.000)	0.008 (0.013)
3 x 4	3	0.000 (0.000)	0.000 (0.008)	0.000 (0.002)	0.000 (0.000)	0.012 (0.016)
3 x 5	3	0.000 (0.000)	0.001 (0.019)	0.002 (0.031)	0.000 (0.000)	0.023 (0.034)
3 x 6	3	0.000 (0.000)	0.001 (0.011)	0.001 (0.015)	0.001 (0.023)	0.017 (0.024)
4 x 2	2	0.000 (0.000)	0.001 (0.003)	0.000 (0.004)	0.000 (0.001)	0.004 (0.006)
4 x 3	3	0.000 (0.000)	0.001 (0.014)	0.001 (0.013)	0.000 (0.001)	0.009 (0.012)
4 x 4	3	0.000 (0.000)	0.001 (0.011)	0.000 (0.000)	0.000 (0.000)	0.016 (0.020)
4 x 5	4	0.000 (0.000)	0.000 (0.006)	0.000 (0.000)	0.000 (0.000)	0.022 (0.033)
4 x 6	4	0.000 (0.000)	0.001 (0.014)	0.000 (0.010)	0.002 (0.022)	0.020 (0.026)
5 x 2	3	0.000 (0.000)	0.000 (0.001)	0.000 (0.003)	0.000 (0.002)	0.008 (0.013)
5 x 3	3	0.000 (0.000)	0.008 (0.022)	0.010 (0.032)	0.003 (0.024)	0.018 (0.026)
5 x 4	4	0.000 (0.000)	0.001 (0.006)	0.000 (0.000)	0.000 (0.000)	0.008 (0.013)
5 x 5	4	0.000 (0.000)	0.000 (0.003)	0.000 (0.002)	0.000 (0.003)	0.007 (0.010)
5 x 6	5	0.000 (0.000)	0.000 (0.000)	0.000 (0.000)	0.000 (0.000)	0.011 (0.016)

Table 3.4: Unfairness for Setting C.

Setting D Revenue: Mean (StDev)						
b	Myr	d				
-	-	1.00	0.75	0.50	0.25	0.00
0.00	2.125	2.043 (0.396)	2.018 (0.381)	1.999 (0.376)	1.992 (0.373)	1.989 (0.376)
0.25	2.582	2.544 (0.397)	2.515 (0.383)	2.486 (0.376)	2.489 (0.375)	2.494 (0.377)
0.50	3.066	3.037 (0.395)	3.016 (0.384)	2.990 (0.376)	2.988 (0.376)	2.997 (0.379)
0.75	3.562	3.540 (0.394)	3.515 (0.384)	3.489 (0.376)	3.487 (0.376)	3.493 (0.380)
1.00	4.062	4.037 (0.392)	4.010 (0.383)	3.987 (0.375)	3.988 (0.377)	3.996 (0.379)

Table 3.5: Revenue for Setting D – a 3 bidder x 4 item auction with items 1 and 2 valued at $U[0,1]$ and items 3 and 4 at $U[0,1]+b$ by all bidders. Fair allocations are enforced for item pairs (1,3) and (2,4).

Setting D Regret: Mean (StDev)					
-	1.00	0.75	0.50	0.25	0.00
0.00	0.004 (0.006)	0.004 (0.004)	0.002 (0.003)	0.002 (0.002)	0.001 (0.002)
0.25	0.004 (0.005)	0.004 (0.004)	0.002 (0.003)	0.003 (0.004)	0.001 (0.001)
0.50	0.005 (0.005)	0.003 (0.004)	0.002 (0.003)	0.002 (0.002)	0.001 (0.002)
0.75	0.006 (0.008)	0.003 (0.004)	0.003 (0.004)	0.002 (0.002)	0.002 (0.002)
1.00	0.006 (0.006)	0.003 (0.004)	0.004 (0.007)	0.001 (0.002)	0.001 (0.002)

Table 3.6: Regret for Setting D.

Again, our fairness constraint for some fairness parameter d is

$$d(j,j',f_i) = d' + (1-d')|f_i(u_j) - f_i(u_{j'})|,$$

Suppose the auction is for software engineer hiring ads, f_1 denotes a feature (e.g. gender) which is irrelevant to the auction, and f_2 is a relevant feature (e.g. a computer science degree). We are interested in how ProportionNet handles the fairness constraint when valuations are artificially raised for users with the f_1 feature, which can be viewed as discriminatory. The new valuation range is $[0,1] + b|f_1(u_j) - f_1(u'_j)|$ for all three bidders, where $b \in \mathbb{R}$ is a parameter that adjusts the level of discriminatory bidding behavior. We tested a grid of values $b,d = 0, 0.25, ..., 1.00$.

Although a 3-bidder, 4-item auction is far outside the reach of analysis, these results allow us to conjecture as to the effects of the different fairness constraints we are applying. Viewing Setting D as a baseline, we see that Setting E yields slightly more revenue as fairness is more strict ($d \geq 0.50$).

Setting D Unfairness: Mean (StDev)					
-	1.00	0.75	0.50	0.25	0.00
0.00	0.000 (0.000)	0.000 (0.006)	0.000 (0.003)	0.000 (0.000)	0.002 (0.003)
0.25	0.000 (0.000)	0.000 (0.004)	0.000 (0.000)	0.000 (0.000)	0.001 (0.002)
0.50	0.000 (0.000)	0.000 (0.005)	0.000 (0.000)	0.000 (0.000)	0.001 (0.002)
0.75	0.000 (0.000)	0.000 (0.005)	0.000 (0.000)	0.000 (0.000)	0.001 (0.003)
1.00	0.000 (0.000)	0.035 (0.001)	0.000 (0.000)	0.000 (0.000)	0.001 (0.002)

Table 3.7: Unfairness for Setting D.

Setting E Revenue: Mean (StDev)						
b	Myr	d				
-	-	1.00	0.75	0.50	0.25	0.00
0.00	2.125	2.043 (0.396)	2.031 (0.389)	2.039 (0.392)	2.041 (0.392)	2.033 (0.393)
0.25	2.582	2.544 (0.397)	2.521 (0.388)	2.528 (0.390)	2.537 (0.394)	2.525 (0.393)
0.50	3.066	3.037 (0.395)	3.025 (0.385)	3.005 (0.384)	3.006 (0.383)	3.010 (0.385)
0.75	3.562	3.540 (0.394)	3.517 (0.386)	3.505 (0.383)	3.511 (0.384)	3.509 (0.385)
1.00	4.062	4.037 (0.392)	4.019 (0.381)	3.999 (0.382)	4.015 (0.382)	4.006 (0.385)

Table 3.8: Revenue for Setting E – identical to setting D but fair allocations are enforced for the user triplet (2, 3, 4).

This is likely because the users u_1 and u_3 required similar ad allocations in Setting D (due to the shared value of $f_2 = 0$). Changing u_1's f_2 feature from 0 to 1 in Setting E relaxes this constraint and allows the auctioneer to extract more value from bids on u_3. Setting F also shows an increase in revenue, though not as pronounced as Setting E. Likewise, we interpret this as a result of the relaxing of fairness constraints. In Setting F, allocations to bidder 3 of $u_{1,2}$ can be similarly low, and that of $u_{3,4}$ similarly high; such allocations would have been considered unfair in Setting D.

Setting E Regret: Mean (StDev)					
b	d				
-	1.00	0.75	0.50	0.25	0.00
0.00	0.004 (0.006)	0.005 (0.006)	0.003 (0.003)	0.003 (0.003)	0.002 (0.003)
0.25	0.004 (0.005)	0.003 (0.003)	0.002 (0.003)	0.003 (0.003)	0.003 (0.003)
0.50	0.005 (0.005)	0.005 (0.006)	0.002 (0.003)	0.003 (0.004)	0.003 (0.005)
0.75	0.006 (0.008)	0.006 (0.007)	0.002 (0.003)	0.003 (0.004)	0.002 (0.004)
1.00	0.006 (0.006)	0.006 (0.006)	0.003 (0.004)	0.003 (0.003)	0.003 (0.004)

Table 3.9: Regret for Setting E.

Setting E Unfairness: Mean (StDev)					
-	1.00	0.75	0.50	0.25	0.00
0.00	0.000 (0.000)	0.000 (0.005)	0.000 (0.000)	0.000 (0.000)	0.002 (0.003)
0.25	0.000 (0.000)	0.000 (0.006)	0.000 (0.000)	0.000 (0.000)	0.001 (0.003)
0.50	0.000 (0.000)	0.000 (0.005)	0.000 (0.000)	0.000 (0.000)	0.001 (0.003)
0.75	0.000 (0.000)	0.000 (0.003)	0.000 (0.000)	0.000 (0.000)	0.002 (0.005)
1.00	0.000 (0.000)	0.000 (0.000)	0.000 (0.000)	0.000 (0.000)	0.002 (0.005)

Table 3.10: Unfairness for Setting E.

Setting F Revenue: Mean (StDev)						
b	Myr	d				
-	-	1.00	0.75	0.50	0.25	0.00
0.00	2.125	2.043 (0.396)	2.022 (0.385)	2.004 (0.377)	1.992 (0.373)	1.992 (0.375)
0.25	2.582	2.544 (0.397)	2.525 (0.387)	2.494 (0.377)	2.492 (0.375)	2.495 (0.375)
0.50	3.066	3.037 (0.395)	3.022 (0.386)	2.991 (0.375)	2.987 (0.376)	2.994 (0.377)
0.75	3.562	3.540 (0.394)	3.514 (0.382)	3.494 (0.376)	3.492 (0.375)	3.493 (0.377)
1.00	4.062	4.037 (0.392)	4.012 (0.384)	3.988 (0.379)	3.992 (0.376)	3.993 (0.377)

Table 3.11: Revenue for Setting F – identical to setting D but bidder 3 must receive fair allocations for user pairs (1,2) and (3,4).

Setting F Regret: Mean (StDev)					
b	d				
-	1.00	0.75	0.50	0.25	0.00
0.00	0.004 (0.006)	0.004 (0.004)	0.003 (0.004)	0.003 (0.004)	0.003 (0.003)
0.25	0.004 (0.005)	0.005 (0.005)	0.004 (0.005)	0.001 (0.002)	0.002 (0.003)
0.50	0.005 (0.005)	0.005 (0.005)	0.003 (0.004)	0.004 (0.005)	0.004 (0.004)
0.75	0.006 (0.008)	0.004 (0.005)	0.003 (0.004)	0.002 (0.002)	0.003 (0.003)
1.00	0.006 (0.006)	0.004 (0.005)	0.003 (0.004)	0.002 (0.002)	0.003 (0.003)

Table 3.12: Regret for Setting F.

Setting F Unfairness: Mean (StDev)					
b	d				
-	1.00	0.75	0.50	0.25	0.00
0.00	0.000 (0.000)	0.000 (0.005)	0.000 (0.003)	0.000 (0.000)	0.003 (0.006)
0.25	0.000 (0.000)	0.000 (0.007)	0.000 (0.002)	0.000 (0.000)	0.004 (0.007)
0.50	0.000 (0.000)	0.000 (0.007)	0.000 (0.002)	0.000 (0.000)	0.007 (0.010)
0.75	0.000 (0.000)	0.000 (0.004)	0.000 (0.004)	0.000 (0.000)	0.004 (0.006)
1.00	0.000 (0.000)	0.000 (0.004)	0.000 (0.004)	0.000 (0.000)	0.004 (0.006)

Table 3.13: Unfairness for Setting F.

D, F	f_1	f_2
u_1	0	0
u_2	0	1
u_3	1	0
u_4	1	1

E	f_1	f_2
u_1	0	1
u_2	0	1
u_3	1	0
u_4	1	1

Table 3.14: User features for settings D, E, and F.

Chapter 4: PreferenceNet: Encoding Human Preferences in Auction Design with Deep Learning

Joint work with Neehar Peri, John Dickerson, Sam Dooley. Appeared in NeurIPS 2021.

4.1 Introduction

Auctions are an essential tool in many marketplaces, including those in electricity [47], advertising [59], and telecommunications [48, 107]. The design of auctions with desirable properties is thus an important theoretical and practical problem. A typical assumption is that bidders may choose to bid strategically and will successfully anticipate the behavior of other bidders. This results in a potentially complicated Bayes-Nash equilibrium which may be difficult to predict. To evade this problem, a common requirement is that an auction should be strategyproof (i.e. bidders should be incentivized to truthfully share their valuations regardless of other bidders' actions).

If the goal of an auction is to maximize the total welfare of all participants, the Vickrey-Clarke-Groves (VCG) mechanism is both strategyproof and welfare-maximizing [41, 83, 160]. Intuitively, in many cases the VCG mechanism corresponds to a second-price auction. If the goal is to maximize the revenue gained, the problem is more challenging. Myerson's work completely characterizes strategyproof, revenue-maximizing auctions for a single item [123]. Beyond this case, results are more limited. Some results are known for the "multiple-good monopolist" problem, in which the auctioneer sells multiple items to a single bidder [53, 54, 76, 86, 98, 115, 129]. In addition, designing auctions for the related but weaker notion of Bayes-Nash incentive compatibility is reasonably well understood [35, 36, 37].

There has been significant difficulty in designing strategyproof auctions involving multiple items and multiple bidders. [167] shares significant, but limited results which solve the case in which items may have at most 2 values. Due to the apparent difficulty of analytically designing strategyproof, revenue maximizing auctions, recent methods instead approximate optimal auctions using machine learning approaches [50, 58, 68, 77, 104, 132, 133]. [152] proposes an method which guarantees exact strategyproofness in the single-agent setting. [31, 156] also use neural networks in the design of auctions. These methods primarily focus on standard mechanism design goals of revenue or welfare maximization, but these may not be the only goals of an auction. An auction might be conducted for public goods (i.e. electromagnetic spectrum distribution [48]), where the auctioneer must consider the effect not only on the auctioneer and bidders but on third-parties as well [125]. In other cases, such as auctions involving job or credit advertisements, it might be necessary to additionally constrain the auction mechanism to ensure fairness with respect to protected characteristics [39, 95]. Recent work has considered the problem of determining revenue-maximizing, strategyproof, fair auctions, either from a specific class [38] or with a general neural network approach [104].

The underlying notions of “fairness” used in these papers (e.g. total-variation fairness [95]) are defensible but somewhat arbitrary – they are attempts to formally capture human intuition. It might instead be better to attempt to capture, from data, human intuitions about fairness or other ethical requirements that are not explicitly defined [73, 154].

In this paper, we introduce PreferenceNet, an extension to RegretNet that directly encodes socially desirable constraints from data, and captures noisy human preferences. We are motivated by advertising auctions where the allocations of the auction mechanism must satisfy human preference constraints alongside the typical goals of strategyproofness and revenue maximization. We conduct a number of experiments using synthetic data (as is typical for neural network based auction mechanisms [58]) and evaluate our method on different auction settings and fairness constraints. We show that PreferenceNet is able to effectively capture each fairness constraint and matches the performance of standard approaches. We also conduct two surveys to further study human preferences. In the first survey (n=140), when given a specific definition of fairness, we ask participants to determine if a given auction setting is fair in order to test the noise in human judgements. In the second survey (n=345), we elicit judgements of pairwise comparisons of two auction settings to determine preferences without priming the participants with any particular definition of fairness. We train PreferenceNet on these data and show that our approach can capture nuanced human preferences in auction design.

4.2 Background and Related Work

RegretNet. [58] presents RegretNet, a neural-network architecture for learning approximately strategyproof auctions that maximize revenue. RegretNet treats the auction mechanism as a function from bids to allocations and payments, parameterized as a neural network. Revenue is optimized via gradient descent on sampled truthful bid profiles; strategyproofness is enforced by computing strategic bids in an adversarial manner to minimize violations. RegretNet has been modified and extended in a variety of ways, particularly to enforce additional desirable constraints. [50, 68, 77, 104, 132, 156].

Special Case: Single-Bidder Auctions. We highlight a special case in which deep learning for auctions has been particularly successful – auctions with a single bidder. In the single-bidder setting, the set of strategyproof mechanisms can be easily characterized [137], and it is possible to design neural network architectures which will always lie in this set. [58, 152] present two different learning-based solutions. Both methods are guaranteed to be strategyproof, and revenue can be maximized by unconstrained optimization. There are some known optimal single-agent mechanisms (we highlight those of Manelli-Vincent [115] and Pavlov [129], as they are relevant to auction settings we study below). Moreover, the theory of single-bidder mechanisms is well understood [53, 54, 76], so it is also possible to learn empirically strong mechanisms using these approaches and then prove their optimality [58, 152]. Unfortunately, when moving beyond the single-agent setting, it is necessary to use more general neural network architectures for which these guarantees do not apply. Because we are interested in such settings, we focus on these general architectures in this work.

Fairness and Human Preferences. As mentioned, while revenue, strategyproofness, and individual rationality are the classic goals of auction design, it might also be necessary for allocations made by an auction to satisfy certain other requirements such as fairness [38, 104]. However, it is not always clear if these mathematical definitions of these concepts actually capture human intuitions. [148, 154] considers human reactions to different definitions of fairness in a classification setting. [143] tests the extent to which human participants are able to understand and apply fairness metrics. [73] considers the problem of learning to perform fair clustering from human-provided demonstrations. As discussed below, we also crowdsource human opinions on fairness in auctions and analyze the results in Section 4.6.

4.3 Problem Setting

Auction Model. An auction is defined as a set of agents $N = \{1, \ldots, n\}$ bidding for items $M = \{1, \ldots, m\}$. Each agent $i \in N$ has a corresponding private valuation v_i, randomly drawn from a set of n valuations as $v = (v_1, \ldots v_n) \in V_i$. In general v_i may be functions over the power set of items 2^M. However, we only consider simpler cases with **additive** valuations and **unit-demand** valuations, where the valuation is simply a vector $v_i \in \mathbb{R}^m$ of values per item.

Each agent reports a bid vector b_i to the auctioneer, which may differ from the private valuation v_i. Given the profile of bids $b = (b_1, \ldots, b_n)$ of all agents, the auction has allocation and payment rules $g(b) : \mathbb{R}^{mn} \to [0,1]^{nm}$ and $p(b) : \mathbb{R}^{mn} \to \mathbb{R}^n$. We will refer to the matrix of allocation probabilities, whose rows must sum to 1, as $g(b) = z$. Likewise agent i's value of the jth item is $v_{i,j}$, and bidder i has payment function p_i. Moreover, for unit-demand auctions, we restrict the allocation to allow each bidder to win, in expectation, at most 1 item. Given the allocation, each agent receives a utility which can in either case be represented in linear form as $u_i(v) = \sum_j v_{i,j} z_{i,j} - p_i$.

Desirable Auction Properties. A mechanism is individually rational (IR) when an agent is guaranteed non-negative utility: $u_i(v_i; v) \geq 0 \; \forall i \in N, v \in V$. A mechanism is dominant-strategy incentive-compatible (DSIC) or strategyproof if every agent maximizes their own utility by bidding truthfully, regardless of the other agents' bids. We can define regret, the difference in utility between the bid player i actually made and the best possible strategic bid:

$$\mathrm{rgt}_i(v) = \max_{b_i} u_i(b_i, v_i, v_{-i}) - u_i(v_i, v_i, v_{-i})$$

. In addition to satisfying the IR and DSIC constraints, the auctioneer seeks to maximize their expected revenue. If the auction is truly DSIC, players will bid truthfully, and as a result revenue is simply $E_{v \sim V}[\sum_{i \in N} p_i(v)]$.

4.4 PreferenceNet: Encoding Human Preferences

We first explore a new metric to evaluate the adherence to socially desirable constraints in item allocations. Next, we describe the implementation details of PreferenceNet and important considerations when training the model.

Evaluation Metrics. There are limited evaluation criteria that quantitatively measures an auction mechanism's ability to enforce constraints on item allocations. If the constraints are not known explicitly, one can qualitatively examine the allocation graphs to evaluate the underlying allocation function $g(b)$. However, visual inspection does not scale to larger auction settings. As shown in Figure 4.1, our proposed metric is not only able to capture the same insights as qualitative analysis, but also scales to arbitrarily large auction mechanisms.

Given the limitations of existing analysis techniques, we propose Preference Classification Accuracy (PCA), a new metric to evaluate how well a learned auction model satisfies an arbitrary constraint. For a set of test bids b and allocations $g(b) : \mathbb{R}^{mn} \to [0,1]^{nm}$ generated by our learned auction model, we assign a label $s(b) \in \{1, 0\}$ to each allocation according to a ground truth labeling

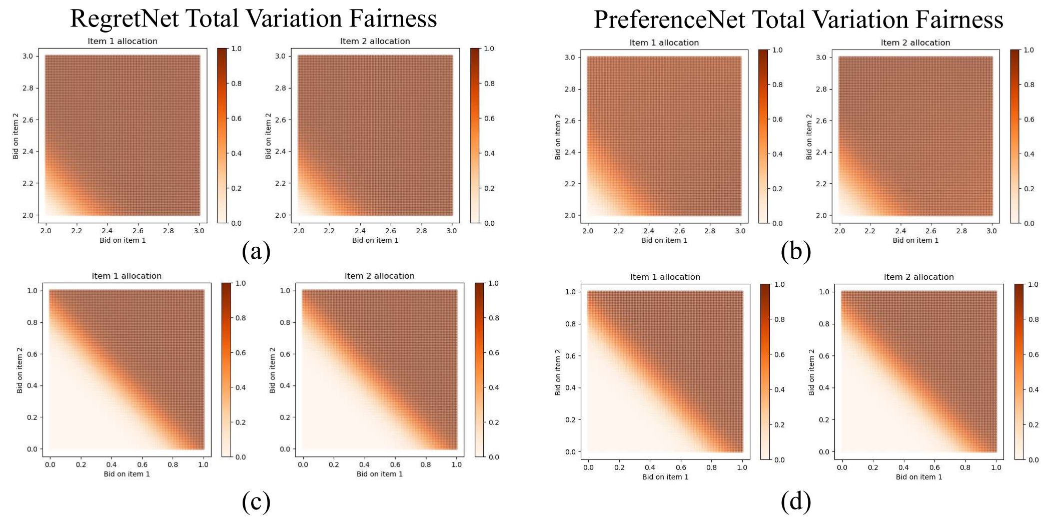


Figure 4.1: We compare the allocation plots of standard RegretNet approaches in enforcing total variation fairness (TVF) [95, 104] (sub-figures a and c) with our proposed approach (sub-figures b and d) learned through exemplars of desirable allocations that satisfy TVF. Visually, the allocations for both the unit-demand (sub-figures a and b) and additive auctions (sub-figures c and d) are identical. In this case, our proposed metric verifies our visual inspection, indicating that allocations from all four models satisfy TVF with 100% accuracy. However, this qualitative analysis does not extend to larger auction settings. In contrast, our proposed metric allows us to quantify the adherence of an auction mechanism to an enforced constraint for arbitrarily large auctions.

function based on the underlying preference. For each test bid b, $s(b)$ is 1 if the learned auction network satisfies the ground truth constraint. PCA is calculated by averaging the value of $s(b)$ over n test bids. Note that this metric remains valid in cases when we know the underlying preference function (e.g. total variation fairness, entropy) as well as when we are sampling from an unknown distribution (e.g. human preference elicitation). For cases where the underlying preference function is known, we can directly compute the preference score for a given allocation and apply a threshold to obtain a label. For cases where the preference function is unknown, as is the case in human preference elicitation, we can use the ground truth allocation-label pair to assign preference labels

$s(b)$ to new allocations based on the nearest neighbor in the ground truth set. This metric gives us a formal way to measure the degree to which preference constraints are violated, which crucially can be used whether or not the constraints follow an explicitly-known function.

