Deep Learning for Multi-Facility Location Mechanism Design

Noah Golowich[1], **Harikrishna Narasimhan**[1], **David C. Parkes**[1]
[1]Harvard University, SEAS, 33 Oxford Street, Cambridge, MA, 02138
ngolowich@college.harvard.edu, hnarasimhan@g.harvard.edu, parkes@eecs.harvard.edu

Abstract

Moulin [1980] characterizes the single-facility, deterministic strategy-proof mechanisms for social choice with single-peaked preferences as the set of generalized median rules. In contrast, we have only a limited understanding of multi-facility strategy-proof mechanisms, and recent work has shown negative worst-case results for social cost. Our goal is to design strategy-proof, multi-facility mechanisms that minimize expected social cost. We first give a PAC learnability result for the class of multi-facility generalized median rules, and utilize neural networks to learn mechanisms from this class. Even in the absence of characterization results, we develop a computational procedure for learning almost strategy-proof mechanisms that are as good as or better than benchmarks from the literature, such as the best percentile and dictatorial rules.

1 Introduction

In many multi-agent systems, the use of payments is either not possible or unethical, for example in deciding about a public good, or in organizational settings where norms preclude payments. Moreover, mechanism design without money is essentially impossible outside of structured domains: by the Gibbard-Sattherwaite theorem [Gibbard, 1973; Satterthwaite, 1975], any strategy-proof mechanism that chooses between two or more outcomes and for which agents can have all possible strict preferences must be dictatorial.

The most studied domain for which there are positive results is that of *single-peaked preferences*. Given a space $\Omega \subseteq \mathbb{R}^d$ of locations, consider the problem of locating one or more facilities at one of these locations. An agent i has single-peaked preferences if she has one location, $\tau_i \in \Omega$ (the *peak*), where she most prefers a facility to be located, and if her preference decreases as the location of the closest facility moves farther away from her peak. The facility location problem can occur in many real-world scenarios. For instance, if $\Omega \subseteq \mathbb{R}^2$, it could represent the problem of deciding where to place a public facility, such as a swimming pool, in a town.

We have a complete understanding of this problem when only one facility must be chosen. If the outcome space is 1-dimensional (e.g. $\Omega = [0, 1]$) it has long been known [Black, 1948] that choosing the median of the agents' peaks is the strategy-proof mechanism that minimizes the sum of the agents' distances to the outcome. Moulin [1980] developed a complete characterization of all strategy-proof social choice rules in the 1-dimensional, 1-facility setup, which was later generalized to multi-dimensional outcome spaces [Border and Jordan, 1983; Barberà *et al.*, 1993].

The design problem for the multi-facility setting is both less well understood, and has negative approximation results. For n agents and 2-facilities, Procaccia and Tenneholz [2013] give a lower-bound of $3/2 - O(1/n)$ and an upper-bound of $n-1$, for deterministic, strategy-proof mechanisms. Amongst follow-on work [Lu *et al.*, 2010; Wang and Lu, 2009; Alon *et al.*, 2010; Fotakis and Tzamos, 2014; Sui *et al.*, 2013], of particular note is a proof that tightens the result of Procaccia and Tenneholz to show that the best approximation ratio is exactly $n - 2$ [Fotakis and Tzamos, 2014]. Moreover, for $K > 2$ facilities and anonymous strategy-proof mechanisms, the approximation ratio becomes unbounded [Fotakis and Tzamos, 2014].

In light of these negative, worst-case approximation results, we adopt the goal of minimizing expected social cost, both unweighted and weighted. We assume that there is a joint distribution $\mathcal{D}$ over the agents' single-peaked utility functions $(u_1, \ldots, u_n)$. As with most works in the multi-facility literature, we take Ω to be a subset of the real line.

Our solution approach uses the framework of machine learning to find optimal mechanisms. While there have been previous works that have used machine learning for the design of mechanisms without money [Narasimhan *et al.*, 2016; Narasimhan and Parkes, 2016], none of them present a practical, flexible approach for designing general mechanisms.

Building on the recent success of using deep learning for the design of revenue-optimal auctions [Dütting *et al.*, 2017; Feng *et al.*, 2018], we model a mechanism as a feed-forward network, and optimize its parameters with the social cost as the loss function. The following are our contributions:

- We show that the class of multi-facility generalized median rules is PAC learnable under product distributions (Section 3). This result leverages the characterization result of Moulin [1980] as well as the recent framework of Cai and Daskalakis [2017] to take advantage of the independence of agents' reports.

- We develop a neural network, *MoulinNet*, for learning from the class of generalized median rules (Section 4). This is analogous to characterization-based approach of Dütting et al. However, unlike their networks, we need to deal with an exponential blow-up in parameters, requiring new tricks to make the training tractable. For a single facility problem, our network can represent to an arbitrary accuracy any strategy-proof mechanism. This is however not the case for $K > 1$ facilities.
- For $K > 1$, we provide a more general approach that does not rely on characterization results and learns mechanisms that yield very low ex-post regret (Section 5). We model a mechanism as a fully-connected network, the *RegretNet-nm*, and impose explicit constraints for strategy-proofness on the training problem. Our approach builds on the agnostic framework of Dütting et al. and is the first to successfully apply deep learning for the design of mechanisms without money.
- In our experiments (Section 6), RegretNet-nm learns mechanisms that are essentially strategy-proof, with social costs comparable to or lower than the best percentile and dictatorial rules. The RegretNet-nm mechanisms are often better than the ones learned by MoulinNet. The generality of RegretNet-nm is particularly beneficial when the design objective is the weighted social cost, or when the utility distribution is non-independent. These are settings where the less flexible mechanisms in the literature do not perform as well.

1.1 Related Work

Conitzer and Sandholm [2002] introduced the paradigm of *automated mechanism design* (AMD). In this work, design was achieved through integer linear programming techniques, with explicit, enumerative representation of functions and incentive constraints. This made their approach difficult to scale. Since then, several papers have adopted machine learning for economic design problems.

