Integrating Machine Learning and Optimization to Boost Decision Making

Ferdinando Fioretto
Syracuse University
ffiorett@syr.edu

Abstract

This paper presents a conceptual review of our recent advancements on the integration of machine learning and optimization. It focuses on describing new hybrid machine learning and optimization methods to predict fast, approximate, solutions to combinatorial problems and to enable structural logical inference.

1 Introduction

Constrained optimization (CO) is in daily use in our society, with applications ranging from supply chains and logistics to electricity grids, organ exchanges, marketing campaigns, and manufacturing to name only a few. Although these problems are often NP-hard and computationally challenging even for medium-sized instances, they constitute fundamental building blocks for the optimization of many industrial processes and are key to the stability of their operations with profound effects on our society and economy. The AI and Operations Research communities have devised a wide spectrum of techniques and algorithms to effectively leverage the problem structure and solve many hard CO problems instances within a reasonable time and with high accuracy. While this success has made possible the deployment of CO solutions in many real-world contexts, the complexity of these problems often prevent them to be adopted in contexts of repeated (e.g., involving expensive simulations, multi-year planning studies) or real-time nature, or when they depend in nontrivial ways on empirical data.

In many practical settings, we must solve many problem instances sharing similar patterns. This is the case with power system applications, for example, where the same problem is solved repeatedly with different inputs (e.g., loads and generation bids), or in industrial manufacturing scheduling, where the different inputs represent different tasks to be scheduled on a constant set of machines. Therefore, the application of deep learning methods to aid in solving computationally challenging constrained optimization problems is a promising approach and has gained traction in the nascent area at the intersection between constrained optimization and machine learning (ML) [Bengio *et al.*, 2020; Kotary *et al.*, 2021b; Vesselinova *et al.*, 2020]. However, while deep learning has proven its power for unconstrained problem settings, it has struggled to perform as well in domains where it is necessary to satisfy hard constraints. For example, in power systems, materials science, fluid dynamics, and many other areas, the data follows well-known physical laws, and violation of these laws can lead to unreliable and unusable approximations. *There is thus a critical need for fast deep learning approximators that can operate in settings where traditional optimizers are slow, yet where feasibility criteria must be satisfied.*

The goal of this paper it to summarize our recent research efforts made to address this need.

2 Problem Definition

Our research uses deep learning to approximate the optimal solutions to optimization problems of the form

$$P(\boldsymbol{d}): \quad \operatorname*{argmin}_{x} f(x, \boldsymbol{d}) \quad \text{subject to:} \quad x \in \mathcal{C}(\boldsymbol{d}), \qquad (1)$$

where $x \in \mathbb{R}^n$ is a vector of decision variables, $\boldsymbol{d} \in \mathbb{R}^m$ a vector of input data, and $f : \mathbb{R}^{n+m} \to \mathbb{R}_+$ is the problem objective. The set of constraints $\mathcal{C}$ is arbitrary. A recurrent class of problems we consider are nonlinear nonconvex programs defined by the constraint set $\mathcal{C}(\boldsymbol{d}) = \{x \mid H(x, \boldsymbol{d}) = 0;\ G(x, \boldsymbol{d}) \leq 0\}$, where H and G describe p equality and q inequality constraints, respectively, $h_i(x, \boldsymbol{d}) = 0$ $(i \in [p])$ and $g_i(x, \boldsymbol{d}) \leq 0$ $(i \in [q])$, each defined on $\mathbb{R}^n \times \mathbb{R}^m$.

The training task is given a dataset $D = \{(\boldsymbol{d}_i, \mathring{x}_i \in P(\boldsymbol{d}_i))\}_{i=1}^N$ of N samples, where $\mathring{x}_i$ is a (possibly) optimal solution to Problem (1) for input $\boldsymbol{d}_i$. The dataset is used to estimate a parametrized model represented by the function $\mathcal{M}$. Given its parameters vector θ and input $\boldsymbol{d}_i$, the model predicts $\hat{x}_i = \mathcal{M}_\theta(\boldsymbol{d}_i) \in \mathbb{R}^n$, a vector of values from the output space of x.

