End-to-End Learning for Stochastic Optimization: A Bayesian Perspective

Yves Rychener [1] **Daniel Kuhn** [1] **Tobias Sutter** [2]

Abstract

We develop a principled approach to end-to-end learning in stochastic optimization. First, we show that the standard end-to-end learning algorithm admits a Bayesian interpretation and trains a posterior Bayes action map. Building on the insights of this analysis, we then propose new end-to-end learning algorithms for training decision maps that output solutions of empirical risk minimization and distributionally robust optimization problems, two dominant modeling paradigms in optimization under uncertainty. Numerical results for a synthetic newsvendor problem illustrate the key differences between alternative training schemes. We also investigate an economic dispatch problem based on real data to showcase the impact of the neural network architecture of the decision maps on their test performance.

1. Introduction

Most practical decision problems can be framed as stochastic optimization models that minimize the expected value of a loss function impacted by one's decisions and by an exogenous random variable Y. Stochastic optimization techniques are routinely used, for example, in portfolio selection (Markowitz & Todd, 2000) or economic dispatch and pricing (Cournot, 1897; Wong & Fuller, 2007) among many other areas. However, the probability distribution $\mathbb{P}_Y$ of the uncertain problem parameter Y is generically unknown and needs to be inferred from finitely many training samples $\widehat{Y}_1, \ldots, \widehat{Y}_N$. In *empirical risk minimization* (ERM), $\mathbb{P}_Y$ is simply replaced with the empirical (uniform) distribution on the given samples. Unfortunately, ERM is susceptible to overfitting, that is, it leads to decisions that exploit artefacts of the training samples but perform poorly on test data. This effect is also referred to as the "optimizer's curse" in decision theory (Smith & Winkler, 2006). Various regularization techniques have been proposed to combat this effect. *Distributionally robust optimization* (DRO) (Delage & Ye, 2010; Wiesemann et al., 2014), for example, seeks decisions that are worst-case optimal in view of a large family of distributions that could have generated the training samples. Alternatively, any additional information or beliefs about $\mathbb{P}_Y$ may be used as a prior that is updated upon observing training data like in Bayesian estimation. The use of prior information also has a regularizing effect, and the resulting optimal decision is referred to as the *posterior Bayes action* (Berger, 2013). ERM and DRO implicitly assume that *independent* samples from $\mathbb{P}_Y$ form the *only* source of information available to the decision-maker. However, this assumptions often fails to hold in practice. Financial asset returns are not stationary, and their distribution is correlated with slowly varying macroeconomic factors (Li, 2002). Similarly, the distribution of wind energy production levels depends on meteorological conditions and is strongly correlated with wind speeds. *Contextual stochastic programs* (Bertsimas & Kallus, 2020) exploit contextual information such as macroeconomic factors or wind speeds to inform decision-making. Existing approaches to contextual stochastic optimization can be grouped into two categories. *Predict-then-optimize* approaches (see, e.g., (Mišić & Perakis, 2020) for a survey) first use some method from statistics or machine learning to learn a parametric or non-parametric model of the distribution $\mathbb{P}_Y$. In a second step, the learned distribution model is used in an optimization model to compute a decision. A key drawback of this approach is that the machine learning method used for predicting $\mathbb{P}_Y$ (e.g., maximum likelihood estimation) is agnostic of the downstream optimization model. Some ideas to remedy this shortcoming are discussed in (Elmachtoub & Grigas, 2022). *End-to-end* learning approaches, on the other hand, train a decision map directly from the available data without the detour of first estimating a model for $\mathbb{P}_Y$ (Donti et al., 2017).

In this work, we investigate the usage of neural networks in end-to-end learning. The relation between a trained neural network and the corresponding Bayes optimal classifier (Baum & Wilczek, 1987; Wan, 1990; Papoulis & Pillai, 2002; Kline & Berardi, 2005) as well as the universal approximation capabilities of neural networks (Cybenko, 1989; Lu et al., 2017) are well understood in traditional

[1]Risk Analytics and Optimization Chair, École Polytechnique Fédérale de Lausanne, Switzerland [2]Department of Computer and Information Science, University of Konstanz, Germany. Correspondence to: Yves Rychener <yves.rychener@epfl.ch>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

regression and classification. In the context of end-to-end learning, however, similar results are lacking. In this work, we aim to close this gap.

Related Work: The concept of *end-to-end learning* for stochastic optimization was first introduced by Donti et al. (2017). Instead of minimizing a generic loss function such as the mean-square error, end-to-end learning directly minimizes the task loss over a class of neural networks that embed the underlying stochastic optimization model in their architecture. Despite their unorthodox structure, such neural networks can be differentiated by exploiting the Karush-Kuhn-Tucker optimality conditions of the embedded optimization model (Agrawal et al., 2019). Hence, they remain amenable to gradient-based training schemes. More recent approaches to end-to-end learning relax the requirement that the neural network must contain an optimization layer and use simpler architectures to approximate the decision map (Zhang et al., 2020; Butler & Kwon, 2021; Uysal et al., 2021a). This paper complements these efforts. Instead of introducing new algorithms to improve performance, we aim to advance our theoretical understanding of the existing algorithms and extend them to broader problem classes.

The term *end-to-end* is used slightly differently across different machine learning communities. We adopt the same convention as Donti et al. (2017) whereby a decision model is "end-to-end" if it is trained on the task loss. In this case the feature extractor and prescriptor are trained jointly and directly on the task loss of interest. In traditional deep learning, on the other hand, end-to-end learning usually refers to the training of a deep network without hand-crafted features that processes the data input in its original format, such as audio spectrograms (Amodei et al., 2016), images (Wang et al., 2011; He et al., 2016) or text (Wang et al., 2017).

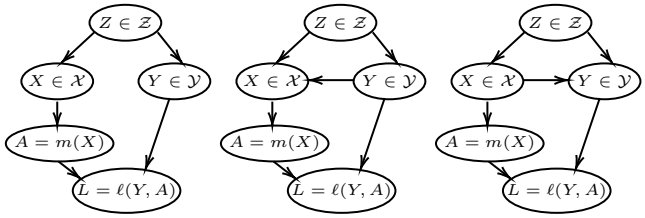
End-to-end learning is also related to *reinforcement learning*, which seeks a policy that maps an observable state to an action in a *dynamic* decision-making context. Reinforcement learning agents have been successfully trained to play board games (Silver et al., 2017; 2018) as well as video games (Vinyals et al., 2019) or to steer self-driving cars (Kiran et al., 2021). However, a key distinguishing feature of reinforcement learning applications is their dynamic nature. The decision-maker interacts with an unknown environment over multiple periods, and the action chosen in a particular period affects the state of the environment in the next period. The decision-maker thus seeks an action that not only incurs a small loss in the current period but also leads to a favorable state in the next period. In contrast, end-to-end learning focuses on *static* decision problems. We highlight that end-to-end learning is also related to *contextual bandits* (Chu et al., 2011; Lattimore & Szepesvàri, 2020), where an agent sequentially chooses among a finite set of actions, whose expected loss or payoff depends on an unknown distribution conditioned on an observable context. End-to-end learning differs from contextual bandits and reinforcement learning in that training is performed offline. Only the immediate loss after training is relevant, which eliminates the notorious exploration-exploitation trade-off (Audibert et al., 2009; Graves & Jaitly, 2014).

Contributions: Our contributions are summarized below.
- We develop a general and versatile modeling framework for end-to-end learning in stochastic optimization.
- We show that the widely used standard algorithm for end-to-end learning outputs a posterior Bayes action map.
- Leveraging the insights of our Bayesian analysis, we propose new end-to-end learning algorithms for training decision maps that output solutions of ERM and DRO problems.
- We show that existing universal approximation results for neural networks extend to decision maps of end-to-end learning models with projection and optimization layers.
- We experimentally compare different approaches to end-to-end learning and different types of contextual information in the framework of a newsvendor problem with synthetic data and an economic dispatch problem with real data.

Notation: All random objects are defined on an abstract probability space $(\Omega, \mathcal{F}, \mathbb{P})$, and $\mathbb{E}[\cdot]$ denotes the expectation with respect to $\mathbb{P}$. Random objects are denoted by capital letters and their realizations by the corresponding lowercase letters. Given two random vectors X and Y, we use $\mathbb{P}_Y$ and $\mathbb{P}_{Y|X}$ to denote the marginal distribution of Y and the conditional distribution of Y given X, respectively.