Procaccia et al. [2009] showed learnability of specific classes of voting rules, but without requiring strategy-proofness. Xia [2013] introduced specific axiomatic properties still without incentive considerations. The first paper to use learning in the context of design under incentive constraints is Dütting et al. [2015], who use support vector machines (SVMs) to learn payment rules that are maximally strategy-proof with respect to a given outcome function.

Our paper builds on the framework of Dütting et al. [2017], extending it for the design of mechanisms without money. In regard to facility location, Narasimhan et al. [2016] have previously adopted an SVM-based approach to automated design without money, but specialize to a specific family of rules, and to single facility only. We search over a richer space of mechanisms and handle multiple facilities.

Recent work has also studied the sample complexity of learning revenue-optimal auctions [Cai and Daskalakis, 2017; Morgenstern and Roughgarden, 2016], and assignment mechanisms with and without money [Narasimhan and Parkes, 2016]. To the best of our knowledge, we provide the first sample complexity results for the facility location problem.

2 Preliminaries

There are a set of agents $N = \{1, \dots, n\}$ and a set of locations Ω. In this paper, we work with $\Omega = [0, 1]$. We are interested in the problem of locating K facilities among locations in Ω. Each agent i has a preference over Ω for where a facility needs to be located, represented by a utility function $u_i : \Omega \to \mathbb{R}$ that associates a real value with each location. Let U_i be the set of permissible utility functions. Let $u = (u_1, \dots, u_n)$ denote a profile of utilities, and denote the set of all such tuples by $U = \prod_{i=1}^n U_i$. Further, let $u_{-i} = (u_1, \dots, u_{i-1}, u_{i+1}, \dots, u_n)$ denote all utility functions other than u_i, and $U_{-i} = \prod_{j \neq i}^n U_j$.

Each agent cares about the location of its closest facility. As such, an agent's utility function induces a utility for an outcome, $o = (o_1, \dots, o_K)$, which specifies a location for each facility. By a slight abuse of notation, we write $u_i(o) = \max_{k \in \{1,\dots,K\}} u_i(o_k)$. We require *single peaked* preferences.

Definition 1 (Single-peaked preferences). *For locations $\Omega \subseteq \mathbb{R}$, a function $u_i : \Omega \to \mathbb{R}_{\geq 0}$ is* **single-peaked** *if and only if there exists a unique point $a \in \Omega$ (the* **peak** *of u), denoted by $a = \tau(u_i)$, such that for all $x, y \in \Omega$, if either $y > x > a$ or $y < x < a$, then $u_i(y) < u_i(x) < u_i(a)$.*

As one moves away from the peak $\tau(u_i)$ of utility function u_i, the location becomes less preferred. Following from Procaccia and Tennenholtz [2013] and subsequent work, we assume $u(x) = -|x - a|$, where $a = \tau(u)$ is the peak.

A *mechanism* for facility location $f : U \to \Omega^K$ takes reports of agent utilities as inputs, and outputs the locations in Ω for the K facilities. We use $f_k(u)$ to denote the location of the k-th facility for input u. A mechanism is *strategy-proof* if agent i cannot strictly increase her utility by misreporting her utility, whatever the inputs from others, i.e. for all $i \in N, u \in U, u'_i \in U_i$, we have $u_i(f(u'_i, u_{-i})) \leq u_i(f(u))$.

We denote by $\mathcal{M}_{sp} \subseteq \mathcal{M}$ the space of all strategy-proof mechanisms for the facility location problem (leaving K implicit). We assume that the agent utilities are sampled from a joint distribution $\mathcal{D}$ over U, with full support on U. We will sometimes assume that $\mathcal{D}$ is a product distribution, i.e. $\mathcal{D} = \prod_{i=1}^n \mathcal{D}_i$ for independent distributions $\mathcal{D}_i$ on U_i.

The design objective is to find the strategy-proof mechanism $f \in \mathcal{M}_{sp}$ that minimizes expected social cost (or a weighted variant): $g_{sc}(f; \mathcal{D}) = -\mathbb{E}_{u \sim \mathcal{D}}\left[\frac{1}{n}\sum_{i=1}^n u_i(f(u))\right]$. We do not have direct access to $\mathcal{D}$, but are provided a sample of profiles $\mathcal{S} = \{u^{(1)}, \dots, u^{(R)}\}$ drawn i.i.d. from $\mathcal{D}$.

3 Learnability of Generalized Median Rules

As a first step, we ask if known classes of strategy-proof mechanisms in the literature are learnable. Moulin provided a characterization of unanimous strategy-proof mechanisms for a single facility, later generalized by Border and Jordan:

Theorem 1 (1-facility generalized median rules [Moulin, 1980; Border and Jordan, 1983]). *A unanimous mechanism $f : U \to \Omega$ is strategy-proof if and only if it is a* generalized median rule, *i.e. for each $S \subseteq \{1, \dots, n\}$, there exists some*

$a_S \in \Omega$ *s.t. for all* $(u_1, \ldots, u_n) \in U$,

$$f(u) = \min_{S \subseteq \{1,\ldots,n\}} \max \left\{ \max_{i \in S} \{\tau(u_i)\}, a_S \right\}. \quad (1)$$

There is no complete characterization of strategy-proof mechanisms for multi-facility location.[1] Still, we can leverage Moulin's result and consider a rich (if incomplete) family, by combining single-facility mechanisms in the natural way.

