E2E-AT: A Unified Framework for Tackling Uncertainty in Task-Aware End-to-End Learning

Wangkun Xu[1], Jianhong Wang[2], Fei Teng[1]

[1]Department of EEE, Imperial College London, UK
[2]Center for AI Fundamentals, The University of Manchester, UK
wangkun.xu18@imperial.ac.uk, jianhong.wang@manchester.ac.uk, fei.teng@imperial.ac.uk

Abstract

Successful machine learning involves a complete pipeline of data, model, and downstream applications. Instead of treating them separately, there has been a prominent increase of attention within the constrained optimization (CO) and machine learning (ML) communities towards combining prediction and optimization models. The so-called end-to-end (E2E) learning captures the task-based objective for which they will be used for decision making. Although a large variety of E2E algorithms have been presented, it has not been fully investigated how to systematically address uncertainties involved in such models. Most of the existing work considers the uncertainties of ML in the input space and improves robustness through adversarial training. We extend this idea to E2E learning and prove that there is a robustness certification procedure by solving augmented integer programming. Furthermore, we show that neglecting the uncertainty of COs during training causes a new trigger for generalization errors. To include all these components, we propose a unified framework that covers the uncertainties emerging in both the input feature space of the ML models and the COs. The framework is described as a robust optimization problem and is practically solved via end-to-end adversarial training (E2E-AT). Finally, the performance of E2E-AT is evaluated by a real-world end-to-end power system operation problem, including load forecasting and sequential scheduling tasks.

Introduction

ML-based prediction algorithms are widely used in real-world applications, such as load forecasting in power system operation, demand forecasting in retailing, and inventory stock forecasting in commerce[*]. Although training such models is straightforwardly a supervised learning problem, the training criteria may not be aligned with the ultimate goal of the downstream decision-making tasks. For instance, the forecast load is further used for generator dispatch in the power system, and the demand forecasting can be used to guide manufacturing. That is, the system designer first trains an ML forecaster using standard statistic training loss, e.g. the mean squared error (MSE) or cross entropy (CE), and then applies the forecast as input to the decision making. In fact, task-aware cost, which is modeled as a constrained optimization (CO) problem, can benefit from training a more economical forecast model. Therefore, task-aware end-to-end learning is proposed by combining ML and COs. However, the uncertainties involved in E2E learning have not been fully understood. This paper aims to bridge the gap from the perspective of robust optimization and adversarial robustness.

In the literature, E2E learning is also named as (smart) predict-then-optimize (Elmachtoub and Grigas 2022), integrated learning and optimization (Sadana et al. 2023), as well as decision-focused learning (Wilder, Dilkina, and Tambe 2019). In the paradigm of contextual optimization, it appears as early as in (Bengio 1997) and has recently shown a surge of interest. The E2E model is integrated to minimize the task-aware cost subject to task constraints. As a result, the CO is first solved in the forward pass to obtain the optimal decision. In the backward pass, the Jacobian from the optimal decision to the forecast value needs to be calculated. Among many approaches, this paper adopts the implicit differentiation method, which is more efficient than the unrolling approach (Domke 2012) and more accurate than the surrogate loss function approach (Elmachtoub and Grigas 2022). Practically, OptNet (Amos and Kolter 2017) applies the implicit function theorem (Krantz and Parks 2002) to denote the Jacobian after formulating the Karush-Kuhn-Ticher (KKT) condition of parametric quadratic programming (QP), which is further extended to disciplined convex programming in CvxpyLayers (Agrawal et al. 2019).

E2E learning has been applied to many critical industrial activities, such as better management of the power system (Stratigakos et al. 2022; Vohra, Rajaei, and Cremer 2023), constraint satisfaction in control (Chen et al. 2021), and routing behavior of the transportation network (Liu et al. 2023). Therefore, it is essential to understand the vulnerability of this learning framework and to study the associated robust enhancement. In detail, the E2E model contains three parts: (a). a parametric forecast model that maps the contextual information (e.g. the input feature) to the interest of forecast; (b). CO models that take the forecast as input and return decisions; and (c). a task-aware loss function that encodes the ultimate goal of decision making. Meanwhile, the parameters of COs and task-aware loss can be classified into

[*]Our code is available at https://github.com/xuwkk/E2E-AT. The appendix of this paper can be found at https://arxiv.org/abs/2312.10587.

predictable and unpredictable parameters. While the uncertainty of predictable parameters can be modeled by the forecast model, the uncertainty of unpredictable parameters are overlooked in literature. Conventionally, the uncertainty of the data is only defined by the input of the forecast model, the worst case of which can be found by adversarial attack and treated by adversarial training (Madry et al. 2017). In the E2E model, the definition of data is augmented by the unpredictable parameters of COs[*]. Our main contribution is to treat multiple sources of uncertainty uniformly as a robust optimization problem, which can be practically solved by end-to-end adversarial training (E2E-AT). Finally, our method can be viewed as a natural interconnection between adversarial training, the implicit layer, and E2E learning.

Preliminaries

Adversarial Training and Certified Robustness

In supervised learning, a parametric model $\boldsymbol{f}(\boldsymbol{x};\boldsymbol{\theta})$ can be defined to map from input $\boldsymbol{x}$ to label $\boldsymbol{y}$ for $(\boldsymbol{x},\boldsymbol{y}) \in \mathcal{D}$ by minimizing the following empirical loss:

$$\min_{\boldsymbol{\theta}} \sum_{i \in \mathcal{D}} \mathcal{L}(\boldsymbol{f}(\boldsymbol{x}^i;\boldsymbol{\theta}),\boldsymbol{y}^i) \qquad (1)$$

In this paper, we use $(\boldsymbol{x},\boldsymbol{y}) \in \mathcal{D}$ and $i \in \mathcal{D}$ interchangeably to denote a sample of the dataset.

It has been well studied that the deep neural network is prone to small perturbations on its input. In security-critical applications, robustness has become an emerging factor. Adversarial training has been shown to improve the robustness of NN, which transforms (1) into a robust optimization (Madry et al. 2017):

$$\min_{\boldsymbol{\theta}} \sum_{i \in \mathcal{D}} \max_{\boldsymbol{\delta}_i \in \Delta} \mathcal{L}(\boldsymbol{f}(\boldsymbol{x}^i + \boldsymbol{\delta}^i;\boldsymbol{\theta}),\boldsymbol{y}^i) \qquad (2)$$

where $\Delta = \{\boldsymbol{\delta} : \|\boldsymbol{\delta}\|_\infty \leq \epsilon\}$ is the attack budget set for some small $\epsilon > 0$.

Robust optimization (2) is practically solved by iterative approaches in which the adversarial perturbation is first solved by inner minimization through gradient ascent. To keep the attack within Δ, projected gradient descent (PGD) (Madry et al. 2017) is adopted:

$$\boldsymbol{\delta}_{t+1}^i = \mathcal{P}_\Delta \left(\boldsymbol{\delta}_t^i + \gamma \cdot \mathrm{sign} \left(\nabla_{\boldsymbol{\delta}} \mathcal{L} \left(\boldsymbol{f} \left(\boldsymbol{x}^i + \boldsymbol{\delta}_t^i \right), \boldsymbol{y}^i \right) \right) \right) \quad (3)$$

where $\mathcal{P}_\Delta$ is the projector on Δ and γ is the step size.