2. End-to-End Learning

We consider a decision problem impacted by a random vector $Y \in \mathcal{Y}$, and we assume that the decision-maker has access to an observation $X \in \mathcal{X}$ that provides information about the distribution of Y. In order to express all possible causal relationships between Y and X, we further assume that there is an unobservable confounder $Z \in \mathcal{Z}$. For example, Z could represent a parameter that uniquely determines the joint distribution $\mathbb{P}_{(X,Y)}$ of X and Y. All Bayesian network structures of interest are shown in Figure 1. In Figure 1a, X and Y have the same parent and no cross influence. This is the case, for instance, if, conditional on Z, $X = [\widehat{Y}_1, \ldots, \widehat{Y}_N]$ consists of multiple independent and identically distributed (i.i.d.) copies of Y. In Figure 1b, there is an additional direct causal link from Y to X. This is the case, for instance, if X represents a noisy measurement of Y. Finally, in Figure 1c, there is an additional direct causal link from X to Y. This is the case, for instance, if Y represents the current state and X the previous state of a Markov chain or if X captures contextual information.



(a) Common Parent (b) Dependent X (c) Influential X

Figure 1. Bayesian networks visualizing the possible relationships between Y and X. The decision map is denoted by m, the loss function by ℓ. Both the decision A and loss L are random variables because they are deterministic mappings of random variables.

The decision maker aims to solve the stochastic program

$$\min_{a \in \mathcal{A}} \mathbb{E}[\ell(Y, a)|X]. \tag{1}$$

This problem minimizes the expected value of a differentiable, bounded loss function $\ell : \mathcal{Y} \times \mathcal{A} \to \mathbb{R}$, which depends on the uncertain problem parameter Y and the decision $a \in \mathcal{A}$, conditional on the observation X. The objective function of (1) is commonly referred to as the Bayesian posterior loss (Berger, 2013, Definition 8). In an energy dispatch problem, for example, $Y \in \mathcal{Y} = \mathbb{R}$ denotes the uncertain wind energy production, $a \in \mathcal{A} = \times_{j=1}^{J}[0, \bar{a}_j]$ represents the energy outputs of J conventional generators with respective capacities $\bar{a}_j$, $j = 1, \ldots, J$, and

$$\ell(Y, a) = c^\top a + p \cdot \max\left\{d - Y - \textstyle\sum_{j=1}^{J} a_j, 0\right\}$$

captures the production cost $c^\top a$ of the generators and a penalty for unmet demand. Here, d stands for the total demand, $Y + \sum_{j=1}^{J} a_j$ represents the total energy production, and $p > 0$ is a prescribed penalty parameter. In this example, the observation X can have different meanings.

Figure 1a: Assume that the confounder $Z \sim \mathcal{N}(\mu_Z, \sigma_Z^2)$ represents the (unknown) mean of $Y \sim \mathcal{N}(Z, \sigma_Y^2)$. In this case $X = [\widehat{Y}_1, \ldots, \widehat{Y}_N]$ may represent a collection of N historical wind production levels. If $\widehat{Y}_n \sim \mathcal{N}(Z, \sigma_Y^2)$, $n = 1, \ldots, N$, are i.i.d. samples from $\mathbb{P}_{Y|Z}$, then one can show that the posterior distribution of Y given X is

$$\mathcal{N}\left(\left(\tfrac{\mu_Z}{\sigma_Z^2} + \tfrac{\sum_{n=1}^{N} \widehat{Y}_n}{\sigma_Y^2}\right)/(\tfrac{1}{\sigma_Z^2} + \tfrac{N}{\sigma_Y^2}), \sigma_Y^2 + (\tfrac{1}{\sigma_Z^2} + \tfrac{N}{\sigma_Y^2})^{-1}\right).$$

Figure 1b: The wind power $Y \sim \mathcal{N}(\mu_Y, \sigma_Y^2)$ may be indirectly observable through the output $X \sim \mathcal{N}(Y, \sigma_X^2)$ of a noisy power meter. Hence, the posterior distribution of Y given X is $\mathcal{N}(\mu_Y + \sigma_Y^2(\sigma_Y^2 + \sigma_X^2)^{-1}(X - \mu_Y), \sigma_Y^2 - \sigma_Y^4(\sigma_Y^2 + \sigma_X^2)^{-1})$, and Y becomes observable as σ_X^2 drops.

Figure 1c: The observation X could represent the wind speed at a nearby location, which has a causal impact on Y. Exploiting such contextual information leads to better decisions. Assuming normality, the posterior distribution of Y given X adopts a similar form as above. Details are omitted.

By the interchangeability principle (Rockafellar & Wets, 2009, Theorem 14.60), problem (1) is equivalent to

$$\min_{m \in \mathcal{M}} \mathbb{E}[\ell(Y, m(X)], \tag{2}$$

where $\mathcal{M}$ denotes the space of all measurable decision maps m from $\mathcal{X}$ to $\mathcal{A}$. Specifically, $m^\star$ solves (2) if and only if $a^\star = m^\star(X)$ solves (1) for every realization of X.

We assume from now on that the joint distribution of X, Y and Z is unknown. In the special situation displayed in Figure 1a, a decision map feasible in (2) can be computed from the observation X alone if $X = [\widehat{Y}_1, \ldots, \widehat{Y}_N]$ consists of sufficiently many i.i.d. samples from $\mathbb{P}_{Y|Z}$. Indeed, $m(X)$ can be defined as a solution of the ERM problem

$$\min_{a \in \mathcal{A}} \tfrac{1}{N} \textstyle\sum_{i=1}^{N} \ell(\widehat{Y}_i, a). \tag{3}$$

Alternatively, $m(X)$ can be defined as a solution of the DRO problem (Delage & Ye, 2010; Wiesemann et al., 2014)

$$\min_{a \in \mathcal{A}} \max_{\mathbb{Q} \in \mathcal{U}(X)} \textstyle\int_{\mathcal{Y}} \ell(y, a)\, \mathrm{d}\mathbb{Q}(y), \tag{4}$$

where $\mathcal{U}(X)$ represents an ambiguity set, that is, a family of distributions $\mathbb{Q}$ of Y that are sufficiently likely to have generated the samples $\widehat{Y}_1, \ldots, \widehat{Y}_N$. Popular choices of the ambiguity set $\mathcal{U}(X)$ are surveyed in (Rahimian & Mehrotra, 2019). However, both ERM and DRO do not readily extend to the situations depicted in Figures 1b and 1c.

In the remainder of the paper we assume that we have access to i.i.d. training samples $\{(X_k, Y_k)\}_{k=1}^{K}$ (recall that the confounder Z is *not* observable). A feasible decision map can then be obtained via the *predict-then-optimize* approach, which trains a regression model to predict Y from X and defines $m(X)$ as an action that minimizes the loss of the prediction, see, e.g., (Mišić & Perakis, 2020). The resulting decisions are tailored to the point prediction at hand but may incur high losses when Y deviates from its prediction. In addition, the regression model used for the prediction is usually agnostic of the downstream optimization model (1); a notable exception being (Elmachtoub & Grigas, 2022). Instead of predicting Y from X, one can use machine learning methods to predict the conditional distribution $\mathbb{P}_{Y|X}$ of Y given X and define $m(X)$ as a solution of (1) under the estimated distribution (Bertsimas & Kallus, 2020). However, the methods that are used for estimating $\mathbb{P}_{Y|X}$ are again agnostic of the downstream optimization model (1).

All approaches reviewed so far have the shortcoming that evaluating $m(X)$ necessitates the solution of a potentially large optimization problem. In contrast, *end-to-end learning*

(Donti et al., 2017; Fu et al., 2018; Agrawal et al., 2019; Uysal et al., 2021b; Zhang et al., 2021) trains a parametric decision map m that is near-optimal in (2) *without* the detour of first estimating Y or its distribution. This is achieved by applying stochastic gradient descent (SGD) directly to (2). Thus, end-to-end learning *(i)* avoids the artificial separation of estimation and optimization characteristic for competing methods, *(ii)* enjoys high scalability because the decision map is trained using SGD and can be evaluated efficiently without solving any optimization model, and *(iii)* can even handle unstructured observations X such as text or images.

In the following, we describe several key aspects of an end-to-end learning model. In Section 3 we discuss neural network architectures that lend themselves to representing decision maps. In Section 4 we then review a popular SGD-based training method and prove that the resulting decision map approximately minimizes the Bayesian posterior loss $\mathbb{E}[\ell(Y,a)|X]$ under the prior $\mathbb{P}_{(X,Y)}$ and the observation X.