Definition 2 (Multi-facility generalized median rules)**.** *Let $\mathcal{M}_{GM}$ denote the class of* multi-facility generalized median rules *given by $f = (f_1, \ldots, f_K)$, where each f_k is a 1-facility generalized median rule for parameters $a_S^k \in \Omega$, $S \subseteq N$.*

The set of multi-facility generalized median rules is a strict superset of the set of multi-facility percentile rules considered in [Sui *et al.*, 2013]; all multi-facility percentile rules are anonymous, but this is not true of all rules in $\mathcal{M}_{\text{GM}}$. It is immediate that $\mathcal{M}_{\text{GM}}$ is strategy-proof for all $K \geq 1$. To see this, suppose an agent i can improve her cost by misreporting her utility. Let $f_k(u)$ and $f_{k'}((u_i', u_{-i}))$ be the closest facility to the agent under a truthful report and the misreport. Clearly, $f_{k'}((u_i', u_{-i}))$ also yields lower cost to the agent than $f_{k'}(u)$, which is a contradiction to the strategy-proofness of $f_{k'}$.

Definition 3 (PAC learnability)**.** *A class of mechanisms $\mathcal{M}$ on U is* probably approximately correct (PAC) learnable over product distributions *if there exists an algorithm $\mathcal{A}$, such that for any product distribution $\mathcal{D}$ over U, any $\epsilon > 0$, and any $\delta \in (0,1)$, running $\mathcal{A}$ on an i.i.d. sample from $\mathcal{D}$ of size* $\text{poly}(1/\epsilon, 1/\delta)$ *gives a function $\widehat{f} \in \mathcal{F}$ s.t. $g_{sc}(\widehat{f}; \mathcal{D}) \leq \inf_{f \in \mathcal{M}} g_{sc}(f; \mathcal{D}) + \epsilon$, with probability $1 - \delta$ over the sample.*

Despite there being an exponential number of parameters a_S^k in defining a mechanism in class $\mathcal{M}_{\text{GM}}$, we have:

Theorem 2. *$\mathcal{M}_{GM}$ is PAC learnable over prod. distributions with $O\left(\frac{n^2(K^2 \log^2 K + n^2)}{\epsilon^2} \cdot \log\left(\frac{n^3(K \log K + n)}{\epsilon\delta}\right)\right)$ samples.*

Proof Sketch. Consider the class of social cost functions $\{u \mapsto \sum_{i=1}^n \min_{1 \leq k \leq K} |\tau(u_i) - f_k(u)| : f \in \mathcal{M}\}$ from mechanisms in $\mathcal{M}$. The proof involves establishing a uniform convergence sample complexity bound for this class. Specifically, we use the recent framework of Cai and Daskalakis [2017] to exploit the product structure of $\mathcal{D}$ and break down the overall analysis into an analysis of the following projected functions from the class for each $i, j \in [n]$: $\mathcal{H}_{i,j} = \{u_j \mapsto \min_{1 \leq k \leq K} |\tau(u_i) - f_k(u_j, u_{-j})| : f \in \mathcal{M}, u_{-i} \in U_{-i}\}$

Each function in $\mathcal{H}_{i,j}$ captures the effect agent j's reports have on agent i's social cost, fixing the reports of agents other than j. It can be shown that the functions in $\mathcal{H}_{i,j}$ are piece-wise linear with $O(K)$ pieces if $i = j$ and O(1) pieces if $i \neq j$, and correspondingly have a pseudo-dimension[2] $O(K \log K)$ if $i = j$ and $O(1)$ if $i \neq j$. Standard results [Anthony and Bartlett, 2009] allow us to derive a uniform convergence bound for each $\mathcal{H}_{i,j}$. An application of the result of Cai and Daskalakis then completes the proof. □

[1]Ehlers and Gordon [2011] and Heo [2013] provide characterizations for the special case $K = 2$ under different assumptions.

[2]The pseudo dimension of a class $\mathcal{F}$ of functions $f : X \to \mathbb{R}$ is the VC-dimension of the class $\{(x, y) \mapsto \text{sign}(f(x) - y) : f \in \mathcal{F}\}$.

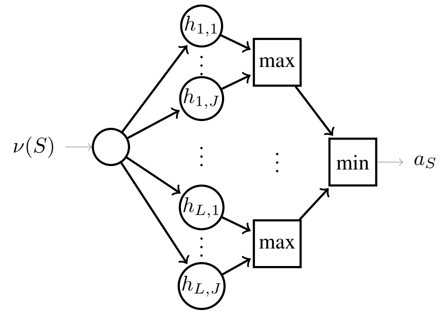


Figure 1: Modeling a_S using a monotone network $h^{\mathbf{w},b}$. The input $\nu(S)$ is a boolean encoding of a set $S \subseteq N$.

4 MoulinNet for Generalized Median Rules

Equipped with the above learnability result, we now develop a computational procedure for learning from the class of generalized median rules, by modeling these rules using neural networks. We start with the case of a single facility.

Recall that a generalized median rule depends only on the agent peaks $\tau(u_i)$. For the single facility location on $[0,1]$ it is w.l.o.g. to only consider mechanisms that operate on agent peaks [Border and Jordan, 1983]. It is also w.l.o.g. to assume that the parameters a_S are monotone, i.e. $a_S \geq a_T$, $\forall S \subset T$.

One computational approach would search direclty over the space of parameters $\{a_S\}_{S \subseteq N}$. But this does not scale because there are exponentially many parameters. Instead we parametrize each a_S using a monotone feed-forward neural network. For this, we use a neural network $h : \{-1,1\}^n \to \mathbb{R}$ that maps a n-dimensional binary representation of set S to a real value. Any $S \subseteq N$ can be represented as a binary vector $x \in \{-1,1\}^n$, with $x_i = 1$ if and only if $i \in S$. We use $\nu(S)$ to denote the binary encoding of S, and define $a_S = h(\nu(S))$. With this, the parameters to optimize are those that parametrize neural network h.