Although adversarial training is effective in improving robustness, it may give the wrong sign of security as inner maximization is inexactly solved. A certification approach can be made on the feedforward neural network with ReLU activations, e.g., a piecewise linear neural network[†]. Using the big-M method, the neural network with d layers can be represented by the following set (Tjeng, Xiao, and Tedrake

2017)

$$\mathcal{C}_{\mathrm{nn}}(\boldsymbol{x};\boldsymbol{\theta}) = \left\{ \boldsymbol{y} : \begin{array}{l} \boldsymbol{z}_1 = \boldsymbol{x}, \boldsymbol{z}_{i+1} \geq \boldsymbol{W}_i \boldsymbol{z}_i + \boldsymbol{b}_i, \\ \boldsymbol{z}_{i+1} \geq \boldsymbol{0}, \boldsymbol{u}_i \cdot \boldsymbol{v}_i \geq \boldsymbol{z}_{i+1} \\ \boldsymbol{W}_i \boldsymbol{z}_i + \boldsymbol{b}_i \geq \boldsymbol{z}_{i+1} + (\boldsymbol{1} - \boldsymbol{v}_i)\boldsymbol{l}_i, \\ \boldsymbol{v}_i \in \{0,1\}^{|\boldsymbol{v}_i|}, \quad i = 1, \cdots, d-2 \\ \boldsymbol{y} = \boldsymbol{W}_{d-1} \boldsymbol{z}_{d-1} + \boldsymbol{b}_{d-1} \end{array} \right\} \qquad (4)$$

where $\boldsymbol{\theta}_i = (\boldsymbol{W}_i, \boldsymbol{b}_i)$ is the weights of the i-th layer. $\boldsymbol{u}_i$ and $\boldsymbol{l}_i$ are the upper and lower bounds of the output of i-th layer, which can be efficiently estimated by interval bound propagation (IBP) (Gowal et al. 2018). $\boldsymbol{v}_i$ is an integer vector that controls the activation of ReLUs.

Based on (4), the inner maximization of (2) becomes:

$$\max_{\boldsymbol{\delta}^i} \mathcal{L}(\hat{\boldsymbol{y}}^i, \boldsymbol{y}^i), \text{ subject to } \boldsymbol{\delta}^i \in \Delta, \hat{\boldsymbol{y}}^i \in \mathcal{C}_{\mathrm{nn}}(\boldsymbol{x}^i + \boldsymbol{\delta}^i; \boldsymbol{\theta}) \qquad (5)$$

As $\mathcal{C}_{\mathrm{nn}}(\boldsymbol{x};\boldsymbol{\theta})$ is defined by mixed integer linear constraints, (5) can be solved exactly for certain types of loss function. When cross-entropy loss is used, we can choose to maximize the output of each logit, and if all the logits are not greater than the ground-truth logit within the attack budget, the NN is said to be certified robust at the candidate sample. For a regression task, (5) can be formulated as mixed-integer linear programming (MILP) to maximize or minimize the forecast value (Xu and Teng 2023).

End-to-End Machine Learning

End-to-end learning takes the fusion of prediction (ML) and decision making (CO), which aims to find the map from the data to the optimal decision such that the learned ML model can optimally reflect the CO via training. Meanwhile, the E2E learning distinguishes itself from separate supervised learning followed by the CO in which the prediction divergence and the task-aware cost can have a large mismatch (Sadana et al. 2023).

Referring to a recent review (Kotary et al. 2021), the CO can be modelled as

$$\boldsymbol{z}^\star = \mathrm{argmin}_{\boldsymbol{z}} \ell(\boldsymbol{z};\boldsymbol{y}) \text{ subject to } \boldsymbol{z} \in \mathcal{C}(\boldsymbol{z};\boldsymbol{y}) \qquad (6)$$

where $\ell(\cdot)$ is the objective function, $\boldsymbol{z}$ is the decision variable, $\boldsymbol{y}$ is the parameter, and $\mathcal{C}$ is the constraint set.

Let $\hat{\boldsymbol{y}}$ be the output of $\boldsymbol{f}(\boldsymbol{x};\boldsymbol{\theta})$ and let $\hat{\boldsymbol{z}}^\star$ be the minimizer of (6) parameterized by $\hat{\boldsymbol{y}}$. There are two options to learn the E2E model guided by (6). In the supervised learning setting, a suitable loss function $\mathcal{M}$ can be chosen to minimize the difference to ground-truth decision, e.g. $\mathcal{M}(\hat{\boldsymbol{z}}^\star, \boldsymbol{z}^\star)$ or $\mathcal{M}(\ell(\hat{\boldsymbol{z}}^\star; \hat{\boldsymbol{y}}), \ell(\boldsymbol{z}^\star; \boldsymbol{y}))$ (Kong et al. 2022). Alternatively, the regret function (Elmachtoub and Grigas 2022) can be implemented:

$$\mathrm{regret} = \ell(\hat{\boldsymbol{z}}^\star; \hat{\boldsymbol{y}}) - \ell(\boldsymbol{z}^\star; \boldsymbol{y}) \qquad (7)$$

The ground truth decision $\boldsymbol{z}^\star$ in (7) is not necessary to compute, as $\ell(\boldsymbol{z}^\star; \boldsymbol{y})$ is a constant (Wilder, Dilkina, and Tambe 2019). Therefore, we argue that the regret function (7) is more intrinsic as the training loss has the same format as the CO objective (6).

---

[*]We merge the unpredictable parameter of task-aware cost as part of CO.

[†]Convolution layer and other piecewise linear activations, such as leaky ReLU, are also piecewise linear.

A Heuristic Example

Before we move into a detailed formulation of the robustness of E2E learning, we first highlight a misleading formulation, which violates the intention of E2E learning and introduces model uncertainties between training and inference.

Consider two E2E learning problems based on the supervised setting:

$$\operatorname{argmin}_{\boldsymbol{\vartheta}, \hat{\boldsymbol{z}}} \quad M(\hat{\boldsymbol{z}}) := \sum_{i \in \mathcal{D}} \mathcal{M}(\ell(\hat{\boldsymbol{z}}^i; \hat{\boldsymbol{y}}^i), \ell(\boldsymbol{z}^{i\star}; \boldsymbol{y}^i))$$
$$\text{subject to} \quad \hat{\boldsymbol{z}}^i \in \mathcal{C}(\boldsymbol{z}^i; \hat{\boldsymbol{y}}^i), \quad i \in \mathcal{D}$$
(8)

whose minimum point is denoted as $(\boldsymbol{\theta}_1^\star, \hat{\boldsymbol{z}}_1^\star)$. And,

$$\operatorname{argmin}_{\boldsymbol{\vartheta}, \hat{\boldsymbol{z}}} \quad M(\hat{\boldsymbol{z}}) := \sum_{i \in \mathcal{D}} \mathcal{M}(\ell(\hat{\boldsymbol{z}}^i; \hat{\boldsymbol{y}}^i), \ell(\boldsymbol{z}^{i\star}; \boldsymbol{y}^i))$$
$$\text{subject to} \quad \hat{\boldsymbol{z}}^i \in \operatorname{argmin}_{\boldsymbol{z}^i} \{ \ell(\boldsymbol{z}^i; \hat{\boldsymbol{y}}^i) : \boldsymbol{z}^i \in \mathcal{C}(\boldsymbol{z}^i; \hat{\boldsymbol{y}}^i) \},$$
$$i \in \mathcal{D}$$
(9)

with minimum point $(\boldsymbol{\theta}_2^\star, \hat{\boldsymbol{z}}_2^\star)$.