3 Learning CO Approximations

Given an optimization problem P of the form specified in Equation (1), the objective is to learn a parametric model $\mathcal{M}_\theta : \mathbb{R}^m \to \mathbb{R}^n$ that, given the problem input data $\boldsymbol{d}$, predicts the optimal values $\hat{x}$ for the decision variables. The learning task must solve the following empirical risk optimization:

$$\mathring{\theta} = \operatorname*{argmin}_{\theta} \sum_{i=1}^{N} \mathcal{L}\left(\mathcal{M}_\theta(\boldsymbol{d}_i), \mathring{x}_i\right) \qquad (2)$$

$$\text{such that:} \quad \mathcal{C}\left(\mathcal{M}_\theta(\boldsymbol{d}_i), \boldsymbol{d}_i\right) \quad \forall i \in [N], \qquad (3)$$

Test Case		Dist. to feasible sol. (%) DC	$\mathcal{M}^-$	$\mathcal{M}$	Dist. to AC-OPF sol. (%) DC	$\mathcal{M}^-$	$\mathcal{M}$
30_ieee	p^g	2.6972	2.0793	**0.0007**	0.1907	2.1353	**0.0058**
	v	1.2929	83.138	**0.0037**	3.4931	6.2996	**0.0086**
118_ieee	p^g	0.2011	0.1071	**0.0038**	0.5865	0.1353	**0.0368**
	v	1.9971	3.4391	**0.0866**	2.2780	4.5972	**0.1335**
300_ieee	p^g	0.1336	0.0447	**0.0084**	0.1717	0.0644	**0.0175**
	v	3.8526	31.698	**0.1994**	0.6854	2.9985	**0.2196**
	p^g	0.7751	0.9843	**0.0197**	0.6090	0.5694	**0.0356**
	v	2.4284	36.288	**0.1995**	1.7870	3.3879	**0.2136**
		Total Average			Total Average		

(a) Solution Quality.

Test Case	Dist. to AC-OPF cost (%) DC	$\mathcal{M}^-$	$\mathcal{M}$	Runtime (sec.) AC	DC	$\mathcal{M}$
30_ieee	7.9894	2.9447	**0.0180**	0.1024	0.0148	$\mathbf{< 10^{-4}}$
118_ieee	4.7455	1.0973	**0.5408**	0.4207	0.0785	**0.0001**
300_ieee	4.7508	1.9543	**0.3011**	8.0645	0.2662	**0.0001**
	4.5733	2.3706	**0.2124**	1x	30.3x	$\mathbf{> 10^4}$x
	Total Average			Min Speedup		

(b) Objective cost distances and runtime.

Table 1: Lagrangian dual deep learning: Each test case represents a benchmark set with different load congestion scenarios. Results are averaged across all scenarios and best results are shown in bold fonts.

to obtain an approximation $\hat{P} = \mathcal{M}_{\hat{\theta}}$ of P. The loss function $\mathcal{L}$ characterizes the similarity of the predictions $\hat{x}_i = \mathcal{M}_\theta(\boldsymbol{d}_i)$ to the ground truth $\mathring{x}_i$ and $\mathcal{C}(\hat{x}, \boldsymbol{d})$ holds if the problem constraints $\hat{x} \in \mathcal{C}(\boldsymbol{d})$ are satisfied. *However, this framework faces a fundamental challenge: deep learning frameworks cannot naturally handle the problem constraints*.

Lagrangian Dual Deep Learning. To drive the DNN predictions toward satisfying the problem constraints, our work has proposed several integrations of optimization techniques with deep learning [Kotary *et al.*, 2021b; Fioretto *et al.*, 2020b; Kotary *et al.*, 2022b]. In particular, one approach exploits Lagrangian duality to integrate trainable and weighted regularization terms that encapsulate constraints violations [Fioretto *et al.*, 2020a]. Indeed, the ML loss function should be guided by both data and the task at hand. The Lagrangian relaxation of problem P with $\mathcal{C} = \{x | H(x) = 0;\ G(x) \leq 0\}$ is a weighted sum of objective and constraint functions $f_L : \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^q \to \mathbb{R}$, defined as:

$$f_L(x, \lambda^h, \lambda^g) = f(x) + \sum_{i=1}^{p} \lambda_i^h\, h_i(x) + \sum_{i=1}^{q} \lambda_i^g\, g_i(x), \quad (4)$$

where λ_i^h and $\lambda_i^g \geq 0$ are the Lagrangian multipliers associated with constraints $h_i(x) = 0$ and $g_i(x) \leq 0$, respectively. The presentation omits the input data $\boldsymbol{d} \in \mathbb{R}^m$ in the various functions and constraints.