Remark 2.1 (Generalized Data Sets). All results of this paper extend to training sets of the form $\{(X_k, \widehat{\mathbb{P}}_k)\}_{k=1}^K$, where $\widehat{\mathbb{P}}_k$ represents an unbiased estimator for the conditional distribution $\mathbb{P}_{Y|X_k}$ in the sense that

$$\mathbb{E}[\textstyle\int_{\mathcal{Y}} \ell(y,a)\,\mathrm{d}\widehat{\mathbb{P}}_k(y)|X_k] = \mathbb{E}[\ell(Y,a)|X_k]\;\forall k = 1,\ldots,K.$$

Note that if (X_k, Y_k) is sampled from $\mathbb{P}_{(X,Y)}$, for example, then the Dirac distribution $\widehat{\mathbb{P}}_k = \delta_{Y_k}$ constitutes an unbiased estimator for $\mathbb{P}_{Y|X_k}$. Hence, the dataset $\{(X_k, \widehat{\mathbb{P}}_k)\}_{k=1}^K$ strictly generalizes the standard dataset $\{(X_k, Y_k)\}_{k=1}^K$.

3. Model Architecture

A fundamental design choice for end-to-end learning models is the architecture of the decision map $m : \mathcal{X} \to \mathcal{A}$. Throughout this section, we represent m as a neural network obtained by combining a feature extractor $f : \mathcal{X} \to \mathcal{R}$ with a prescriptor $p : \mathcal{R} \to \mathcal{A}$, where $\mathcal{R}$ denotes the feature space. The complete network is thus given by $m = p \circ f$. Below we review possible choices for both f and p.

3.1. Feature Extractor

As in classical machine learning tasks, the choice of the architecture for the feature extractor is mainly informed by the data format and by the desired symmetry and invariance properties. Thus, the feature extractor may include linear layers (Rumelhart et al., 1986), convolutional layers (Zhang et al., 1988; LeCun et al., 1989), attention layers (Vaswani et al., 2017) and recurrent layers (Elman, 1990; Hochreiter & Schmidhuber, 1997; Cho et al., 2014) etc., combined with activation functions and regularization layers.

3.2. Prescriptor

We propose three different architectures for the prescriptor. **(A) Multi-Layer Perceptron (MLP).** Uysal et al. (2021b) use classical neural network layers for the prescriptor. In this case the decision map reduces to a MLP, which enjoys great expressive power thanks to various universal approximation theorems; see (Cybenko, 1989; Barron, 1991; Lu et al., 2017; Delalleau & Bengio, 2011) and references therein. **(B) Constraint-Aware Layers.** The stochastic program (1) often involves constraints that ensure compliance with physical or regulatory requirements (such as maximum driving voltage constraints in robotics or short-sales constraints in portfolio selection). To ensure that the decision map satisfies all constraints, we use an output layer that maps any input to the corresponding feasible set. This can be achieved in two different ways. Sometimes, one can manually design activation functions tailored to the constraint set at hand (Zhang et al., 2021). For example, the smooth softmax function maps any input into the probability simplex. However, more complicated constraint sets require a more systematic treatment. If the constraint set is closed and convex, for example, then one can construct an output layer that projects any input onto the feasible set. Unfortunately, this approach suffers from a gradient projection problem outlined in Section 4.4. **(C) Optimization Layers.** In view of (1), it is natural to define the prescriptor as the 'argmin' map of a parametric optimization model. Specifically, the prescriptor may output the solution of a *deterministic* optimization model that minimizes the loss at a point estimate of Y. Alternatively, it may output the solution of a *stochastic* optimization model that minimizes the *expected* loss under an estimator for the conditional distribution $\mathbb{P}_{Y|X}$. In both cases, the Jacobian of the prescriptor with respect to the estimator, which is an essential ingredient for SGD-type methods, can quite generally be derived from the problem's KKT conditions (Donti et al., 2017; Agrawal et al., 2019; Uysal et al., 2021b). A key advantage of optimization layers is their ability to capture prior structural information. They are also highly interpretable because the features can be viewed as predictions of Y or $\mathbb{P}_{Y|X}$. Like constraint-aware layers, however, optimization layers suffer from a gradient projection problem that is easy to overlook; see Section 4.4.

3.3. Approximation Capabilities

It is well known that MLPs can uniformly approximate any continuous function even if they only have one hidden layer (Cybenko, 1989; Lu et al., 2017). We will now show that the decision maps considered in this paper inherit the universal approximation capabilities from the feature extractor.

Proposition 3.1 (Universal Approximation of m)**.** *Assume that $m = p \circ f$ combines a feature extractor $f : \mathcal{X} \to \mathcal{R}$ with a prescriptor $p : \mathcal{R} \to \mathcal{A}$ and that p is L_p-Lipschitz*

continuous. Then, the following hold.

(i) If there exists a neural network $f_w : \mathcal{X} \to \mathcal{R}$ with $\sup_{x\in\mathcal{X}} \|f(x) - f_w(x)\| \leq \varepsilon$, then $m_w = p \circ f_w$ satisfies $\sup_{x\in\mathcal{X}} \|m(x) - m_w(x)\| \leq L_p\varepsilon$.

(ii) If there exists a neural network $f_w : \mathcal{X} \to \mathcal{R}$ with $\mathbb{E}[\|f(X)-f_w(X)\|^q] \leq \varepsilon$ for some $q \geq 1$, then $m_w = p \circ f_w$ satisfies $\mathbb{E}[\|m(X) - m_w(X)\|^q] \leq L_p^q\varepsilon$.

The following corollary shows that neural networks can not only be used to approximate m but also its expected loss.

Corollary 3.2 (Universal Approximation of the Loss)**.** *Assume that $m = p \circ f$ combines a feature extractor $f : \mathcal{X} \to \mathcal{R}$ with a prescriptor $p : \mathcal{R} \to \mathcal{A}$, that f is continuous and that p is L_p-Lipschitz continuous. If $\mathcal{X}$ is bounded and the loss function $\ell(y, a)$ is L_ℓ-Lipschitz in a uniformly across all $y \in \mathcal{Y}$, then for every $\varepsilon > 0$ there exists a neural network $f_w : \mathcal{X} \to \mathcal{R}$ with sigmoid activation that satisfies*

$$|\mathbb{E}[\ell(Y, m(X)] - \mathbb{E}[\ell(Y, m_w(X))]| \leq \varepsilon.$$

Corollary 3.2 implies that, for every $\varepsilon > 0$, there exists a neural network f_w and a decision map $m_w = p \circ f_w$ with

$$\mathbb{E}[\ell(Y, m(X)] \leq \mathbb{E}[\ell(Y, m_w(X))] + \varepsilon.$$

In other words, there exists a neural network-based map m_w whose the expected loss exceeds that of m at most by ε.

4. Training Process and Loss Function

The decision map $m^\star$ that solves (2) under the unknown true distribution of X and Y is inaccessible. However, we can train a neural network m_w parametrized by $w \in \mathbb{R}^d$ that approximates $m^\star$. In the following we review a popular SGD-type algorithm for training m_w. While the intimate relation between this widely used algorithm and problems (1) and (2) has not yet been investigated, we prove that m_w maps any observation X to an approximate posterior Bayes action corresponding to X. When X represents a collection of i.i.d. samples from $\mathbb{P}_{Y|Z}$, finally, we outline alternative training methods under which m_w maps X to an approximate minimizer of an ERM or a DRO problem akin to (1).

4.1. SGD-Type Algorithm

End-to-end learning problems are commonly addressed with Algorithm 1 below (Uysal et al., 2021b; Zhang et al., 2021).

Algorithm 1 End-to-End Learning

for $k \leftarrow 1, \ldots, K$ **do**
  $g_k \leftarrow \nabla_w \ell(Y_k, m_w(X_k))|_{w=w_{k-1}}$
  $w_k \leftarrow w_{k-1} - \eta_k g_k$
end for

Note that $\nabla_w \ell(Y_k, m_w(X_k))$ constitutes an unbiased estimator for $\nabla_w \mathbb{E}[\ell(Y, m_w(X))]$. Thus, Algorithm 1 can be viewed as an SGD method for training the decision map m_w. Note also that the step size $\eta_k > 0$ may depend on time.

Remark 4.1 (Generalized Data Sets). Given a generalized dataset $\{(X_k, \widehat{\mathbb{P}}_k)\}_{k=1}^K$ as described in Remark 2.1, we can use $\nabla_w \int_{\mathcal{Y}} \ell(y, m_w(X_k)) \mathrm{d}\widehat{\mathbb{P}}_k(y)|_{w=w_{k-1}}$ as an unbiased gradient estimator in Algorithm 1 (if it is well-defined).

4.2. Bayesian Interpretation of Algorithm 1

We now show that if the decision map m_w is trained via Algorithm 1, then $m_w(X)$ constitutes an approximate posterior Bayes action. That is, it approximately solves problem (1), which minimizes the Bayesian posterior loss. The Bayesian posterior $\mathbb{P}_{Y|X}$ reflects the information available from a given prior $\mathbb{P}_{(X,Y)}$ and an observation X. It is widely used in various decision-making problems (Kalman, 1960; Stengel, 1994; Pezeshk, 2003; Long et al., 2010).