E2E learning problem (8) takes the sample-wise constraints of (6) as its constraints while (9) is a bilevel optimization problem whose upper level minimizes the difference to the ground-truth objective $\ell(\boldsymbol{z}^{i\star}; \boldsymbol{y}^i)$ and each lower level solves the sample-wise decision-making problem in parallel.

In the **inference stage**, we first predict $\hat{\boldsymbol{y}} = f(\boldsymbol{x}; \boldsymbol{\theta}^\star)$ and then solve the optimization problem (6) parameterized by $\hat{\boldsymbol{y}}$. The optimal decision variables are denoted as $\hat{\boldsymbol{z}}_r^\star(\boldsymbol{\theta}_1^\star)$ and $\hat{\boldsymbol{z}}_r^\star(\boldsymbol{\theta}_2^\star)$, respectively.

Proposition 1 *Given two formulations* (8) *and* (9) *of E2E learning,* $M(\hat{\boldsymbol{z}}_1^\star) \leq M(\hat{\boldsymbol{z}}_2^\star) = M(\hat{\boldsymbol{z}}_r^\star(\boldsymbol{\theta}_2^\star)) \leq M(\hat{\boldsymbol{z}}_r^\star(\boldsymbol{\theta}_1^\star))$.

The proof can be found in the appendix.

It shows that formulation (8) can result in different decisions at the training and inference stages. Therefore, Proposition 1 demonstrates that a misformulation of CO can possibly lead to misled decision making at inference. Broadly speaking, the ignorance of the optimization model contributes to a new source of uncertainties, causing generalization error, in addition to the well-studied uncertainties in the dataset (e.g. out-of-distribution, adversarial attack, etc.). This motivates us to take into account the uncertainties in both the input space and the CO formulation.

Sequential E2E Learning as Multi-Level Optimization

Proposition 1 implies formulating E2E learning as a multilevel optimization problem by respecting the sequence of downstream tasks. Given $\boldsymbol{\theta}$, sequential decision makings can be denoted as lower-level problems after the prediction.

Formally, at the inference stage, the cost of one-time decision making on $(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{D}$ can be written as:

$$\min \quad \mathcal{L}(\hat{\boldsymbol{z}}_1, \cdots, \hat{\boldsymbol{z}}_m; \boldsymbol{y}, \boldsymbol{\phi}_0)$$
$$\text{subject to} \quad \hat{\boldsymbol{z}}_i \in \operatorname{argmin}_{\boldsymbol{z}_i} \{ \ell_i(\boldsymbol{z}_i; \boldsymbol{y}, \boldsymbol{\phi}_i) :$$
$$\boldsymbol{z}_i \in \mathcal{C}_i(\boldsymbol{z}_i; \boldsymbol{y}, \hat{\boldsymbol{z}}_{i-1}, \boldsymbol{\phi}_i) \}, \quad i = 2, \cdots, m$$
$$\hat{\boldsymbol{z}}_1 \in \operatorname{argmin}_{\boldsymbol{z}_1} \{ \ell_1(\boldsymbol{z}_1; \boldsymbol{y}, \boldsymbol{\phi}_1) :$$
$$\boldsymbol{z}_1 \in \mathcal{C}_1(\boldsymbol{z}_1; \boldsymbol{y}, \hat{\boldsymbol{y}}, \boldsymbol{\phi}_1) \}$$
$$\hat{\boldsymbol{y}} = f(\boldsymbol{x}; \boldsymbol{\theta})$$
(10)

Inference (10) contains m downstream tasks with objective $\ell_i(\cdot)$s, constraint set $\mathcal{C}_i(\cdot)$s, and **unpredictable** parameter $\boldsymbol{\phi}_i$s. It can be compactly denoted as

$$\min_{\hat{\boldsymbol{z}}} \quad \mathcal{L}(\hat{\boldsymbol{z}}; \boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\phi})$$
$$\text{subject to} \quad \hat{\boldsymbol{z}} \in \mathcal{C}_{\text{E2E}}(\boldsymbol{z}; \boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\phi})$$
(11)

where $\hat{\boldsymbol{z}} = [\hat{\boldsymbol{z}}_1, \cdots, \hat{\boldsymbol{z}}_m]$ and $\boldsymbol{\phi} = [\boldsymbol{\phi}_0, \cdots, \boldsymbol{\phi}_m]$. $\mathcal{C}_{\text{E2E}}(\cdot)$ represents the constraint set representing all constraints of (10). Note that we consider the COs as parametric functions that can be uniquely modeled by $(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\phi})$.

To train $f(\cdot; \boldsymbol{\theta})$, the empirical training loss can be minimized:

$$\min_{\boldsymbol{\theta}} \quad \sum_{i \in \mathcal{D}} \mathcal{L}(\hat{\boldsymbol{z}}^i; \boldsymbol{\theta}, \boldsymbol{x}^i, \boldsymbol{y}^i, \boldsymbol{\phi})$$
$$\text{subject to} \quad \hat{\boldsymbol{z}}^i \in \mathcal{C}_{\text{E2E}}(\boldsymbol{z}^i; \boldsymbol{\theta}, \boldsymbol{x}^i, \boldsymbol{y}^i, \boldsymbol{\phi}), \quad i \in \mathcal{D}$$
(12)

Since this paper does not focus on solving optimizations, we restrict downstream optimizations to quadratic programming (QP). Furthermore, QP has been widely implemented in many industrial applications and is mostly discussed in the E2E learning literature (Kotary et al. 2021). Meanwhile, since QP is convex and if the Slater condition holds, the Karush–Kuhn–Tucker (KKT) condition is sufficient for optimality (Boyd and Vandenberghe 2004). Therefore, optimizations at the lower level can be replaced by the corresponding KKT conditions, known as the mathematical program with equilibrium constraints (MPEC) (Luo, Pang, and Ralph 1996). Therefore, if a linear parametric model is considered, it is possible to solve (12) exactly using optimization software.