To approximate problem P with a DNN model $\mathcal{M}_\theta$ while accounting for the problem constraints, we parametrize the model loss function $\mathcal{L}$ by the Lagrangian multipliers $\lambda = \{\lambda_c\}_{c \in \mathcal{C}}$, as $\mathcal{L}_\lambda(\hat{x}, \mathring{x}) = \mathcal{L}^O(\hat{x}, \mathring{x}) + \mathcal{L}_\lambda^C(\hat{x})$, where $\hat{x} = \mathcal{M}_\theta(\boldsymbol{d})$ represents the model prediction, $\mathcal{L}^O(\hat{x}, \mathring{x})$ measures the prediction error and $\mathcal{L}_\lambda^C(\hat{x}) = \sum_{c \in \mathcal{C}} \lambda_c \nu_c(\hat{x})$ the constraint violations of the prediction. For multipliers λ, solving the optimization problem

$$\mathring{\theta}(\lambda) = \operatorname*{argmin}_\theta \sum_{i=1}^{N} \mathcal{L}_\lambda(\mathcal{M}_\theta(\boldsymbol{d}_i), \mathring{x}_i) \quad (5)$$

produces an approximation $\widehat{P}_\lambda = \mathcal{M}_{\mathring{\theta}(\lambda)}$ of P. The Lagrangian dual computes the optimal multipliers, i.e.,

$$\mathring{\lambda} = \operatorname*{argmax}_\lambda \min_\theta \sum_{i=1}^{N} \mathcal{L}_\lambda(\mathcal{M}_\theta(\boldsymbol{d}_i), \mathring{x}_i) \quad (6)$$

to obtain, $\widehat{P}^* = \mathcal{M}_{\mathring{\theta}(\mathring{\lambda})}$, i.e., the strongest Lagrangian relaxation of P. Learning $\widehat{P}^*$ relies on an iterative scheme that interleaves the learning of a number of Lagrangian relaxations (for various multipliers) with a subgradient method to learn the best multipliers and described in [Fioretto *et al.*, 2020a].

A Glimpse of its Effectiveness. The effectiveness of the Lagrangian dual learning framework was shown in several domains, including learning power flows in energy systems [Fioretto *et al.*, 2020b; Mak *et al.*, 2021], learning optimal compression settings in gas networks [Fioretto *et al.*, 2020a], trasnprecision computing [Fioretto *et al.*, 2020a], ML model pruning [Kaur *et al.*, 2022], and fair machine learning [Tran *et al.*, 2021b, 2022; Nagar *et al.*, 2021].

In the following we provide an overview of the results obtained to the problem of finding the least cost generator dispatch to meet the demands in a power network (AC-OPF). The problem is required to satisfy the nonconvex nonlinear AC power flow equations that are a core building block in many power system applications. Table 1 summarizes results for power systems with up to 1,000+ lines [Fioretto *et al.*, 2020b; Chatzos *et al.*, 2020]. It compares the Lagrangian dual method $\mathcal{M}$ against the DC-OPF model (DC)–which is a linear approximation of the AC-OPF–and a DNN model $\mathcal{M}^-$ whose loss function only includes the prediction error $\mathcal{L}^O(\hat{x}, \mathring{x})$ component (i.e., it does not encourage constraint satisfaction). The DNN architectures are based on a feed-forward multi-task network to predict the various physical quantities (generators power and voltage) [Fioretto *et al.*, 2020b]. Table 1a compares the accuracy of the proposed DNN model against the approximation found by the DC-OPF model and the *optimal* AC-OPF solutions. The solution quality is measured by first finding the closest AC feasible solution to the predictions returned by the DC or by the DNN models to restore feasibility. Then, the dispatch values are compared to the original ones. The table reports the average distances (%) for the active power (p^g) and voltage magnitude (v) against the model predictions (left sub-table) and the AC-OPF solutions (right sub-table). The table reports the results for three benchmark networks (with 30, 118, and 300 buses) and the last row reports the average results for an extensive set of medium-size networks [Fioretto *et al.*, 2020b; Chatzos *et al.*, 2020]. The results illustrate that the Lagrangian dual model is up to two orders of magnitude more precise