We say h is monotonically decreasing if $\forall x, x' \in \{-1,1\}^n$, $i \in [n]$, $(x_i \geq x_i') \wedge (x_{-i} = x_{-i}') \implies h(x) \leq h(x')$. If h is monotone then the resulting $\{a_S\}_{S \subseteq N}$ are also monotone. We appeal to a characterization of Sill [1998] to construct monotone neural networks. Sill shows that any bounded, monotone function $h : \{-1,1\}^n \to \mathbb{R}$ can be approximated to arbitrary precision using the following parameterization, for suitable choice of positive integers L, J, parameters $w_{lj} \in \mathbb{R}_-^n$, $b_{lj} \in \mathbb{R}$, $l \in [L], j \in [J]$:

$$a_S = h^{\mathbf{w},\mathbf{b}}(\nu(S)) = \min_{l \in [l]} \max_{j \in [J]} \{\langle w_{lj}, \nu(S) \rangle + b_{lj}\}. \quad (2)$$

Figure 1 contains a representation of this function as a feed-forward network, taking $\nu(S)$ as input and outputting a_S.

Thus for any choice of $w_{lj} \in \mathbb{R}_-^n$, $b_{lj} \in \mathbb{R}$, $j \in [J]$, $l \in [L]$, $L, J \in \mathbb{N}$, the following single-facility mechanism is strategy-proof:

$$f^{\mathbf{w},\mathbf{b}}(u) = \min_{S \subseteq N} \left\{ \max_{i \in S} \left\{ \tau(u_i), \, h^{\mathbf{w},\mathbf{b}}(\nu(S)) \right\} \right\}. \quad (3)$$

By choosing parameters L and J to tune the complexity of the function class, (3) gives us a way to parametrize all 1-facility generalized median rules.

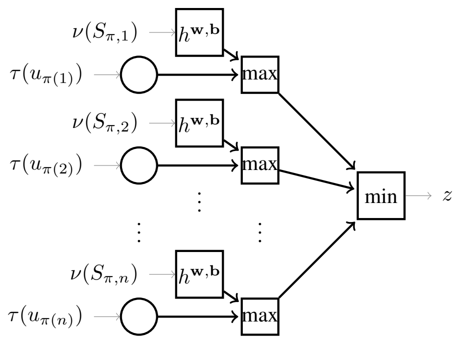


Figure 2: MoulinNet for 1 facility. $h^{\mathbf{w},\mathbf{b}}$ is shown in Fig 1. We precompute π and $S_{\pi,i}$ for each example during training.

The method in (3) still involves a minimum over $|S| = 2^n$ functions of w_{lj}, b_{lj}. We can compute (3) efficiently by exploiting the fact that the a_S values are monotone. In particular, let π denote a permutation of the reported peaks in sorted order, i.e. $\tau(u_{\pi(1)}) \leq \cdots \leq \cdots \leq \tau(u_{\pi(n)})$. For $1 \leq i \leq n$, we let $S_{\pi,i} = \{\pi(1), \ldots, \pi(i)\}$. We can show that:

$$f^{\mathbf{w},\mathbf{b}}(u) = \min_{1\leq i\leq n} \left[\max[h^{\mathbf{w},\mathbf{b}}(\nu(S_{\pi,i})), \tau(u_{\pi(i)})]\right].$$

Theorem 3. *(3) can be computed in time $O(z \cdot n + n\log n)$, where z is the time taken for a single invocation of $h^{\mathbf{w},\mathbf{b}}$.*

The proof is omitted due to lack of space. The resulting mechanism for a single facility is given by a feed-forward neural network that takes peaks $\tau(u_1), \ldots, \tau(u_n)$ as inputs and outputs the location of the facility. This architecture, referred to as *MoulinNet*, is shown in Figure 2. When there are $K \geq 2$ facilities, we use K replicas of the network, with K parameter sets, each deciding the location of a facility. We know that this K-facility generalized median rule is strategy-proof. We optimize the network parameters with the social cost as the loss function, which is estimated from sample S.

5 RegretNet-nm for General Mechanisms

We now develop a general approach that is not limited by existing characterization results, and which learns mechanisms that have low ex-post regret. For this, we extend the agnostic framework of Dütting et al. [2017]. We model a mechanism as a feed-forward neural network, which may not be strategy-proof for all parameter assignments. We optimize the network parameters to minimize a given cost objective, while seeking to achieve strategy-proofness. The search space can be controlled by varying the depth and width of the neural network.

To measure the deviation of a mechanism f from strategy-proof, we adapt the notion of *regret* to facility location mechanisms [Dütting *et al.*, 2015; 2017]. We define the *expected ex post regret* from a mechanism f to an agent i as the (expected) maximum gain in utility that the agent can get by misreporting her preferences:

$$\text{rgt}_i(f) = \mathbb{E}_{u\sim D}\left[\max_{u_i'\in U_i} u_i(f(u_i', u_{-i})) - u_i(f(u_i, u_{-i}))\right].$$

The network architecture for a mechanism is described in Figure 3. The network is fully-connected, has L hidden layers, takes n agent peaks $\tau(u_1), \ldots, \tau(u_n)$ as input and outputs the locations of the K facilities $z_1, \ldots, z_K$. Specifically,

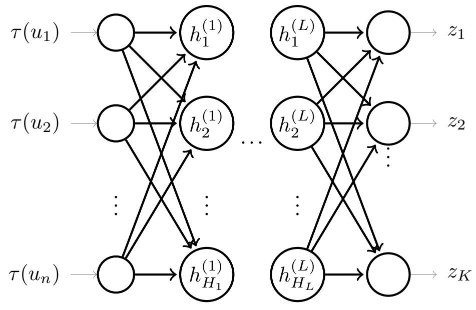


Figure 3: RegretNet-nm Architecture

$$f^{\mathbf{w}}(u) := \mathbf{w}_L\bar{\sigma}(\mathbf{w}_{L-1}\cdots\mathbf{w}_2(\bar{\sigma}(\mathbf{w}_1\cdot\tau(u)))\cdots), \tag{4}$$

where $\tau(u) = (\tau(u_1), \ldots, \tau(u_n))$, $\mathbf{w}_i \in \mathbb{R}^{H_i\times H_{i-1}}$, with $H_0 = n, H_{L+1} = K$, are the parameters of the network, $H_1, \ldots, H_L$ are the number of units in each hidden layer of the network, and $\sigma : \mathbb{R} \to \mathbb{R}$ is a nonlinear activation function, and $\bar{\sigma}(A)$ applies σ to every entry of a real matrix A. We use the ReLU activation function $\sigma(x) = \max\{0, x\}$.