4.2.1. Theoretical Analysis of Algorithm 1

Even though Algorithm 1 is commonly used and conceptually simple, the training loss it minimizes has not yet been investigated. We now analyze Algorithm 1 theoretically. The following standard assumption is required for convergence results of all methods under consideration.

Assumption 4.2 (Smoothness)**.** The loss function $\ell(y, a)$ is smooth in a for all $y \in \mathcal{Y}$, the decision map $m_w(x)$ is smooth in w for all $x \in \mathcal{X}$, and their gradients are bounded.

Replacing the space of all measurable decision maps by the set of all neural network-based decision maps m_w with parameter $w \in \mathbb{R}^d$ yields the following approximation of (2).

$$\min_{w\in\mathbb{R}^d} \mathbb{E}[\ell(Y, m_w(X))] \tag{5}$$

From now on we use $\varphi(w) = \mathbb{E}[\ell(Y, m_w(X))]$ as a shorthand for the objective function of (5). Next, we prove that Algorithm 1 converges to a stationary point of problem (5).

Theorem 4.3 (Bayesian Interpretation of Algorithm 1)**.** *Algorithm 1 solves problem (5) in the following sense.*

(i) The vector g_k computed in Algorithm 1 constitutes an unbiased stochastic gradient for $\varphi(w)$ at $w = w_{k-1}$.

(ii) If Assumption 4.2 holds and Algorithm 1 uses step sizes $\eta_k \propto 1/\sqrt{k}$, then the random iterate $\widehat{w}_K$ sampled from $\{w_k\}_{k=1}^K$ with respective probabilities $\{p_k\}_{k=1}^K$, $p_k \propto \eta_k^{-1}$, satisfies $\mathbb{E}[\|\nabla\varphi(\widehat{w}_K)\|_2^2] = O(1/\sqrt{K})$.

Theorem 4.3 implies that if Assumption 4.2 holds, then the gradient of $\varphi(w)$ at the random iterate $\widehat{w}_K$ generated by Algorithm 1 is small in expectation and—by virtue of Markov's inequality—also small with high probability.

Thus, $\widehat{w}_K$ converges in probability to a stationary point $w^\star$ of problem (5) as K tends to infinity. Throughout the subsequent discussion we assume that $w^\star$ is in fact a minimizer of (5). Theorem 4.3 then suggests that the neural network $m_{w^\star}$ maps any observation X to an *approximation* of the posterior Bayes action corresponding to X. To see this, recall that any minimizer $m^\star$ of (2) maps any observation X to the *exact* posterior Bayes action $m^\star(X)$, which solves the original stochastic optimization problem (1). If the family of neural networks $\mathcal{M}_w = \{m_w : w \in \mathbb{R}^d\}$ is rich enough to contain a minimizer $m^\star$ of (2), then (2) and (5) are clearly equivalent. A sufficient condition for this is that for every function $m_0 \in \mathcal{M}_w$, every measurable set $\mathcal{B} \subseteq \mathcal{X}$ and every bounded measurable function $m_1 \in \mathcal{M}$, the function $m : \mathcal{X} \to \mathcal{A}$ defined through $m(x) = m_0(x)$ if $x \in \mathcal{B}$ and $m(x) = m_1(x)$ if $x \notin \mathcal{B}$ is also a member of $\mathcal{M}_w$; see (Rockafellar & Wets, 2009, Theorem 14.60). The next corollary describes a situation in which this condition holds.

Proposition 4.4 (Finite Observation Spaces)**.** *Assume that $\mathcal{X} = \{x_1, \ldots, x_n\}$ is finite and that the family of neural networks $\mathcal{M}_w = \{m_w : w \in \mathbb{R}^d\}$ is rich enough such that for every $a \in \mathbb{R}^n$ there exists $w \in \mathbb{R}^d$ with $a_i = m_w(x_i)$ for all $i = 1, \ldots, n$. Then, problems (2) and (5) are equivalent.*

Under the assumptions of Proposition 4.4 the family $\mathcal{M}_w$ of neural networks is able to model a look-up table on $\mathcal{X}$. This implies that if Algorithm 1 converges to the global minimizer $w^\star$ of (5), then $m_{w^\star}$ coincides exactly with the posterior Bayes action map. Otherwise, Algorithm 1 uses SGD to approximate $m^\star$ or, in other words, to seek an architecture-regularized approximation of $m^\star$.

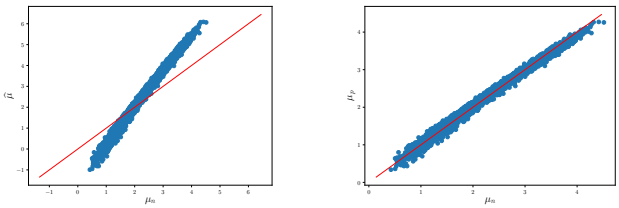
The above insights highlight the importance of choosing a training dataset that is reflective of one's prior belief about the distribution of the unobserved confounder Z. Put differently, it is crucial that the dataset $\{(X_k, Y_k)\}_{k=1}^K$ is consistent with the prior data distribution $\mathbb{P}_{(X,Y)}$ during deployment. Indeed, a strong prior induced by the dataset may significantly bias the decisions of the model. In addition, the above insights provide some justification for augmenting the dataset with rare corner cases. Indeed, excluding corner cases amounts to setting their prior probabilities to 0, which may have undesirable consequences. In the context of algorithmic fairness (Barocas et al., 2019), for instance, using a training dataset in which one demographic group is underrepresented amounts to working with a biased prior distribution and results in subpar predictions for the underrepresented group (Buolamwini & Gebru, 2018).

4.2.2. Minimum Mean-Square Estimation

In order to gain further intuition, consider the problem of estimating the mean Z of a Gaussian random variable $Y \sim \mathcal{N}(Z, 4)$ based on an observation $X = [\widehat{Y}_1, \ldots, \widehat{Y}_{20}]$ of 20 i.i.d. samples drawn from $\mathbb{P}_{Y|Z}$. Assume that $Z \sim \mathcal{N}(2, 0.25)$. The minimum mean-square estimator coincides with the solution of the stochastic optimization problem

$$\min_{a \in \mathbb{R}} \mathbb{E}[(Y - a)^2 | X], \tag{6}$$

which is readily identified as an instance of (1). In order to compute the minimum mean-square estimator simultaneously for all realizations of X, we should solve the corresponding instance of (2). A simple calculation exploiting our distributional assumptions shows that (2) is solved by $m^\star(X) = (8 + \frac{1}{4}\sum_{i=1}^{20} \widehat{Y}_i)/9$. Absent any information about the joint distribution of X, Y and Z, we have to solve the approximate problem (5) instead. Specifically, we optimize over decision maps of the form $m_w = p \circ f_w = f_w$, where the prescriptor p is the identity function, and the feature extractor f_w is a feed-forward neural network with an input layer accommodating 20 neurons and linear activation functions, two hidden layers with 500 neurons and ReLU-activation functions, and an output layer with 1 neuron and a linear activation function. We solve the resulting instance of (5) using Algorithm 1 with $K = 5 \times 10^6$ training samples $\{(X_k, Y_k)\}_{k=1}^K$ to find an approximate posterior Bayes action map m_{w_K}. Given an independent test sample $X = [\widehat{Y}_1, \ldots, \widehat{Y}_{20}]$, this map outputs a neural network-based approximation $\widehat{\mu}_{\mathrm{NN}} = m_{w_K}(X)$ of the exact posterior mean $\widehat{\mu}_{\mathrm{MMSE}} = m^\star(X)$. We compare it against the sample mean $\widehat{\mu}_{\mathrm{ERM}} = \frac{1}{20}\sum_{n=1}^{20} \widehat{Y}_i$, which minimizes the empirical risk. Figure 2 visualizes the differences between these estimators on 5,000 test samples generated by sampling from $\mathbb{P}_{X|Z}$ for 5,000 equidistant values of Z between 0 and 5. We observe that $\widehat{\mu}_{\mathrm{MMSE}}$ closely approximates the posterior mean because the scatter plot concentrates on the identity line in red. When compared to $\widehat{\mu}_{\mathrm{ERM}}$, $\widehat{\mu}_{\mathrm{NN}}$ displays a bias towards 2 due to the strong prior.



(a) NN vs ERM (b) NN vs Posterior Mean

Figure 2. Comparison of the approximate posterior mean $\widehat{\mu}_{\mathrm{NN}}$ output by Algorithm 1 against $\widehat{\mu}_{\mathrm{ERM}}$ and $\widehat{\mu}_{\mathrm{MMSE}}$. As the prior $\mathbb{P}_Y$ is concentrated around $\mathbb{E}[Y] = 2$, both the posterior mean $\widehat{\mu}_{\mathrm{MMSE}}$ as well as its approximation $\widehat{\mu}_{\mathrm{NN}}$ display a bias towards 2.