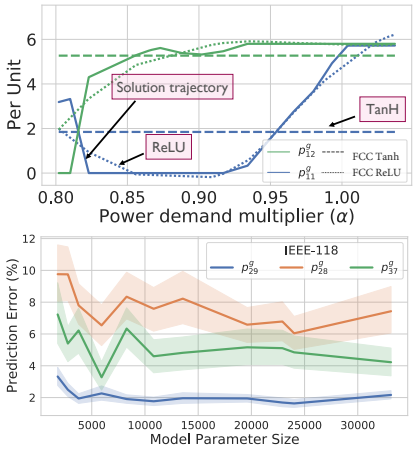

Fig. 1: Solution Trajectories of AC-OPF variables and their learned approximations (top) and prediction errors at varying of the model size (bottom).

than the DC model and the baseline DNN. Table 1b compares accuracy and runtime of the proposed DNN model and DC against the AC-OPF solutions found by a state-of-the-art (SoTA) nonlinear solver. The left sub-table reports the average L1-distances of the dispatch costs against the optimal costs and the right sub-table illustrates the time required to find an AC-OPF solution, a linear DC approximation, and a prediction using the proposed Lagrangian approach. Observe that the proposed model finds dispatches whose costs are at least one order of magnitude closer to the true solution than those returned by the DC while being several orders of magnitude faster. *These results are significant: They suggest that the integration of optimization and deep learning has the potential to replace approximations, such as the DC model in energy systems, and deliver solutions with greater fidelity and computing speed.*

4 Why Constraints help Predictions?

While this line of research has been shown effective to approximate CO solutions, still little is known on when and why these models can predict solutions to CO problems accurately, as well as about their predictions robustness. In an effort to address this knowledge gap, and recognizing that answers to this questions may depend on the target problem and computational budgets, in [Dinh *et al.*, 2021] we focused on AC-OPF predictions.

Solution Trajectories as Piecewise Linear Functions. To answer why DNNs are able to approximate OPF solutions with low errors, we studied the relation between the training data and their target outputs. Figure 1 (top) shows, with solid lines, how generator outputs change as a function of the total demand for selected IEEE-118 generators. Observe that, the solution trajectory associated with the problem instances on various input parameters can be naturally approximated by piecewise linear functions. This approximation is in fact exact for linear programs when the inputs capture incremental changes to the objective coefficients or the right-hand side of the constraints. Additionally, ReLU neural networks have the ability to capture piecewise linear functions [Moon, 2021].

However, while these models are thus compatible with the task of predicting the solutions of an optimization problem, the model capacity required to represent a target piecewise linear function exactly depends directly on the number of constituent pieces. In a [Kotary *et al.*, 2021a] we characterized the error associated with the classes of functions that can be learned by a ReLU network model with fixed capacity with respect to a target trajectory. When many variables have simple solution trajectories, thus, highly accurate approximation can be obtained by a ReLU networks, even if ignoring the problem constraints. In the figure we illustrate that a model equipped with ReLU activations (dotted line) can indeed better approximate a model equipped with TanH activations (dashed lines).

The Importance of Constraints. Givne the result above, it is to be expected that larger DNN models will be better suited to learning more complex solution trajectories. However, this aspect was not consistently observed in our experiments [Dinh *et al.*, 2021]. Figure 1 (bottom) illustrates this surprising behavior for three generators with complex trajectorires, showing that the prediction error does not decrease at increasing of the model size. To understand why this behavior occurs, in [Dinh *et al.*, 2021] we examined the dynamics of highly volatile solution trajectories, which, incidentally, are also associated with high prediction errors, when a DNN (unconstrained) model is used to predict them. In the paper we show that, the highly volatile portions of the solution trajectories are often associated with binding constraint in the problem and that the prediction errors arise as the hidden representation of the DNN does not accurately learn the operational and physical constraints which regulate the behavior of the OPF solutions. When constraints are explicitly encoded in the model, e.g., via the Lagrangian dual method described above, these behaviors largely disappear obtaining highly accurate predictions.

5 Dealing with Data and Discrete Structures

Section 3 focused on endowing deep learning models with the ability to satisfy constraints. In all these studies, the ML frameworks that learn to approximate solutions to hard optimization problems are assumed to have access to supervision labels that can guide the construction of the solutions to target problems. However, when these labels are themselves approximations, when the optimization problem has symmetric solutions, and/or when the optimization solver uses randomization, solutions to closely related instances may exhibit large differences and the learning task can become inherently more difficult. This issue particularly acute when learning proxies to discrete optimization problems, whose solution spaces often present a large number of symmetries.