Let $\mathbf{w} \in \mathbb{R}^d$ denote all the network parameters. The loss function $\mathcal{L}(\mathbf{w})$ is the expected social cost or a weighted variant. Our goal is to minimize the loss subject to the expected *ex post* regret being zero for all agents. Since $\mathcal{D}$ has full support on U, this implies that the network is strategy-proof (except for events with zero measure):

$$\min_{\mathbf{w}\in\mathbb{R}^d} \mathcal{L}(\mathbf{w}) \text{ s.t. } \text{rgt}_i(f^{\mathbf{w}}) = 0, \forall i \in N.$$

Following Dütting et al. [2017], one way to estimate the regret from a sample $u^{(1)}, \ldots, u^{(R)}$ is to search over a subset of misreport peaks in $[0,1]$ for agent i, say at a certain granularity $1/M$; we denote the average of this quantity over the sample by $\hat{\text{rgt}}_i(f^{\mathbf{w}})$. While training, searching over M misreports can be expensive, so we adopt a new, pairwise estimate for training that makes better use of each utility calculation:

$$\begin{aligned}\hat{\text{prgt}}_i(f) &= \frac{1}{R}\sum_{j=1}^{R} \max_{\substack{\tau(u_i'),\tau(u_i'')\in\\ \{\beta_1,\ldots,\beta_{M'}\}}} (\star) \\ (\star) &= u_i'(f(u_i'', u_{-i}^{(j)})) - u_i'(f(u_i', u_{-i}^{(j)})),\end{aligned} \tag{5}$$

where $\beta_1, \ldots, \beta_{M'} \sim [0,1]$ are sampled independently and uniformly. For each example j, (5) computes the maximum utility agent i could gain if her true peak was one of $\beta_1, \ldots, \beta_{M'}$ and she chose to misreport it as one of $\beta_1, \ldots, \beta_{M'}$. To include T misreports per agent, the standard definition requires us to perform $R(T+1)$ utility evaluations, whereas (5) requires at most $R\sqrt{T+1}$ utility evaluations.

During training, we adopt the *augmented Lagrangian* solver in Dütting et al. [2017]. We minimize the empirical loss $\hat{\mathcal{L}}(\mathbf{w}) = -\frac{1}{Rn}\sum_{j=1}^{R}\sum_{i=1}^{n} u_i^{(j)}(f^{\mathbf{w}}(u^{(j)}))$, while seeking to satisfy the constraints $\hat{\text{prgt}}_i(f) = 0$ for each i. The following is the Lagrangian for the above optimization problem, augmented with a quadratic penalty term:

$$\hat{\mathcal{L}}(\mathbf{w}) + \lambda^t \cdot \max_{i\in N} \hat{\text{prgt}}_i(f^{\mathbf{w}}) + \rho\Big(\max_{i\in N}\hat{\text{prgt}}_i(f^{\mathbf{w}})\Big)^2, \tag{6}$$

K	Perc.	Dict.	Cons.	MoulinNet	RegretNet-nm		NonSP
					sc	$\max_i \hat{\text{rgt}}_i$	
1	**0.200**	0.267	0.253	**0.201**	**0.201**	0.0003	0.200
2	**0.0833**	0.126	0.126	**0.0837**	**0.0833**	0.0003	0.0708
3	**0.0335**	0.0609	0.0834	0.0353	0.0376	0.0009	0.0278
4	**0.0171**	0.0236	0.0635	0.0188	**0.0177**	0.0024	0.0083

Table 1: $n = 5$, unweighted social cost. Compares the benchmark mechanisms (percentile, dictator and constant), MoulinNet, RegretNet, and the socially optimal, non-strategyproof mechanism. The best results (to within 0.001) are in **bold**. For RegretNet, we also provide the expected *ex post* regret.

where $\rho > 0$ controls the weight on the quadratic term.

The solver alternates between updates on the parameter λ^t: $\lambda^{t+1} = \lambda^t + \rho \cdot \max_{i \in N} \hat{\text{prgt}}_i(f^{\mathbf{w}})$, and using stochastic gradient descent (SGD) to optimize the network parameters $\mathbf{w}$ to minimize the Lagrangian. The SGD involves multiple updates on $\mathbf{w}$, each computing the gradient of (6) w.r.t. $\mathbf{w}$.

Since the optimization problem (5) is non-convex, and the solver works with estimates of the loss and regret from a sample, the learned mechanism may have a small, residual regret.[3] In our experiments, we find that the mechanisms learned by RegretNet-nm have a non-zero regret on the test sample, but the regret is often negligibly small, indicating that the mechanisms are essentially strategy-proof.

6 Experimental Results

We describe the results of our experiments on learning mechanisms for multi-facility location problems using MoulinNet and RegretNet-nm. MoulinNet serves as as a benchmark for the best that can be achieved with characterization results.

We implement RegretNet-nm with $L = 4$ hidden layers, each with 40 units. In the augmented Lagrangian solver, we performed 1000 updates on λ and 50 gradient updates on $\mathbf{w}$ for every one update on λ. We implement MoulinNet with $L = K = 3$. In both cases we make use of the Tensorflow library. We use the Adam algorithm for optimizing $\mathbf{w}$ (with mini-batches of size 500).[4] We repeat training with 5 random initializations, and choose the network from the run with smallest regret (based only on the training data).