Preference Elicitation. In order to effectively elicit preferences, we rely on pairwise comparisons between allocations to identify both positive and negative exemplars. We compare each input set of of allocations against n other allocations to determine if a particular sample is preferred over these alternatives (see Figure 4.2). This method of group preference elicitation reduces noise and ensures that the learned preference is satisfactory to a majority of the participants. We use this ranking approach to generate training labels in all of our experiments as described below.

Training Algorithm. We present the training algorithm of PreferenceNet below. We use the same additive and unit-demand network architectures as RegretNet for arbitrary numbers of agents and items. Our training algorithm follows closely from RegretNet. PreferenceNet consists of two sub-networks: RegretNet and a 3-layer MLP with a sigmoid activation at the output. We first train the MLP using a uniformly drawn sample of allocations as inputs, and optimize the binary cross entropy loss function using ground truth labels (generated as in Figure 4.2) identifying positive and negative exemplars. Next, we train RegretNet using the standard training procedure with the modified loss function described in Eq. 4.1. Lastly, we sample allocations and payments from the partially trained RegretNet model every c epoch and augment the MLP training set to adapt to the distributional shifts in allocations over the course of training. Our modified loss function $\mathscr{C}_\rho(w;\lambda)$ is defined as:

$$\mathscr{L}_{\text{rgt}} = \sum_{i\in N} \lambda_{(r,i)}\, \text{rgt}_i(w) + \frac{\rho_r}{2}\sum_{i\in N} \text{rgt}_i(w)^2,\ L_{\text{s}} = \sum_{j\in M} \text{s}_j$$

$$\mathscr{C}_\rho(w;\lambda) = -\frac{1}{L}\sum_{l=1}^{L}\sum_{i\in N} p_i^w(v^{(l)}) + \mathscr{L}_{\text{rgt}} - \mathscr{L}_{\text{s}} \tag{4.1}$$

where $\mathscr{L}_{\text{s}}$ is the output of the trained MLP. RegretNet is optimized such that it is strategyproof, revenue-maximizing, satisfies the preference learned by the MLP (i.e. maximizes the output scores of the MLP).

For each configuration of n agents and m items, we train RegretNet for a maximum of 200 epoch using 160,000 training samples. We also train the MLP with 80,000 initial training samples and iteratively retrain the MLP with 5,000 additional samples from the partially trained RegretNet every 5 epoch. For all networks, we use the Adam optimizer and 100 hidden nodes per layer. We apply warm restarts to the MLP optimizer each time we add new training data to to prevent the model from settling in a local minima. We incremented ρ_r every 2500 iterations and λ_r every 25 iterations. Finally, we report the preference classification accuracy, mean regret, and mean payments by simulating the allocations of 20,000 testing samples. We run all our experiments on an NVIDIA Titan X (Pascal) GPU.

MLP Architecture. We learn implicit preference functions using a simple 3-layer multi-layer perceptron that takes as input a mini-batch of allocations and outputs a vector that scores the input $\in [0,1]$ as a measure of how closely the allocation satisfies the ground truth preference. Given that neural networks are universal function approximators, with enough parameters the MLP can represent any arbitrary preference function. In practice, we find that ReLU non-linear activations and batch normalization are essential to effectively train this network.

Class Balanced Sampling. The distribution of positive and negative training examples for arbitrary preferences are often imbalanced – preferred examples may be concentrated in a small region of possible allocations. Often, this imbalance can make it difficult to train a robust model. Commonly, neural networks with improper class balance fail to learn a good decision boundary. In order to compensate for such imbalanced datasets, we can explicitly over-sample sparse classes until there are an equal number of positive and negative training examples. In practice this allows the network to quickly learn the decision boundary and allows us to train with fewer data points.

MLP Co-Training. The distribution of payments and allocations generated by RegretNet shift significantly during the course of training. As a result, the initially trained MLP may not effectively enforce the preference loss as RegretNet continues to train. In order to adapt to the distribution shift, we sample allocations from RegretNet at fixed intervals while it is training, add them back to the training set, and retrain the MLP. We generate noisy labels [108] using the existing MLP to reduce the cost of collecting expensive labels. In practice, we find that this approach effectively reinforces the decision boundary between positive and negative exemplars.

Model Validation and Selection. The optimal model minimizes regret, while maximizing payments and preference classification accuracy. In practice, satisfying all three conditions may be difficult. Often, we find that choosing a fixed epoch to evaluate does not provide consistent results. Rather, we evaluate each checkpoint on a validation set and maximize the following criteria in Eq. 4.2:

$$\alpha * \text{PCA} + \beta * \frac{\bar{p(b)}}{\max(p(b))} + \gamma * (1 - \frac{\bar{\text{rgt}(b)}}{\max(\text{rgt}(b))}) \text{ s.t. } \alpha + \beta + \gamma = 1 \tag{4.2}$$

In our experiments we set $\alpha = 0.45, \beta = 0.1, \gamma = 0.45$. It is important to note that the maximum regret and payments are calculated over the entire training process, while the mean regret, payment, and preference classification accuracy are calculated at each epoch.

4.5 Sampling Synthetic Preferences

Given the lack of publicly available auction data, we generate synthetic bids as in [58, 104, 152]. We first validate PreferenceNet using synthetic preferences, and extend our analysis to real human preferences in Section 4.6. In our synthetic preference experiments, we are interested in three types of fairness: total variation fairness (TVF), entropy, and a quota system. All three definitions of fairness map the allocations $g(b)$ onto $\mathbb{R}$. We train auction models for each of these valuation function and compare RegretNet with our proposed model under uniform additive and unit-demand auction settings in Table 4.1. Note in Table 4.1 that RegretNet is trained with an additional loss function term to explicitly optimize for the preference. PreferenceNet is trained as described in 4.4.

We describe the three definitions of fairness below:

Total Variation Fairness. An auction mechanism satisfies total variation fairness if the ℓ_1-distance between allocations for any two items is at most the distance between those items. That is, total variation fairness is satisfied when

$$\forall k \in \{1,...,c\}, \forall j, j' \in M, \sum_{i \in C_k} \left| g(b)_{i,j} - g(b)_{i,j'} \right| \leq d^k(j, j'). \tag{4.3}$$

We fix $d = 0$ in all of our experiments. We minimize violations of the above constraints.

Entropy. An auction mechanism satisfies the entropy requirement if the entropy of the normalized allocation for a bid profile b

$$-\sum_{i=1}^{n} \sum_{j=1}^{m} \left(\frac{g(b)_{i,j}}{\sum_j g(b)_{ij}} \right) \log \left(\frac{g(b)_{i,j}}{\sum_j g(b)_{ij}} \right) \tag{4.4}$$

is high. We normalize across items, turning the allocation into a probability distribution. This normalization ensures that entropy reflects diversity in the allocations, and not the overall number of items being allocated. Specifically, encouraging entropy ensures that the allocation for a given agent will tend to be more uniformly distributed.

Quota. An auction mechanism satisfies the quota constraint if, for each item in the (normalized) allocation, the smallest allocation to any agent j is greater than some minimum threshold t:

$$\min_j \left(\frac{g(b)_{\cdot,j}}{\sum_i g(b)_{i,j}} \right) > t \tag{4.5}$$

Here, we normalize the allocation so that the allocation per item will be a probability distribution over agents. The intuition for this definition is that each agent must guarantee some floor across every item. Returning to the advertising example, this ensures some minimum percentage of ad impressions are seen by every demographic group.

We train a number of models to compare the performance between RegretNet and PreferenceNet. Despite leveraging an weaker, implicit signal to learn fairness constraints, PreferenceNet is able to

Table 4.1: We evaluate both RegretNet and PreferenceNet over *nxm* auctions ("u" refers to unit-demand and "a" refers to additive), where *n* is the number of agents and *m* is the number of items. We measure three criteria **(a) PCA**, **(b) Regret Mean (STD)**, **(c) Payment Mean (STD)**. Although PreferenceNet learns each preference implicitly, it produces similar performance to our strong baseline.

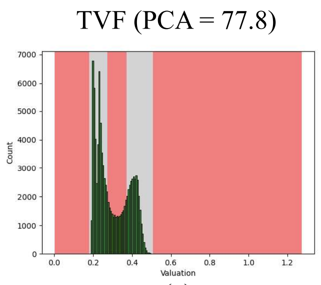
(a)	TVF		Entropy		Quota	
	RegretNet	PreferenceNet	RegretNet	PreferenceNet	RegretNet	PreferenceNet
2x2 u	100.0	100.0	86.4	69.6	100.0	100.0
2x4 u	100.0	100.0	99.7	94.9	100.0	100.0
4x2 u	100.0	100.0	100.0	94.5	.1	100.0
4x4 u	100.0	99.3	100.0	67.4	100.0	100.0
2x2 a	100.0	100.0	99.6	99.5	75.1	100.0
2x4 a	100.0	100.0	100.0	100.0	36.0	100.0
4x2 a	100.0	100.0	99.9	100.0	.1	.1
4x4 a	100.0	99.9	100.0	96.1	0	0

(b)	TVF		Entropy		Quota	
	RegretNet	PreferenceNet	RegretNet	PreferenceNet	RegretNet	PreferenceNet
2x2 u	.012 (.012)	.012 (.012)	.02 (.02)	.011 (.01)	.012 (.013)	.013 (.012)
2x4 u	.008 (.006)	.016 (.013)	.045 (.045)	.022 (.019)	.012 (.01)	.034 (.022)
4x2 u	.015 (.009)	.028 (.013)	.025 (.025)	.056 (.018)	.026 (.017)	.819 (.136)
4x4 u	.033 (.024)	.067 (.037)	.031 (.031)	.029 (.014)	.037 (.023)	.432 (.145)
2x2 a	.005 (.004)	.006 (.004)	.006 (.006)	.013 (.008)	.008 (.007)	.05 (.031)
2x4 a	.008 (.011)	.008 (.009)	.007 (.007)	.008 (.009)	.01 (.01)	.078 (.032)
4x2 a	.038 (.076)	.014 (.011)	.036 (.036)	.162 (.052)	.017 (.013)	.017 (.013)
4x4 a	.424 (.324)	.139 (.104)	.015 (.015)	.257 (.1)	.038 (.017)	.039 (.018)

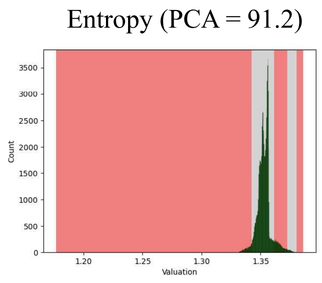
(c)	TVF		Entropy		Quota	
	RegretNet	PreferenceNet	RegretNet	PreferenceNet	RegretNet	PreferenceNet
2x2 u	4.18 (.45)	4.17 (.44)	4.26 (.37)	4.14 (.43)	4.19 (.35)	4.18 (.33)
2x4 u	4.37 (.29)	4.41 (.34)	4.48 (.34)	4.47 (.21)	4.44 (.16)	4.49 (.29)
4x2 u	4.93 (.21)	4.98 (.21)	4.85 (.23)	4.99 (.21)	5.16 (.24)	5.03 (.21)
4x4 u	8.81 (.39)	8.92 (.38)	8.79 (.5)	8.81 (.42)	8.8 (.3)	8.81 (.36)
2x2 a	.87 (.31)	.87 (.32)	.88 (.32)	.9 (.31)	.73 (.37)	.6 (.3)
2x4 a	1.75 (.38)	1.74 (.4)	1.77 (.45)	1.74 (.4)	1.76 (.44)	1.39 (.5)
4x2 a	1.1 (.34)	1.2 (.22)	1.1 (.34)	1.15 (.22)	1.3 (.23)	1.31 (.23)
4x4 a	2.58 (.39)	2.41 (.36)	2.26 (.3)	2.28 (.32)	2.52 (.32)	2.56 (.33)

closely match the performance of RegretNet. Surprisingly, PreferenceNet improves upon RegretNet in some cases, indicating that with careful hyperparameter tuning, further improvements are possible. Of the three fairness constraints examined, enforcing a quota is hardest for additive auctions, as both RegretNet and PreferenceNet struggled for auctions with a large number of agents.

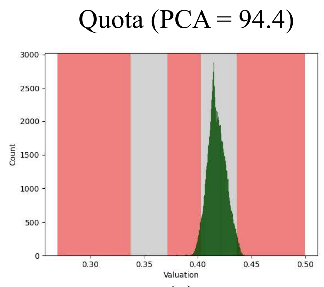
Limitations. Despite its effectiveness, we highlight limitations of our approach. In general, PreferenceNet always optimizes for the simplest function. For example, if learning a piece-wise preference where the positive exemplars are not clustered in a single region as in Figure 4.3, PreferenceNet tends to only satisfy part of the piece-wise function. This is unsurprising, given that neural networks are known to take shortcuts in optimization [74]. Moreover, PreferenceNet has difficulty in generating allocations with tightly clustered preference scores to satisfy a particular constraint. Given that we optimize for preferences implicitly using exemplars, this behavior is understandable. We study these limitations further in the supplemental material.



(a)



(b)



(c)

Figure 4.3: We simulate preference elicitation where positive exemplars are spread along multiple bands. The grey bands represent ground truth regions of positive exemplars, and the red bands represent ground truth regions of negative exemplars. In each plot, the green histogram represents the preference scores of the generated allocations. A model that perfectly enforcing a given preference rule will generate allocations that have valuations entirely within the grey bands.

4.6 Soliciting Human Preferences

In this section, we detail the creation of and results derived from two human subjects surveys. The results of these surveys are used to validate core assumptions of our model and provide data that we use to train an auction model. We find that PreferenceNet is able to adhere to the notion of fairness expressed by the human's preferences of auction outcomes. This encouraging result validates the utility and expressiveness of PreferenceNet.

We conducted both surveys through Cint, a crowdsourcing platform which connected us with English-speaking participants located in the United States. After submitting an application for our human subjects research to our institutional IRB, we were notified that the survey was exempt from IRB review. Our survey protocols can be found in the supplemental materials. Cint compensates per survey completion (regardless of length to complete), and both surveys were set to pay \$1.92. All survey results are anonymized to protect participant privacy. We include these results in the supplemental material.

Measuring Preference Noise. Our first survey was designed to test how participants would interpret and apply a simple fairness definition. The survey protocol was designed as follows: We primed participants to consider an advertising auction that was shown to two different (vague) demographic groups. Each participant was told that an auction would be fair if each ad was presented to each group at equal rates. After some familiarization, we asked the participants to determine if a given scenario was fair. Each participant was asked 30 such questions. The median completion time for the survey was 6 minutes with a median hourly wage of \$18. The 30 questions presented to the participants came from a question bank of 64 randomly generated scenarios, each with an associated TVF score.

Human understanding is an inherently noisy process. Despite providing the same context, we observe that survey participants understand a given definition of fairness and apply it to various scenarios differently. Given this data, we perform a normality test using a Q-Q plot as shown in the supplemental material. We find that our survey data are well correlated with the Gaussian distribution. This is also well supported by our visual inspection of the noise distribution. Experimentally, we observe that participants' choice of what is fair has a decision boundary at approximately a TVF value of 0.7. Interestingly, the noise is maximal near the decision boundary as shown in Figure 4.4. Concretely, we can model this noise using a probit model, so that the probability that that the label is unpertubed for a particular TVF value increases with distance such that $P(Y=1|X)=k\Phi(\frac{|x-\mu|}{\sigma})$, where Φ is the CDF of the Gaussian distribution, μ is the decision boundary, σ is the measured sample standard deviation, and k is an optional parameter that can scale the noise. We can use this noise model to perturb the input training data to the MLP to better simulate real data. We explore this further in the supplemental material.

Group Preference Elicitation. We designed our second survey to ask similar questions to the preference noise survey above. However, there were two primary differences: (1) we did not prime the participants with a fairness definition, and (2) the participants were presented with two scenarios and were asked which they thought was more fair. Each participant was asked 30 of these pairwise questions on the questions described above (full protocol details in the supplemental materials). This survey had 345 participants who had a median completion time of 7 minutes and a median pay of $15 per hour.

Notions of fairness and diversity have neither standardized nor widely accepted formal definitions [93, 143, 148, 154]. The purpose of this survey is to elicit preferences from a group and train an auction model whose allocations resemble the group preference. Using the survey data, we apply our preference elicitation strategy as described in Section 4.4 to generate training labels for each sample. After training PreferenceNet to enforce the group preference for both the unit-demand and additive auction settings, we find that the group preference is more similar to TVF and entropy. As shown in Figure 4.5, human preferences are not perfectly captured by these typical models, both because group preferences rarely converge to a unifying model, and preference elicitation is a noisy process.

4.7 Conclusion

Although surveying people to elicit their preferences can effectively help us model ambiguous definitions of socially beneficial constraints, we must be careful about the framing of the survey questions and the choice of audiences we survey.

Ethical Implications. Humans are inherently biased, so we need to be cognizant of the effects these latent biases might have over preferences for fairness. Moreover, sampling human preferences facilitates opportunities for data poisoning attacks, in which a malicious survey respondent could try to negatively impact the survey collection process. In general, we can mitigate both of these issues by sampling at scale to avoid problems with noisy labels, although this brings additional cost. Most importantly, we must involve stakeholders to ensure that their preferences are validated through the learned model in an iterative process.

In this paper we present PreferenceNet, a novel extension to RegretNet that makes it easier to learn preferences from data to encode socially desirable constraints for auction design. We introduce a new metric to empirically measure how closely a learned mechanism enforces a particular constraint, and show that our proposed method is able to effectively capture human preferences.

4.8 Simulating Noisy Preferences

We revisit survey results from Section 4.6, and explore the impact of label noise on our proposed method. We show that despite significant input perturbation, PreferenceNet is able to effectively capture preferences from noisy labeled examples.

Comparing Survey Data to Probit Model. In order to study noise in preference elicitation, we ask survey participants to interpret and apply a simple fairness definition. We primed participants to consider an advertising auction that was shown to two different (vague) demographic groups. Each participant was told that an auction would be fair if each ad was presented to each group at equal rates. After some familiarization, we asked the participants to determine if a given scenario was fair.

We hypothesize that the distribution of noise should be distributed according to a probit model, defined as $k\Phi(\frac{|x-\mu|}{\sigma})$, where Φ is the CDF of the Gaussian distribution, μ is the decision boundary, σ is the measured sample standard deviation, and k is an optional parameter that can scale the noise. The intuition behind this particular noise model is that participants will have higher uncertainty about allocation fairness closer to the decision boundary, and lower uncertainty farther away from the decision boundary. In practice, the probit model closely follows the survey noise distribution close to the sample mean, but diverges farther away from the decision boundary. The assumption that label noise goes to zero at sufficient distance from the mean does not hold in our human subject research. Rather, we find that there is a minimum amount of uncertainty regardless of the distance from the decision boundary, indicating that human understanding of fairness is inherently noisy.

We can update our probit model to incorporate this fact. We instead estimate the probability of a label flip as $min(k\Phi(\frac{|x-\mu|}{\sigma}), f)$, where f represents the noise floor. We experimentally select $k = 1.05$ and $f = 0.15$ to minimize the difference between the real and synthetic distributions. We compare the real distribution to the probit model in Figure 4.6 using a Q-Q plot. Since we expect the synthetic noise model to closely model the survey noise distribution, we plot the quantile function in comparison to the line $y = x$ (in black), and find that our synthetic model closely models real noise ($R^2 = 0.972$).

We train PreferenceNet, perturbing labels according to their distance from the decision boundary. Despite perturbing more than 25% of the training labels, PreferenceNet is able to capture the underlying preference, while minimizing regret and maximizing revenue.

4.9 Mixing Preferences in Synthetic Experiments

Eliciting preferences from a group is particularly challenging, because these preferences often heterogeneous. In Section 4.6 we train PreferenceNet on real human preferences and find that we can capture human preferences with high accuracy. To further study group preference elicitation, we train several models on a mixture of synthetic preferences to better understand the results of our human subject experiments. Specifically, the training set for the MLP contains allocations drawn from a uniform distribution, which are partitioned into 3 sets and labeled according to a particular definition.

As mentioned in Section 4.5, PreferenceNet often optimizes for the simplest function. In Table 4.2, we can see that almost all models maximize PCA for Entropy and TVF over a quota system irrespective of the mixture of training labels. In general, there does not seem to be a clear correlation between the input weighting of the different preferences and the learned preference function. This may explain why PreferenceNet trained on human preferences generates allocations that optimize for TVF and entropy.

Table 4.2: We simulate preference elicitation of three different definitions of fairness. For a set of allocations, we label non-overlapping partitions according to different preference functions. We calculate the PCA of each constraint independently, and weight them according to the proportions of the training labels. For unit demand auctions, we find that all three constraints have high PCA, indicating that PreferenceNet is able to generate a set of allocations that satisfy all preferences. However, we find that additive auctions have inconsistent results, indicating the ability to encode group preferences in auction allocation is dependent on the auction type.