These challenges are illustrated in Figure 2 (Standard Labels/blue curves). The left subplot shows the L_1-distance between the obtained solution to each training instance (i.e., the start times of the tasks in a job shop scheduling problem), sorted by processing times, and a reference optimal solution for some instance. The curve shows that meaningful patterns

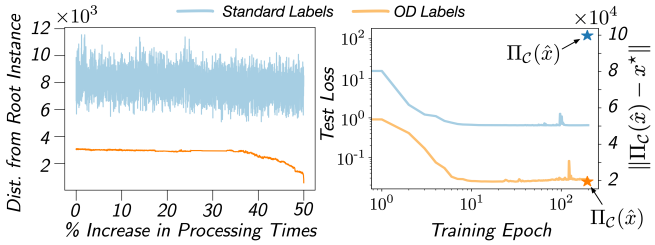

Fig. 2: Data: L1 distance (left) and test loss (right).

can be lost, including the important relationship between an increase in processing times and the resulting solutions. Figure 2 (right) shows that, despite the high volatility, the learning problem appears well behaved in the face of minimizing the test loss. However, when training loss converges, accuracy (measured as the distance between the *projection* $\Pi_{\mathcal{C}}(\hat{x})$ of the prediction $\hat{x}$ onto the schedule-feasible space $\mathcal{C}$ and the real label $\dot{x}$) remains poor (blue star). Similar observations pertain to constraints violations, which remain high. It is worth emphasizing that these volatility issues are further exacerbated when time constraints prevent the solver from obtaining optimal solutions.

In [Kotary *et al.*, 2021a] we have analyzed this critical challenge, connecting the volatility of the training data to the ability of a model with given capacity to approximate it. In addition, we devised a method for producing (exact or approximate) solutions to optimization problems that are more amenable to supervised learning tasks.

Exploiting the Problem Structure This training data generation process was used to learn highly accurate job shop scheduling problems [Kotary *et al.*, 2022a], a notoriously challenging combinatorial optimization problem, due to its disjunctive constraints. When dealing with such problem, we also observe that the network architecture plays a non-trivial role in helping the training process to converge to better minima and, even, to help generating predictions that satisfy the problem constraints. More specifically, in [Kotary *et al.*, 2022a] we devise a network that differentiates three types of layers: Job layers, that process processing times organized by jobs, Machine layers, that process processing times organized by machines, and Shared layers, that process the outputs of the job layers and the machine layers to return a prediction. Such architecture exploits directly the job-shop problem structure and was shown highly beneficial to the learning task. We believe that this observation is highly generalizable to other contexts and decision tasks.

Learning Constrained Policies Despite the high quality results obtained in the learning methods described in the previous sections, their predictions offer no guarantees on the satisfaction of the problem constraints. This issue is often mitigated by a post-prediction step that projects the predictions returned by the learning models onto the feasible space. While these projections steps use a classical optimization solver, they are generally extremely fast, as the solvers can be warm-started with the predicted solutions, which are already very close to the optimal ones.

This issue however, complicates further in the presence of discrete structures, where efficient projections may not exists or where many symmetric solutions exists, as pointed out above. Motivated by this challenge, we have proposed a integration of constrained optimization programs within a deep learning pipeline which is trained end-to-end to produce optimal (constraint-satisfying) solutions. This technique is studied in [Kotary *et al.*, 2022b] in the context of fair learning to rank tasks. The task consists in learning a mapping between a list of items and a permutation of such list, which defines the order in which the items should be ranked in response to a user query while enforcing some desired notion of group fairness over rankings. In addition to providing a certificate on the fairness requirements, this method allows the modeler to enforce a large number of fairness concepts that can be formulated as linear constraints over rankings and is shown to significantly improve current state-of-the-art fair learning-to-rank systems with respect to established performance metrics.

Finally, we have shown how the integration of machine learning and optimization can be used to enforce fairness constraints on predictors and obtain state-of-the-art results when minimizing disparate treatments [Fioretto *et al.*, 2020a]. This technique can be used in combination with differential privacy to achieve a privacy-preserving deep learning algorithm that also encourages the satisfaction of a variety of fairness constraints, as we have shown in [Tran *et al.*, 2021a,b].