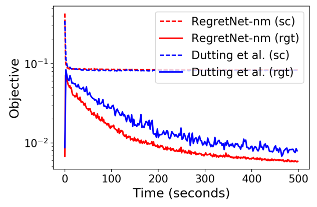
We train RegretNet with $M' = 5$ by generating new instances and misreports for each mini-batch. We train MoulinNet with multiple passes through a single batch of 1000 examples. Both networks take less than half hour to train on a Tesla K20Xm GPU. Both networks are evaluated based on their social cost on a held-out test set of 2000 examples. For RegretNet, we also report its empirical regret: $\max_i \hat{\text{rgt}}_i$, where $\hat{\text{rgt}}_i$ is computed using 50 misreports from $[0, 1]$.

We compare our approach against standard mechanisms from the literature: the best percentile rule [Sui *et al.*, 2013], the best dictatorial rule, and the best constant rule, all found by a brute-force search, which is quick for these rules.[5]

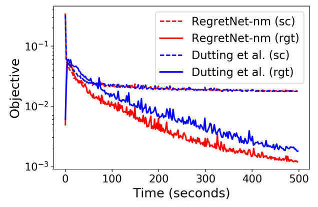
[3]See Nocedal and Wright [1999] for convergence properties of the augmented Lagrangian method on non-convex problems.

[4]The learning rate in Adam was initialized to 0.005 for RegretNet-nm (and decayed by a factor 0.99 every 100 updates) and to 0.1 in MoulinNet. Weights were initialized as i.i.d. draws from $\mathcal{N}(0, 0.01)$. The offsets in RegretNet-nm were initialized to 0.1.

[5]A *percentile rule* locates each facility at a fixed percentile of the reported peaks. A *dictatorial rule* locates each facility at the peak of a fixed agent. A *constant rule* locates each facility at a fixed point.

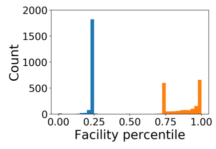


(a) $K = 2$

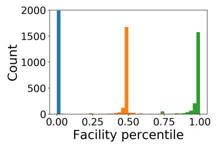


(b) $K = 4$

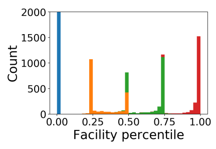
Figure 4: Social cost and regret ($\hat{\text{rgt}}$) on test set as a function of run-time. We compare RegretNet-nm (red), which uses $\hat{\text{prgt}}$ in the learning formulation (6), with the approach of Dütting et al. [2017] (blue) that uses $\hat{\text{rgt}}$ instead.



(a) $K = 2$

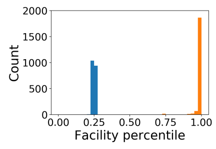


(b) $K = 3$

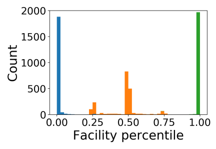


(c) $K = 4$

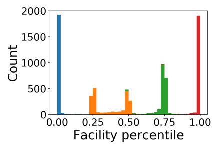
MoulinNet



(d) $K = 2$



(e) $K = 3$



(f) $K = 4$

RegretNet-nm

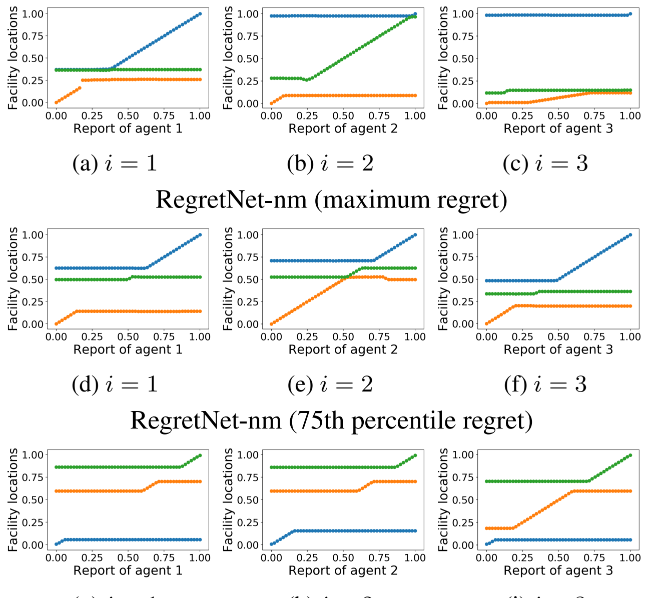
Figure 5: $n = 5$, unweighted social cost. Histograms of facility percentiles chosen by RegretNet and MoulinNet mechanisms. For each training instance, the 4 facility percentiles chosen by the network are sorted and shown in different colors. The blue bar denotes the smallest of the 4 facilities, orange the second smallest, and so on.

Unweighted Social Cost

We begin with social cost as the design objective. We consider $n = 5$ agents, whose peaks are distributed i.i.d. uniformly on $[0, 1]$. We focus primarily on $K = 2, 3$ and 4 facilities; for $K = 1$ both MoulinNet and RegretNet-nm nearly always place the facility very close to the median of the agents' peaks.

Table 1 shows the social cost on the test set for the mechanisms learned by RegretNet-nm, MoulinNet and the other baselines. We also report the social cost for the optimal (non-strategy-proof) mechanism. Both kinds of networks yield similar performance as the best percentile rule. The RegretNet-nm mechanisms have negligible regret, indicating that they are essentially strategy-proof.