Mixture	TVF PCA	Entropy PCA	Quota PCA	Average PCA
50% TVF, 25% Entropy, 25% Quota u	99.7	100.0	99.4	99.9
25% TVF, 50% Entropy, 25% Quota u	93.9	100.0	93.6	98.6
25% TVF, 25% Entropy, 50% Quota u	100.0	100.0	100.0	99.6
33.3% TVF, 33.3% Entropy, 33.3% Quota u	98.3	100.0	98.3	99.4
50% TVF, 25% Entropy, 25% Quota a	61.7	87.5	94.2	78.8
25% TVF, 50% Entropy, 25% Quota a	99.4	100.0	99.9	99.9
25% TVF, 25% Entropy, 50% Quota a	15.0	15.3	18.0	16.0
33.3% TVF, 33.3% Entropy, 33.3% Quota a	59.4	75.7	57.5	64.4

4.10 Augmented Lagrangian Multipliers to Enforce Constraints

In this section we explore the impact of enforcing constraints explicitly using augmented lagrangian multipliers. We we modify Eq. 4.1 as follows:

$$\mathscr{L}_{\text{rgt}} = \sum_{i \in N} \lambda_{(r,i)} \, \text{rgt}_i(w) + \frac{\rho_r}{2} \sum_{i \in N} \text{rgt}_i(w)^2$$

$$\mathscr{L}_{\text{s}} = \sum_{j \in M} \lambda_{(s,j)} \, \text{s}_j + \frac{\rho_s}{2} \sum_{j \in N} \text{s}_j^2 \tag{4.6}$$

$$\mathscr{C}_\rho(w;\lambda) = -\frac{1}{L} \sum_{l=1}^{L} \sum_{i \in N} p_i^w(v^{(l)}) + \mathscr{L}_{\text{rgt}} - \mathscr{L}_{\text{s}}$$

Similarly, we also adapt the loss functions for training RegretNet to explicitly enforce the constraint using augmented Lagrangian multipliers as in [104] and observe the impact on PCA, mean regret, and mean payments. In general, adding augmented Lagrangian multipliers improves overall PCA scores for both RegretNet and PreferenceNet. However, it provides RegretNet with limited improvements in minimizing regret and maximizing payments. Moreover, augmented Lagrangian multipliers have an overall negative impact on PreferenceNet, as mean regret is higher and mean payments are lower on average. We hypothesize that since PreferenceNet enforces preferences constraints implicitly with a learned model rather than an exact signal, preference loss should not be strictly enforced with Lagrangian multipliers.

4.11 Comparing Training Time between PreferenceNet and RegretNet

We compare the training time between PreferenceNet and RegretNet for several models that enforce the TVF constraint. We train each model on an unloaded machine with an RTX 2080 graphics card, and 32 GB of memory. As shown in Table 4.4, PreferenceNet takes nearly 50% longer to train. The training time increases proportionally with the size of the auction, the size of the MLP training set, and the number of times the MLP is retrained. Carefully tuning these hyperparameters could further improve the training time of PreferenceNet.

4.12 Scaling to Larger Auctions

PreferenceNet, much like RegretNet, can scale to relatively large auctions. The largest auction setting tested in the literature is the 5 agent, 10 item auction as described in [58]. Similarly, we replicate this experiment in Table 4.5. We find that many of the trends seen for smaller auctions still hold as the number of agents and items scale up. Surpringly, we find that both RegretNet and PreferenceNet fail to satisfy the quota-constraint for the unit-demand setting.

We also note that as auctions get larger in the number of bidders, a baseline itemwise Myerson auction will in many settings capture a very large percentage of the possible optimal revenue, so perhaps at really huge scales, the optimal auction problem is also less interesting.

4.13 Comparing PreferenceNet and RegretNet for Asymmetrical Valuations

All prior experiments were IID, where all agents sampled bids from the same valuation functions. We soften this requirement and evaluate the performance of RegretNet and PreferenceNet where all agents do not sample bids from the same distribution. Specifically, we study the auction setting with 2 agents, and 2 items. We study the case where agent 1 samples from a distribution n times larger than agent 2, where $n = 2, 4, 8$. We find that PreferenceNet is able to satisfy the preference requirement for TVF, entropy, and quota better than RegretNet. Moreover, the average regret and payment are comparable between the two networks.

4.14 Survey Protocols and Selected Responses

We now report the entirety of the two survey protocols and share select survey responses on how participants made decisions. For each survey, we also include two attention check questions similar to the provided examples to ensure that participants are actively engaged and faithfully completing the survey.

4.14.1 Selected Responses

We asked participants to describe in words why they selected certain auctions as fair. We found that many of these responses can be categorized into two groups: (1) participants with clear definitions of fairness and (2) participants with vague inexpressible preferences. Below are some examples selected from among the first 100 responses in each category:

Clear Preference of Fairness

- "I considered 50/50 most fair, added the percentages both groups were off from this number and had the smaller number was the most fair."
- "I used math, subtraction, and which option had the least amount of difference"
- "I looked at each instance and figured out which case presented the least variance from 50/50"

Inexpressible Preference of Fairness

- "I just read the dialog and decided from there which one was the better choice."
- "I just clicked on what I thought was fair"
- "I went with what looked more even"

The first group described something like an explicit function for determining fairness; the second group did have preferences but could not describe them. PreferenceNet provides a mechanism to capture preferences from both types of survey participants, allowing broader participation in the auction design process.

4.14.2 Measuring Preference Noise: Survey Protocol

The design of this survey is aimed at understanding how you interpret a given definition of fairness, and use this interpretation to decide if allocations of resources are fair with the given definition. You will be exploring this concept in the setting of advertising to different demographics. The survey will have two parts:

(1) familiarization with the given definition of fairness and

(2) answering questions about your understanding of the definition as applied to new scenarios.

This survey does not have correct answers. We would like you to consider each scenario and let us know what you genuinely think.

Let's begin with the familiarization process now. Consider this fairness definition.

Fairness Definition: An allocation is fair if the two similar demographic groups see ads at similar rates.

Let us walk through some examples of this. Consider a company, Company A, who displays an ad.

In our first example, Company A's ad was shown to 45.0% DEMOGRAPHIC1 and 55.0% DEMOGRAPHIC2.

Considering the given definition of fairness, this allocation is **fair**.

In our second example, Company A's ad was shown to 25.0% DEMOGRAPHIC1 and 75.0% DEMOGRAPHIC2.

Considering the given definition of fairness, this allocation is **not fair**.

——————Page Break——————

Now answer these questions based on what you think is fair and what isn't.

——————Page Break——————

[The participants were shown 8 examples of this question with values X and Y randomly generated between 30 and 70.]

Question: Company A's ad was shown to X% DEMOGRAPHIC1 and Y% DEMOGRAPHIC2. Considering the given definition of fairness, is this fair? [Yes/No]

——————Page Break——————

Now, consider a more complex setting where there are two companies, Company A and Company B. We are still interested in what you think is fair. Importantly, when making a determination of fairness, you must consider that the two demographic groups see both Company A and Company B's ads at similar rates.

Let us walk through some examples:

Example 1. Company A's ad was shown to 30.0% DEMOGRAPHIC1 and 70.0% DEMOGRAPHIC2. Company B's ad was shown to 30.0% DEMOGRAPHIC1 and 70.0% DEMOGRAPHIC2. Considering both companies, according to the given definition of fairness, this allocation is **fair**.

Example 2. Company A's ad was shown to 30.0% DEMOGRAPHIC1 and 70.0% DEMOGRAPHIC2. Company B's ad was shown to 30.0% DEMOGRAPHIC1 and 70.0% DEMOGRAPHIC2. Considering both companies, according to the given definition of fairness, this allocation is **not fair**.

——————Page Break——————

[The participants were shown 30 examples of this question with values X, Y, W, and Y randomly generated between 30 and 70.]

Question: Company A's ad was shown to X% DEMOGRAPHIC1 and Y% DEMOGRAPHIC2. Company B's ad was shown to W% DEMOGRAPHIC1 and Z% DEMOGRAPHIC2. Considering the given definition of fairness, is this fair?[Yes/No]

——————Page Break——————

Thanks for submitting answers to those questions.

Question: Can you describe what method or process you used to make your decisions? [Free text Response]

——————Page Break——————

Thank you for your time taking this survey. Please click next to return to Cint.

——————Survey End——————

4.14.3 Group Preference Elicitation: Survey Protocol

The design of this survey is aimed at understanding how you interpret the fairness of ad placements, and how you use this interpretation to decide which allocations of resources are more fair than others. You will be exploring this concept in the setting of advertising to different demographics. The survey will have two parts:

(1) familiarization with setting and

(2) answering questions about your preferences between new scenarios.

This survey does not have correct answers. We would like you to consider each scenario and let us know what you genuinely think.

Let's begin with the familiarization process now by walking through some examples.

Consider two companies, Company A and Company B, who displays an ad.

In our first example, Company A's ad was shown to 45.0% DEMOGRAPHIC1 and 55.0% DEMOGRAPHIC2.

In our second example, Company B's ad was shown to 25.0% DEMOGRAPHIC1 and 75.0% DEMOGRAPHIC2.

In the first part of the survey, you will be asked which of these two scenarios you think is more fair.

——————Page Break——————

Now answer these questions based on what you think is fair and what isn't.

——————Page Break——————

[The participants were shown 8 examples of this question with values X, Y, W, and Y randomly generated between 30 and 70.]

Question: Case 1: Company A's ad was shown to X% DEMOGRAPHIC1 and Y% DEMOGRAPHIC2. Case 2: Company B's ad was shown to W% DEMOGRAPHIC1 and Z% DEMOGRAPHIC2. Which is more fair, Case 1, Case 2? [Case 1/Case 2]

——————Page Break——————

Now, consider a more complex setting where you are comparing the joint placement of Companies A and B with Companies C and D. We are still interested in what you think is fair.

Let us walk through an example:

Example In Case 1: Company A's ad was shown to 30.0% DEMOGRAPHIC1 and 70.0% DEMOGRAPHIC2. Company B's ad was shown to 30.0% DEMOGRAPHIC1 and 70.0% DEMO-

GRAPHIC2.

In Case 2: Company C's ad was shown to 30.0% DEMOGRAPHIC1 and 70.0% DEMOGRAPHIC2. Company D's ad was shown to 30.0% DEMOGRAPHIC1 and 70.0% DEMOGRAPHIC2.

Considering both cases, we are going to ask you which of these two cases do you think is more fair.

—————————Page Break—————————[The participants were shown 30 examples of this question with values X1, Y1, W1, Y1, X2, Y2, W2, and Y2 randomly generated between 30 and 70.]

Question: Case 1:Company A's ad was shown to X1% DEMOGRAPHIC1 and Y1% DEMOGRAPHIC2. Company B's ad was shown to W1% DEMOGRAPHIC1 and Z1% DEMOGRAPHIC2.

Case 2: Company C's ad was shown to X2% DEMOGRAPHIC1 and Y2% DEMOGRAPHIC2. Company D's ad was shown to W2% DEMOGRAPHIC1 and Z2% DEMOGRAPHIC2.

Which is more fair, Case 1, Case 2? [Case 1/Case 2]

—————————Page Break—————————

Thanks for submitting answers to those questions.

Question: Can you describe what method or process you used to make your decisions? [Free text Response]

—————————Page Break—————————

Thank you for your time taking this survey. Please click next to return to Cint.

—————————Survey End—————————

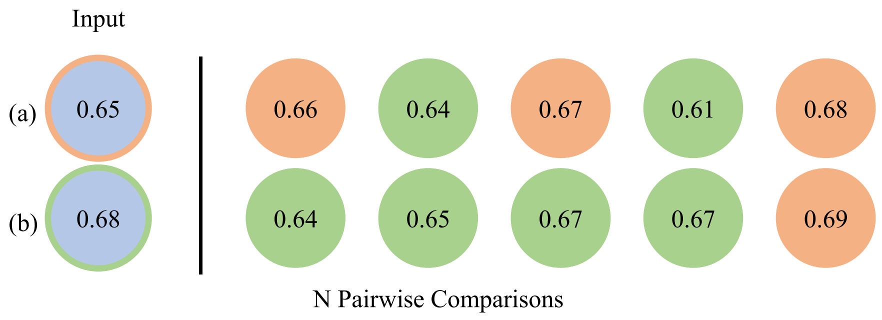


Figure 4.2: To elicit preferences from a group, or from a person without deterministic preferences, we use pairwise comparisons to determine labels for each allocation. In examples (a) and (b), each allocation (represented as a circle) has a preference score (represented by a number inside the circle). We can compare the score of the input against n other valuations to determine the relative ranking of the input data point. If the input data point has a smaller score than a plurality of the points it is compared against, it is a negative exemplar for the implicit preference (as in (a)). Otherwise it is positive (as in (b)).

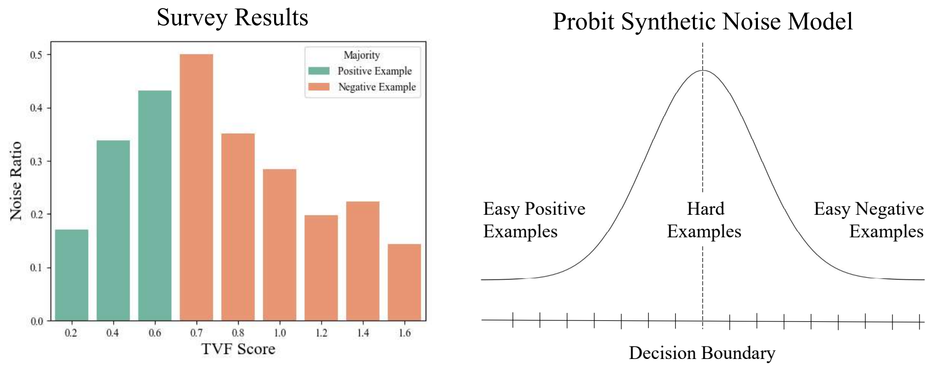
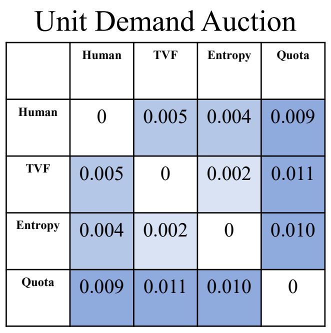
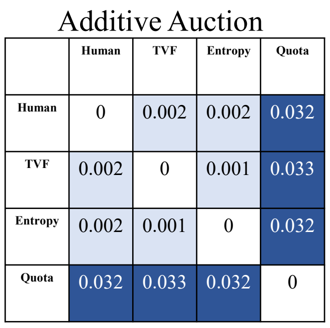


Figure 4.4: We crowdsource human annotators to examine various advertising scenarios and determine if an allocation is fair according to a given definition. We simplify the definition of total variation fairness (TVF) and measure the noise in responses as a function of the ambiguity of the scenario. We expect that the the label noise for a particular TVF value is inversely proportional to the distance from the decision boundary. Concretely, easy allocations will have lower noise ratios, and ambiguous allocations will have high noise ratios. We can leverage this model of uncertainty and apply it to our synthetic experiments to better simulate human preference elicitation.



Unit Demand Auction

	Human	TVF	Entropy	Quota
Human	0	0.005	0.004	0.009
TVF	0.005	0	0.002	0.011
Entropy	0.004	0.002	0	0.010
Quota	0.009	0.011	0.010	0

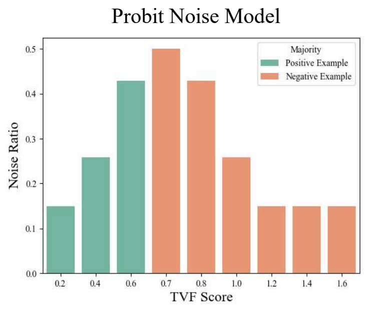


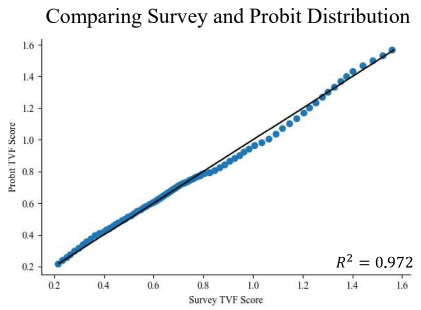
Additive Auction

	Human	TVF	Entropy	Quota
Human	0	0.002	0.002	0.032
TVF	0.002	0	0.001	0.033
Entropy	0.002	0.001	0	0.032
Quota	0.032	0.033	0.032	0

Human Preferences	PCA	Regret Mean (STD)	Payment Mean (STD)
2x2 Unit	100.0	.016 (.015)	4.20 (.37)
2x2 Additive	100.0	.005 (.004)	.87 (.31)

Figure 4.5: We randomly sample 20,000 bids and compute the average L2 distance from each model's learned allocations to identify allocation similarity. We find that human preferences are most similar to both TVF and entropy in both the unit demand and additive auction settings. Moreover, PreferenceNet is able to perfectly capture human preferences with low regret.





(a)	TVF		Entropy		Quota	
	No Noise	Probit Noise	No Noise	Probit Noise	No Noise	Probit Noise
2x2 u	100.0	92.8	69.6	49.6	100.0	98.0
2x2 a	100.0	100.0	99.5	77.8	100.0	6.7

(b)	TVF		Entropy		Quota	
	No Noise	Probit Noise	No Noise	Probit Noise	No Noise	Probit Noise
2x2 u	.012 (.012)	.08 (.042)	.011 (.01)	.045 (.024)	.013 (.012)	.043 (.028)
2x2 a	.006 (.004)	.006 (.005)	.013 (.008)	.007 (.007)	.05 (.031)	.007 (.005)

(c)	TVF		Entropy		Quota	
	No Noise	Probit Noise	No Noise	Probit Noise	No Noise	Probit Noise
2x2 u	4.17 (.44)	4.31 (.35)	4.14 (.43)	4.25 (.27)	4.18 (.33)	4.29 (.36)
2x2 a	.87 (.32)	.88 (.32)	.9 (.31)	.87 (.28)	.6 (.3)	.92 (.36)

Figure 4.6: We measure the performance of PreferenceNet with label noise that follows the Probit model for 2 agent, 2 item auctions (“u” refers to unit-demand and “a” refers to additive). We measure three criteria **(a) PCA**, **(b) Regret Mean (STD)**, **(c) Payment Mean (STD)**. We find that PreferenceNet is robust to noise in most cases, despite 25% of labels being flipped. PCA is most sensitive to noise. We find that the level of performance degredation is dependent on the particular constraint and auction type. In contrast, average regret and payments are comparable irrespective of input label noise.

Table 4.3: We evaluate both RegretNet and PreferenceNet over *nxm* auctions ("u" refers to unit-demand and "a" refers to additive), where *n* is the number of agents and *m* is the number of items. We measure three criteria **(a) PCA**, **(b) Regret Mean (STD)**, **(c) Payment Mean (STD)**. We find that using an augmented Lagrangian approach to enforce constraints do not provide significant improvement to RegretNet, and negatively impact the performance of PreferenceNet.

(a)	TVF		Entropy		Quota	
	RegretNet	PreferenceNet	RegretNet	PreferenceNet	RegretNet	PreferenceNet
2x2 u	100.0	100.0	100.0	100.0	100.0	100.0
2x4 u	100.0	100.0	100.0	100.0	100.0	100.0
4x2 u	100.0	100.0	100.0	81.7	100.0	100.0
4x4 u	100.0	100.0	100.0	69.7	100.0	100.0
2x2 a	100.0	100.0	100.0	100.0	100.0	100.0
2x4 a	100.0	100.0	100.0	100.0	100.0	100.0
4x2 a	100.0	100.0	100.0	100.0	100.0	100.0
4x4 a	100.0	100.0	100.0	71.9	.0	.1

(b)	TVF		Entropy		Quota	
	RegretNet	PreferenceNet	RegretNet	PreferenceNet	RegretNet	PreferenceNet
2x2 u	.011 (.011)	.17 (.082)	.013 (.013)	.027 (.018)	.015 (.015)	.02 (.029)
2x4 u	.01 (.008)	.136 (.086)	.008 (.008)	.061 (.057)	.012 (.017)	.904 (.185)
4x2 u	.016 (.008)	.897 (.167)	.018 (.018)	.662 (.118)	.076 (.062)	.165 (.075)
4x4 u	.029 (.02)	.223 (.07)	.041 (.041)	.283 (.134)	.05 (.041)	1.317 (.303)
2x2 a	.006 (.004)	.001 (.001)	.006 (.006)	.031 (.024)	.007 (.004)	.034 (.016)
2x4 a	.007 (.011)	.063 (.052)	.007 (.007)	.035 (.032)	.007 (.006)	.031 (.013)
4x2 a	.091 (.148)	.031 (.029)	.012 (.012)	.149 (.032)	.12 (.063)	.177 (.036)
4x4 a	.363 (.323)	0 (0)	.026 (.026)	.399 (.057)	.036 (.016)	.042 (.03)

(c)	TVF		Entropy		Quota	
	RegretNet	PreferenceNet	RegretNet	PreferenceNet	RegretNet	PreferenceNet
2x2 u	4.17 (.45)	4.12 (.38)	4.19 (.45)	3.97 (.33)	4.12 (.36)	4.14 (.12)
2x4 u	4.37 (.39)	4.33 (.3)	4.4 (.28)	3.86 (.15)	4.32 (.37)	4.95 (.26)
4x2 u	4.92 (.2)	4.99 (.21)	4.92 (.2)	5.06 (.21)	4.27 (.05)	4.29 (.18)
4x4 u	8.78 (.39)	8.66 (.41)	8.78 (.42)	6.88 (.3)	8.69 (.39)	8.84 (.74)
2x2 a	.89 (.31)	.01 (0)	.89 (.3)	.72 (.23)	.47 (.28)	.51 (.19)
2x4 a	1.78 (.4)	1.74 (.35)	1.76 (.39)	.99 (.07)	1.1 (.47)	.04 (.02)
4x2 a	1.16 (.38)	.07 (.01)	1.17 (.22)	.2 (.04)	.34 (.05)	.18 (.04)
4x4 a	2.58 (.39)	0 (0)	2.22 (.3)	1.98 (.29)	2.51 (.32)	1.99 (.31)

Table 4.4: We evaluate both RegretNet and PreferenceNet over *n*x*m* auctions ("u" refers to unit-demand and "a" refers to additive), where *n* is the number of agents and *m* is the number of items. We measure the total training time of both models and find PreferenceNet takes 50% longer to train.

TVF	1x2 a	2x2 a	2x4 a	4x2 a	4x4 a
RegretNet	28m	29m	34m	32m	37m
PreferenceNet	43m	46m	48m	49m	54m

Table 4.5: We evaluate both RegretNet and PreferenceNet over *n*x*m* auctions ("u" refers to unit-demand and "a" refers to additive), where *n* is the number of agents and *m* is the number of items. We measure three criteria **(a) PCA**, **(b) Regret Mean (STD)**, **(c) Payment Mean (STD)**. We find that PreferenceNet can replicate the performance of RegretNet for larger auctions.