4.3. Alternative Decision Models

While minimizing the Bayesian posterior loss is uncommon in the literature, ERM and DRO are widely used if the observation $X = [\widehat{Y}_1, \ldots, \widehat{Y}_N]$ consists of N i.i.d. samples from the unknown distribution $\mathbb{P}_{Y|Z}$. We now propose new neural network-based end-to-end learning algorithms that

Table 1. Overview of main results.

Decision Model	Algorithm	Guarantees
Bayesian	Algorithm 1	Theorem 4.3
ERM	Algorithm 2	Theorem 4.5
DRO	Algorithm 3	Theorem 4.6

output approximate solution maps for problems (3) and (4). An overview of our main results is provided in Table 1.

4.3.1. Empirical Risk Minimization

Given observations of the form $X = [\widehat{Y}_1, \ldots, \widehat{Y}_N]$, it is more common to solve the ERM problem (3) instead of the Bayesian optimization problem (1). Algorithm 2 below learns a parametric decision map m_w with the property that $m_w(X)$ approximately solves (3) for every realization of X.

Algorithm 2 End-to-End Learning for ERM

for $k \leftarrow 1, \ldots, K$ **do**
$g_k \leftarrow \nabla_w \frac{1}{N} \sum_{n=1}^N \ell(\widehat{Y}_{k,n}, m_w(X_k))|_{w=w_{k-1}}$
$w_k \leftarrow w_{k-1} - \eta_k g_k$
end for

Unlike Algorithm 1, which evaluates the loss in the gradient computations at a single sample Y_k, which is independent of X_k conditional on the unobserved confounder Z_k, Algorithm 2 evaluates the loss at N samples $\{\widehat{Y}_{k,n}\}_{n=1}^N$, which are the components of X_k. Thus, Algorithm 2 uses a training set that consists only of observations $\{X_k\}_{k=1}^K$ but not of the corresponding problem parameters $\{Y_k\}_{k=1}^K$.

Using the interchangeability principle (Rockafellar & Wets, 2009, Theorem 14.60) and the law of iterated conditional expectations, one can show that problem (3) is equivalent to

$$\min_{m \in \mathcal{M}} \mathbb{E}[\tfrac{1}{N} \textstyle\sum_{n=1}^N \ell(\widehat{Y}_n, m(X))]. \tag{7}$$

Thus, $m^\star$ solves (7) if and only if $m^\star(X)$ solves (3) for all realizations of X (Lemma B.1). Next, we show that Algorithm 2 targets the following approximation of (7).

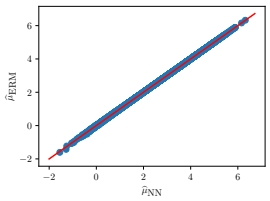
$$\min_{w \in \mathbb{R}^d} \mathbb{E}[\tfrac{1}{N} \textstyle\sum_{n=1}^N \ell(\widehat{Y}_n, m_w(X))] \tag{8}$$

Below we abbreviate the objective function of (8) by $\psi(w)$.

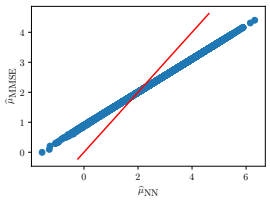
Theorem 4.5 (ERM Interpretation of Algorithm 2)**.** *Algorithm 2 solves problem (8) in the following sense.*

(i) The vector g_k computed in Algorithm 2 constitutes an unbiased stochastic gradient for $\psi(w)$ at $w = w_{k-1}$.

(ii) If Assumption 4.2 holds and Algorithm 2 uses step sizes $\eta_k \propto 1/\sqrt{k}$, then the random iterate $\widehat{w}_K$ sampled from $\{w_k\}_{k=1}^K$ with respective probabilities $\{p_k\}_{k=1}^K$, $p_k \propto \eta_k^{-1}$, satisfies $\mathbb{E}[\|\nabla\psi(\widehat{w}_K)\|_2^2] = O(1/\sqrt{K})$.



(a) NN vs ERM



(b) NN vs Posterior Mean

Figure 3. Comparison of the approximate sample mean $\widehat{\mu}_{\mathrm{NN}}$ output by Algorithm 2 against $\widehat{\mu}_{\mathrm{ERM}}$ and $\widehat{\mu}_{\mathrm{MMSE}}$.

In analogy to Section 4.2.2, we can use Algorithm 2 to train an approximate ERM action map m_{w_K}, which assigns each observation $X = [\widehat{Y}_1, \ldots, \widehat{Y}_{20}]$ an approximate sample mean $\widehat{\mu}_{\mathrm{NN}} = m_{w_K}(X)$. Figure 3 compares $\widehat{\mu}_{\mathrm{NN}}$ against the exact sample mean $\widehat{\mu}_{\mathrm{ERM}}$ and the posterior mean $\widehat{\mu}_{\mathrm{MMSE}}$.

4.3.2. Distributionally Robust Optimization

Another generalization of Algorithms 1 and 2 is Algorithm 3 below, which learns a parametric decision map m_w with the property that $m_w(X)$ approximately solves the DRO problem (4) for every realization of X. Much like Algorithm 2, Algorithm 3 uses a training set that consists only of observations but not of the corresponding problem parameters.

Algorithm 3 End-to-End Learning for DRO

for $k \leftarrow 1, \ldots, K$ **do**
$\mathbb{Q}_k^\star \in \arg\max_{\mathbb{Q} \in \mathcal{U}(X_k)} \int_{\mathcal{Y}} \ell(y, m_{w_{k-1}}(X_k))\, \mathrm{d}\mathbb{Q}(y)$
$g_k \leftarrow \nabla_w \int_{\mathcal{Y}} \ell(y, m_w(X_k))\, \mathrm{d}\mathbb{Q}_k^\star(y)|_{w=w_{k-1}}$
$w_k \leftarrow w_{k-1} - \eta_k g_k$
end for

One can show (Lemma B.2) that problem (4) is equivalent to

$$\min_{m \in \mathcal{M}} \mathbb{E}\left[\max_{\mathbb{Q} \in \mathcal{U}(X)} \int_{\mathcal{Y}} \ell(y, m(X))\, \mathrm{d}\mathbb{Q}(y)\right]. \tag{9}$$

Algorithm 3 then targets the following approximation of (9).

$$\min_{w \in \mathbb{R}^d} \mathbb{E}\left[\max_{\mathbb{Q} \in \mathcal{U}(X)} \int_{\mathcal{Y}} \ell(y, m_w(X))\, \mathrm{d}\mathbb{Q}(y)\right]. \tag{10}$$

Below we abbreviate the objective function of (10) by $\chi(w)$.

Theorem 4.6 (DRO Interpretation of Algorithm 3)**.** *Algorithm 3 solves problem (4) in the following sense. If Assumption 4.2 holds, $\mathcal{U}(X) \neq \emptyset$ is weakly compact and the maximization problem over $\mathbb{Q}$ in (10) has $\mathbb{P}$-almost surely a unique solution, then the vector g_k computed in Algorithm 3 is an unbiased stochastic gradient for $\chi(w)$ at $w = w_{k-1}$.*

The assumption that the maximization problem over $\mathbb{Q}$ has a unique solution is violated by popular ambiguity sets such as the Wasserstein ambiguity set (Kuhn et al., 2019). It is satisfied, however, by the Kullback-Leibler (Hu & Hong, 2013) and Sinkhorn (Wang et al., 2021a) ambiguity sets.

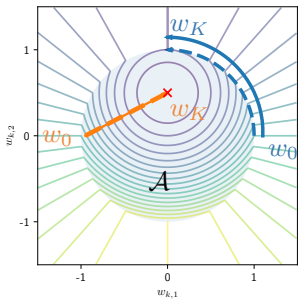


Figure 4. Contours of the expected loss $\varphi(w) = \mathbb{E}[\ell(Y, m_w(X))]$. If the gradient $\nabla\varphi(w)$ exists, then it is perpendicular to the contours of $\varphi(w)$. Thus, the iterates of Algorithm 1 converge to the global minimum $(0, \frac{1}{2})$ of $\varphi(w)$ if $w_0 \in \mathcal{A}$ (solid red line) or to a local minimum in $\{(0, t) : t > 1\}$ if $w_0 \notin \mathcal{A}$ (solid blue line).

4.4. Gradient Projection Phenomenon

Despite their conceptual merits, optimization and projection layers in the decision map m_w may impede the convergence of gradient-based training algorithms. Indeed, if the current iterate $w \notin \mathcal{A}$ is *in*feasible, then optimization and projection layers in m_w have a tendency to push the gradient of $\varphi(w)$ to a subspace of $\mathbb{R}^d$ that is (approximately) perpendicular to the shortest path from w to $\mathcal{A}$. Thus, gradient-based methods like Algorithm 1 may circle around $\mathcal{A}$ without ever reaching a feasible point. To our best knowledge, this phenomenon has not yet been studied or even recognized.