Figure 4 plots the average social cost and empirical regret for RegretNet-nm on the test set as a function of running time. With increasing number of solver updates, the regret can be seen to approach zero. The plot also includes the approach of [Dütting *et al.*, 2017], which uses the standard regret definition $\hat{\text{rgt}}$ in the Lagrangian formulation (6). This method was trained with 20 misreports per agent, which is effectively the number of misreports per agent RegretNet-nm obtains with



(a) $i = 1$ (b) $i = 2$ (c) $i = 3$

RegretNet-nm (maximum regret)

(d) $i = 1$ (e) $i = 2$ (f) $i = 3$

RegretNet-nm (75th percentile regret)

(g) $i = 1$ (h) $i = 2$ (i) $i = 3$

MoulinNet

Figure 6: $n = 5, K = 3$, unweighted social cost. Plots of mechanism outputs as a function of report of a single agent $i \in \{1, 2, 3\}$, keeping the reports of other agents fixed. The location of each facility is shown in a different color. (a)–(c): Plots for RegretNet-nm on instances where $\hat{\text{rgt}}_i$, $i = 1, 2, 3$ are respectively maximum (in a sample of size 40). (d)–(f): Plots for RegretNet-nm on instances where $\hat{\text{rgt}}_i$, $i = 1, 2, 3$ are in the 75-th percentile of the regret values. (g)–(i): Plots for MoulinNet (strategy-proof) on a random instance.

$M' = 5$, Clearly, by using the pairwise regret $\hat{\text{rgt}}$ in (6), RegretNet-nm achieves a notable speed-up.

We next take a closer look at the learned mechanism. Figure 5 shows the histograms of the percentiles of the facilities chosen by the RegretNet-nm and MoulinNet mechanisms for the test set examples for $K = 2, 3, 4$.[6] The peaks on percentiles $0, 1/(K-1), 2/(K-1), \ldots, 1$ indicate that the networks use the percentiles of agent reports to make decisions. For $K = 2, 3$, the mechanisms learned by RegretNet-nm approximate the best percentile rules. For $K = 4$, the percentiles at which RegretNet-nm and MoulinNet place facilities are not the same for all instances.

To understand the inputs on which the RegretNet-nm mechanisms incur a small regret, we visualize the misreports that are useful for agents, for $K = 3$. Figure 6 plots the locations of the 3 facilities as a function of the report of a single agent i, holding the reports u_{-i} of all other agents fixed.

For fixed u_{-i}, let $R_{u_{-i}}$ denote the set of all locations where a facility is placed by a mechanism f for some report u_i of agent i. Then f is strategy-proof iff for every u_i, $f(u_i, u_{-i})$ places one facility at the location in $R_{u_{-i}}$ closest to $\tau(u_i)$.

In other words, strategy-proof rules have facilities along the line $y = x$, and $y = \textit{const}$ elsewhere. The plots in row 1 (inputs with max. regret in RegretNet) do not satisfy this property. Suppose in (c), agent 3's true peak is at 0.06. Then there is a value in $[0.4, 0.6]$ that she can deviate to and have one of the facilities (orange) placed at her peak. In row 3 (strategy-proof MoulinNet), all the plots satisfy this property.

[6]If $p_1, \ldots, p_n$ are the agent peaks in sorted order, then a facility at location x has percentile 0 if $x \le p_1$, has percentile 1 if $x \ge p_n$, and has percentile $\frac{i-1}{n-1} + \frac{x-p_i}{(n-1)(p_{i+1}-p_i)}$ if $p_i \le x < p_{i+1}$.

Distr.	Perc.	Dict.	Cons.	MoulinNet	RegretNet-nm		NonSP
					sc	$\max_i \hat{\text{rgt}}_i$	
Unif	0.056	0.053	0.085	0.043	**0.041**	0.0005	0.032
$\mathcal{D}^1_{dep}$	0.0216	0.0351	0.0937	0.0232	**0.0204**	0.0003	0.0124
$\mathcal{D}^2_{dep}$	0.0333	**0.0173**	0.0943	0.0194	**0.0174**	0.0001	0.0157

Table 2: Row 1: uniform distribution, $n = 9, K = 3$, weighted social cost. Rows 2 and 3: non-independent distribution, $n = 5, K = 3$, unweighted social cost.

Weighted Social Cost

In order to demonstrate the flexibility of our framework, we also consider weighted social cost as the design objective: $-\mathbb{E}_{u\sim\mathcal{D}}\left[\frac{1}{\sum_{i=1}^n \gamma_i} \cdot \sum_{i=1}^n \gamma_i u_i(f(u))\right]$, where γ_i is a weight on agent i and may correspond to the importance of agent i to society. In this setup, there are $n = 9$ agents, and all valuations are i.i.d. uniform on $[0, 1]$. Agents 1 and 2 are assigned a weight of 5, while the remaining agents are assigned a weight of 1. As shown in Table 2 (first row), RegretNet-nm and MoulinNet yield significantly smaller social cost than the baseline mechanisms.

Non-Independent Valuations

Finally, we show that our approach can be used to find optimal mechanisms for non-product distributions. Let $n = 4, K = 3$. We consider two distributions $\mathcal{D}^1_{dep}$ and $\mathcal{D}^2_{dep}$, where it is not immediately clear what the optimal strategy-proof mechanism is. In both, the peak X_1 for agent 1 is drawn from $U([0, 1])$, and the peak for each agent $i = 2, 3, 4$ is set to one of the following piece-wise linear functions: $X_1 + \sigma_i$, $f_1(X_1) + \sigma_i$, $f_2(X_1) + \sigma_i$, where $\sigma_i \sim \mathcal{N}(0, 0.01)$.[7] In $\mathcal{D}^1_{dep}$, the agent chooses one of the 3 functions with equal probability, independent of the others. In $\mathcal{D}^2_{dep}$, the selection is made uniformly over all possibilities where each function is chosen by exactly one agent. The results are shown in Table 2 (rows 2–3). The flexibility of RegretNet-nm allows it to outperform all the benchmarks. For $\mathcal{D}^2_{dep}$, the dictatorial rule that places the facilities at the peaks of agents $2, 3, 4$ performs well since each of the functions $X_1, f_1(X_1), f_2(X_1)$ is chosen by exactly one agent.