6 Opportunities and Challenges

Our research on integrating constrained optimization with end-to-end machine learning is part of a larger effort embarked by a nascent community at the intersection between operations research and artificial intelligence [Kotary *et al.*, 2021b]. Despite the encouraging results, a number of challenges remain that must be addressed to allow an integration that lives up to its full potential. **(1)** In predicting solutions to constrained optimization problems, the current methods cannot robustly guarantee arbitrary problem constraints to be satisfied. This critical shortcoming may be addressed by integrating ML approaches with methods from the robust optimization literature or by developing ad-hoc layers to project the predictions onto the feasible space. **(2)** Despite the variety of approaches, the success of integrating an optimization solver within a ML model in the loop has been demonstrated on a relatively limited set of optimization problems and, focusing mostly on linear programming formulations. Challenges posed by the parametrization of constraints stand in the way of broader applications. **(3)** Issues associated with the runtime of combinatorial solvers in-the-loop still make some potential applications impractical. **(4)** Finally, this area still lacks theoretical results providing guarantees on the classes of optimization problems that can be approximated via a learning proxy and their performance.

In summary, the integration of constrained optimization and learning promises to produce a new generation of solvers that will enable novel operation assessments at unprecedented scales (e.g., over multi-years simulation studies) and may transform the current concepts of robustness and high-quality solutions in many engineering and scientific applications.

Acknowledgments

This research is partially supported by NSF grant 2007164, NSF CAREER Award 2143706, and a Google Research Scholar Award. Its views and conclusions are those of the authors only.

References

Yoshua Bengio, Andrea Lodi, and Antoine Prouvost. Machine learning for combinatorial optimization: a methodological tour d'horizon. *European Journal of Operational Research*, 2020.

Minas Chatzos, Ferdinando Fioretto, Terrence W. K. Mak, and Pascal Van Hentenryck. High-fidelity machine learning approximations of large-scale optimal power flow. *CoRR*, abs/2006.16356, 2020.

My H. Dinh, Ferdinando Fioretto, Mostafa Mohammadian, and Kyri Baker. Towards understanding the unreasonable effectiveness of learning AC-OPF solutions. *CoRR*, abs/2111.11168, 2021.

Ferdinando Fioretto, Pascal Van Hentenryck, Terrence W. K. Mak, Cuong Tran, Federico Baldo, and Michele Lombardi. Lagrangian duality for constrained deep learning. In *European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD)*, volume 12461, pages 118–135, 2020.

Ferdinando Fioretto, Terrence W.K. Mak, and Pascal Van Hentenryck. Predicting ac optimal power flows: Combining deep learning and lagrangian dual methods. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 630–637, 2020.

Sawinder Kaur, Ferdinando Fioretto, and Asif Salekin. Deadwooding: Robust global pruning for deep neural networks. *CoRR*, abs/2202.05226, 2022.

James Kotary, Ferdinando Fioretto, and Pascal Van Hentenryck. Learning hard optimization problems: A data generation perspective. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

James Kotary, Ferdinando Fioretto, Pascal Van Hentenryck, and Bryan Wilder. End-to-end constrained optimization learning: A survey. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 4475–4482, 2021.

James Kotary, Ferdinando Fioretto, and Pascal Van Hentenryck. Fast approximations for job shop scheduling: A lagrangian dual deep learning method. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, page to appear, 2022.

James Kotary, Ferdinando Fioretto, Pascal Van Hentenryck, and Ziwei Zhu. End-to-end learning for fair ranking systems. In *Proceedings of the ACM Web Conference (WWW)*, pages 3520–3530, 2022.

Terrence W. K. Mak, Ferdinando Fioretto, and Pascal Van Hentenryck. Load embeddings for scalable AC-OPF learning. *CoRR*, abs/2101.03973, 2021.

Sunghwan Moon. Relu network with bounded width is a universal approximator in view of an approximate identity. *Applied Sciences*, 11(1):427, 2021.

Anudit Nagar, Cuong Tran, and Ferdinando Fioretto. Privacy-preserving and accountable multi-agent learning. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 1605–1606, 2021.

Cuong Tran, My H. Dinh, and Ferdinando Fioretto. Differentially private empirical risk minimization under the fairness lens. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 27555–27565, 2021.

Cuong Tran, Ferdinando Fioretto, and Pascal Van Hentenryck. Differentially private and fair deep learning: A lagrangian dual approach. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 9932–9939, 2021.

Cuong Tran, Keyu Zhu, Ferdinando Fioretto, and Pascal Van Hentenryck. SF-PATE: scalable, fair, and private aggregation of teacher ensembles. *CoRR*, abs/2204.05157, 2022.

Natalia Vesselinova, Rebecca Steinert, Daniel F Perez-Ramirez, and Magnus Boman. Learning combinatorial optimization on graphs: A survey with applications to networking. *IEEE Access*, 8:120388–120416, 2020.