(a)	TVF		Entropy		Quota	
	RegretNet	PreferenceNet	RegretNet	PreferenceNet	RegretNet	PreferenceNet
5x10 u	100.0	100.0	100.0	100.0	97.7	100.0
10x5 u	100.0	99.8	100.0	0	100.0	100.0
5x10 a	100.0	98.4	100.0	99.8	0	0
10x5 a	100.0	49.8	100.0	100.0	0	0
(b)	TVF		Entropy		Quota	
	RegretNet	PreferenceNet	RegretNet	PreferenceNet	RegretNet	PreferenceNet
5x10 u	.108 (.072)	.082 (.035)	.051 (.051)	.044 (.021)	.093 (.036)	2.502 (.207)
10x5 u	.082 (.023)	.117 (.024)	.085 (.085)	.164 (.041)	2.446 (.198)	2.466 (.2)
5x10 a	.142 (.3)	.054 (.036)	.293 (.293)	.504 (.141)	.521 (.371)	.517 (.383)
10x5 a	.144 (.226)	.459 (.153)	.26 (.26)	.404 (.08)	.293 (.103)	.299 (.108)
(c)	TVF		Entropy		Quota	
	RegretNet	PreferenceNet	RegretNet	PreferenceNet	RegretNet	PreferenceNet
5x10 u	10.11 (1.22)	11.54 (.39)	11.43 (.49)	11.4 (.43)	11.52 (.4)	12.53 (.21)
10x5 u	12.43 (.23)	12.56 (.23)	12.52 (.25)	12.72 (.24)	12.51 (.21)	12.51 (.21)
5x10 a	6.07 (.6)	5.63 (.46)	5.94 (.62)	5.64 (.53)	6.09 (.59)	6.09 (.59)
10x5 a	3.32 (.4)	3.02 (.38)	3.17 (.44)	2.93 (.26)	4.4 (.21)	4.41 (.21)

Table 4.6: We evaluate both RegretNet and PreferenceNet over 2 agent, 2 item auctions with asymmetric input valuations. We scale the distribution for input bids of agent 1 by n times the distribution for agent 2 (Denoted by n:1). We measure three criteria **(a) PCA**, **(b) Regret Mean (STD)**, **(c) Payment Mean (STD)**. Importantly, PreferenceNet maintains parity with RegretNet.

(a)	TVF		Entropy		Quota	
	RegretNet	PreferenceNet	RegretNet	PreferenceNet	RegretNet	PreferenceNet
2x2 2:1	100.0	100.0	93.5	99.3	39.2	100.0
2x2 4:1	100.0	100.0	80.0	98.6	27.3	100.0
2x2 8:1	96.5	100.0	72.9	96.2	17.1	100.0

(b)	TVF		Entropy		Quota	
	RegretNet	PreferenceNet	RegretNet	PreferenceNet	RegretNet	PreferenceNet
2x2 2:1	.007 (.006)	.008 (.006)	.008 (.008)	.015 (.009)	.011 (.01)	.064 (.036)
2x2 4:1	.01 (.007)	.011 (.007)	.011 (.011)	.062 (.035)	.035 (.026)	.135 (.069)
2x2 8:1	.019 (.013)	.021 (.016)	.02 (.02)	.054 (.072)	.02 (.014)	.151 (.082)

(c)	TVF		Entropy		Quota	
	RegretNet	PreferenceNet	RegretNet	PreferenceNet	RegretNet	PreferenceNet
2x2 1:2	1.37 (.62)	1.36 (.61)	1.38 (.61)	1.39 (.61)	1.36 (.7)	.94 (.51)
2x2 1:4	2.44 (1.33)	2.4 (1.34)	2.46 (1.34)	2.59 (1.31)	2.58 (1.37)	1.76 (.97)
2x2 1:8	4.66 (2.73)	4.57 (2.77)	4.62 (2.8)	4.35 (2.77)	4.71 (2.83)	3.07 (1.75)

Chapter 5: Learning Revenue-Maximizing Auctions With Differentiable Matching

Joint work with Michael Curry, Uro Lyi, Tom Goldstein, John Dickerson. Appeared in AISTATS 2022.

5.1 Introduction

Auctions have been held for millennia, since at least Classical antiquity [103], and have played an important complementary role to set-price sales and bargaining to exchange goods. In recent decades, the advent of computation has resulted in a surge of large-scale fielded auctions in a variety of important industries, such as broadcasting [107], advertising [59], electricity markets [47], and many others [121, 140]. These developments have made auctions not only of theoretical interest, but also of great practical importance.

Auction *design* is the problem faced by an auctioneer who, given uncertain knowledge of the demands of auction participants, wishes to set the rules of the auction to ensure, via proper incentive structure, a desirable outcome. In the usual theoretical model of auctions [127], the bidders have some private valuation of the items, but the distribution from which these valuations are drawn is common knowledge. The auctioneer solicits bids from participants, awards the items up for sale to the winners, and charges some amount of money to each. Bidders may, however, choose to strategically lie about their valuation given their knowledge of the auction rules and anticipated behavior of other participants, resulting in a Bayes-Nash equilibrium which may be very hard for the auction designer to predict.

To avoid this problem, an auctioneer may simply wish to design a strategyproof (or truthful) mechanism in which participants are incentivized to truthfully reveal their valuations. The distribution of bids is then simply the valuation distribution itself, so it becomes easy to predict the results of the auction in expectation. While satisfying the strategyproofness constraint, the auctioneer can additionally optimize total utility, their own revenue, or other desirable properties.

If the auctioneer's goal is to maximize the welfare of all participants while maintaining strategyproofness, the Vickrey-Clarke-Groves (VCG) mechanism always gives a solution [41, 83, 160]. By contrast, if the auctioneer wishes to maximize revenue, strategyproof mechanisms are much harder to find. Some revenue-maximizing strategyproof mechanisms are known in limited cases: when selling a single item, the Myerson auction is strategyproof and maximizes revenue [123], and for selling multiple items to one agent, some results are known [54, 98, 115, 129]. But even for the simple case of selling *two* items to *two* agents, there has been almost no progress, with the major exception of a breakthrough for the special case where items take on at most two discrete valuations [167].

The theoretical difficulty of devising revenue-maximizing strategyproof auctions has resulted in a number of attempts to approximate them. Recently, Dütting et al. [58] presented a method, RegretNet, for learning approximately incentive compatible mechanisms given samples from the bidder valuations. They parameterize the auction as a neural network, and learn to maximize revenue, and maintain strategyproofness, by gradient descent. This learning approach to auction design can replicate some known-optimal auctions, and can learn good auctions in settings where the optimal auction is not known. It has been extended in a variety of ways [50, 68, 77, 104, 130, 132, 133, 156].

The RegretNet architecture has architectural features to ensure that no item is over-allocated, and that each bidder receives the correct amount of goods. This is accomplished through a combination of softmax and minimum operations, depending on the exact bidder demand constraints.

However, this approach only works when some ad-hoc combination of these operations is sufficient to enforce the constraints. This is not always possible. Consider the case where the auctioneer needs to ensure that every participant receives exactly k items. Such equality constraints cannot be enforced using the min-of-softmax approach.

We observe that an auction allocation for bidders whose utilities are linear functions of their valuations amounts to a matching assigning items to bidders. Using this observation, we present an alternative approach which explicitly applies matching constraints on the output, using the Sinkhorn algorithm [51] to solve a discrete matching problem as part of an end-to-end differentiable architecture.

This approach is among the first attempts to use the techniques of differentiable optimization [4] for learned mechanism design – concurrent work [111] uses a differentiable sorting operator to learn generalized second-price auctions. By simply modifying the constraints in our matching problem, our proposed approach can produce feasible allocations without changes to the overall network architecture for many different bidder demand constraints. Our approach successfully recovers known optimal mechanisms in multiple settings including settings (such as exactly-one-demand auctions) in which RegretNet would be unable to guarantee that its output allocations adhere to the constraints.

5.2 Related Work

Auction Design and Learning Auction mechanisms are functions from bids to allocations and payments; if one assumes sample access to the valuation distribution, it is natural to treat auction design as a learning problem, and there has been much work in this area.

One thread of work has been learning-theoretic, determining the sample complexity for various known families of auctions to estimate properties like revenue [22, 42, 122] or incentive compatibility [24].

Another thread of work, sometimes called "differentiable economics", represents auction mechanisms using general parametric function approximators, and attempts to optimize them using gradient descent. [58] introduce several neural network architectures to find revenue-maximizing auctions, including RochetNet, which works for single-agent auctions and is strategyproof by constructions, and RegretNet, which represents auctions as a general neural network and includes a term in the loss function to enforce strategyproofness.

Further work has built directly on both RochetNet and RegretNet [50, 104, 132, 133]. Others have taken a similar approach but applied to different mechanism design problems, including finding welfare-maximizing auctions [156] and facility location [77]. Other applications of gradient-based methods to problems in mechanism design include [89, 164].

Single-agent auction learning The case of selling multiple items to a single agent is reasonably well understood. Rochet [137] gives a characterization of strategyproof single-agent mechanisms – their utility functions must be monotone and convex. Based on this characterization, Dütting et al. [58] design RochetNet, a network architecture for single-agent mechanism learning which is guaranteed to be perfectly strategyproof. Shen, Tang, and Zuo [152] also provide network architectures for the single-agent setting which can be made strategyproof by construction. The authors of both works are able to learn mechanisms and then prove them optimal using the theoretical framework of Daskalakis, Deckelbaum, and Tzamos [54].

We wish to explicitly contrast these architectures with our own approach. In a single-agent setting, they work better than a RegretNet-style architecture, and as mentioned can always guarantee perfect strategyproofness. In this work, however, we focus on general architectures which can work in both single- and multi-agent settings.

Optimal Transport and the Sinkhorn Algorithm Optimal transport [97] is the problem of moving one set of masses to another mass while minimizing some cost function. In its infinite dimensional form, where it amounts to finding the cost-minimizing joint distribution between two marginal continuous probability distributions, it has wide applications in pure mathematics [161] as well as machine learning [14, 75]. Interestingly, Daskalakis, Deckelbaum, and Tzamos [54] and Kash and Frongillo [98] use the mathematical tools of infinite-dimensional optimal transport in the context of mechanism design theory.

Our work here is not directly related. By contrast, we use the discrete form of optimal transport, which is essentially a formulation of minimum cost bipartite matching (in our case, between agents and items). This discrete problem can be numerically solved in a number of ways, see Peyré, Cuturi, et al. [131] for an overview. In particular, we focus on the Sinkhorn algorithm [51]: a fast, GPU-parallelizable iterative method for solving the entropy-regularized version of the optimal transport problem, which can be used as a differentiable bipartite matching operation [52, 62, 82, 119, 158]. We use the Sinkhorn algorithm to compute matchings between agents and items.

Differentiable optimization and deep learning Recently, there has been broad interest in mixing convex optimization problems with deep learning. For many convex optimization problems, the derivative of the optimal solution with respect to parameters of the objective or constraints is well defined, so it is possible to define a neural network layer that will output a feasible, optimal solution to some optimization problem. This is useful if one wants to use optimization with deep learning in a "predict and optimize" pipeline [69, 165], to enforce that network outputs satisfy some constraints, or if interesting operations can be formulated in terms of optimization problems [79].

One family of approaches [e.g., 4, 8] involves solving the convex optimization problem using standard solvers, then using the implicit function theorem to compute the gradient for the backward pass. In a contrasting approach, when using an iterative method to solve the optimization problem, it is also possible to use automatic differentiation to simply backpropagate through the numerical operations performed by the solver. We employ this latter approach with the aforementioned Sinkhorn algorithm in order to compute feasible matchings in a differentiable manner

5.3 Differentiable Economics and Combinatorial Optimization

General auction setting We consider an auction setting with n bidders and m items. Each bidder i has a private valuation function, $v_i : [0,1]^m \to \mathbb{R}_{\geq 0}$, that maps any subset of the items to a real number. We assume that v_i is drawn from a known distribution F_i, but the realized valuation v_i is private and unknown to the auctioneer. Each bidder i then submits their bids $b_i \in \mathbb{R}^m$. Let $b = (b_1, b_2, ..., b_n)$ be the bids from all bidders. The auction mechanism is then a combination of an allocation mechanism and payment mechanism, $(g(b), p(b))$. The g outputs an allocation $(g_1, g_2, ..., g_n)$ where g_{ij} is the probability of allocating item j to bidder i. The payment mechanism p outputs $(p_1, p_2, ..., p_n)$ where p_i is how much bidder i is charged. Each bidder then receives some utility $u_i(v_i, b) = v_i(g_i(b)) - p_i(b)$.

Quasilinear utilities, strategyproofness, and individual rationality Allowing a distinct valuation for every combination of items may result in a combinatorial explosion. A simplifying assumption is that a bidder may have a single valuation for each item. Then their valuation v_i is simply a vector in $\mathbb{R}^m$, and their utility is simply $u_i(v_i, b) = \langle v_i, g_i(b) \rangle - p_i(b)$.

Since a bidder's true valuation is private they are free to strategically report bids to maximize their utility. Thus, a we require our mechanism to be **strategyproof** or **dominant-strategy incentive compatible (DSIC)**, meaning each bidder's utility is maximized by reporting truthfully i.e. $u_i(v_i, (v_i, b_{-i})) \geq u_i(v_i, (v_i', b_{-i})$ for any other v_i' where b_{-i} are all the bids except the ith bidder. **Regret** is defined by the following equation: $\text{rgt}_i(v_i, b) = \max_{v_i'} u_i(v_i, (v_i', b_{-i})) - u_i(v_i, b)$, and represents the utility the bidder could have gained from lying in their bid. We can then also say that when a mechanism is DSIC, the regret for truthful bidding should be 0 for every bidder i with truthful valuation v_i, for any opponent bids b_{-i}.

Another desirable property for an auction for it to be (ex post) **individually rational**. This means each bidder receives a non-negative utility from the auction i.e. $u_i(v_i, (v_i, b_{-i})) \geq 0$ for all bidders i, all possible valuations v_i, and all other bids b_{-i}. Without this guarantee bidders might choose not to participate in the auction at all because they are concerned about being left worse off than they were before the auction.

5.3.1 Bidder Demand Types

Here we define common bidder demand constraints which are used in the experiments.

The value of a bundle of goods for a **unit-demand** bidder is equal to the maximum value of a single item in the bundle: $v_i(S) = \max_{j \in S} v_i(j)$. In this sort of auction, there is no need to consider allocations of more than one item to each bidder since those provide that same utility as only allocating the most desirable item within that bundle to that bidder. Thus, by restricting allocations to allocate at most one item per bidder, we can again treat bidder utility as quasilinear. More generally, there can be k-unit-demand auctions where a bidder's value of a bundle of goods is the sum of the top k items in the bundle [168].

We also consider **exactly-one demand**, where each bidder must be assigned exactly one item. In this setting, the bidder cannot be assigned more or fewer than one item and receives the value of the assigned item. Additionally, this demand-type can be extended to a more general exactly-k demand setting where each bidder must be allocated k items.

In all these settings, in general allocations might be randomized so that bidders can receive items with different probabilities. This corresponds to allowing fractional allocations, so that in the unit-demand and exactly-one-demand settings, the one item the bidder receives can be a mixture of fractions of multiple items.

5.3.2 RegretNet

The RegretNet architecture consists of two neural networks, the allocation network (denote it g) which outputs a matrix representing the allocation of each item to each agent and payment network (denote it p) which outputs the payments for each bidder [58].

RegretNet guarantees individual rationality by making the payment network use a sigmoid activation, outputting a value $\tilde{p} \in [0,1]$. The final payment is then $\left(\sum_j v_{ij} g_{ij}\right) \tilde{p}$, ensuring that the utility of each bidder is non-negative.

[58] present network architectures to learn under additive, unit-demand and combinatorial valuations. Their architecture utilizes a traditional feed-forward neural network with modifications in the output layer to ensure a valid allocation matrix. In the additive setting, the allocation probabilities for each item must be at most one: this is enforced by taking a row-wise softmax on the network outputs. For unit-demand auctions, each bidder wants at most one item, so the network takes a row-wise softmax of one matrix and a column-wise softmax of another to ensure

item allocation is less than one. The final output is then the elementwise min of these two, ensuring that both item allocation and unit-demand constraints are enforced.

For training, the RegretNet architecture uses gradient descent on an augmented Lagrangian which includes terms to maximize payment while also containing terms to enforce the constraint that regret should be zero:

$$\mathscr{L}_{\boldsymbol{\lambda}}(\boldsymbol{v}) = -\sum_i p_i(\boldsymbol{v}) + \sum_i \lambda_i \widehat{\mathrm{rgt}}_i(\boldsymbol{v}) + \frac{\rho}{2}\sum_i \left(\widehat{\mathrm{rgt}}_i(\boldsymbol{v})\right)^2 \tag{5.1}$$

Here $\widehat{\mathrm{rgt}}_i$ denotes an empirical estimate of regret produced by running gradient ascent on player i's portion of the network input to maximize their utility, computing their possible benefit from strategically lying. During training, the Lagrange multipliers λ are gradually maximized, incentivizing the minimization of regret.

5.4 Bipartite Matching for Auctions

The auction allocation problem is equivalent to finding a minimum-cost bipartite matching between bidders and items, for some cost matrix. We solve this bipartite matching problem by formulating it as an optimal transport problem. Using the Sinkhorn algorithm, we are able to solve this optimal transport problem in an end-to-end differentiable way, yielding a valid matching that our network can use to learn an optimal allocation mechanism.

5.4.1 Matching Linear Program

For marginal vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ – here representing constraints for agents and items respectively, each with a dummy component that represents no item or no agent – the optimal transport problem is as follows.

$$\min_{\boldsymbol{P}} \sum_{i=1}^{n+1}\sum_{j=1}^{m+1} \boldsymbol{P}_{ij}\boldsymbol{C}_{ij} \text{ s.t. } \boldsymbol{P}\mathbb{1}_{m+1} = \boldsymbol{a},\ \mathbb{1}_{n+1}^T\boldsymbol{P} = \boldsymbol{b} \tag{5.2}$$

By specifying $\boldsymbol{a}$ and $\boldsymbol{b}$, we can specify the demand constraints of the problem:

- k-demand: $\boldsymbol{a}_i = k$, $\boldsymbol{a}_{n+1} = m$, $\boldsymbol{b}_j = 1$, $\boldsymbol{b}_{m+1} = kn$

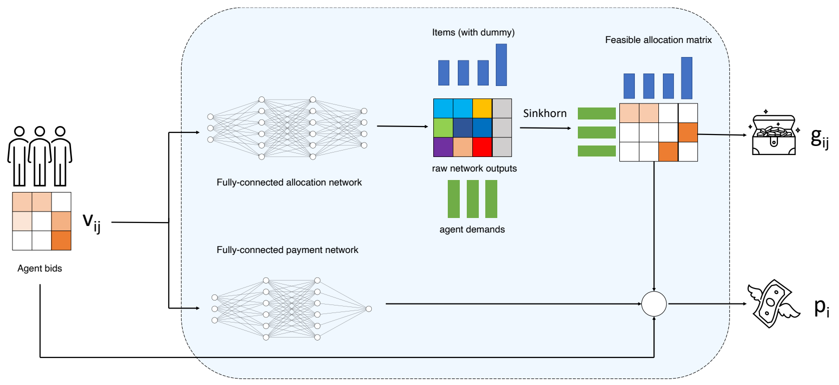

Items (with dummy)
Feasible allocation matrix
Sinkhorn
raw network outputs
agent demands
Fully-connected allocation network
Fully-connected payment network
v_{ij}
Agent bids
g_{ij}
p_i

Figure 5.1: A schematic showing the Sinkhorn-based mechanism network. The agents' bids (here, assumed to be truthful) on each item act as input to a feedforward network, whose output is used as the cost matrix $\boldsymbol{C}_{ij}$ for the Sinkhorn algorithm. Marginals, specified separately, ensure that the Sinkhorn output is a feasible allocation. The payment network charges a fraction of the value of items that each agent wins.

- exactly-k-demand: $\boldsymbol{a}_i = k$, $\boldsymbol{a}_{n+1} = m - kn$, $\boldsymbol{b}_j = 1$, $\boldsymbol{b}_{m+1} = 0$

Here the $n+1$ and $m+1$ values of the marginals represent the dummy agent and item respectively. Concretely, the marginal constraint for agents represents how many items that agent must receive; if there is a dummy item, it can be fully or partially met by assigning that dummy item, representing receiving nothing. Likewise, the constraints for items represent how much of each item must be allocated; if there are dummy agents, these constraints can be met by allocating to the dummy agent (representing allocation to nobody).

More complex constraints, such as different k for different agents, can also be accommodated. Also, note in the exactly-k-demand settings there must be at least kn items, so $m \geq kn$. An optimal solution to the matching LP will always be a permutation matrix [28], or more generally on one of the extreme points of the constraint set.

5.4.2 Entropy-Regularized Optimal Transport and the Sinkhorn Algorithm

The problem in Equation (5.2) can be solved as a linear program. However, Cuturi [51] observed that by adding an entropy penalty to the problem, it can be solved in a practical and scalable way using the matrix-scaling Sinkhorn algorithm. The new objective of the problem is

$$\min_{\boldsymbol{P}} \sum_{i=1}^{N} \sum_{j=1}^{M} \boldsymbol{P}_{ij} \boldsymbol{C}_{ij} + \varepsilon \sum_{i=1}^{N} \sum_{j=1}^{M} \boldsymbol{P}_{ij} \log \boldsymbol{P}_{ij}$$

Given marginals and a cost matrix, Cuturi [51] provides an iterative algorithm to solve this problem. It can be implemented in a variety of ways; we use the following stabilized log-domain updates [131]:

$$\boldsymbol{f}_i = -\varepsilon \log \left(\sum_j \exp \left(-\frac{C_{ij} - \boldsymbol{g}_j}{\varepsilon} \right) \right) + \varepsilon \log \boldsymbol{a}$$
$$\boldsymbol{g}_j = -\varepsilon \log \left(\sum_i \exp \left(-\frac{C_{ij} - \boldsymbol{f}_i}{\varepsilon} \right) \right) + \varepsilon \log \boldsymbol{b}$$

where $\boldsymbol{P}_{i,j} = e^{\boldsymbol{f}_i/\varepsilon} e^{-C_{ij}/\varepsilon} e^{\boldsymbol{g}_j/\varepsilon}$. In addition to enabling the use of the iterative algorithm, the regularization term makes the problem strongly convex. Thus, the derivative of the optimal solution with respect to the cost matrix is well-defined. To approximate this derivative, we choose to unroll several iterations of the Sinkhorn updates (which consist only of differentiable operations) and use automatic differentiation.

The ε parameter controls the strength of the regularization. If it is large the output $\boldsymbol{P}$ will be nearly uniform; as it goes to zero we recover the original problem and $\boldsymbol{P}$ will be a permutation matrix. For our purposes, we set ε large enough to avoid vanishing gradients, but small enough that the output allocations are still nearly permutation matrices.

5.4.3 Network Architecture

Our architecture, like RegretNet, uses a pair of networks to compute allocations and payments. We denote the allocation and payment networks g^w and p^w respectively, where w are the network weights.

The main distinction from RegretNet is in the allocation mechanism. Like RegretNet we utilize a traditional feedforward neural network that takes all given bids as input. We use the output of the network as the cost matrix $\boldsymbol{C}_{ij}$ to the Sinkhorn algorithm (if there are dummy agents or items, they receive cost 0). Marginals are specified separately depending on the problem setting.

Finally, the output of the Sinkhorn algorithm (after truncating the row and column for dummy variables) provides the final probabilities g_{ij}^{w} of allocating item j to bidder i.

As in RegretNet, the final payment for bidder i is $\left(\sum_j v_{ij} g_{ij}\right) \tilde{p}_i^w$ where $\tilde{p}_i^w \in [0,1]$ is the ith output from a feedforward payment network. This ensures that the mechanism is individually rational as the bidders will never be charged more than their utility gained from the allocated items.