As an example, consider an instance of problem (2) with $\mathcal{X} = \mathbb{R}$, $\mathcal{Y} = \mathbb{R}^2$, $\mathcal{A} = \{a \in \mathbb{R}^2 : \|a\|_2 \leq 1\}$ and $\ell(y, a) = \|y - a\|_2^2$. Assume further that $\mathbb{P}_X = \delta_1$ and $\mathbb{P}_{Y|X} = \delta_{(0,\frac{1}{2})}$. In this case the constant decision map $m(X) = (0, \frac{1}{2})$ is optimal in (2). We now approximate (2) by problem (5), which minimizes over parametric decision maps of the form $m_w = p \circ f_w$, and we assume that the prescriptor consists of an optimization layer that maps any feature $r \in \mathbb{R}^2$ to

$$p(r) = \arg\min_{a \in \mathcal{A}} \|a - r\|_2,$$

while the feature extractor parametrized by $w \in \mathbb{R}^2$ maps any observation x to a feature $f_w(x) = xw = w$ $\mathbb{P}$-almost surely. Training the decision map m_w via Algorithm 1 with an initial iterate $w_0 \notin \mathcal{A}$ generates a sequence of iterates that stay outside of $\mathcal{A}$ as visualized in Figure 4 (solid blue line). The corresponding predictions satisfy $m_{w_k}(X) = p(w_k)$ $\mathbb{P}$-almost surely. Thus, they stay on the boundary of $\mathcal{A}$ (dashed blue line). If $w_0 \in \mathcal{A}$, on the other hand, then the iterates converge to the global minimum (solid red line). In this case, the corresponding predictions satisfy $m_{w_k}(X) = p(w_k) = w_k$ $\mathbb{P}$-almost surely. Thus, they coincide with the underlying iterates (dashed red line).

5. Experiments

We benchmark the discussed algorithms for end-to-end learning against simple baselines as well as the predict-then-optimize approach (Mišić & Perakis, 2020) in the context of a newsvendor problem and an economic dispatch problem. Implementation details are given in Appendix C, and the code underlying all experiments is provided on GitHub.[1]

5.1. Newsvendor Problem

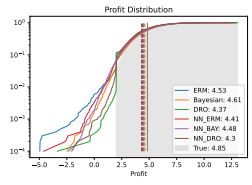
We first compare the Bayesian model addressed by Algorithm 1 against the ERM and DRO models. To this end, we consider the decision problem of the seller of a perishable good (e.g., a newspaper). At the beginning of each day, the newsvendor buys a number $a \in \mathcal{A} = \{1, \ldots, d\}$ of items from the supplier at a wholesale price $p > 0$. During the day she sells the items at the retail price $q > p$ until the supply a is exhausted or the random demand $Y \in \mathcal{Y} = \mathcal{A}$ is covered. The salvage value of unsold items is 0. Hence, the newsvendor's total cost amounts to $\ell(Y, a) = pa - q\min\{a, Y\}$. We assume that the unobserved confounder $Z \in \mathbb{R}^d$ represents the demand distribution, that is, $Z_j = \mathbb{P}_{Y|Z}(Y = j)$ for all $j = 1, \ldots, d$. If the newsvendor observes N independent historical demands $X = [\widehat{Y}_1, \ldots, \widehat{Y}_N]$ sampled from $\mathbb{P}_{Y|Z}$, then she aims to solve the following instance of (1)

$$\min_{a \in \mathcal{A}} \mathbb{E}[\ell(Y, a)|X] = \min_{a \in \mathcal{A}} \textstyle\sum_{j=1}^d \ell(j, a)\mathbb{E}[Z_j|X].$$

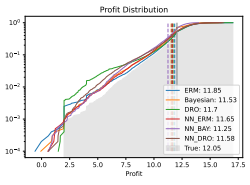
In the following we fix a common neural network architecture and train decision maps via Algorithms 1, 2 and 3 with $K = 5 \times 10^6$ samples. Specifically, in Algorithm 3 we set $\mathcal{U}(X)$ to the Kullback-Leibler ambiguity set of radius 0.025 around the empirical distribution on the demand samples contained in X. We designate the strategies output by the three algorithms as "NN_BAY", "NN_ERM" and "NN_DRO", respectively. The strategies obtained by solving the Bayesian problem (1), the ERM problem (3) and the DRO problem (4) *exactly* are designated as "BAY", "ERM" and "DRO", respectively. Finally, the strategy output by an oracle with perfect knowledge of Z is designated as "True".

Figure 5 visualizes the out-of-sample cumulative distribution functions of the profit (negative loss) generated by different data-driven strategies. In line with Theorems 4.3, 4.5 and 4.6, the neural network-based decisions display a similar performance as the corresponding optimization-based decisions. Figure 5 further shows that a correct choice of the prior is crucial for the Bayesian strategies (BAY and NN_BAY). Finally, while the expected profits generated by different strategies are similar, the lower tails of the profit distributions vary dramatically. For example, Figure 5a indicates that the risk of a loss (negative profit) is almost 10 times higher under the ERM strategy than under any other

[1] https://github.com/RAO-EPFL/end2end-SO



(a) Correct Prior



(b) Incorrect Prior

Figure 5. Cumulative distribution functions (solid lines) and expected values (dashed lines) of the out-of-sample profit generated by different data-driven strategies.

strategy. Appendix C provides a more detailed comparison between the neural network-based strategies and the corresponding optimization-based strategies.

5.2. Economic Dispatch Problem

In the second experiment, we revisit the stylized economic dispatch problem described in Section 2. Here we assume that there are $J = 6$ traditional generators and that the penalty for unmet demand amounts to $p = 100$. The capacities $\bar{a}_j$ of the six generators are given by 1, 0.5, 1, 1, 1 and 0.5, and the respective unit production costs c_j are set to 15, 20, 15, 20, 30 and 25. We use historical wind power production and weather records[2] as samples from $\mathbb{P}_{(X,Y)}$.

Observations Contextual stochastic programs account for side information such as wind speed measurements or weather forecasts that can help to make better decisions. Any such side information is captured by the random variable X, which is observable when the generation decisions $a \in \mathcal{A}$ must be selected. In the following experiment we distinguish three different possible observations. The myopic observation **(Myopic)** consists of current temperature, wind speed and wind direction measurements. The incomplete myopic observation **(Myopic incomp.)** consists only of the current wind direction. The most informative observation **(Historical)** consists of the current temperature, wind speed and wind direction measurements as well as of the temperature, wind speed, wind direction, and wind energy production measurements at the last two timesteps.

Baselines We compare the performance of all data-driven strategies to be described below against that of an ideal baseline strategy that has oracle access to the future wind power production level. This oracle strategy thus observes $X = Y$, in which case the stochastic program (1) collapses to a deterministic optimization problem. The resulting strategy is *infeasible* in practice and serves merely as a basis for comparison. A related *feasible* baseline strategy is obtained by pretending that Y equals the wind energy production quantity observed in the last period and by solving the corresponding deterministic optimization problem akin to (1).

[2] https://www.kaggle.com/datasets/theforcecoder/wind-power-forecasting

Table 2. Average test costs generated by different data-driven strategies for the economic dispatch problem.

Approach	Observation	10 minute freq.	30 minute freq.
Baseline	Oracle	60.69	60.69
	Lag-1	64.96	67.71
MLE	Myopic	281.44(1.65)	280.10(0.75)
	Myopic Incomp.	348.41(0.0)	348.43(0.0)
	Historical	304.56(11.47)	277.74(2.38)
E2E-CAL	Myopic	68.25(3.73)	66.57(3.07)
	Myopic Incomp.	72.62(0.00)	74.11(3.00)
	Historical	67.09(4.50)	77.06(5.59)
E2E-OPL-Softplus	Myopic	71.33(1.87)	72.53(0.13)
	Myopic Incomp.	72.60(0.01)	72.60(0.00)
	Historical	72.60(0.0)	72.60(0.0)

Maximum Likelihood Estimation (MLE) We construct a predict-then-optimize strategy by solving a least squares regression model for predicting the wind energy production level Y and then solving the deterministic version of problem (1) corresponding to this point prediction.

End-to-End (E2E) We also design end-to-end learning strategies, which are obtained by using Algorithm 1 to train the feature extractor together with the prescriptor. We distinguish two different neural network architectures. On the one hand, we can define the prescriptor as an optimization problem layer (OPL) and the feature extractor as a predictor of Y. A Softplus activation function in the output layer of the feature extractor ensures that the prediction of Y is nonnegative. On the other hand, we can define the prescriptor as a constraint-aware layer (CAL), which uses rescaled sigmoid activation functions to force its output into $\mathcal{A}$.