In addition to the linear model, stochastic gradient descent (SGD) needs to be applied on the mini-batches of $\mathcal{D}$ to train the NN model. SGD requires 1) a forward pass in which the optimizations are solved and 2) a backward pass to update the NN parameters. Denote the equality part of KKT condition of the i-th optimization as

$$g(\boldsymbol{z}_i^\star; \hat{\boldsymbol{z}}_{i-1}, \boldsymbol{y}, \boldsymbol{\phi}_i) = \boldsymbol{0}$$
(13)

which can be viewed as differentiable layer (OptNet) (Amos and Kolter 2017) by the implicit function theorem (Krantz and Parks 2002):

$$\frac{\partial \boldsymbol{z}_i^\star}{\partial \hat{\boldsymbol{z}}_{i-1}} = - \left(\frac{\partial g(\boldsymbol{z}_i^\star; \hat{\boldsymbol{z}}_{i-1}, \boldsymbol{y}, \boldsymbol{\phi}_i)}{\partial \boldsymbol{z}_i} \right)^{-1} \frac{\partial g(\boldsymbol{z}_i^\star; \hat{\boldsymbol{z}}_{i-1}, \boldsymbol{y}, \boldsymbol{\phi}_i)}{\partial \hat{\boldsymbol{z}}_{i-1}}$$
(14)

Note that in (13) and (14), the optimal dual variables $(\boldsymbol{\lambda}^\star, \boldsymbol{\nu}^\star)$ in the KKT conditions are omitted for simplicity. As long as the Jacobian matrix is not singular, the gradient of the output $\boldsymbol{z}_i^\star$ to the input $\hat{\boldsymbol{z}}_{i-1}$ exists, allowing backpropagation through the differentiable layers.

Unified Robustness Framework

When treating the E2E framework as an integrated model, the data source includes both conventionally defined data samples $(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{D}$ and the unpredictable parameter $\boldsymbol{\phi}$ of COs. Small input uncertainties have been shown to cause a significant performance drop, and it is reasonable to draw a similar conclusion for the unpredictable parameter. In fact, Proposition 1 shows that any mismatches between the COs
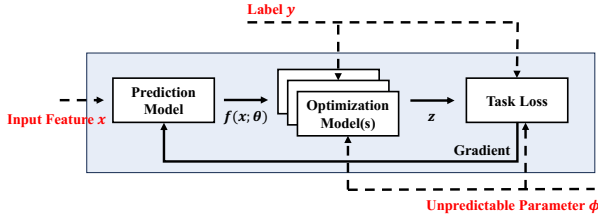
Figure 1: An illustration of E2E learning. We consider the uncertainties in the input data $(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{D}$ and the uncertainties in the COs, specifically, the unpredictable parameter $\boldsymbol{\phi}$.

used for training and inference should be explicitly considered. Although COs can take an infinite number of formulations, without loss of generality, we restrict the uncertainties of COs in the parameters of objective and constraints. We argue that the unpredictable parameter used during training may not be the same as that used for real-time decision-making. For example, in power system operation, the production costs of the generators can vary over time, and the resistance and susceptance of transmission lines can be altered both intentionally and unintentionally. These parameters are usually not known to the system operator in advance or at least not fully aware when training the forecast model. See Fig. 1 for an illustration.

Formulation

Consider the uncertainty of the input $\boldsymbol{x} + \boldsymbol{\delta}_x \in \mathcal{X}$ and the unpredictable parameter $\boldsymbol{\phi} + \boldsymbol{\delta}_{\phi} \in \Phi$. Denote $\boldsymbol{\psi} = (\boldsymbol{x}, \boldsymbol{\phi}) \in \Psi := \mathcal{X} \times \Phi$. The worst scenario, which maximizes the task-aware objective, can be formulated as

$$\begin{aligned} \max_{\boldsymbol{\psi} \in \Psi} \quad & \mathcal{L}(\hat{\boldsymbol{z}}; \boldsymbol{\theta}, \boldsymbol{\psi}, \boldsymbol{y}) \\ \text{subject to} \quad & \hat{\boldsymbol{z}} \in \mathcal{C}_{\text{E2E}}(\boldsymbol{z}; \boldsymbol{\theta}, \boldsymbol{\psi}, \boldsymbol{y}) \end{aligned} \quad (15)$$

in which $\boldsymbol{\theta}$ is fixed.

Consequently, a robust optimization can be formulated. A unified E2E adversarial training (E2E-AT) considering both input and CO uncertainties becomes

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & \sum_{i \in \mathcal{D}} \max_{\boldsymbol{\psi}^i \in \Psi^i} \mathcal{L}(\{\hat{\boldsymbol{z}}^i; \boldsymbol{\theta}, \boldsymbol{\psi}^i, \boldsymbol{y}^i) \\ \text{subject to} \quad & \hat{\boldsymbol{z}}^i \in \mathcal{C}_{\text{E2E}}(\boldsymbol{z}^i; \boldsymbol{\theta}, \boldsymbol{\psi}^i, \boldsymbol{y}^i), \quad i \in \mathcal{D} \end{aligned} \quad (16)$$

where the constrains are subject to both minimization and maximization. Adversarial training can be adopted to solve (16). Similarly to the implicit function theorem (14), the gradient of the constraint exists, regardless of the minimization or maximization. Therefore, Danskin's theorem can be used by first solving the inner maximization through gradient ascent (with $\boldsymbol{\theta}$ fixed) and then for the outer gradient descent (with $\boldsymbol{\psi}$ fixed). Although using the Danskin theorem requires one to exactly solve the inner maximization, it can give a descent direction for suboptimal $\boldsymbol{\psi}$, e.g. solved by PGD (3) and is applicable to various adversarial training algorithms (Madry et al. 2017; Zhang et al. 2019; Dong et al. 2020).

Certified Robustness

Although PGD (3) is effective for E2E-AT, it cannot verify the robustness, as it only finds the local maximum (Xiao

et al. 2018). In particular, a robustness certification on (11) verifies if an adversarial example exists within the budget Δ such that the task-aware objective is altered by a certain amount. The key is to find the exact adversarial attack in (15). We show that for specific type of objective and COs (e.g. affine-parametric QPs), optimal solution to (15) can be solved exactly, which extends the certified robustness in piecewise linear neural network (4).