5.4.4 Settings That RegretNet Cannot Represent

RegretNet provides three different architectures depending on bidder demand type and valuation functions. For its unit-demand and combinatorial architectures, it employs a “min-of-softmax” approach to ensure both that agents receive at most 1 item or bundle, and items are not overallocated.

We particularly emphasize the case of exactly-k demand, which describes the situation where the auction designer is obligated to ensure that every participant receives k items, no matter what they bid. This captures, for instance, an offline version of the classic Display Ads setting [66], with a special case of no-free-disposal requirements, where every ad buyer is contractually guaranteed a certain number of ad slots. For multiple agents and multiple items, RegretNet cannot represent this demand type via the min-of-softmax approach.

(In some special cases, ad-hoc changes could be made to the original RegretNet architecture to support settings beyond additive and unit-demand. For example, to enforce “at-most-k” demand, one can just scale the unit-demand outputs by a factor of k. In the special single-agent case, it is possible to produce exactly-k demand by removing the dummy item which allows for the possibility of allocating no item.)

By contrast, in multi-agent settings with exactly-k demand, the Sinkhorn-based architecture can ensure feasible outputs by simply specifying the correct marginals. Our network architecture is also independent of the demand type since only the marginals change rather than the entire network output structure.

5.4.5 Optimization and Training

Like RegretNet, our training objective is Equation (5.1). For each batch we estimate the empirical regret, $\widehat{\mathrm{rgt}}_i(w)$ by performing gradient ascent for each bidder on the network inputs, approximating a misreport for each bidder.

In our experiments, the network for both the payment and allocation network had 2 hidden layers of 128 nodes with Tanh activation functions, optimized using the Adam optimizer [100] with a learning rate of 10^{-3}. The training set consisted of $2^{19} = 524,288$ valuation profiles. We used batch sizes of 4,096 and ran a total of 100 epochs. For each batch, we computed the best misreport using 25 loops of gradient ascent with a learning rate 0.1. We also incremented ρ every two batches and and updated the Lagrange multipliers, λ, every 100 batches. Experiments were all run on single 2080Ti GPUs, on either a compute node or workstation with 32GB RAM, using PyTorch [128].

For evaluation, we used 1,000 testing examples and optimized the misreports of these examples for 1,000 iterations with learning rate 0.1. For each of the 1,000 samples we use ten random initialization points for the misreport optimization and use the maximum regret from these. At both train and test time we used the Sinkhorn algorithm with an ε parameter of 0.05, and a stopping criterion of a relative tolerance of 0.01. Additionally we used an ε schedule (as suggested by Cuturi, Teboul, and Vert [52] and Schmitzer [149]) of 10 steps from 1 down to the final value of 0.05. Pseudocode for computing the allocation network output is given in algorithm 2.

5.5 Experiments

5.5.1 Optimal single-agent mechanisms

We start by recovering two known optimal mechanisms in the single bidder case. We emphasize that if the only goal were to learn single-agent auctions, architectures such as RochetNet [58] or

Algorithm 2 Sinkhorn allocation procedure

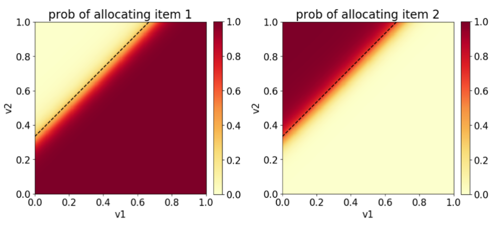
```
**Input:** Bid $\boldsymbol{v} \in \mathbb{R}^{m\times n}$, allocation network output $f(\boldsymbol{v};\theta)$, marginals $\boldsymbol{a} \in \mathbb{R}^{n+1}$ and $\boldsymbol{b} \in \mathbb{R}^{m+1}$, Sinkhorn epsilon schedule $\varepsilon_1, \cdots, \varepsilon_T$, tolerance $t$
**Output:** Feasible allocation matrix $\boldsymbol{g}_{ij} \in \mathbb{R}^{m\times n}$
**Initialize:** $\boldsymbol{f} = 0^{n+1}, \boldsymbol{g} = 0^{m+1}, \boldsymbol{C} = f(\boldsymbol{v};\theta)$
**for** $\varepsilon$ in $\varepsilon_0, \cdots, \varepsilon_T$ **do**
    $\boldsymbol{P}_{ij} = e^{\boldsymbol{f}_i/\varepsilon} e^{-\boldsymbol{C}_{ij}/\varepsilon} e^{\boldsymbol{g}_j/\varepsilon}$
    **while** $\max_i |\sum_j \boldsymbol{P}_{ij} - \boldsymbol{a}_i|/\boldsymbol{a}_i \geq t$ **do**
        $\boldsymbol{f}_i = -\varepsilon \log\left(\sum_j \exp\left(-\frac{\boldsymbol{C}_{ij}-\boldsymbol{g}_j}{\varepsilon}\right)\right) + \varepsilon \log \boldsymbol{a},\ \boldsymbol{g}_j = -\varepsilon \log\left(\sum_i \exp\left(-\frac{\boldsymbol{C}_{ij}-\boldsymbol{f}_i}{\varepsilon}\right)\right) + \varepsilon \log \boldsymbol{b}$
        $\boldsymbol{P}_{ij} = e^{\boldsymbol{f}_i/\varepsilon} e^{-\boldsymbol{C}_{ij}/\varepsilon} e^{\boldsymbol{g}_j/\varepsilon}$
$\boldsymbol{g}_{ij} = \boldsymbol{P}_{ij}$ for $i = 1..n, j = 1..m$
```

that of [152] would work better. Our architecture works for both single-agent and multi-agent auctions. Following previous work [58, 132, 133], we use these single-agent experiments as a test case to make sure it recovers some known optimal mechanisms, before continuing on to multi-agent settings where optimal auctions are not known and where RochetNet or [152] cannot work.

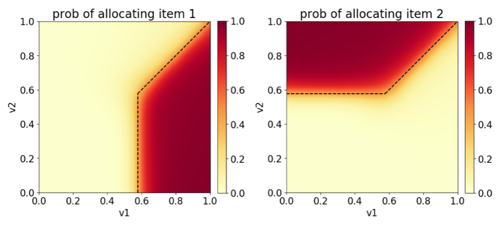
Unit-demand We first recover the optimal mechanism in the unit-demand single-agent two-item setting with item values drawn from $U[0,1]$. The optimal mechanism, also approximately recovered in Dütting et al. [58], is from [129]: it is to offer each good for a price of $\frac{\sqrt{3}}{3}$. The learned allocation probabilities are shown in Figure 5.2b with the boundary of the optimal analytic mechanism denoted by a dotted line. The x and y axis are the valuation of the bidder for item one and two respectively. The color represents the allocation probability output by the learned mechanism with the darker color corresponding to higher probability. Quantitatively, the learned mechanism has small regret and slightly higher than the optimal mechanism likely due to the small amount of regret.

Exactly-one demand The second optimal mechanism was a deterministic mechanism in the same single agent, two-item setting with valuations on $U[0,1]$. However, this time the agent will be allocated exactly one item (instead of at most one). Kash and Frongillo [98] shows that the optimal mechanism is to offer one of the items for free and the other for a price of $\frac{1}{3}$.

The boundary of this optimal mechanism is shown in Figure 5.2a as the black dotted line. The revenue in Figure 5.1 is slightly higher than the optimal, again likely due to the presence of small



(a) Exactly-One



(b) Unit-Demand

Figure 5.2: Heatmaps of learned mechanisms for single-bidder two-item with $v_1, v_2 \sim U[0,1]$ are shown in (a)-(b). The optimal mechanism boundaries are the black dotted lines.

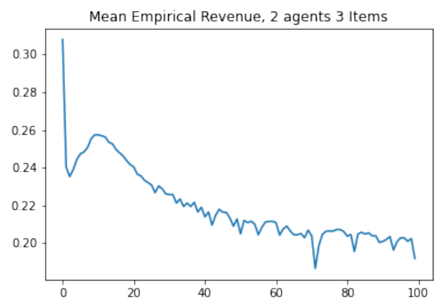
Setting	Rev	Regret	Opt Rev	Train Time
Unit-demand	0.397 (0.248)	< 0.001 (< 0.001)	0.393	3h25m
RegretNet Unit-Demand	0.381 (0.258)	< 0.001 (< 0.001)	0.393	18m34s
Exactly-one	0.079 (0.127)	0.001 (0.001)	0.069	1h21m

Table 5.1: Table of mean revenue and regret from learned mechanism in test set alongside revenue of optimal mechanism for 1 agent, 2 items, with item valuations distributed independently on $U[0,1]$.

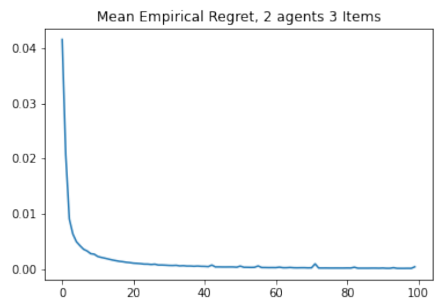
regret.

5.5.2 Multi-agent setting

We now experiment in settings where the optimal strategyproof mechanism is unknown. Specifically, we study 2 bidders and either 3 or 15 items where valuations for each item are drawn from $U[0,1]$. A typical analytic baseline is a separate Myerson auction for each item, but this only works in additive settings. Instead, as a baseline, we compute the revenue from an [41, 83, 160]. The VCG auction is strategyproof, and maximizes welfare rather than revenue. It works by charging each bidder the harm they cause to other bidders, which in the settings we test may be very small or even close to zero. Table 5.2 contains the revenues for both mechanism (unsurprisingly higher for the learned auction) as well as the average observed regret on the testing sample, while figures 5.3a and 5.3b show training plots for the two agent, 3 item, exactly-one demand case.



(a) Mean revenue for exactly-one demand, 2 agent 3 item setting



(b) Mean empirical regret for exactly-one demand, 2 agent 3 item setting

Setting	Agents	Items	Rev	Regret	VCG Rev	Time
Unit	2	3	.876 (.322)	.001 (.001)	.048	6h20m
RegretNet Unit	2	3	.878 (.337)	$<.001$ ($<.001$)	.048	20m28s
Unit	2	15	.886 (.113)	.001 ($<.001$)	.002	3h10m
RegretNet Unit	2	15	.999 (.123)	$<.001$ ($<.001$)	.002	25m45s
Exactly-1	2	3	.194 (.064)	.004 (.008)	.049	2h38m
RegretNet Exactly-1	2	3	N/A	N/A	.049	N/A
Exactly-1	2	15	.571 (.030)	.003 (.015)	.002	1h29m
RegretNet Exactly-1	2	15	N/A	N/A	.002	N/A

Table 5.2: Table of mean revenue and regret from learned mechanism in test set along with revenue from VCG auction, for 2 agents with valuations distributed independently on $U[0,1]$

5.5.3 Comparison To RegretNet

RegretNet is capable of representing unit-demand auctions, but not the exactly-one setting. We compare performance to it in this case; results are shown in Table 5.2. (Runtimes are significantly faster than those reported in Dütting et al. [58] likely due to our much larger batch sizes.) For the one-agent, two item setting, both architectures approximate the optimal mechanism so performance is similar. Performance is also similar for the 2 agent, 3 item setting. In the 15-item setting, revenue with the Sinkhorn architecture is somewhat smaller. In all cases the computational cost of RegretNet is significantly lower, as it requires only two softmax operations instead of many iterations of the Sinkhorn algorithm.

5.6 Strengths, Limitations, and Potential Impacts

Structurally, our architecture has the benefit of remaining the same for any quasi-linear bidder utilities and constraints, with only the Sinkhorn marginals changing. The use of the Sinkhorn algorithm allows us to enforce these constraints explicitly and even tackle new settings, such as exactly-k demand, that the existing RegretNet architecture would be unable to handle.

Our Sinkhorn-based architecture has a higher computational cost than the comparable RegretNet architecture, as the Sinkhorn algorithm requires many iterations. The Sinkhorn algorithm also requires two additional hyperparameters, ε and a tolerance before the algorithm terminates.

We find that the ε parameter plays a crucial role in the training of the algorithm. If the ε is too small, training fails, likely due to vanishing gradients. However, with too large of an ε the mechanism becomes too smooth and unable to approximate the sharp boundaries that tend to show up in optimal mechanisms leading to suboptimal revenue. (For further discussion of ε, see section 5.8 at the end of this chapter.) Our architecture has an inductive bias towards deterministic allocations. This may pose a limitation where revenue-maximizing mechanisms are nondeterministic, but it may be an advantage if they are deterministic or if determinism is desirable. By setting ε small enough it is possible to ensure near-determinism in allocations. However, because the mechanisms are trained under a higher ε, there are regions where the price charged becomes too high, increasing regret. Because training directly with small ε is difficult, we cannot directly guarantee that we train deterministic mechanisms.

Following previous work that uses neural networks to approximate optimal auctions [e.g. 58, 132, 133, 152], we train on synthetic data. While in principle deep-learning-based methods could be used to train on truthful bids from real-world bidders, and doing so would be extremely interesting, such data is hard to come by. We see learned auctions as, at least in part, a tool for pushing theory forward, so using synthetic data remains interesting when it is drawn from valuation distributions for which analytic solutions have remained out of reach.

Our work, like most work in differentiable economics, uses sample-based approximations to estimate revenue and regret. It would be desirable to understand the error due to sampling. Some generalization bounds in previous mechanism design papers (such as the quantile regret from Dütting et al. [58]) apply directly to our case; others (such as the techniques in Balcan, Sandholm, and Vitercik [24]) do not but might be extended.

There are clear ethical implications of revenue-maximizing auction design, in terms of potential impact on bidders and society as a whole. We do release code which can in principle be used for revenue-maximizing mechanism design. However, since our work is still closely to the theoretical models discussed above, we do not see any direct ethical implications – it is very unlikely that deep-learned auctions will be directly deployed in the near future.

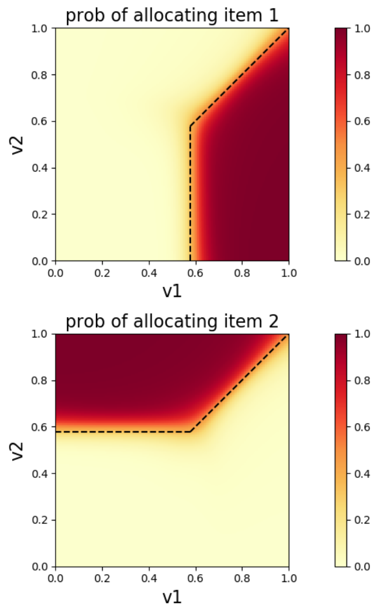
5.7 Conclusion and Future Work

We have presented a new architecture for learned auctions which uses the Sinkhorn algorithm to perform a differentiable matching operation to compute an allocation. Our architecture works for a variety of bidder demand constraints by encoding them into the marginals used in the Sinkhorn algorithm. This new architecture allows the network to guarantee valid allocations in settings where RegretNet could not.

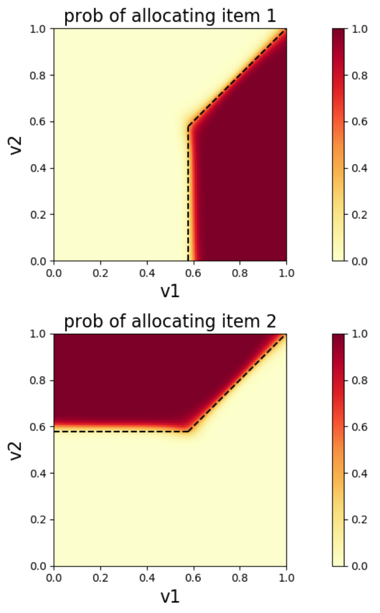
We show that our approach successfully recovers optimal mechanisms in settings where optimal mechanisms are known, and achieves good revenue and low regret in larger settings. Future work might include extending the Sinkhorn architecture to more directly support randomized allocations, further computational improvements, or further extensions to other mechanism design problems where the allocation decision can be expressed as a matching.

5.8 Effect of Sinkhorn Epsilon

Effect of Sinkhorn ε Figures 5.4b and 5.4a highlights the effect of the Sinkhorn parameters on the final mechanism. The lower the ε in the Sinkhorn algorithm the sharper the boundary becomes, as with less entropy regularization, the optimal matching is closer to deterministic. However, we found that very small values of epsilon led to problems during training; the learned mechanism would choose to never allocate items, likely due to vanishing gradients. One can make an almost-perfectly-deterministic mechanism by decreasing ε at test time, but this can increase regret as the learned payments no longer agree with the allocations.



(a) $\varepsilon = 0.05$



(b) $\varepsilon = 0.02$

Figure 5.4: The following figures illustrate the effect of Sinkhorn temperature on allocation mechanism (c) higher Sinkhorn ε value and (d) lower Sinkhorn ε value.

Chapter 6: Differentiable Economics for Randomized Affine Maximizer Auctions

Joint work with Tuomas Sandholm, John Dickerson.

6.1 Introduction

Auctions are a widely-used mechanism for allocating scarce items that are for sale, in which a centralized auctioneer solicits bids from auction participants, and based on those bids, allocates the items (possibly keeping some of them) and charges some payments. The auctioneer may wish to design the auction to achieve some goal. The usual assumption is that the auctioneer has access to a prior distribution over bidders' valuations. Typically, it is also desired that the auction be strategyproof, that is, there should be no incentive for bidders to be untruthful in their bids about their valuations.

When the auctioneer wants to maximize the total welfare of the bidders, the *Vickrey-Clarke-Groves (VCG)* mechanism, which is always strategyproof, is also optimal [41, 83, 160]. When the auctioneer instead wants to maximize her revenue (or profit), the problem is significantly more challenging.

Myerson [123] settled the revenue-maximizing strategyproof auction problem when there is one item for sale. Maskin and Riley [118] generalized that mechanism to the case of multiple copies of a single item. However, four decades later, the multi-item revenue-maximizing auction is still unknown. Special cases of the two-item setting have been solved [15, 18]. There is some theory of strong duality [54, 98] for selling multiple items to a single agent. In some such cases it may be advantageous to offer *lotteries* – i.e. award certain items to participants with some fractional probability. There have also been some successes for the weaker notion of Bayesian incentive

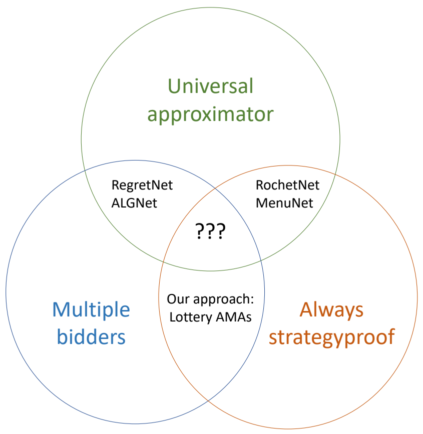


Figure 6.1: Our architecture in relation to other techniques from differentiable economics for multi-item revenue-maximizing auction design. The “holy grail” in the middle of the Venn diagram—that is, techniques that can represent (i) any auction for (ii) general numbers of bidders and items while (iii) guaranteeing strategyproofness—has not been achieved; however, we show that our method achieves (iii) strategyproofness-by-design for (ii) general numbers of items and bidders while still improving revenue over baselines.

compatibility [35, 36, 37]. But for designing dominant-strategy incentive compatible mechanisms that sell multiple items to multiple agents there has been little progress despite decades of research. Yao [167] presents a result for one special case, giving an explicit example of a revenue gap between the best dominant-strategy incentive compatible mechanism and the best Bayes-Nash incentive compatible mechanism.

Nevertheless, the problem is wide open. Even for the seemingly trivial case of two agents with i.i.d. uniform valuations over two items, the optimal selling mechanism is not known.

In part motivated by the fact that the theory on this question has essentially gotten stuck for decades, Conitzer and Sandholm [45] and Sandholm [145] introduced the idea of *automated mechanism design (AMD)*: designing the mechanism computationally for the problem instance at hand, as opposed to trying to analytically derive a general form for the revenue-maximizing multi-item auction. AMD has since become a popular research topic. Three different high-level approaches to AMD have been introduced: 1) designing the mechanism from scratch in tabular form [45], 2) conducting search over the parameters of a mechanism class where all the mechanisms in the class have some desirable properties such as strategyproofness and individual rationality (the latter incentivizes buyers to participate) [109, 110, 147], and 3) *incremental mechanism design* where the design starts from some (typically well-known but not strategyproof) mechanism and then keeps making changes to the mechanism to improve it [46].

A recent form of incremental mechanism design that capitalizes on the modern power of deep learning is called *differentiable economics*. Dütting et al. [58] introduced the use of deep neural networks as function approximators to learn auctions. Their RegretNet architecture learns approximately strategyproof auctions for multi-bidder multi-item auctions. MenuNet [152] and RochetNet [58] are restricted to a single bidder, but enforce strategyproofness at the architectural level.

6.2 Our Contributions

Ideally, we would like an auction architecture that 1) supports multiple agents and items, 2) is perfectly strategyproof by construction, and 3) is always rich enough to represent the true optimal auction, given enough parameters. Such an architecture does not yet exist. RegretNet achieves 1 and 3 only; RochetNet and MenuNet achieve 2 and 3. In our work, we present an approach that achieves 1 and 2, though not 3 – **a multi-bidder, multi-item auction architecture which is always perfectly strategyproof**.

Consider a classic tool for automated mechanism design – the family of *affine maximizer auctions (AMAs)* [136]. AMAs are essentially versions of the VCG mechanism, modified by associating a positive "weight" to each bidder's welfare and adding potentially different "boosts" to all the possible allocations. AMAs are always strategyproof and individually rational like VCG, but revenue can be significantly increased over VCG by tuning these parameters (weights and boosts). Importantly, this can be done by just using *samples* of the valuation distribution [109, 110, 147] rather than the traditional mechanism design approach of taking the full valuation distribution as input, which would be prohibitively complex in these combinatorial settings. Later work considers the number of samples needed for this in a learning-theoretic sense [22, 23, 25].

Our contribution is to revisit the problem of learning AMAs, now with differentiable economics.

One can view the paper from at least the following perspectives:

1. It can be seen as an extension of previous work on learning AMAs, now **allowing for lottery allocations**. This means not only learning the weights and boosts, but also learning over the (continuous) set of lotteries to offer. Randomization can increase revenue.

2. It can be seen as a **multi-bidder generalization of RochetNet and MenuNet**. Restricting our lottery AMAs to a single bidder essentially recovers these architectures, and for multiple bidders, strategyproofness is still guaranteed by construction. (However, for general multi-bidder combinatorial auction settings, AMAs cannot represent every truthful mechanism; there is no guarantee they can represent an optimal one.)