7 Conclusion

We have shown that neural networks can be successfully applied to design near-optimal, low ex-post regret mechanisms for the multi-facility location problem. This opens the door for using deep learning to design mechanisms for other settings without money such as matching and allocation problems. In the future, it would be interesting to consider cost functions that are general functions of distance to nearest facility, to study generalization properties of the *ex post* regret,

[7]$f_1(x) = 2/3 \cdot \mathbf{1}[0 \le x < 1/3] + 1/4 \cdot \mathbf{1}[1/3 \le x < 2/3]$, and $f_2(x) = 1/3 + 2/3 \cdot \mathbf{1}[0 \le x < 1/3] + 5/12 \cdot \mathbf{1}[1/3 \le x < 2/3]$.

to develop networks that are invariant to the number of agents, and to allow for randomized mechanisms.

Acknowledgments

We thank Dimitris Fotakis for a useful discussion, Zhe Feng for helpful suggestions, and the anonymous reviewers for their comments. The first author is grateful for funding from the Harvard College Research Program.

References

[Alon *et al.*, 2010] Noga Alon, Michal Feldman, Ariel D. Procaccia, and Moshe Tennenholtz. Strategyproof approximation of the minimax on networks. *Mathematics of Operations Research*, 35(3):513–526, 2010.

[Anthony and Bartlett, 2009] Martin Anthony and Peter L. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, 2009.

[Barberà *et al.*, 1993] Salvador Barberà, F Gul, and E Stacchetti. Generalized median voter schemes and committees. *Journal of Economic Theory*, 61:262–289, 1993.

[Black, 1948] Duncan Black. On the Rationale of Group Decision-making. *Journal of Political Economy*, 56(1):23–34, 1948.

[Border and Jordan, 1983] Kim C. Border and J. S. Jordan. Straightforward Elections, Unanimity and Phantom Voters. *Review of Economic Studies*, 50(1):153, 1983.

[Cai and Daskalakis, 2017] Yang Cai and Constantinos Daskalakis. Learning multi-item auctions with (or without) samples. In *Proceedings of the IEEE 58th Annual Symposium on Foundations of Computer Science*, 2017.

[Conitzer and Sandholm, 2002] Vincent Conitzer and Tuomas Sandholm. Complexity of mechanism design. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, 2002.

[Dütting *et al.*, 2015] Paul Dütting, Felix Fischer, Pichayut Jirapinyo, John K. Lai, Benjamin Lubin, and David C. Parkes. Payment Rules through Discriminant-Based Classifiers. *ACM Transactions on Economics and Computation*, 3(1):1–41, 2015.

[Dütting *et al.*, 2017] Paul Dütting, Zhe Feng, Harikrishna Narasimhan, and David C. Parkes. Optimal Auctions through Deep Learning. *ArXiv:1706.03459 [cs]*, 2017.

[Ehlers and Gordon, 2011] Lars Ehlers and Sidartha Gordon. Strategy-proof provision of two public goods: the LexMax extension. Technical report, Mimeo Université de Montréal, 2011.

[Feng *et al.*, 2018] Zhe Feng, Harikrishna Narasimhan, and David C. Parkes. Deep Learning for Revenue-Optimal Auctions with Budgets. In *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems*, 2018. To appear.

[Fotakis and Tzamos, 2014] Dimitris Fotakis and Christos Tzamos. On the power of deterministic mechanisms for facility location games. *ACM Transactions on Economics and Computation*, 2(4):15, 2014.

[Gibbard, 1973] Allan Gibbard. Manipulation of voting schemes: a general result. *Econometrica*, 41(4):587–601, 1973.

[Heo, 2013] Eun Jeong Heo. Strategy-proof rules for two public goods: double median rules. *Social Choice and Welfare*, 41(4):895–922, 2013.

[Lu *et al.*, 2010] Pinyan Lu, Xiaorui Sun, Yajun Wang, and Zeyuan Allen Zhu. Asymptotically Optimal Strategy-proof Mechanisms for Two-facility Games. In *Proceedings of the 11th ACM Conference on Electronic Commerce*. ACM, 2010.

[Morgenstern and Roughgarden, 2016] Jamie Morgenstern and Tim Roughgarden. Learning Simple Auctions. In *Proceedings of the Conference on Learning Theory*, 2016.

[Moulin, 1980] H Moulin. On strategy-proofness and single-peakedness. *Public Choice*, 35(4):55–74, 1980.

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

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

[Nocedal and Wright, 1999] Jorge Nocedal and Stephen Wright. Numerical optimization. *Springer*, 1999.

[Procaccia and Tennenholtz, 2013] Ariel D. Procaccia and Moshe Tennenholtz. Approximate Mechanism Design without Money. *ACM Transactions on Economics and Computation*, 1(4):1–26, 2013.

[Procaccia *et al.*, 2009] A.D. Procaccia, A.Zohar, Y. Peleg, and J.S. Rosenschein. The learnability of voting rules. *Artificial Intelligence*, 173:1133–1149, 2009.

[Satterthwaite, 1975] Mark Allen Satterthwaite. Strategy-proofness and Arrow's conditions: Existence and correspondence theorems for voting procedures and social welfare functions. *Journal of economic theory*, 10(2):187–217, 1975.

[Sill, 1998] Joseph Sill. Monotonic networks. In *Advances in Neural Information Processing Systems*, 1998.

[Sui *et al.*, 2013] Xin Sui, Craig Boutilier, and Tuomas W. Sandholm. Analysis and optimization of multi-dimensional percentile mechanisms. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence*, 2013.

[Wang and Lu, 2009] Yajun Wang and Pinyan Lu. Tighter Bounds for Facility Games. In *Proceedings of the 5th International Workshop on Internet and Network Economics*, 2009.

[Xia, 2013] Lirong Xia. Designing social choice mechanisms using machine learning. In *Proceedings of the international conference on Autonomous agents and multi-agent systems*, 2013.