Proposition 2 *The affine-parametric QP:*

$$\begin{aligned} \boldsymbol{z}_{i+1} := \quad & \arg\min_{\boldsymbol{z}} \quad \tfrac{1}{2}\boldsymbol{z}^T \boldsymbol{Q} \boldsymbol{z} + \boldsymbol{q}^T \boldsymbol{z} \\ & \text{subject to} \quad \boldsymbol{A}\boldsymbol{z} + \boldsymbol{G}\boldsymbol{z}_i \le \boldsymbol{b} \\ & \quad\quad\quad\quad\quad \boldsymbol{C}\boldsymbol{z} + \boldsymbol{H}\boldsymbol{z}_i = \boldsymbol{d} \end{aligned} \quad (17)$$

can be equivalently written as the set of mixed integer linear constraints:

$$\boldsymbol{Q}\boldsymbol{z}_{i+1} + \boldsymbol{q} + \boldsymbol{A}^T \boldsymbol{\lambda}_{i+1} + \boldsymbol{C}^T \boldsymbol{\nu}_{i+1} = 0$$

$$\boldsymbol{C}\boldsymbol{z}_{i+1} + \boldsymbol{H}\boldsymbol{z}_i - \boldsymbol{d} = \boldsymbol{0}$$

$$\boldsymbol{\lambda}_{i+1} \ge \boldsymbol{0}$$

$$\boldsymbol{A}\boldsymbol{z}_{i+1} + \boldsymbol{G}\boldsymbol{z}_i - \boldsymbol{b} \le \boldsymbol{0}$$

$$\boldsymbol{\lambda}_{i+1} \le \boldsymbol{\varphi} M, \; \boldsymbol{A}\boldsymbol{z}_{i+1} + \boldsymbol{G}\boldsymbol{z}_i - \boldsymbol{b} \ge (\boldsymbol{\varphi} - \boldsymbol{1})M, \; \boldsymbol{\varphi} \in \{0,1\}^{|\boldsymbol{\varphi}|}$$

where $\boldsymbol{Q}, \boldsymbol{q}, \boldsymbol{A}, \boldsymbol{G}, \boldsymbol{b}, \boldsymbol{C}, \boldsymbol{H}, \boldsymbol{d}$ are the parameters with proper dimensions. $\boldsymbol{\varphi}$ is binary vector and M is a large positive number. The equalities and inequalities are element-wise.

The proof can be found in the appendix.

Due to the complexity of integer programming, we restrict the original settings in (Amos and Kolter 2017) by assuming linearity in uncertain terms for certified robustness. For example, when the uncertainty in the input feature is considered, the parameter $\boldsymbol{z}_i$ that represents the optimal decision of the previous task is decoupled from the variable $\boldsymbol{z}$ and is affine so that the reformulation is linear. When the uncertainty of CO is considered, we assume that the uncertain unpredictable parameter is decoupled from the variable as well. We note that this setting follows the disciplined parametrized programming (DPP) (Agrawal et al. 2019).

Since both affine-parametric QP (17) and neural network (4) can be represented by mixed-integer linear constraints, maximizing an affine function subject to these constraints becomes mixed-integer linear programming (MILP), which can be effectively solved and used to certify the worst possible cost.

Discussion on the Robustness

Previously, the uncertainty involved in COs has been considered in many E2E learning algorithms. From the perspective of contextual optimization, E2E learning applies ML to predict the uncertain parameter (Sadana et al. 2023). In such setting, probabilistic forecast and stochastic program can be implemented (Donti, Amos, and Kolter 2017). However, we view the entire E2E learning as an integrated model such that the uncertainty of the intermediate variable can be merged within the E2E training objective. Alternatively, we separate the parameters of COs into two parts. The first part (predictable parameter) is forecasted by the ML while the

second part is unpredictable. Indirectly, the uncertainty of the predictable parameter is tackled by adversarial training on the input feature, while the uncertainty of unpredictable parameter needs to be tackled as well.

We note that the physical meaning of the uncertainty of the unpredictable parameter is different from that of the adversarial attack in the input feature space. Although the ultimate goal is to improve the robustness of the ML, the adversarial attack assumes that there exists a malicious party that can find the worst attack. However, the uncertainty of the unpredictable parameter always exists regardless of the malicious party. Inspired by the previous work (Donti et al. 2021; Agarwal et al. 2022), we can alternatively view the E2E-AT on the unpredictable parameter as the following optimization problem:

$$
\begin{aligned}
\min_{\boldsymbol{\theta}} \quad & \alpha \sum_{i \in \mathcal{D}} \mathcal{L}(\bar{\boldsymbol{z}}^i; \boldsymbol{\theta}, \boldsymbol{x}^i, \boldsymbol{y}^i, \bar{\boldsymbol{\phi}}) \\
& +(1-\alpha) \sum_{i \in \mathcal{D}} \mathbf{E}_{(\hat{\boldsymbol{z}}^i, \boldsymbol{\phi}^i)}[\mathcal{L}(\hat{\boldsymbol{z}}^i; \boldsymbol{\theta}, \boldsymbol{x}^i, \boldsymbol{y}^i, \boldsymbol{\phi}^i)] \\
\text{subject to} \quad & \bar{\boldsymbol{z}}^i \in \mathcal{C}_{\text{E2E}}(\boldsymbol{z}^i; \boldsymbol{\theta}, \boldsymbol{x}^i, \boldsymbol{y}^i, \bar{\boldsymbol{\phi}}), \ i \in \mathcal{D} \\
& \hat{\boldsymbol{z}}^i \in \mathcal{C}_{\text{E2E}}(\boldsymbol{z}^i; \boldsymbol{\theta}, \boldsymbol{x}^i, \boldsymbol{y}^i, \boldsymbol{\phi}^i), \ \boldsymbol{\phi}^i \in \Phi^i, i \in \mathcal{D}
\end{aligned}
\tag{19}
$$

where $\bar{\boldsymbol{z}}^i$ is the decision variable from the COs parameterized by the nominal unpredictable parameter $\bar{\boldsymbol{\phi}}$.

It can be argued that the NN forecaster is trained by considering the expected task-aware cost due to uncertainties. The learning objective (19) takes the nominal unpredictable parameter (denoted as $\bar{\boldsymbol{\phi}}$) and the expected uncertainties over $\boldsymbol{\Phi}^i$ into account, which are balanced by the hyperparameter α. As shown in (19), this stochastic program can be solved by sampling $\boldsymbol{\phi}^i \in \boldsymbol{\Phi}^i$ during training. In addition, if the uncertainty set of the unpredictable parameter is independent of the sample, $\boldsymbol{\Phi}$ is not subject to index i.

Final Training Objective

In E2E-AT, (19) is solved by robust optimization by finding the maximum over $\boldsymbol{\psi}^i \in \Psi^i$, as in (16). We also take the input uncertainty into account:

$$
\begin{aligned}
\min_{\boldsymbol{\theta}} \quad & \alpha \cdot \sum_{i \in \mathcal{D}} \mathcal{L}(\bar{\boldsymbol{z}}^i; \boldsymbol{\theta}, \bar{\boldsymbol{\psi}}^i, \boldsymbol{y}^i) \\
& +(1-\alpha) \cdot \sum_{i \in \mathcal{D}} \max_{\boldsymbol{\psi}^i \in \Psi^i} \mathcal{L}(\hat{\boldsymbol{z}}^i; \boldsymbol{\theta}, \boldsymbol{\psi}^i, \boldsymbol{y}^i) \\
\text{subject to} \quad & \bar{\boldsymbol{z}}^i \in \mathcal{C}_{\text{E2E}}(\boldsymbol{z}^i; \boldsymbol{\theta}, \bar{\boldsymbol{\psi}}, \boldsymbol{y}^i) \\
& \hat{\boldsymbol{z}}^i \in \mathcal{C}_{\text{E2E}}(\boldsymbol{z}^i; \boldsymbol{\theta}, \boldsymbol{\psi}^i, \boldsymbol{y}^i), \quad i \in \mathcal{D}
\end{aligned}
\tag{20}
$$

The new adversarial training objective provides an upper bound on the expectation part of (19). Meanwhile, similar to adversarial training on image tasks (Zhang et al. 2019), α can be used to balance the clean and adversarial accuracies. When $\alpha \to 1$, (20) becomes the original E2E learning and when $\alpha \to 0$, it becomes pure adversarial training. In addition, clean and adversarial training losses may not directly reflect clean and robust accuracy for image tasks, causing an unbalanced training objective. In E2E-AT, the two objectives are defined by the task, which is the exact metric during decision making.