3. It provides a **more interpretable** family of mechanisms to learn using differentiable economics. RegretNet-style auctions are opaque: they map bid profiles to outcomes in an arbitrary way. In contrast, the rules for determining outcomes of an AMA are easy to explain. Moreover, by the end of training, our learned mechanisms typically have a small number of possible outcomes which are easily summarized.

6.3 Related work

Differentiable economics Dütting et al. [58] use the tools of modern deep learning to learn revenue-maximizing mechanisms. In particular, they present the RegretNet neural architecture. The idea is to treat an auction mechanism as a function mapping bid profiles to allocations and payments, and directly approximate this function using a neural network. The loss function consists of a term for revenue maximization, and another term for minimizing *regret* – violations of strategyproofness. RegretNet works quite well, approximately recovering some known optimal auctions and outperforming other approaches.

However, its approach has several limitations. In particular, the learned auctions are only approximately strategyproof – there is still some small presence of regret, and moreover the presence of regret can only be measured empirically. Curry et al. [50] provides a way to exactly compute regret, which mitigates this latter limitation. But the former problem remains – a mechanism learned using the RegretNet approach is not guaranteed to be perfectly strategyproof.

Characterizing strategyproof mechanisms Rochet [137] shows that for any mechanism *with a single agent*, strategyproof mechanisms can be identified with convex utility functions (as a function of the agent's true type). Any strategyproof pair of allocation and payment rules will induce a convex utility function. An allocation rule can be derived from any convex utility function by simply taking its gradient (which also fixes the payment rule).

Characterizing strategyproof mechanisms for multiple agents is not so straightforward. Rochet's characterization still holds in this case: fixing other bids, agent i's utility must be convex as a function of their type, and this must hold for all agents and for any choice of opponent bids. However, coming up with some universal approximator for the entire class of functions that has this property is difficult.

Strategyproof architectures Alongside RegretNet, Dütting et al. [58] also presents the RochetNet architecture, which is restricted to a single bidder but is perfectly strategyproof.[1] Shen, Tang, and Zuo [152] concurrently present MenuNet, another single-bidder architecture which is perfectly strategyproof.

Both MenuNet and RochetNet offer possibly-randomized sets of menu items at different prices. The bidder maximizes over all offered menu items, inducing a convex utility as a function of the bidder's type. As such, MenuNet and RochetNet will always represent a strategyproof mechanism for any setting of their parameters. And given enough parameters, they are universal approximators for strategyproof mechanisms.

For single-bidder auction design, there is a strong duality result which can be used to prove optimality of a proposed mechanism [54, 98]. The authors of Dütting et al. [58] and Shen, Tang, and Zuo [152] apply these results to their learned auctions, and discover some previously-unknown optimal auctions.

[1] In the appendix, they also present MyersonNet, which is restricted to 1 item.

Further work in differentiable economics Many papers have built on RegretNet. ALGNet [132] gives an improved loss function, which has fewer hyperparameters, and an improved training algorithm. We use it as a point of comparison below. Other papers apply the same general approach to auctions with fairness or budget constraints [68, 104, 130], add new inductive biases to the architecture [49, 57, 96, 133], or apply similar techniques to other mechanism design problems [32, 77, 134].

Another line of work uses neural networks to model agent preferences over possible bundles [19, 30, 31, 156, 164]. Bichler et al. [27] uses ML techniques to compute equilibrium strategies for non-incentive-compatible auctions.

Automated mechanism design and learning theory for auctions Affine maximizer auctions (AMAs) are classic tools for automated mechanism design [109, 110, 147]. In essence, AMAs are just weighted versions of the celebrated Vickrey-Clarke-Groves (VCG) mechanism [41, 83, 160]. VCG chooses the welfare-maximizing allocation; an AMA maximizes a rescaled and shifted version of the welfare. By choosing the parameters of the AMA carefully, performance on metrics other than welfare maximization can be improved without sacrificing strategyproofness. Previous work considers the problem of learning high-performing AMAs from samples using gradient based methods [110, 147], albeit using different techniques and without considering lotteries. Guo, Hata, and Babar [84] computes AMA parameters via linear programming for a particular problem setting. Other works consider the sample complexity of learning AMAs, treating them as a parameterized function class [22, 23, 25]. Tang and Sandholm [157] considers a subset of AMAs for which the optimal revenue can be computed in closed form. Deng et al. [56] tunes the parameters of a class of AMAs to improve performance in an online advertising application.

Lotteries and menu size complexity There are a number of theoretical results showing that offering lotteries can improve revenue [33, 54, 129]. Hart and Nisan [88] analyze this phenomenon and give an interesting perspective – in the most general sense, it is not offering lotteries *per se* that improves revenue.

Rather, it is that offering more menu items can improve revenue by allowing finer price discrimination, and there are always fewer deterministic allocations than possible lotteries.

These results, however, give worst-case revenue gaps across whole classes of valuations, not a guarantee for any specific instance. As discussed below, we find that even when our mechanisms can improve their revenue by offering lotteries, they offer relatively few menu items, so performance improvements are not due to increased menu size.

Expressiveness of AMAs and Roberts's Theorem To what extent can the class of affine maximizer auctions actually express the optimal strategyproof auction? As mentioned, Rochet [137] shows that all single-agent strategyproof mechanisms can be identified with convex functions. For multi-agent multi-item settings with unrestricted valuations (meaning every agent may get any positive or negative utility from any outcome, and may even care about which particular items other agents receive), Roberts [136] shows that every strategyproof mechanism must take the form of an AMA.

The settings we consider here do not have unrestricted valuations, so Robert's theorem does not apply. In particular, Roberts's theorem does not hold for deterministic combinatorial auctions where valuations are monotonically increasing in receiving more items, and the empty set has zero value. All the valuations we consider have these properties. On the other hand, for many settings, [106] shows that any implementable allocation rule which satisfies certain natural conditions must be "almost" an AMA in a certain technical sense.

6.4 Affine Maximizer Auctions

6.4.1 Combinatorial Auction Setting

Consider a setting in which m auction participants are bidding on n items. Each bidder has a private type $v_i \in \mathbb{R}^n$ denoting how much they value each item.

Allocations consist of matrices $a \in \mathbb{R}_+^{mn}$, where a_{ij} denotes the amount of item j given to bidder

i. We require that $\sum_i a_{ij} \leq 1$, so that no item is overallocated. For deterministic auctions, we require that $a_{ij} \in \{0,1\}$. For unit-demand auctions, we also require that every bidder receives at most 1 item: $\sum_j a_{ij} \leq 1$. Denote the set of feasible allocations for a given setting by $A \subset \mathbb{R}^{mn}$. We will often treat A as a set with elements a_k. Payments p_i are simply positive scalars. Given an allocation, bidder i receives utility $u_i = \sum_j a_{ij} v_{ij} - p_i$.

The regret for player i under a given bid profile is defined as the difference in utility between bidding truthfully and the best strategic misreport: $\text{rgt}_i(v) = \max_{b_i} u_i(b_i, v_{-i}) - u_i(v_i)$. When regret is 0 for every player, and for every bid profile, the auction is dominant-strategy incentive compatible (DSIC). In this work, all our auctions have guaranteed zero regret, but some of our baselines may have positive regret. In addition to requiring our auctions to be DSIC, we also require individual rationality (IR) – that is, $u_i \geq 0$ for every bidder, or equivalently, no truthful bidder will ever pay more than the value of the items they receive.

6.4.2 Affine Maximizer Auction Mechanism

Affine maximizer auctions have parameters consisting of weights w_i for each bidder and boosts b_k associated with each allocation a_k. Given some bids $\boldsymbol{v}$ for each bidder, the affine maximizer auction chooses the allocation (and boost) a_k, b_k that will maximize the weighted, boosted welfare: $k^* = \arg\max_k \sum_i w_i \sum_j (a_k)_{ij} \boldsymbol{v}_{ij} + b_k$. Let $a(\boldsymbol{v}) = a_{k^*}, b(\boldsymbol{v}) = b_{k^*}$.

Then, to compute a payment p_i for bidder i, it considers the counterfactual auction result where bidder i did not participate. The total decrease in all other bidder' welfare (weighted and boosted) between this counterfactual auction and the new auction is p_i:

$$p_i = \frac{1}{w_i}\left(\sum_{\ell \neq i}\sum_j w_\ell a(\boldsymbol{v}_{-i})_{\ell j} \boldsymbol{v}_{\ell j} + b(\boldsymbol{v}_{-i})\right) - \frac{1}{w_i}\left(\sum_{\ell \neq i}\sum_j w_\ell a(\boldsymbol{v})_{\ell j} \boldsymbol{v}_{\ell j} + b(\boldsymbol{v})\right) \tag{6.1}$$

As mentioned above, AMAs (like the VCG mechanism) are always DSIC. To see why this is the case, observe that for any fixed set of bids v_{-i}, agent i's utility $u_i(v_i) = \sum_j a(v_i, v_{-i})_{ij} v_{ij} - p_i(v_i, v_{-i})$ will be a pointwise maximum over a set of affine functions (one per possible allocation), and thus convex.

The choice of the above payment rule also ensures IR. We additionally require that allocating nothing and charging nothing always be among the possible outcomes a_k, although this is not strictly required to ensure IR.

Our Approach Generalizes RochetNet and MenuNet When there is only one bidder, without loss of generality we can fix the weights to one and assume welfare when the single bidder is removed is zero, recovering the max-over-affine representation of a strategyproof single-bidder mechanism. Thus our approach of learning allocations and boosts by gradient descent directly generalizes RochetNet [58] and MenuNet [152].

6.5 Learning Affine Maximizers Via Differentiable Economics

AMAs have three types of parameters: the bidder weights w_i, the boosts b_k, and the allocations a_k. (Treating the allocations of AMAs as learned parameters along with the weights and boosts is a contribution of our work.) We assume access to sampled truthful valuations, and learn these parameters jointly via gradient descent on the objective $-\sum_i p_i$.

During training, we use the softmax function as a differentiable surrogate for the max and argmax operations: that is,

$$\arg\max_k f(a_k) \approx \langle \text{softmax}_\tau(f(a_1), \cdots, f(a_K)), \boldsymbol{a} \rangle$$

and

$$\max_k f(a_k) \approx \langle \text{softmax}_\tau(f(a_1), \cdots, f(a_K)), \boldsymbol{f}(\boldsymbol{a}) \rangle$$

. As the softmax temperature parameter τ approaches 0, this approach recovers the true argmax.

Using this soft version of the AMA definition, we directly compute the total payment and differentiate it with respect to the parameters via the Jax autograd system [29] along with Haiku [90] and optax [91]. At test time, we use the learned parameters in the exact AMA definition, using the regular max operator.

For deterministic auctions, we fix the set a_k to be the set of all feasible allocations. For lottery auctions, we randomly initialize a large (typically $|A| = 4096$) set of allocations – although by the end of training, very few of these are actually used (discussed below).

We parameterize these allocations a_k to ensure that they are always feasible. Following the approach from [58], for additive allocations, each allocation is represented by an m by $n+1$ matrix of unrestricted parameters – the extra column is for a dummy item representing "no allocation". We take an item-wise softmax and truncate the dummy column to generate a feasible allocation. For unit-demand allocations, we follow the approach used in [134], applying the softplus operation to two matrices of m by n parameters, normalizing row- and column-wise respectively, and taking the minimum of the result.

6.6 Results

6.6.1 Hyperparameters and training

For lottery AMAs, we allow 2048 or 4096 allocations. The softmax temperature is 100; we use an Adam optimizer with learning rate of 0.01. We train all auctions for 9000 steps, with 2^{15} fresh valuation samples per gradient update, on cluster nodes with 2080Ti GPUs. All reported test revenues are on 100000 sampled valuations. Because the valuation distributions are symmetric, in the cases tested below we fix bidder weights to 1. To determine which allocations are actually used, we sample 100000 test valuations, and include any allocation that was chosen for even one bid profile. For baselines, we compare against previously reported results from RegretNet [58], ALGNet [132], and AMAs trained using other methods [147], as well as theoretical revenues from Myerson auctions of separate items and of the grand bundle. We also include the lowest-regret version of the RegretFormer approach of Ivanov et al. [96]. (Other versions trade off slightly relaxed enforcement of regret for even higher revenue than ALGNet and RegretNet.) Since we only use previous results, please refer to the previous papers for training details of the baselines.

AMA Type	Max Revenue	Min Revenue	Mean Revenue	Std Revenue
Lottery	2.158	1.87	2.06	0.098
Deterministic	1.462	0.627	0.842	0.279

Table 6.1: Results from 8 random parameter initializations, with 2048 allocations, on the spherical valuation distribution. The worst lottery mechanism outperforms the best deterministic mechanism. Moreover, the lottery mechanisms do actually learn to randomize.

6.6.2 Revenue performance

Spherical distribution In order to demonstrate a revenue improvement by offering lotteries, we consider a particular valuation distribution which we refer to as the "spherical distribution" for lack of a better name – this is a distribution on a number of discrete, random points, scaled and normalized according to the proof construction in [33].

We construct such a distribution for 4 items with 5 valuation points and consider a setting with two unit-demand bidders, each of whose valuations are sampled i.i.d from this distribution. We would expect a large gap between revenue extracted by lotteries and by a deterministic mechanism.

Indeed, we find that this is the case – when we train our lottery AMA with 2048 allocations on this distribution, it gets more than twice the revenue of a deterministic AMA (see 6.1). Figure 6.2, left, shows the final offered allocations and boosts from a representative mechanism – the auction is actually taking advantage of randomization.

2 bidder, 2 item uniform We also consider a 2 bidder, 2 item additive auction where item values are independently distributed on $U[0,1]$. This seems like the most trivial possible multi-bidder multi-item auction setting, yet it is so far completely beyond current theory – this makes it an interesting test case for automated mechanism design. We train a lottery AMA on this setting and find revenue competitive with both previous AMA approaches [147, 157] as well as the RegretNet neural network approach (which performs better but is not perfectly strategyproof) [58]. Interestingly, we outperform the lowest-regret variant of the RegretFormer [96], while also having 0 regret. An interesting observation, though, is that even though our lottery AMA is free to offer

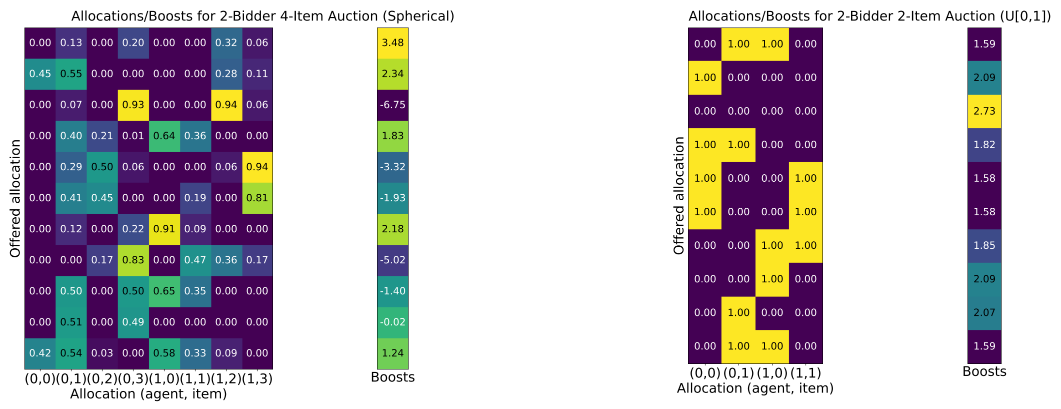


Figure 6.2: **Left**: The ten lottery allocations (and their boosts) actually used after training an auction. (The auction has many more parameters, but the 2038 other allocations are never chosen for any of the sampled bids.) One can see that the mechanism does typically offer lotteries. **Right**:All allocations actually used after training for a 2x2 U[0,1] additive auction (the same setting as compared to in Likhodedov and Sandholm [110] and Dütting et al. [58]). Here, although the mechanism space is that of randomized mechanisms, the algorithm learns to offer deterministic allocations. The revenue is comparable to results in Likhodedov and Sandholm [110].

lotteries, it does not do so – all allocations actually offered by the end of training are deterministic, as seen in Figure 6.2, right.

3 bidder, 10 item uniform Finally, we consider one of the much larger auction settings from [58] – 3 additive bidders with 10 $U[0,1]$ items. We give our network parameters for 4096 allocations, many fewer than the number of possible deterministic allocations in this setting.

Results are shown in Table 6.3, along with baselines. While we do not match the performance of RegretNet and related approaches, we do at least exceed the performance of the separate Myerson and grand bundling approaches. Some, though probably not most, of the extra revenue gained by RegretNet and others may be due to non-zero regret.

We also attempted to train a lottery AMA for the 5 bidder, 10 item uniform case, but found that after several attempts it failed to outperform the separate Myerson baseline.

Number of allocations used We observe that although our auctions are initialized with many parameters, the number of possible deterministic outcomes may be quite large, the number of allocations used on any actual valuation profile is typically quite small. Results are summarized in

Auction	Best Revenue	Regret
Lottery AMA (ours)	0.868	0
Combinatorial AMA	0.862	0
Separate Myerson	0.833	0
Grand Bundle	0.839	0
MBARP	0.871	0
RegretNet	0.878	< 0.001
ALGNet	0.879	0.00058
RegretFormer (low regret budget)	0.818	0.00002

Table 6.2: Revenue comparison for 2 bidder, 2 item U[0,1] additive auction. Our approach is competitive with other approaches. Combinatorial AMA refers to results from Sandholm and Likhodedov [147]. MBARP is a subset of AMA from Tang and Sandholm [157] where the optimal parameters have been computed (only for 2 items). RegretNet achieves higher revenue, but possibly due to a small strategyproofness violation. We manage to outperform the lowest-regret RegretFormer from [96]. Note that Sandholm and Likhodedov [147] present many variants, some of which beat our revenue, although all are comparable.

Auction	Best Revenue	Regret
Lottery AMA (ours)	5.345	0
Separate Myerson	5.31	0
Grand bundle	5.009	0
RegretNet	5.541	0.002
ALGNet	5.562	0.002
RegretFormer (low regret budget)	5.371	0.00017

Table 6.3: Revenue comparison for 3 bidder, 10 item U[0,1] additive auction. We train a lottery AMA with 4096 allocations. It underperforms RegretNet and others (although these approaches have a small strategyproofness violation), but outperforms the separate Myerson and grand bundling baselines.

Auction	Min	Max	# at Initialization	# Deterministic
Lottery spherical	8	15	2048	20
Deterministic spherical	6	9	20	20
2x2 U[0,1]	7	10	4096	9
3x10 U[0,1]	58	64	4096	2^{20}

Table 6.4: The number of allocations actually used after 9000 steps of training, for the experiments given above. Except for the smallest setting, these quantities are smaller than the number of initial outcomes as well as the number of possible deterministic outcomes.

	Mean Rev.	Best Rev.
Winning Ticket (2x2)	0.870	0.872
Small Random (2x2)	0.772	0.777
Winning Ticket (Spherical)	1.836	1.842
Small Random (Spherical)	1.197	1.572

Table 6.5: We take the actually-used allocations from the best-performing 2x2 uniform and spherical models – the values of these parameters *before training* are the "winning ticket". We initialize a lottery AMA using the winning ticket initializations, and train on 4 random data seeds. We also test 4 different random initializations of the same small number of allocations, and find lower performance.

Table 6.4 for all experiments mentioned above.

Effects of parameter initialization Motivated by the *lottery ticket hypothesis* in neural network training [70], we consider the effects of overparameterization and parameter initialization on performance.

First, we consider training from the same parameter initialization, under a different source of randomness for the data. We find that starting from the same initialization typically results in nearly the same allocation indices being chosen, with Jaccard similarities of .64, .67, and .82 across the indices chosen under the new random data. Starting from a different parameter initialization, there was no overlap in the indices chosen. We find that the results are quite similar, which suggests that parameter initialization is important in determining the end results.

We also consider the opposite approach: take the final actually-used allocations, look at what values those parameters took at initialization before training, and retrain using only those parameters. In other words, we train a model with very few parameters "from scratch", but with an initialization we hope will perform well – the “winning lottery ticket”. Results are summarized in Table 6.5. We indeed find a large gap in performance between the good initialization and randomly-initialized models with the same number of parameters.

6.7 Discussion

We see our approach as a first step towards strategyproof architectures for multi-agent differentiable economics. On the one hand, it is a natural generalization of RochetNet and MenuNet. On the other hand, it is also a natural generalization of classic work on AMAs.

Beyond the obvious advantage of perfect strategyproofness, there are other reasons one might prefer this approach over RegretNet. In particular, AMAs are interpretable – it’s easy to simply inspect which allocations are being offered as possibilities. However, it’s unclear when and whether our approach can actually represent the true optimal mechanism – that remains an open theory question. Regardless, we see it as a useful tool for automated mechanism design in multi-bidder multi-item settings.

Lottery ticket hypothesis Our networks are quite sensitive to initialization – there’s a relatively wide range of performance between reinitialized instances of the same architecture shown the same sequence of training data. Moreover, we found that starting out with a large number of parameters improves performance, even though by the end of training only a tiny number of these parameters were actually used. A dependence on initialization, a benefit from overparameterization, and a final model which is effectively sparse all bring to mind the lottery ticket hypothesis [70] in deep learning. Indeed, our experimental results in section 6.6.2 suggest that some version of the lottery ticket hypothesis is in play here. We also observe that Curry et al. [50] was able to significantly distill learned auction networks without harming performance.

6.8 Strengths and Limitations

Limitations compared to other differentiable economics approaches The most obvious limitation of our work is that in most settings, there are probably strategyproof non-AMAs which outperform the best AMA, but we cannot learn these. This is most likely why ALGNet and RegretNet outperform our mechanisms in terms of revenue. On the other hand, we think that ensuring perfect strategyproofness is a real advantage, and a more flexible neural architecture that preserves this property while going beyond AMAs remains out of reach.

Limitations compared to other AMA/AMD approaches Our approach works for settings with additive and unit-demand valuations, where the bidders' valuations are just vectors so that the total value of a bundle can be expressed as an inner product. It is not straightforwardly well-suited to complex combinatorial valuations where bidders may value bundles very differently depending on the presence of specific items. Moreover, we solve the winner determination problem by explicitly computing the (weighted, boosted) welfare for every possible allocation. This works because at any given time, our auctions are only offering a restricted set of learned allocations, which is usually much smaller than the overall set of possible allocations. But in a setting where it was important to offer something closer to the full set of allocations, our approach would quickly become intractable.

Experiment scale Palfrey [126] shows that as the number of bidders grow large, separate Myerson auctions become optimal, making optimal mechanism design less advantageous in the case of large numbers of bidders. Although some works [57, 58, 132] train 5x10 auction networks, 3 agents by 10 items is among the largest auction sizes considered by most differentiable economics papers. More generally, in terms of the number of items, 10 items is already quite large for a combinatorial auction. Thus, we see the scale of our experiments as both the right size to be interesting and commensurate with other work in the space.