Table 2 reports the test costs of the different strategies. All strategies are compared under different data frequency models. That is, data is either observed every 10 minutes or every 30 minutes. We observe that the MLE method incurs the highest costs. This may be attributed to the symmetric training loss, which ignores that it is better to underestimate energy production because unmet demand is heavily penalized. If new data is observed every 10 minutes, then the lag-1 baseline performs best, while the end-to-end strategy with access to historical data and with constraint-aware layers performs best among all neural network-based approaches. If new data is observed every 30 minutes, then the stationarity assumption is violated and the lag-1 baseline is no longer competitive. In this case, the most useful observations are the wind speed measurements, which cannot be leveraged by the baselines. End-to-end learning strategies based on optimization layers often fail to train, which we attribute to the gradient projection problem. The myopic observation is most useful, but the incomplete myopic observation does not contain enough information to accurately estimate the wind power production. The historical information is overly rich and therefore leads to overfitting (Ying, 2019).

References

Agrawal, A., Amos, B., Barratt, S., Boyd, S., Diamond, S., and Kolter, Z. Differentiable convex optimization layers. In *Advances in Neural Information Processing Systems*, 2019.

Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., Chen, G., et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International Conference on Machine Learning*, 2016.

Audibert, J.-Y., Munos, R., and Szepesvári, C. Exploration–exploitation tradeoff using variance estimates in multi-armed bandits. *Theoretical Computer Science*, 410(19): 1876–1902, 2009.

Barocas, S., Hardt, M., and Narayanan, A. *Fairness and Machine Learning*. 2019. http://www.fairmlbook.org.

Barron, A. R. Approximation and estimation bounds for artificial neural networks. In *Fourth Annual Workshop on Computational Learning Theory*, 1991.

Baum, E. and Wilczek, F. Supervised learning of probability distributions by neural networks. In *Neural Information Processing Systems*, 1987.

Ben-Tal, A., Den Hertog, D., De Waegenaere, A., Melenberg, B., and Rennen, G. Robust solutions of optimization problems affected by uncertain probabilities. *Management Science*, 59(2):341–357, 2013.

Berger, J. O. *Statistical Decision Theory and Bayesian Analysis*. Springer, 2013.

Bertsimas, D. and Kallus, N. From predictive to prescriptive analytics. *Management Science*, 66(3):1025–1044, 2020.

Buolamwini, J. and Gebru, T. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on Fairness, Accountability and Transparency*, 2018.

Butler, A. and Kwon, R. H. Integrating prediction in mean-variance portfolio optimization. *arXiv preprint arXiv.2102.09287*, 2021.

Cho, K., Van Merriënboer, B., Bahdanau, D., and Bengio, Y. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

Chu, W., Li, L., Reyzin, L., and Schapire, R. Contextual bandits with linear payoff functions. In *International Conference on Artificial Intelligence and Statistics*, 2011.

Cournot, A. A. *Researches into the Mathematical Principles of the Theory of Wealth*. Macmillan Company, 1897.

Cybenko, G. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, 1989.

Delage, E. and Ye, Y. Distributionally robust optimization under moment uncertainty with application to data-driven problems. *Operations Research*, 58(3):595–612, 2010.

Delalleau, O. and Bengio, Y. Shallow vs. deep sum-product networks. In *Advances in Neural Information Processing Systems*, 2011.

Donti, P. L., Amos, B., and Kolter, J. Z. Task-based end-to-end model learning in stochastic optimization. In *International Conference on Neural Information Processing Systems*, 2017.

Durrett, R. *Probability: Theory and Examples*. Cambridge University Press, 2010.

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

Elman, J. L. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990.

Fu, S.-W., Wang, T.-W., Tsao, Y., Lu, X., and Kawai, H. End-to-end waveform utterance enhancement for direct evaluation metrics optimization by fully convolutional neural networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(9):1570–1584, 2018.

Graves, A. and Jaitly, N. Towards end-to-end speech recognition with recurrent neural networks. In *International Conference on Machine Learning*, 2014.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Hu, Z. and Hong, L. J. Kullback-Leibler divergence constrained distributionally robust optimization. *Optimization Online*, 2013.

Kalman, R. E. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Kiran, B. R., Sobh, I., Talpaert, V., Mannion, P., Al Sallab, A. A., Yogamani, S., and Pérez, P. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 23 (6):4909–4926, 2021.

Kline, D. M. and Berardi, V. L. Revisiting squared-error and cross-entropy functions for training neural network classifiers. *Neural Computing & Applications*, 14(4): 310–318, 2005.

Kuhn, D., Mohajerin Esfahani, P., Nguyen, V. A., and Shafieezadeh-Abadeh, S. Wasserstein distributionally robust optimization: Theory and applications in machine learning. In *Operations Research & Management Science in the Age of Analytics*, pp. 130–166. INFORMS, 2019.

Lattimore, T. and Szepesvàri, C. *Bandit Algorithms*. Cambridge University Press, 2020.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

Li, L. Macroeconomic factors and the correlation of stock and bond returns. *Available at SSRN 363641*, 2002.

Long, B., Chapelle, O., Zhang, Y., Chang, Y., Zheng, Z., and Tseng, B. Active learning for ranking through expected loss optimization. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, 2010.

Lu, Z., Pu, H., Wang, F., Hu, Z., and Wang, L. The expressive power of neural networks: A view from the width. In *Advances in Neural Information Processing Systems*, 2017.

Markowitz, H. M. and Todd, G. P. *Mean-variance analysis in portfolio choice and capital markets*. John Wiley & Sons, 2000.

Mišić, V. V. and Perakis, G. Data analytics in operations management: A review. *Manufacturing & Service Operations Management*, 22(1):158–169, 2020.

Murphy, K. P. Conjugate Bayesian analysis of the Gaussian distribution. Technical report, University of British Columbia, 2007.

Papoulis, A. and Pillai, S. U. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, 2002.

Pezeshk, H. Bayesian techniques for sample size determination in clinical trials: a short review. *Statistical Methods in Medical Research*, 12(6):489–504, 2003.

Rahimian, H. and Mehrotra, S. Distributionally robust optimization: A review. *arXiv preprint arXiv:1908.05659*, 2019.

Rockafellar, R. T. and Wets, R. J.-B. *Variational Analysis*. Springer, 2009.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

Shapiro, A., Dentcheva, D., and Ruszczynski, A. *Lectures on Stochastic Programming: Modeling and Theory*. SIAM, 2021.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359, 2017.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

Smith, J. E. and Winkler, R. L. The optimizer's curse: Skepticism and postdecision surprise in decision analysis. *Management Science*, 52(3):311–322, 2006.

Stengel, R. F. *Optimal Control and Estimation*. Courier Corporation, 1994.

Uysal, A. S., Li, X., and Mulvey, J. M. End-to-end risk budgeting portfolio optimization with neural networks. *arXiv preprint arXiv.2107.04636*, 2021a.

Uysal, A. S., Li, X., and Mulvey, J. M. End-to-end risk budgeting portfolio optimization with neural networks. *arXiv preprint arXiv:2107.04636*, 2021b.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575 (7782):350–354, 2019.

Wan, E. A. Neural network classification: A Bayesian interpretation. *IEEE Transactions on Neural Networks*, 1 (4):303–305, 1990.

Wang, J., Gao, R., and Xie, Y. Sinkhorn distributionally robust optimization. *arXiv preprint arXiv:2109.11926*, 2021a.

Wang, K., Babenko, B., and Belongie, S. End-to-end scene text recognition. In *International Conference on Computer Vision*, 2011.

Wang, X., Magnússon, S., and Johansson, M. On the convergence of step decay step-size for stochastic optimization. *Advances in Neural Information Processing Systems*, 2021b.

Wang, Y., Skerry-Ryan, R., Stanton, D., Wu, Y., Weiss, R. J., Jaitly, N., Yang, Z., Xiao, Y., Chen, Z., Bengio, S., Le, Q., Agiomyrgiannakis, Y., Clark, R., and Saurous, R. A. Tacotron: A fully end-to-end text-to-speech synthesis model. *arXiv preprint arXiv:1703.10135*, 2017.

Wiesemann, W., Kuhn, D., and Sim, M. Distributionally robust convex optimization. *Operations Research*, 62(6): 1358–1376, 2014.

Wong, S. and Fuller, J. D. Pricing energy and reserves using stochastic optimization in an alternative electricity market. *IEEE Transactions on Power Systems*, 22(2): 631–638, 2007.

Ying, X. An overview of overfitting and its solutions. In *Journal of Physics: Conference Series*, 2019.

Zhang, C., Zhang, Z., Cucuringu, M., and Zohren, S. A universal end-to-end approach to portfolio optimization via deep learning. *arXiv preprint arXiv:2111.09170*, 2021.