Previously in (Donti et al. 2021; Agarwal et al. 2022), the authors reformulate a bilevel optimization problem into robust optimization and use the implicit function theorem for constraint satisfaction. Connecting (19) to (20), we extend (Donti et al. 2021) to training ML models. It can be seen

that for each mini-batch in E2E-AT, a similar robust optimization is solved as (Donti et al. 2021). The implicit function theorem is also used to learn the task-aware objective while satisfying the constraints.

'Free' E2E Adversarial Training

In E2E-AT, the number of forward and backward passes is equal to `no_batch`×(`no_pgd`+1)×`no_epoch` (assuming that all minibatches have the same size). Adversarial training is computationally ineffective due to intensive backpropagation (controlled by the complexity of the neural network). The computational burden is even higher in E2E-AT as in each forward pass, the COs need to be solved (controlled by the complexity of the COs). To save training time, we adopt the gradient reuse strategy. In the *adversarial training for free* (Shafahi et al. 2019), the attack vector and the model parameter are repeatedly updated in the same mini-batch for `no_pgd` times. Then, `epoch_no` is divided by `no_pgd` to maintain the total number of model updates unchanged. This results in `no_batch` × `no_epoch` numbers of forward and backward passes, which is the same as the clean E2E training.

Experiment

In the experiment, the NN is trained to forecast the load in the power system. The robustness of various E2E-AT settings is explored. Detailed experiment settings and results can be found in the Appendix.

Power System Operations

A practical power system operation problem, named as network constrained economic dispatch (NCED), is considered, which has been widely used in the US and can be formulated as two-stage QP or LP (Conejo and Baringo 2018). In stage one (also known as dispatch), the set points of the generator are determined based on the **forecast** load. The goal of the first stage is to minimize the generator cost while meeting the physical constraints of the grid. When the generator has been dispatched, we consider a realization on the actual load by solving the second stage problem (also known as redispatch), in which any mismatches on the load and generation from stage one, as well as the violation of the physical constraints of the grid, will be penalized by extra cost:

$$
\boldsymbol{P}_g^\star = \text{Dispatch}(\boldsymbol{f}(\boldsymbol{x}; \boldsymbol{\theta}), \boldsymbol{b})
\tag{21a}
$$

$$
\boldsymbol{P}_{ls}^\star, \boldsymbol{P}_{gs}^\star = \text{Redispatch}(\boldsymbol{y}, \boldsymbol{P}_g^\star, \boldsymbol{b})
\tag{21b}
$$

where $\boldsymbol{P}_g^\star$ is the optimal generator dispatch, $\boldsymbol{P}_{ls}^\star$ is the load shedding and $\boldsymbol{P}_{gs}^\star$ is the power storage. $\boldsymbol{b}$ is the susceptance of the transmission line (when the resistance is close to zero, the susceptance is reciprocal to the reactance). The task-aware objective is defined as

$$
\mathcal{L}(\boldsymbol{\theta}) = \boldsymbol{c}_g^T \boldsymbol{P}_g + \boldsymbol{c}_{ls}^T \boldsymbol{P}_{ls} + \boldsymbol{c}_{gs}^T \boldsymbol{P}_{gs}
\tag{22}
$$

where $\boldsymbol{c}_g$, $\boldsymbol{c}_{ls}$, and $\boldsymbol{c}_{gs}$ are the coefficients such that $\boldsymbol{c}_{ls} \gg \boldsymbol{c}_{gs} \gg \boldsymbol{c}_g$. That is, the load shedding is more costly.

Training Method			Clean	Input Attack, ϵ_x			CO Attack, ϵ_ϕ			Integrated Attack, $(\epsilon_x, \epsilon_\phi)$		
ϵ_x	ϵ_ϕ	α	N/A	0.01	0.02	0.03	0.05	0.10	0.15	(0.01,0.05)	(0.02,0.10)	(0.03,0.15)
NAT: Natural Training with Task Loss												
0	0	0	201.6	653.6	1188.5	1792.1	754.6	2044.3	3468.4	1153.9	3090.8	5214.4
AT-MSE: Adversarial Training with MSE Loss												
0.02	N/A	N/A	396.5	676.6	978.3	1278.4	1223.8	2576.7	3923.6	1475.4	3120.1	4778.3
0.03	N/A	N/A	437.0	653.6	886.3	1125.5	1208.8	2517.2	3881.6	1410.5	2956.5	4624.0
AT-INPUT: Adversarial Training with Task Loss on the Input Uncertainty												
0.02	0	1.0	219.1	445.2	745.0	1106.1	672.4	1929.0	3334.02	934.4	2706.3	4854.5
		0.5	203.1	438.9	790.1	1246.7	660.7	1932.7	3349.6	914.8	2763.9	4930.5
0.03	0	1.0	239.4	448.6	705.0	1022.1	638.7	1787.5	3220.6	855.5	2445.9	4507.0
		0.5	227.6	556.8	976.6	1435.3	783.7	2074.9	3503.1	1106.8	2948.5	4999.8
AT-PARA: Adversarial Training with Task Loss on the CO Uncertainty												
0	0.05	1.0	212.3	398.84	726.8	1197.4	214.9	590.1	1931.3	399.1	939.1	2620.0
		0.5	206.7	419.3	798.2	1299.9	215.6	741.2	2120.4	419.8	1114.8	3072.6
0	0.15	1.0	214.0	488.7	900.1	1434.5	214.0	221.0	453.2	493.0	925.8	1502.0
		0.5	216.5	423.5	804.6	1309.8	216.5	223.1	465.6	429.6	817.8	1324.6
AT-BOTH: Adversarial Training with Task Loss on the Integrated Uncertainties												
0.02	0.05	1.0	228.2	399.5	655.7	992.0	254.2	823.5	2156.7	400.5	1128.7	2956.2
		0.5	212.1	490.1	879.9	1336.5	289.4	1257.9	2711.8	516.0	1793.9	3903.8
0.03	0.15	1.0	274.7	551.0	856.5	1226.3	274.7	279.0	495.7	550.6	883.4	1274.0
		0.5	239.2	429.1	666.3	977.6	239.2	250.8	508.3	429.0	667.6	1053.7

Table 1: Performances of the E2E-AT.