Ethical concerns Our technique does not result in any new ethical concerns beyond any that might already exist for automated mechanism design techniques.

Strength: interpretability A major advantage of our approach is in interpretability. Neural-network-based approaches are almost totally opaque – there's no way to summarize or explain the learned mechanism to the bidders, other than simply giving them the network parameters. By contrast, in our approach, we can simply describe the possible allocations, their boosts, and each bidder's weights, and it is immediately clear how the mechanism works.

6.9 Future Research

We focus on auctions because there is a large body of techniques in automated mechanism design and differentiable economics which provide useful baselines for performance. We expect that the approach described here could be extended to other mechanism design problems as long as 1) VCG-style mechanisms can be used, 2) feasible mechanism outcomes can be parameterized in a way amenable to gradient-based learning, and 3) the welfare of an outcome as a function of agent types can be computed in a way that preserves differentiability. Exploring the use of learned AMAs in new mechanism design settings is a fruitful direction for future work.

Chapter 7: Future Work and Open Questions

In this chapter, we describe some promising future directions at the intersection of machine learning and mechanism design. Some are already in progress, some have not yet been attempted, and some are simply interesting approaches being taken by other researchers.

7.1 Reinforcement Learning and Dynamic Mechanism Design

There are other cases in which learning is useful as well – [169] uses RL to remove simplifying assumptions from macroeconomic models, and [89] presents techniques for learning good bidder strategies in a non-strategyproof mechanism. A line of work exemplified by Brero, Lubin, and Seuken [31] uses machine learning to model agents' complicated utility functions, sidestepping the need to communicate an exponential amount of information to the mechanism. Another interesting direction is *dynamic* mechanism design – Brero et al. [32] considers using reinforcement learning to optimize mechanisms that interact with myopic bidders in sequential rounds, and this direction could be pushed further. It's also worth noting that in the context of credible auctions (discussed below), sequential mechanisms could provide some advantages, and it might be worth learning them.

7.2 Other direct-revelation mechanisms

There is much room for future work. We have focused mainly on the setting of private-value auctions – here the goal is to learn a mechanism, represented as a function, which has good performance in expectation over the prior on agents' types.

Mechanisms that buy and sell A recent line of work has considered constant-function market makers (CFMMs) [10, 11, 12], a particular class of automated market makers [3]. CFMMs start with some initial pool of assets and will trade with a counterparty as long as the trade preserves some invariant function over that pool. Angeris et al. [12] shows that, under a model where an arbitrageur with linear utility knows the true price and trades opportunistically, there is a CFMM that can replicate any concave payoff curve as a function of the true price.

Since the game between the arbitrageur and the market maker is zero sum, this is equivalent to saying that there is a CFMM that will produce any convex utility function for the arbitrageur.

From this perspective, there is a clear connection to truthful mechanism design. Every DSIC single-agent mechanism must have a convex utility function for the participant, whose (sub)gradients are the allocations [137]. Here, the CFMM is the mechanism, and the difference from traditional auction design settings is that the gradients may go negative, representing situations where the mechanism is willing to buy goods from the arbitrageur. (This description is less general, but simpler and more directly related to CFMMs, than the connections between scoring rules, automated market makers, and mechanisms laid out in Frongillo and Kash [71].)

Can the tools of single-agent auction design, also be applied fruitfully to CFMM design? In some preliminary work, it appears that the answer is yes. We can use a RochetNet/MenuNet style approach to learn CFMMs; and with some small modifications, the theory of strong duality following [54, 98] seemingly goes through.

The main challenge is unfortunately more fundamental – it's unclear what a realistic mechanism design objective even is. Maximizing revenue recovers the optimal auction design problem. Assuming that the auctioneer disagrees with the arbitrageur about the value of the goods results in interesting solutions, but requires fairly implausible assumptions about arbitrageur behavior (a truly smart arbitrageur would be able to infer the auctioneer's value by looking at the published mechanism and incorporate this new information).

Two-sided matching Ravindranath et al. [134] attempts to use deep learning for two-sided matching problems. Here, there is no question of revenue maximization. Instead, they use as baselines two famous algorithms for two-sided matching: random serial dictatorship [1], which offers lotteries over matchings and is strategyproof, and the classic deferred acceptance algorithm [72], which is stable. Since achieving both strategyproofness and stability at once is impossible [141], the idea of the paper is to use deep learning to learn mechanisms that trade off between these properties, hopefully pushing the "Pareto frontier" outwards.

The main limitation of this work (as they point out in the conclusion) is that they don't completely cope with the fact that ordinal preference profiles for matching are discrete objects (unlike typical type profiles in auctions). Thus, there is no obvious way to use a gradient-based method to search for worst-case misreports of preferences; they instead enumerate all possible misreports. But this severely limits the scale of matching problems to which the technique can be applied. Overcoming this limitation would be a worthwhile direction for future research.

7.3 Other mechanism design goals

Credibility Credibility is a new desideratum for auctions introduced by Akbarpour and Li [5]. The intuitive idea behind credibility is that in some auction formats, auctioneers can lie to one bidder about the reported bids of other bidders. For example, in a second-price auction, the auctioneer might claim the second-highest bid was much higher in order to charge the winner a higher price. In a credible auction, there should be no incentive for the auctioneer to do this – a first-price auction, for example, has this property. Thus credibility is a kind of truthfulness property for the auctioneer rather than the bidders.

Akbarpour and Li [5] presents a trilemma: it is impossible for auctions to simultaneously be credible, strategyproof for the bidders, and static (meaning the auction happens in a single round).

Applying differentiable economics to the problem of credible auction design in the static setting is an interesting challenge. It might be a tool for exploring what credible multi-item mechanisms look like. Since in the static setting achieving both strategyproofness and credibility is impossible, it might allow moving along the Pareto frontier between these two properties, similar to the approach for two-sided matching mentioned above. And, as mentioned above, sequential mechanism design is another interesting research direction, albeit further afield.

7.4 Automated duality theory

For single-agent mechanism design, there is a theory of optimal-transport based duality – it is always possible in principle to check the optimality of a proposed single-agent mechanism, under a variety of different allocation constraints [54, 98]. The general idea is that, appealing to Rochet's classic result [137], each mechanism can be identified with its utility function. The objective (expected revenue) can be rewritten to simply involve integrating this utility against a signed measure constructed from the valuation distribution – in other words, it is a linear optimization problem in the space of functions. Dual to this problem is an optimal transport problem – a solution to this problem is a cost-minimizing transport plan between pairs of possible bidder types. Then all one needs is to find a pair of such solutions, which is guaranteed to exist.

Actually solving either side of this problem is not particularly easy. Standard approaches involve sitting and thinking very hard to come up with good mechanisms, which is an unreliable approach, especially for those who are less good at thinking. This is a large part of the appeal of RochetNet and MenuNet – after a modest amount of training, these approaches will give a very plausible solution for the primal side of the problem.

There is not currently any such approach for the dual side of the problem. In [54, 98], there are some "recipes" given that allow decomposing the problem into easier subproblems, but still require some geometric creativity. And for dealing with cases where there are more than two items, things are even more difficult.

The creators of MenuNet and RochetNet use their learning-based methods to find candidate optimal mechanisms. But to prove these methods optimal, they still simply think very hard to come up with dual solutions. Ideally, there would be a numerical or learning-based method for solving the dual side of the problem as well – then this would truly be automated mechanism design.

There are many effective numerical techniques for solving optimal transport problems, such as the Sinkhorn algorithm [51]. However, the optimal transport problems that show up in mechanism design have an unusual wrinkle – they don't have the typical exact marginal constraints of OT. Instead, there is a relaxation – some amount of mass can be shifted around. This relaxation makes it unclear how exactly to apply existing techniques for OT. One possible direction is connecting the relaxed OT problem to problems like unbalanced optimal transport [40] or martingale optimal transport [20]. But there is no obvious technique in the literature that is directly applicable to the relaxed OT problem from mechanism design.

Working through the connections and finding an automatic way to solve the dual mechanism design problem, and therefore to check optimality of proposed mechanisms, would be a useful complement to techniques like MenuNet and RochetNet, as well as probably useful to mechanism design theorists.

(It's worth noting an alternate, and much simpler, approach. This is to finely discretize the space of possible types, and then directly write down a linear program to maximize expected revenue subject to DSIC constraints. This is equivalent to the primal program above, and it has a dual program which is also just a normal LP. But, as shown e.g. in Shen, Tang, and Zuo [152], this approach scales quite poorly. You need n^2 DSIC constraints, where n is the number of discretized grid points, a number that itself will grow exponentially in the dimension of the types for the same degree of accuracy. The dual problem has a variable for each of these constraints. The hope is to find some approach that, in practice, manages to dodge these scaling issues, as MenuNet/RochetNet are able to do on the primal side.)

7.5 The Ultimate Auction Architecture

As described in Chapter 6, RegretNet and related neural architectures cannot guarantee perfect strategyproofness. There are approaches restricted to the single-bidder case that are in fact perfectly strategyproof; and our differentiable-AMA approach works for multiple bidders but cannot always represent the optimal auction.

It would be desirable to have an architecture which is capable of representing exactly the set of feasible strategyproof mechanisms. Then, one would simply need to train on the single objective of revenue maximization – and supervised learning on a single objective with potentially infinite data should be very easy. However, this goal seems difficult to achieve. Below we summarize some of the theoretical properties required of such a mechanism, and partial solutions to some related practical problems.

7.5.1 Implementable allocation rules

Rochet [137] shows that, when considering a single bidder, every allocation rule that can be part of a DSIC mechanism – that is "implementable in dominant strategies" – must be the gradient of some convex utility function, so that we have $u(v_i) = \langle \nabla u(v_i), v_i \rangle - p_i(v_i) = \langle a(v_i), v_i \rangle - p_i(v_i)$. Assuming for the sake of simplicity that the "zero type" receives nothing and pays nothing, the payment rule to produce a DSIC mechanism is then $p(v_i) = \langle a(v_i), v_i \rangle - \int_0^1 a(tv_i) \cdot v_i \, dt$ (a multidimensional version of Myerson's lemma [123]).

What sorts of functions are the gradients of convex functions? In the nicest case, where everything is twice differentiable, we know that convex functions must have positive semidefinite Hessians; so a vector-valued function with a PSD Jacobian will be the gradient of a convex function.

In general, we expect there to be "sharp" non-differentiable points in mechanism utility functions, so this definition isn't completely satisfactory. Rockafellar [138] showed that (sub)gradients of convex functions must satisfy a property known as cyclic monotonicity, and that every cyclically monotone set-valued map is at least contained in some larger map that is the subgradient map of a convex function. Cyclic monotonicity holds when, for any sequence of types $v_1, \cdots, v_k, v_{k+1} = v_1$:

$$\sum_{i=0}^{k} \langle v_{i+1} - v_i, a(v_{i+1}) - a(v_i) \rangle \geq 0$$

[13].

(It's possible to think of this as something like a discrete version of a closed-loop line integral –

so it is analogous to the condition that, if a vector field is the gradient of some potential, all closed line integrals must integrate to 0 [139].)

Roberts' theorem [136] shows that for "unrestricted domains", meaning domains where bidders have arbitrary real-valued valuations on outcomes, and are not indifferent to what other agents receive, the only allocation rules which can be implementable are those that maximize some affine transformation of the total welfare (i.e. the affine maximizers).

Other theoretical work considers domains where less stringent requirements imply cyclic monotonicity and therefore implementability. For example, Saks and Yu [144] shows that, for convex type spaces and mechanisms with a finite number of possible allocations, a property known as weak monotonicity – just cyclic monotonicity with sequences of only length 2 – is sufficient for implementability. Various results in this direction have been further extended and generalized [13, 16, 71].

Multi-bidder extensions These definitions can be straightforwardly extended to multi-bidder auctions. For a multi-bidder auction to be DSIC, it must be DSIC for every bidder i, for any fixed value v_{-i} of the other bidders' bids. Therefore, for a multi-bidder allocation rule to be implementable in dominant strategies, it must be the case that $a(\cdot, v_{-i})_i$ is cyclically monotone for every i and for every possible v_{-i}. The payment rule is also simply $p(v_i, v_{-i})_i = \langle a(v_i, v_{-i})_i, v_i \rangle - \int_0^1 a(tv_i, v_{-i})_i \cdot v_i \, dt$. (Actually making use of this extension is not so easy.)

7.5.2 Practical solutions

Allocation rule Thus, we need to ensure that for each agent, the allocation rule they face is always the gradient of a convex function (i.e. cyclically monotone). This is, somewhat surprisingly, easy to do in several ways.

Perhaps the simplest would be to simply define the allocation rule for each agent (i.e. a row of an allocation matrix) to be the gradient of an input-convex neural network (ICNN) [9]. This approach implicitly involves taking second derivatives with respect to network parameters during training, but modern autograd frameworks handle this reasonably well, and similar approaches have worked for other applications [114]. An approach with a more mechanism-design-inspired story might be to parameterize a neural network that, for each bidder, outputs a menu of allocations and prices as a function of only opponent bids v_{-i}; bidder i then receives their utility-maximizing menu item. (This is essentially RochetNet/MenuNet but varying as a function of opponent bids.)

One also has to meet allocation constraints: people should receive positive allocations between 0 and 1, no item should be overallocated, and so on. For the ICNN approach, in many cases it should be possible to enforce bidder-wise constraints by constraining the gradients of the ICNN (e.g. by normalizing weight matrix rows/columns). For the RochetNet-inspired approach, one can simply directly enforce bidder-wise constraints.

The challenge is itemwise constraints – i.e. ensuring that, across all bidders, no item is overallocated. In the RegretNet approach, one applies the softmax function independently to each row of the matrix to ensure they sum to 1 (adding a dummy agent to receive unallocated items). Unfortunately, composing this operation with a function that is row-wise cyclically monotone does not preserve the cyclic monotonicity property – it doesn't even preserve weak monotonicity. There is no obvious way to enforce the itemwise feasibility constraints without destroying monotonicity.

Payment rule The two above approaches construct the utility function for each agent, so it's straightforward to compute the DSIC payment rule. However, it's possible there is some promising approach where this does not hold. But given a neural architecture that represents an implementable allocation rule (and doesn't automatically specify a payment rule or induced utility), it is possible to directly compute the Myerson payment by simply numerical computing the line integral. At train time, this would have to be done in a differentiable manner. Somewhat surprisingly, this is fairly easy, using for instance differentiable quadrature [163] or a differentiable ODE solver (e.g. [99]).

Making this work The problem of allocation constraints is where this approach falls apart – there is not any obvious way to ensure that items are not overallocated without destroying cyclic monotonicity. From a certain perspective, it's not so surprising that this problem should be very difficult: for each bidder we have an entire family of truthful mechanisms (indexed by opponent bid profiles). Then, all of these families have to be carefully glued together while remaining compatible at each point. There may not be any simple architectural tricks to make this work.

One compromise might be to just ignore allocation constraints – then one would have a perfectly DSIC mechanism that might nonsensically overallocate items. The training loss could incorporate a term for revenue maximization and a term (probably with an increasing Lagrange multiplier) for allocation feasibility, hopefully resulting in a feasible mechanism with good revenue. This is a promising incremental direction. Even though there are still multiple moving parts in the training procedure, it seems plausible that, as far as training dynamics go, enforcing allocation feasibility (a property which can be checked pointwise) might be easier than enforcing low regret (a property which depends simultaneously on all possible types). Indeed, in chapters 4 and 3 we already show the ability to train-time encourage certain allocation constraints.

Chapter 8: Conclusion

In the private-value setting, it's natural to interpret mechanism design as a learning problem, where the prior over agent types is the data distribution, the mechanism design goal (revenue, welfare, or something else) is the loss function, and properties like strategyproofness can be thought of in terms of robustness to adversaries. Dütting et al. [58] takes advantage of this connection to learn state-of-the-art optimal auctions.

In this work, we have explored several new research directions. In Chapter 2, we computed explicit bounds on the regret suffered by RegretNet-style models, correcting one of the major limitations of that approach. In Chapters 3 and 4, we extended the approach to mechanism design goals beyond revenue, possibly learned from vague human preferences. In Chapter 5, we extended the scope of RegretNet to capture more constraints on the allocations. In Chapter 6, we limited the scope of differentiable economics, accepting a reduced ability to represent auctions in exchange for the desirable property of perfect strategyproofness in multi-agent settings. And in Chapter 7, we described several promising, or at least plausible, future research directions.

This and other research has shown that applied machine learning has a role to play in automated mechanism design. But there is still further progress to be made. With luck and persistence, it should be possible to improve these tools to be even better and more reliable, and use them to design real-world mechanisms that unlock real economic value.

Bibliography

[1] Atila Abdulkadiroğlu and Tayfun Sönmez. “Random serial dictatorship and the core from random endowments in house allocation problems”. In: *Econometrica* 66.3 (1998), pp. 689–701.

[2] Atila Abdulkadiroglu et al. *Changing the Boston School Choice Mechanism*. Working Paper 11965. National Bureau of Economic Research, Jan. 2006. DOI: `10.3386/w11965`.

[3] Jacob D Abernethy and Rafael M Frongillo. “A characterization of scoring rules for linear properties”. In: *Conference on Learning Theory*. 2012.

[4] Akshay Agrawal et al. “Differentiable Convex Optimization Layers”. In: *Neural Information Processing Systems (NeurIPS)*. 2019.

[5] Mohammad Akbarpour and Shengwu Li. “Credible auctions: A trilemma”. In: *Econometrica* 88.2 (2020), pp. 425–467.

[6] Muhammad Ali et al. “Discrimination through Optimization”. In: *Proceedings of the ACM on Human-Computer Interaction* (2019). DOI: `10.1145/3359301`. URL: `http://dx.doi.org/10.1145/3359301`.

[7] Alphabet. *Alphabet Announces Second Quarter 2020 Results*. `https://abc.xyz/investor/static/pdf/2020Q2_alphabet_earnings_release.pdf`. 2020.

[8] Brandon Amos and J Zico Kolter. “Optnet: Differentiable optimization as a layer in neural networks”. In: *International Conference on Machine Learning (ICML)*. 2017.

[9] Brandon Amos, Lei Xu, and J Zico Kolter. “Input Convex Neural Networks”. In: (Sept. 2016). arXiv: `1609.07152 [cs.LG]`.

[10] Guillermo Angeris and Tarun Chitra. “Improved Price Oracles: Constant Function Market Makers”. In: (Mar. 2020). arXiv: `2003.10001 [q-fin.TR]`.

[11] Guillermo Angeris, Alex Evans, and Tarun Chitra. *Replicating monotonic payoffs without collateral*. `https://web.stanford.edu/~guillean/papers/cfmm-monotone.pdf`. 2021.

[12] Guillermo Angeris et al. “Constant function market makers: Multi-asset trades via convex optimization”. In: (July 2021). arXiv: `2107.12484 [math.OC]`.

[13] Aaron Archer and Robert Kleinberg. “Truthful germs are contagious: a local-to-global characterization of truthfulness”. In: *Games and Economic Behavior* 86 (2014), pp. 340–366.

[14] Martin Arjovsky, Soumith Chintala, and Léon Bottou. “Wasserstein Generative Adversarial Networks”. In: *International Conference on Machine Learning (ICML)*. 2017.

[15] Mark Armstrong. “Optimal Multi-Object Auctions”. In: *Review of Economic Studies* 67 (2000), pp. 455–481.

[16] Itai Ashlagi et al. “Monotonicity and implementability”. In: *Econometrica* 78.5 (2010), pp. 1749–1772.

[17] Anish Athalye, Nicholas Carlini, and David A. Wagner. “Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples”. In: *International Conference on Machine Learning (ICML)*. 2018.

[18] Christopher Avery and Terrence Hendershott. “Bundling and Optimal Auctions of Multiple Products”. In: *Review of Economic Studies* 67 (2000), pp. 483–497.

[19] Yoram Bachrach et al. “A Neural Network Auction For Group Decision Making Over a Continuous Space”. In: *International Joint Conference on Artificial Intelligence (IJCAI)*. 2021.

[20] Julio Backhoff-Veraguas and Gudmund Pammer. “Stability of martingale optimal transport and weak optimal transport”. In: *The Annals of Applied Probability* 32.1 (2022), pp. 721–752.

[21] Baidu. *Baidu Announces First Quarter 2020 Results*. `http://ir.baidu.com/news-releases/news-release-details/baidu-announces-first-quarter-2020-results`. 2020.

[22] Maria-Florina F Balcan, Tuomas Sandholm, and Ellen Vitercik. “Sample complexity of automated mechanism design”. In: *Advances in Neural Information Processing Systems*. 2016, pp. 2083–2091.

[23] Maria-Florina Balcan, Tuomas Sandholm, and Ellen Vitercik. “A general theory of sample complexity for multi-item profit maximization”. In: *Economics and Computation (EC)*. 2018.

[24] Maria-Florina Balcan, Tuomas Sandholm, and Ellen Vitercik. “Estimating Approximate Incentive Compatibility”. In: *Economics and Computation (EC)*. 2019.

[25] Maria-Florina Balcan et al. “How much data is sufficient to learn high-performing algorithms? Generalization guarantees for data-driven algorithm design”. In: *ACM Symposium on Theory of Computing (STOC)*. 2021.

[26] Solon Barocas and Andrew D Selbst. “Big data’s disparate impact”. In: *Calif. L. Rev.* 104 (2016), p. 671.

[27] Martin Bichler et al. “Learning equilibria in symmetric auction games using artificial neural networks”. In: *Nature Machine Intelligence* 3.8 (2021), pp. 687–695.

[28] Garrett Birkhoff. “Tres observaciones sobre el algebra lineal”. In: *Univ. Nac. Tucuman, Ser. A* 5 (1946), pp. 147–154.

[29] James Bradbury et al. *JAX: composable transformations of Python+NumPy programs*. Version 0.2.5. 2018. URL: `http://github.com/google/jax`.

[30] Gianluca Brero, Sébastien Lahaie, and Sven Seuken. "Fast Iterative Combinatorial Auctions via Bayesian Learning". In: *AAAI Conference on Artificial Intelligence*. 2019.

[31] Gianluca Brero, Benjamin Lubin, and Sven Seuken. "Machine Learning-powered Iterative Combinatorial Auctions". In: *arXiv preprint arXiv:1911.08042* (2019).

[32] Gianluca Brero et al. "Reinforcement Learning of Sequential Price Mechanisms". In: *AAAI Conference on Artificial Intelligence*. 2021.

[33] Patrick Briest et al. "Pricing randomized allocations". In: *ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 2010.

[34] Rudy Bunel et al. "A Unified View of Piecewise Linear Neural Network Verification". In: *Neural Information Processing Systems (NeurIPS)*. 2018.

[35] Yang Cai, Constantinos Daskalakis, and S Matthew Weinberg. "An algorithmic characterization of multi-dimensional mechanisms". In: *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*. 2012, pp. 459–478.