Training Settings

We use an open source load forecasting dataset from the Texas Backbone Power System (Lu et al. 2023) on a modified IEEE bus-14 system. We randomly collect 1.0k samples and use a feedforward neural network with three hidden layers to forecast the load[‡]. We do E2E-AT with **a).** input feature uncertainties δ_x, **b).** uncertainty of the unpredictable parameters δ_ϕ, and **c).** integrated uncertainties of both (δ_x, δ_ϕ). We first implement natural (or clean) E2E learning, based on which we warm-start the E2E-ATs. Adam optimizer is used, and 'Adversarial training for free' (Shafahi et al. 2019) is applied to reuse the gradients for PGD.

In detail, **a).** Since meteorological features have been normalized into $[0,1]$, we attack with a normalized budget $\epsilon_x \in \{0.02, 0.03\}$. The inner maximization is solved with 7 PGD steps and the step size is dynamically set as $\epsilon_x/7 \times 2$. We summarize this setting from (Zhang et al. 2019). We clamp the attacked input into $[0,1]$ whenever it is updated. **b).** We consider uncertainties on the susceptance b in the **redispatch** problem. Since each transmission line can have different nominal susceptance, we set the budget $\epsilon_\phi \in \{0.05, 0.15\}$ as the proportion to the individual nominal value, which is consistent with the common operation range of susceptance (Xu, Jaimoukha, and Teng 2022). **c).** We do E2E-AT with $(\epsilon_x, \epsilon_\phi) \in \{(0.02, 0.05), (0.03, 0.15)\}$ for the integrated uncertainties. The other settings are the same as in (a) and (b).

Performance of E2E-AT

Multi-run adversarial attacks are evaluated in Table 1. For each sample and attack scenario, we randomly select three

---

[‡]More experiment can be found in the appendix.

starting points within the attack budget and report the **worst** task-aware cost (22) to reduce the variance. Specifically, five training algorithms are compared. **NAT**: natural training with task-aware loss; **AT-MSE**: adversarial training with MSE loss; **AT-INPUT**: adversarial training with task-aware loss and input uncertainties; **AT-PARA**: adversarial training with task-aware loss and unpredictable CO uncertainties; and **AT-BOTH**: adversarial training with task-aware loss and integrated uncertainties.

We first highlight that uncertainties in COs can significantly increase the cost, e.g. 15 times higher than the clean cost when $\epsilon_\phi = 0.15$ for NAT. The AT-MSE performs better for input attacks, compared to the NAT, but performs poorly on the CO attacks. This is because AT-MSE is only trained with input uncertainties. Second, the performance of E2E-AT is similar to conventional adversarial training for image classification (Madry et al. 2017; Zhang et al. 2019). For instance, training with a larger attack budget can result in better robustness on attacks with a smaller budget but can inevitably increase the clean cost. Meanwhile, the hyperparameter α gives a trade-off between clean and robust accuracy in most cases.

In addition to common findings on adversarial robustness of image tasks, some unique findings of E2E-AT are highlighted. First, AT-INPUT and AT-PARA are more effective in the uncertainty with which they are trained. However, it is observed that E2E-AT based on one source of uncertainty can also improve the robustness of the other. For example, in AT-PARA, the cost of the input attack is even lower than that trained by AT-INPUT, when ϵ_x is small. Moreover, AT-PARA is more effective in the integrated attack than AT-INPUT. In fact, any input uncertainty eventually feeds into the COs, which becomes the uncertainties of the predictable parameters in COs. Finally, AT-BOTH not only improves
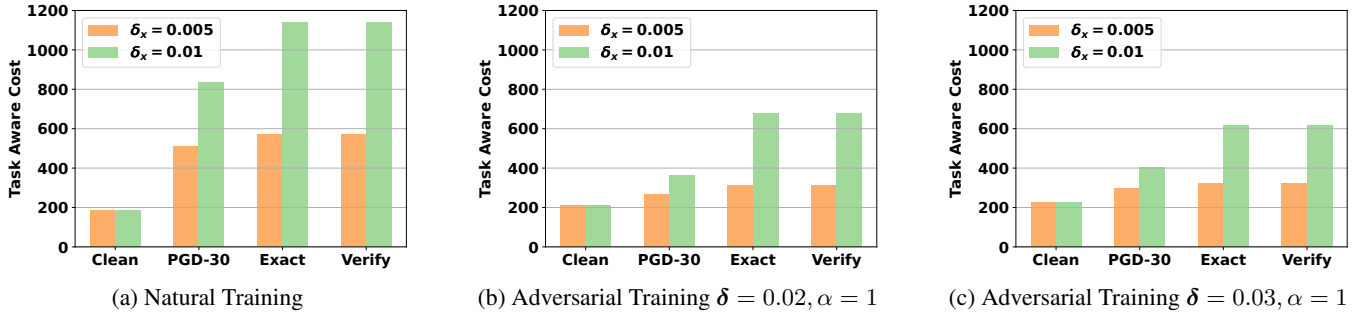
Figure 2: Input space adversarial attack using the exact MILP formulation. 'Clean': cost of the clean sample; 'PGD-30': cost of adversarial sample found by 30 PGD steps; 'Exact': cost of adversarial sample found by exact MILP; 'Verify': cost of the adversarial sample by feeding the MILP solutions into the dispatch and redispatch problems.

Training Method	Clean	$\epsilon = 0.05$		$\epsilon = 0.1$		$\epsilon = 0.15$	
		Random	PGD-7	Random	PGD-7	Random	PGD-7
NAT	201.6	267.2	754.6	396.6	2044.3	655.5	3468.4
AT-PARA $\epsilon_\phi = 0.05, \alpha = 1.0$	212.3	212.3	214.9	218.4	590.1	313.3	1931.3
AT-PARA $\epsilon_\phi = 0.05, \alpha = 0.5$	206.7	206.7	215.6	222.6	741.2	313.1	2020.4
AT-PARA $\epsilon_\phi = 0.15, \alpha = 1.0$	214.0	214.0	214.0	214.0	221.0	222.9	453.2
AT-PARA $\epsilon_\phi = 0.15, \alpha = 0.5$	216.5	216.5	216.5	216.5	223.1	219.4	465.6

Table 2: CO uncertainties with random sampling of b. The adversarial attack reported in Table 1 is denoted as PGD-7.

the robustness of integrated uncertainty, but improves the robustness of the individual's. All of the findings demonstrate that the uncertainties of both sources can be treated together in a unified way.

Certified Robustness

Certification on the robustness of the input space is considered, as susceptance b is coupled with the decision variable and Proposition 2 is not applicable. Using Proposition 2 and the mixed integer reformulation of NN (4), the exact input space adversarial attack on sample (x, y) can be found by a mixed integer linear program (MILP). Interval bound propagation (IBP) (Gowal et al. 2018) is used to estimate the bounds of layers in NN. As for COs, we set $M = 10^5$ by experience. Due to the large computation burden, we randomly sample 15 **same** samples and solve the robust certification using `Gurobi`.

First, the *branch-and-bound* algorithm is applied whose optimality is guaranteed if feasible. Meanwhile, the MILP formulation is verified as the same task-aware cost is achieved when solving the downstream COs parameterized by the optimal attack vector. Second, it can be observed that the exact attack vector causes the worse cost degradation, compared to the PGD-30 attacks. Finally, AT-INPUT can effectively reduce the task-aware cost on the exact input space attack.

Parameter Uncertainties in COs

As shown in (19), the uncertainties of CO can be modeled by stochastic CO. To model stochastic susceptance b, we randomly alter the susceptance (random attack) for each sample and report the task-aware cost in Table 2. Under the random attack, the task-aware cost of NAT increases, clearly

demonstrating the need for adversarial training. After E2E-AT, task-aware cost under random attacks is significantly reduced and is equal to the corresponding clean cost when the attack budget is small. Although the clean cost increases, it is still lower than the average cost under random attacks. The task-aware cost is also empirically upper bound by the adversarial attack (e.g. PGD-7), which verifies our argument on connecting the stochastic COs with E2E-AT (16) and (20).

Conclusion

This paper proposes a unified framework for tackling uncertainties in task-aware E2E learning. We argue that the uncertainties occur at both the input feature of ML and the unpredictable parameter of COs. A robust program is formulated, which is practically solved by adversarial training (E2E-AT). Through theoretical analysis and experiment, we demonstrate that 1). the CO uncertainty can cause significant generalization degradation which has been overlooked before; 2). The optimal adversarial attack on affine-parametric QP can be found by solving the mixed integer (linear) program; and 3). adversarial training can effectively improve the robustness of E2E learning in a unified way.

Acknowledgments

References

Agarwal, A.; Donti, P. L.; Kolter, J. Z.; and Pileggi, L. 2022. Employing adversarial robustness techniques for large-scale stochastic optimal power flow. *Electric Power Systems Research*, 212: 108497.

Agrawal, A.; Amos, B.; Barratt, S.; Boyd, S.; Diamond, S.; and Kolter, J. Z. 2019. Differentiable convex optimization layers. *Advances in neural information processing systems*, 32.

Amos, B.; and Kolter, J. Z. 2017. Optnet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning*, 136–145. PMLR.

Bengio, Y. 1997. Using a financial training criterion rather than a prediction criterion. *International journal of neural systems*, 8(04): 433–443.

Boyd, S. P.; and Vandenberghe, L. 2004. *Convex optimization*. Cambridge university press.

Chen, B.; Donti, P. L.; Baker, K.; Kolter, J. Z.; and Bergés, M. 2021. Enforcing policy feasibility constraints through differentiable projection for energy optimization. In *Proceedings of the Twelfth ACM International Conference on Future Energy Systems*, 199–210.

Conejo, A. J.; and Baringo, L. 2018. *Power system operations*, volume 11. Springer.

Domke, J. 2012. Generic methods for optimization-based modeling. In *Artificial Intelligence and Statistics*, 318–326. PMLR.

Dong, Y.; Deng, Z.; Pang, T.; Zhu, J.; and Su, H. 2020. Adversarial distributional training for robust deep learning. *Advances in Neural Information Processing Systems*, 33: 8270–8283.

Donti, P.; Agarwal, A.; Bedmutha, N. V.; Pileggi, L.; and Kolter, J. Z. 2021. Adversarially robust learning for security-constrained optimal power flow. *Advances in Neural Information Processing Systems*, 34: 28677–28689.

Donti, P.; Amos, B.; and Kolter, J. Z. 2017. Task-based end-to-end model learning in stochastic optimization. *Advances in neural information processing systems*, 30.

Elmachtoub, A. N.; and Grigas, P. 2022. Smart "predict, then optimize". *Management Science*, 68(1): 9–26.

Gowal, S.; Dvijotham, K.; Stanforth, R.; Bunel, R.; Qin, C.; Uesato, J.; Arandjelovic, R.; Mann, T.; and Kohli, P. 2018. On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv preprint arXiv:1810.12715*.

Kong, L.; Cui, J.; Zhuang, Y.; Feng, R.; Prakash, B. A.; and Zhang, C. 2022. End-to-end stochastic optimization with energy-based model. *Advances in Neural Information Processing Systems*, 35: 11341–11354.

Kotary, J.; Fioretto, F.; Van Hentenryck, P.; and Wilder, B. 2021. End-to-end constrained optimization learning: A survey. *arXiv preprint arXiv:2103.16378*.

Krantz, S. G.; and Parks, H. R. 2002. *The implicit function theorem: history, theory, and applications*. Springer Science & Business Media.

Liu, Z.; Yin, Y.; Bai, F.; and Grimm, D. K. 2023. End-to-end learning of user equilibrium with implicit neural networks. *Transportation Research Part C: Emerging Technologies*, 150: 104085.

Lu, J.; Li, X.; Li, H.; Chegini, T.; Gamarra, C.; Yang, Y.; Cook, M.; and Dillingham, G. 2023. A Synthetic Texas Backbone Power System with Climate-Dependent Spatio-Temporal Correlated Profiles. *arXiv preprint arXiv:2302.13231*.

Luo, Z.-Q.; Pang, J.-S.; and Ralph, D. 1996. *Mathematical programs with equilibrium constraints*. Cambridge University Press.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Sadana, U.; Chenreddy, A.; Delage, E.; Forel, A.; Frejinger, E.; and Vidal, T. 2023. A Survey of Contextual Optimization Methods for Decision Making under Uncertainty. *arXiv preprint arXiv:2306.10374*.

Shafahi, A.; Najibi, M.; Ghiasi, M. A.; Xu, Z.; Dickerson, J.; Studer, C.; Davis, L. S.; Taylor, G.; and Goldstein, T. 2019. Adversarial training for free! *Advances in Neural Information Processing Systems*, 32.

Stratigakos, A.; Camal, S.; Michiorri, A.; and Kariniotakis, G. 2022. Prescriptive trees for integrated forecasting and optimization applied in trading of renewable energy. *IEEE Transactions on Power Systems*, 37(6): 4696–4708.

Tjeng, V.; Xiao, K.; and Tedrake, R. 2017. Evaluating robustness of neural networks with mixed integer programming. *arXiv preprint arXiv:1711.07356*.

Vohra, R.; Rajaei, A.; and Cremer, J. L. 2023. End-to-end learning with multiple modalities for system-optimised renewables nowcasting. *arXiv preprint arXiv:2304.07151*.

Wilder, B.; Dilkina, B.; and Tambe, M. 2019. Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 1658–1665.

Xiao, K. Y.; Tjeng, V.; Shafiullah, N. M.; and Madry, A. 2018. Training for faster adversarial robustness verification via inducing relu stability. *arXiv preprint arXiv:1809.03008*.

Xu, W.; Jaimoukha, I. M.; and Teng, F. 2022. Robust moving target defence against false data injection attacks in power grids. *IEEE Transactions on Information Forensics and Security*, 18: 29–40.

Xu, W.; and Teng, F. 2023. Availability Adversarial Attack and Countermeasures for Deep Learning-based Load Forecasting. In *2023 IEEE Belgrade PowerTech*, 01–06.

Zhang, H.; Yu, Y.; Jiao, J.; Xing, E.; El Ghaoui, L.; and Jordan, M. 2019. Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning*, 7472–7482. PMLR.