[36] Yang Cai, Constantinos Daskalakis, and S Matthew Weinberg. "Optimal multi-dimensional mechanism design: Reducing revenue to welfare maximization". In: *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*. IEEE. 2012, pp. 130–139.

[37] Yang Cai, Constantinos Daskalakis, and S Matthew Weinberg. "Understanding incentives: Mechanism design becomes algorithm design". In: *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*. IEEE. 2013, pp. 618–627.

[38] L Elisa Celis, Anay Mehrotra, and Nisheeth K Vishnoi. "Toward controlling discrimination in online ad auctions". In: *International Conference on Machine Learning (ICML)*. 2019.

[39] Shuchi Chawla and Meena Jagadeesan. "Fairness in ad auctions through inverse proportionality". In: *arXiv preprint arXiv:2003.13966* (2020).

[40] Lenaic Chizat et al. "Scaling algorithms for unbalanced optimal transport problems". In: *Mathematics of Computation* 87.314 (2018), pp. 2563–2609.

[41] Edward H Clarke. “Multipart pricing of public goods”. In: *Public choice* (1971), pp. 17–33.

[42] Richard Cole and Tim Roughgarden. “The sample complexity of revenue maximization”. In: *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*. 2014, pp. 243–252.

[43] Benoıt Colson, Patrice Marcotte, and Gilles Savard. “An overview of bilevel optimization”. In: *Annals of Operations Research* 153.1 (Apr. 2007), pp. 235–256.

[44] Vincent Conitzer and Tuomas Sandholm. “Applications of Automated Mechanism Design”. In: *UAI-03 workshop on Bayesian Modeling Applications*. 2003.

[45] Vincent Conitzer and Tuomas Sandholm. “Complexity of Mechanism Design”. In: *Uncertainty in Artificial Intelligence (UAI)*. 2002.

[46] Vincent Conitzer and Tuomas Sandholm. “Incremental Mechanism Design”. In: *International Joint Conference on Artificial Intelligence (IJCAI)*. 2007.

[47] Peter Cramton. “Electricity market design”. In: *Oxford Review of Economic Policy* 33.4 (2017), pp. 589–612.

[48] Peter Cramton. “The FCC spectrum auctions: An early assessment”. In: *Journal of Economics & Management Strategy* 6.3 (1997), pp. 431–495.

[49] Michael J Curry et al. “Learning Revenue-Maximizing Auctions With Differentiable Matching”. In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*. 2021.

[50] Michael Curry et al. “Certifying Strategyproof Auction Networks”. In: *Neural Information Processing Systems (NeurIPS)*. 2020.

[51] Marco Cuturi. “Sinkhorn distances: Lightspeed computation of optimal transport”. In: *Neural Information Processing Systems (NeurIPS)*. 2013.

[52] Marco Cuturi, Olivier Teboul, and Jean-Philippe Vert. “Differentiable Ranking and Sorting using Optimal Transport”. In: *Neural Information Processing Systems (NeurIPS)*. 2019.

[53] Constantinos Daskalakis, Alan Deckelbaum, and Christos Tzamos. “Mechanism design via optimal transport”. In: *Proceedings of the fourteenth ACM conference on Electronic commerce*. 2013, pp. 269–286.

[54] Constantinos Daskalakis, Alan Deckelbaum, and Christos Tzamos. “Strong Duality for a Multiple-Good Monopolist”. In: *Econometrica* 85.3 (2017), pp. 735–767.

[55] Amit Datta, Michael Carl Tschantz, and Anupam Datta. “Automated Experiments on Ad Privacy Settings”. In: *Proceedings on Privacy Enhancing Technologies* 2015.1 (2015), pp. 92–112. DOI: `https://doi.org/10.1515/popets-2015-0007`. URL: `https://content.sciendo.com/view/journals/popets/2015/1/article-p92.xml`.

[56] Yuan Deng et al. “Towards Efficient Auctions in an Auto-bidding World”. In: *The Web Conference*. 2021.

[57] Zhijian Duan et al. “A Context-Integrated Transformer-Based Neural Network for Auction Design”. In: *arXiv preprint arXiv:2201.12489* (2022).

[58] Paul Dütting et al. “Optimal auctions through deep learning”. In: *International Conference on Machine Learning*. 2019.

[59] Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz. “Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords”. In: *American Economic Review* 97.1 (2007), pp. 242–259.

[60] EEOC. *Uniform Guidelines on Employment Selection Procedures, 29 C.F.R. §1607.4(D) (1978)*. 1978.

[61] Rüdiger Ehlers. “Formal Verification of Piece-Wise Linear Feed-Forward Neural Networks”. In: *Automated Technology for Verification and Analysis*. Springer, 2017.

[62] Patrick Emami and Sanjay Ranka. “Learning permutations with sinkhorn policy gradient”. In: *arXiv preprint arXiv:1805.07010* (2018).

[63] Facebook. *About Ad Auctions*. https://www.facebook.com/business/help/430291176997542?id=561906377587030.

[64] Facebook. *Facebook Reports First Quarter 2020 Results*. https://s21.q4cdn.com/399680738/files/doc_news/Facebook-Reports-First-Quarter-2020-Results-2020.pdf. 2020.

[65] Irfan Faizullabhoy and Aleksandra Korolova. “Facebook’s advertising platform: New attack vectors and the need for interventions”. In: *arXiv preprint arXiv:1803.10099* (2018).

[66] Jon Feldman et al. “Online Ad Assignment with Free Disposal”. In: *Internet and Network Economics*. Springer Berlin Heidelberg, 2009, pp. 374–385.

[67] Michael Feldman et al. “Certifying and removing disparate impact”. In: *International Conference on Knowledge Discovery and Data Mining (KDD)*. 2015.

[68] Zhe Feng, Harikrishna Narasimhan, and David C. Parkes. “Deep Learning for Revenue-Optimal Auctions with Budgets”. In: *International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*. 2018.

[69] Aaron Ferber et al. “Mipaal: Mixed integer program as a layer”. In: *AAAI Conference on Artificial Intelligence*. 2020.

[70] Jonathan Frankle and Michael Carbin. “The lottery ticket hypothesis: Finding sparse, trainable neural networks”. In: *International Conference on Learning Representations (ICLR)*. 2019.

[71] Rafael M Frongillo and Ian A Kash. “General truthfulness characterizations via convex analysis”. In: *Games and Economic Behavior* 130 (2021), pp. 636–662.

[72] David Gale and Lloyd S Shapley. “College admissions and the stability of marriage”. In: *The American Mathematical Monthly* 69.1 (1962), pp. 9–15.

[73] Sainyam Galhotra, Sandhya Saisubramanian, and Shlomo Zilberstein. “Learning to Generate Fair Clusters from Demonstrations”. In: (Feb. 2021). arXiv: 2102.03977 [stat.ML].

[74] Robert Geirhos et al. “Shortcut learning in deep neural networks”. In: *Nature Machine Intelligence* 2.11 (2020), pp. 665–673.

[75] Aude Genevay, Gabriel Peyré, and Marco Cuturi. “Learning generative models with sinkhorn divergences”. In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*. 2018.

[76] Yiannis Giannakopoulos and Elias Koutsoupias. “Duality and optimality of auctions for uniform distributions”. In: *Economics and Computation (EC)*. 2014.

[77] Noah Golowich, Harikrishna Narasimhan, and David C. Parkes. “Deep Learning for Multi-Facility Location Mechanism Design”. In: *International Joint Conference on Artificial Intelligence (IJCAI)*. 2018.

[78] Google. *Ad Rank thresholds: Definition*. `https://support.google.com/google-ads/answer/7634668`.

[79] Stephen Gould, Richard Hartley, and Dylan Campbell. “Deep declarative networks: A new hope”. In: *arXiv preprint arXiv:1909.04866* (2019).

[80] Sven Gowal et al. “On the Effectiveness of Interval Bound Propagation for Training Verifiably Robust Models”. In: *CoRR* abs/1810.12715 (2018). arXiv: `1810.12715`.

[81] Griggs. *Griggs v. Duke Power Co., 401 U.S. 424 (1971)*. 1971.

[82] Aditya Grover et al. “Stochastic optimization of sorting networks via continuous relaxations”. In: *arXiv preprint arXiv:1903.08850* (2019).

[83] Theodore Groves. “Incentives in teams”. In: *Econometrica: Journal of the Econometric Society* (1973), pp. 617–631.

[84] Mingyu Guo, Hideaki Hata, and Ali Babar. “Optimizing affine maximizer auctions via linear programming: an application to revenue maximizing mechanism design for zero-day exploits markets”. In: *International Conference on Principles and Practice of Multi-Agent Systems*. 2017.

[85] Gurobi Optimization, LLC. *Gurobi Optimizer Reference Manual*. 2020. URL: `http://www.gurobi.com`.

[86] Nima Haghpanah and Jason D. Hartline. “Reverse Mechanism Design”. In: *CoRR* abs/1404.1341 (2014). arXiv: `1404.1341`.

[87] Alex Hanna et al. “Towards a critical race methodology in algorithmic fairness”. In: *Conference on Fairness, Accountability, and Transparency (FAccT)*. 2020.

[88] Sergiu Hart and Noam Nisan. “Selling multiple correlated goods: Revenue maximization and menu-size complexity”. In: *Journal of Economic Theory* 183 (2019), pp. 991–1029.

[89] Stefan Heidekrüger et al. “Equilibrium Learning in Combinatorial Auctions: Computing Approximate Bayesian Nash Equilibria via Pseudogradient Dynamics”. In: *arXiv preprint arXiv:2101.11946* (2021).

[90] Tom Hennigan et al. *Haiku: Sonnet for JAX*. Version 0.0.3. 2020. URL: `http://github.com/deepmind/dm-haiku`.

[91] Matteo Hessel et al. *Optax: composable gradient transformation and optimisation, in JAX!* Version 0.0.1. 2020. URL: `http://github.com/deepmind/optax`.

[92] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. “Distilling the Knowledge in a Neural Network”. In: *CoRR* abs/1503.02531 (2015). arXiv: `1503.02531`.

[93] Kenneth Holstein et al. “Improving fairness in machine learning systems: What do industry practitioners need?” In: *Conference on Human Factors in Computing Systems (CHI)*. 2019.

[94] Ben Hutchinson and Margaret Mitchell. “50 years of test (un) fairness: Lessons for machine learning”. In: *Conference on Fairness, Accountability, and Transparency (FAccT)*. 2019.

[95] Christina Ilvento, Meena Jagadeesan, and Shuchi Chawla. “Multi-Category Fairness in Sponsored Search Auctions”. In: *Conference on Fairness, Accountability, and Transparency (FAccT)*. 2020.

[96] Dmitry Ivanov et al. “Optimal-er Auctions through Attention”. In: *arXiv preprint arXiv:2202.13110* (2022).

[97] Leonid V Kantorovich. “On the translocation of masses”. In: *Dokl. Akad. Nauk. USSR (NS)*. Vol. 37. 1942, pp. 199–201.

[98] Ian A Kash and Rafael M Frongillo. “Optimal auctions with restricted allocations”. In: *Economics and Computation (EC)*. 2016.

[99] Patrick Kidger. “On Neural Differential Equations”. PhD thesis. University of Oxford, 2021.

[100] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[101] Kris Korrel. *sparsemax-pytorch*. Version v1.0.0. May 2020. DOI: `10.5281/zenodo.3860669`. URL: `https://doi.org/10.5281/zenodo.3860669`.

[102] Kris Korrel et al. “Transcoding Compositionally: Using Attention to Find More Generalizable Solutions”. In: *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. 2019.

[103] Vijay Krishna. *Auction theory*. Academic press, 2009.

[104] Kevin Kuo et al. “ProportionNet: Balancing Fairness and Revenue for Auction Design with Deep Learning”. In: *arXiv preprint arXiv:2010.06398* (2020).

[105] Anja Lambrecht and Catherine Tucker. “Algorithmic bias? An empirical study of apparent gender-based discrimination in the display of STEM career ads”. In: *Management Science* 65.7 (2019), pp. 2966–2981.

[106] R Lavi, Ahuva Mu’alem, and N Nisan. “Towards a characterization of truthful combinatorial auctions”. In: *Symposium on Foundations of Computer Science (FOCS)*. Oct. 2003.

[107] Kevin Leyton-Brown, Paul Milgrom, and Ilya Segal. “Economics and computer science of a radio spectrum reallocation”. In: *Proceedings of the National Academy of Sciences (PNAS)* 114.28 (2017), pp. 7202–7209.

[108] Jingling Li et al. “Noisy Labels Can Induce Good Representations”. In: *CoRR* abs/2012.12896 (2020). URL: `https://arxiv.org/abs/2012.12896`.

[109] Anton Likhodedov and Tuomas Sandholm. “Approximating Revenue-Maximizing Combinatorial Auctions”. In: *AAAI Conference on Artificial Intelligence*. 2005.

[110] Anton Likhodedov and Tuomas Sandholm. “Methods for boosting revenue in combinatorial auctions”. In: *AAAI*. 2004, pp. 232–237.

[111] Xiangyu Liu et al. “Neural Auction: End-to-End Learning of Auction Mechanisms for E-Commerce Advertising”. In: *arXiv preprint arXiv:2106.03593* (2021).

[112] Jingyue Lu and M. Pawan Kumar. “Neural Network Branching for Neural Network Verification”. In: *International Conference on Learning Representations (ICLR)*. 2020.

[113] Aleksander Madry et al. “Towards Deep Learning Models Resistant to Adversarial Attacks”. In: *International Conference on Learning Representations (ICLR)*. 2018.

[114] Ashok Makkuva et al. “Optimal transport mapping via input convex neural networks”. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 6672–6681.

[115] Alejandro M. Manelli and Daniel R. Vincent. “Bundling as an optimal selling mechanism for a multiple-good monopolist”. In: *J. Econ. Theory* 127.1 (2006), pp. 1–35.

[116] Padala Manisha, CV Jawahar, and Sujit Gujar. “Learning optimal redistribution mechanisms through neural networks”. In: *International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*. 2018.

[117] André F. T. Martins and Ramón Fernández Astudillo. “From Softmax to Sparsemax: A Sparse Model of Attention and Multi-Label Classification”. In: *International Conference on Machine Learning (ICML)*. 2016.

[118] Eric Maskin and John Riley. “Optimal Multi-Unit Auctions”. In: *The Economics of Missing Markets, Information, and Games*. Ed. by Frank Hahn. Clarendon Press, Oxford, 1989. Chap. 14, pp. 312–335.

[119] Gonzalo Mena et al. "Learning latent permutations with gumbel-sinkhorn networks". In: *arXiv preprint arXiv:1802.08665* (2018).

[120] Jeremy B. Merrill. "Does Facebook Still Sell Discriminatory Ads? –". In: *The Markup* (Aug. 2020). URL: `https://themarkup.org/ask-the-markup/2020/08/25/does-facebook-still-sell-discriminatory-ads`.

[121] Paul Milgrom. *Discovering prices: auction design in markets with complex constraints*. Columbia University Press, 2017.

[122] Jamie Morgenstern and Tim Roughgarden. "Learning simple auctions". In: *Conference on Learning Theory*. 2016, pp. 1298–1318.

[123] Roger B Myerson. "Optimal auction design". In: *Mathematics of Operations Research* 6.1 (1981), pp. 58–73.

[124] Milad Nasr and Michael Carl Tschantz. "Bidding strategies with gender nondiscrimination constraints for online ad auctions". In: *Conference on Fairness, Accountability, and Transparency (FAT*)*. 2020, pp. 337–347.

[125] Congressional Budget Office. *The Budget and Economic Outlook: Fiscal Years 2001–2010*. Appendix B. 2000.

[126] Thomas R Palfrey. "Bundling decisions by a multiproduct monopolist with incomplete information". In: *Econometrica: Journal of the Econometric Society* (1983), pp. 463–483.

[127] Simon Parsons, Juan A Rodriguez-Aguilar, and Mark Klein. "Auctions and bidding: A guide for computer scientists". In: *ACM Computing Surveys (CSUR)* 43.2 (2011), pp. 1–59.

[128] Adam Paszke et al. "Pytorch: An imperative style, high-performance deep learning library". In: *Advances in neural information processing systems* 32 (2019).

[129] Gregory Pavlov. "Optimal mechanism for selling two goods". In: *The BE Journal of Theoretical Economics* 11.1 (2011).

[130] Neehar Peri et al. “PreferenceNet: Encoding Human Preferences in Auction Design with Deep Learning”. In: *Neural Information Processing Systems (NeurIPS)*. 2021.

[131] Gabriel Peyré, Marco Cuturi, et al. “Computational optimal transport: With applications to data science”. In: *Foundations and Trends® in Machine Learning* 11.5-6 (2019), pp. 355–607.

[132] Jad Rahme, Samy Jelassi, and S Matthew Weinberg. “Auction learning as a two-player game”. In: *International Conference on Learning Representations (ICLR)*. 2021.

[133] Jad Rahme et al. “A Permutation-Equivariant Neural Network Architecture For Auction Design”. In: *AAAI Conference on Artificial Intelligence*. 2021.

[134] Sai Srivatsa Ravindranath et al. “Deep Learning for Two-Sided Matching”. In: *arXiv preprint arXiv:2107.03427* (2021).

[135] Ricci. *Ricci v. DeStefano, 557 U.S. 557 (2009)*. 2009.

[136] Kevin Roberts. “The characterization of implementable choice rules”. In: *Aggregation and Revelation of Preferences* 12.2 (1979), pp. 321–348.

[137] Jean-Charles Rochet. “A necessary and sufficient condition for rationalizability in a quasi-linear context”. In: *Journal of mathematical Economics* 16.2 (1987), pp. 191–200.

[138] R Tyrrell Rockafellar. *Convex analysis*. Vol. 18. Princeton university press, 1970.

[139] Ralph Rockafellar. “Characterization of the subdifferentials of convex functions”. In: *Pacific Journal of Mathematics* 17.3 (1966), pp. 497–510.

[140] Alvin E Roth. “Marketplaces, markets, and market design”. In: *American Economic Review* 108.7 (2018), pp. 1609–58.

[141] Alvin E Roth. “The economics of matching: Stability and incentives”. In: *Mathematics of operations research* 7.4 (1982), pp. 617–628.

[142] Tim Roughgarden. “Algorithmic game theory”. In: *Communications of the ACM* 53.7 (2010), pp. 78–86.

[143] Debjani Saha et al. “Measuring Non-Expert Comprehension of Machine Learning Fairness Metrics”. In: *International Conference on Machine Learning (ICML)*. 2020.

[144] Michael Saks and Lan Yu. “Weak monotonicity suffices for truthfulness on convex domains”. In: *Proceedings of the 6th ACM conference on Electronic commerce*. 2005, pp. 286–293.

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

[146] Tuomas Sandholm. “Expressive commerce and its application to sourcing: How we conducted $35 billion of generalized combinatorial auctions”. In: *AI Magazine* 28.3 (2007), pp. 45–45.

[147] Tuomas Sandholm and Anton Likhodedov. “Automated design of revenue-maximizing combinatorial auctions”. In: *Operations Research* 63.5 (2015), pp. 1000–1025.

[148] Nripsuta Ani Saxena et al. “How do fairness definitions fare? Examining public attitudes towards algorithmic definitions of fairness”. In: *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*. 2019, pp. 99–106.

[149] Bernhard Schmitzer. “Stabilized sparse scaling algorithms for entropy regularized transport problems”. In: *SIAM Journal on Scientific Computing* 41.3 (2019), A1443–A1481.

[150] Andrew D Selbst et al. “Fairness and abstraction in sociotechnical systems”. In: *Conference on Fairness, Accountability, and Transparency (FAccT)*. 2019.

[151] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[152] Weiran Shen, Pingzhong Tang, and Song Zuo. “Automated mechanism design via neural networks”. In: *Proceedings of the 18th International Conference on Autonomous Agents and Multiagent Systems*. 2019, pp. 215–223.

[153] Weiran Shen et al. "Reinforcement Mechanism Design, with Applications to Dynamic Pricing in Sponsored Search Auctions". In: *CoRR* abs/1711.10279 (2017). arXiv: 1711.10279.

[154] Megha Srivastava, Hoda Heidari, and Andreas Krause. "Mathematical Notions vs. Human Perception of Fairness: A Descriptive Approach to Fairness for Machine Learning". In: (Feb. 2019). arXiv: 1902.04783 [cs.CY].

[155] Latanya Sweeney. "Discrimination in online ad delivery". In: *Communications of the ACM* 56.5 (2013), pp. 44–54. DOI: 10.1145/2447976.2447990.

[156] Andrea Tacchetti et al. "A neural architecture for designing truthful and efficient auctions". In: *arXiv preprint arXiv:1907.05181* (2019).

[157] Pingzhong Tang and Tuomas Sandholm. "Mixed-bundling auctions with reserve prices." In: *International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*. 2012.

[158] Yi Tay et al. "Sparse sinkhorn attention". In: *International Conference on Machine Learning (ICML)*. 2020.

[159] Vincent Tjeng, Kai Y. Xiao, and Russ Tedrake. "Evaluating Robustness of Neural Networks with Mixed Integer Programming". In: *International Conference on Learning Representations (ICLR)*. 2019.

[160] William Vickrey. "Counterspeculation, auctions, and competitive sealed tenders". In: *The Journal of finance* 16.1 (1961), pp. 8–37.

[161] Cédric Villani. *Topics in optimal transportation*. American Mathematical Soc., 2003.

[162] Shiqi Wang et al. "Enhancing Gradient-based Attacks with Symbolic Intervals". In: *CoRR* abs/1906.02282 (2019). arXiv: 1906.02282.

[163] Antoine Wehenkel and Gilles Louppe. "Unconstrained monotonic neural networks". In: *Advances in neural information processing systems* 32 (2019).

[164] Jakob Weissteiner and Sven Seuken. “Deep Learning—Powered Iterative Combinatorial Auctions”. In: *AAAI Conference on Artificial Intelligence*. 2020.

[165] Bryan Wilder, Bistra Dilkina, and Milind Tambe. “Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization”. In: *AAAI Conference on Artificial Intelligence*. 2019.

[166] Kai Y. Xiao et al. “Training for Faster Adversarial Robustness Verification via Inducing ReLU Stability”. In: *International Conference on Learning Representations (ICLR)*. 2019.

[167] Andrew Chi-Chih Yao. “Dominant-strategy versus bayesian multi-item auctions: Maximum revenue determination and comparison”. In: *Economics and Computation (EC)*. 2017.

[168] Hanrui Zhang and Vincent Conitzer. “Learning the Valuations of a *k*-demand Agent”. In: *International Conference on Machine Learning (ICML)*. 2020.

[169] Stephan Zheng et al. “The AI Economist: Improving Equality and Productivity with AI-Driven Tax Policies”. In: *arXiv preprint arXiv:2004.13332* (2020).