Zhang, W., Tanida, J., Itoh, K., and Ichioka, Y. Shift-invariant pattern recognition neural network and its optical architecture. In *Annual Conference of the Japan Society of Applied Physics*, 1988.

Zhang, Z., Zohren, S., and Roberts, S. Deep learning for portfolio optimization. *The Journal of Financial Data Science*, 2(4):8–20, 2020.

A. Proofs of Section 3

Proof of Proposition 3.1. We first prove Assertion (i). As p is Lipschitz continuous, we have

$$\|p(r_1) - p(r_2)\| \leq L_p \|r_1 - r_2\| \quad \forall r_1, r_2 \in \mathcal{R}.$$

This is notably also true for $r_1 = f(x)$ and $r_2 = f_w(x)$, in which case we obtain

$$\|m(x) - m_w(x)\| = \|p(f(x)) - p(f_w(x))\| \leq L_p \|f(x) - f_w(x)\| \leq L\varepsilon \quad \forall x \in \mathcal{X},$$

where the last inequality follows from the assumption about $f_w(x)$. The claim now follows by maximizing the left hand side across all $x \in \mathcal{X}$. As for Assertion (ii), the first part of the proof readily implies that

$$\mathbb{E}\left[\|m(x) - m_w(x)\|^q\right] \leq L_p^q \mathbb{E}\left[\|f(x) - f_w(x)\|^p\right] \leq L_p^q \varepsilon,$$

where the last inequality follows from the the assumption about $f_w(x)$. □

Proof of Corollary 3.2. Set $\delta = \varepsilon/(L_p L_\ell)$. By (Cybenko, 1989), there exists a neural network f_w with a single hidden layer of sufficient width and with sigmoid activation functions such that $\sup_{x \in \mathcal{X}} \|f(x) - f_w(x)\| \leq \delta$. By the Lipschitz-continuity of p and ℓ, we thus obtain

$$\sup_{x \in \mathcal{X}, y \in \mathcal{Y}} |\ell(y, m(x)) - \ell(y, m_w(x))| \leq \sup_{x \in \mathcal{X}} L_\ell \|m(x) - m_w(x)\| \leq \sup_{x \in \mathcal{X}} L_p L_\ell \|f(x) - f_w(x)\| \leq \varepsilon.$$

The claim now follows because the expected value of a non-negative random variable is upper bounded by its supremum. □

B. Proofs of Section 4

Proof of Theorem 4.3. As for Assertion (i), note that the expected value of g_k conditional on the last iterate w_{k-1} satisfies

$$\begin{aligned}
\mathbb{E}[g_k | w_{k-1}] &= \mathbb{E}[\nabla_w \ell(Y_k, m_w(X_k))|_{w=w_{k-1}} | w_{k-1}] \\
&= \nabla_w \mathbb{E}[\ell(Y_k, m_w(X_k)) | w_{k-1}]|_{w=w_{k-1}} \\
&= \nabla_w \mathbb{E}\big[\ell(Y, m_w(X))\big]|_{w=w_{k-1}} = \nabla \varphi(w_{k-1}),
\end{aligned}$$

where the first equality follows from the definition of g_k. The second equality holds because the gradient with respect to w and the expectation conditional on w_{k-1} can be interchanged thanks to the dominated convergence theorem, which applies thanks to Assumption 4.2. The third equality, finally, exploits the independence of (X_k, Y_k) and w_{k-1}, and the last equality follows from the definition of φ. Assertion (ii) follows from (Wang et al., 2021b, Theorem 3.5), which applies again thanks to Assumption 4.2. Indeed, g_k constitutes an unbiased gradient estimator for $\varphi(w)$ at $w = w_{k-1}$ thanks to Assertion (i). Assumption 4.2 further implies that $\varphi(w)$ is L-smooth for some $L < \infty$. In addition, the stochastic gradients g_k have bounded variance because they are themselves bounded by Assumption 4.2. Finally, for $\varphi^\star = \min_{w \in \mathbb{R}^d} \varphi(w)$, we have that $\mathbb{E}[\varphi(w_k) - \varphi^\star]$ is bounded because ℓ is bounded. Thus, the claim follows. □

The next lemma establishes the equivalence of problems (3) and (7).

Lemma B.1 (Equivalence of (3) and (7))**.** *Problems (3) and (7) are equivalent in the sense that if $m^\star$ solves (7), then $m^\star(X)$ solves (3) $\mathbb{P}$-almost surely and vice versa.*

Proof of Lemma B.1. By the law of iterated conditional expectations (Durrett, 2010, Theorem 5.1.6.), (7) can be recast as

$$\min_{m \in \mathcal{M}} \mathbb{E}\left[\mathbb{E}\left[\frac{1}{N}\sum_{n=1}^N \ell(\widehat{Y}_n, m(X)) \,\middle|\, X\right]\right]. \tag{11}$$

Next, by the interchangeability principle (Rockafellar & Wets, 2009, Theorem 14.60), minimizing over all measurable decision maps $m \in \mathcal{M}$ outside of the outer expectation is equivalent to minimizing over all decisions $a \in \mathcal{A}$ inside the outer expectation. Hence, the above optimization problem is equivalent to

$$\mathbb{E}\left[\min_{a \in \mathcal{A}} \mathbb{E}\left[\frac{1}{N}\sum_{n=1}^N \ell(\widehat{Y}_n, a) \,\middle|\, X\right]\right]. \tag{12}$$

In addition, the interchangeability principle also implies that $m^\star$ solves the minimization problem over $m \in \mathcal{M}$ in (11) if and only if $a^\star = m^\star(X)$ solves the minimization problem over $a \in \mathcal{A}$ in (12) $\mathbb{P}$-almost surely. As $X = [\widehat{Y}_1, \ldots, \widehat{Y}_N]$, the samples $\widehat{Y}_n$ are measurable with respect to the σ-algebra generated by X for all $i = 1, \ldots, N$. Therefore, the inner minimization problem in (12) is in fact equivalent to the ERM problem (3). Thus, the claim follows. □

Proof of Theorem 4.5. The proof widely parallels that of Theorem 4.3. Details are omitted for brevity. □

Similar to Lemma B.1, we provide a Lemma to outline the equivalence of problems (4) and (9).

Lemma B.2 (Equivalence of (4) and (9))**.** *Problems (4) and (9) are equivalent in the sense that if $m^\star$ solves (9), then $m^\star(X)$ solves (4) $\mathbb{P}$-almost surely and vice versa.*

Proof of Lemma B.2. By the law of iterated conditional expectations (Durrett, 2010, Theorem 5.1.6.), (9) can be recast as

$$\min_{m \in \mathcal{M}} \mathbb{E}\left[\mathbb{E}\left[\max_{\mathbb{Q} \in \mathcal{U}(X)} \int_{\mathcal{Y}} \ell(y, m(X))\, \mathrm{d}\mathbb{Q}(y) \,\middle|\, X\right]\right], \tag{13}$$

and by the interchangeability principle (Rockafellar & Wets, 2009, Theorem 14.60), this is equivalent to

$$\mathbb{E}\left[\min_{a \in \mathcal{A}} \mathbb{E}\left[\max_{\mathbb{Q} \in \mathcal{U}(X)} \int_{\mathcal{Y}} \ell(y, a)\, \mathrm{d}\mathbb{Q}(y) \,\middle|\, X\right]\right] = \mathbb{E}\left[\min_{a \in \mathcal{A}} \max_{\mathbb{Q} \in \mathcal{U}(X)} \int_{\mathcal{Y}} \ell(y, a)\, \mathrm{d}\mathbb{Q}(y)\right]. \tag{14}$$

The last equality holds because $X = [\widehat{Y}_1, \ldots, \widehat{Y}_N]$. The interchangeability principle also implies that $m^\star$ solves the minimization problem over $m \in \mathcal{M}$ in (13) if and only if $a^\star = m^\star(X)$ solves the minimization problem over $a \in \mathcal{A}$ in (14) $\mathbb{P}$-almost surely. This observation completes the proof. □

Proof of Theorem 4.6. By Assumption 4.2, the gradient $\nabla_w \int_{\mathcal{Y}} \ell(y, m_w(X_k)) \mathrm{d}\mathbb{Q}(y)$ exists and is continuous in w for every $\mathbb{Q} \in \mathcal{U}(X)$ and for every $k = 1, \ldots, K$. Since the maximization problem over $\mathbb{Q}$ in (10) has $\mathbb{P}$-almost surely a unique solution, Danskin's Theorem (Shapiro et al., 2021, Theorem 7.21) implies that

$$g_k = \nabla_w \left[\int_{\mathcal{Y}} \ell(y, m_w(X_k)\, \mathrm{d}\mathbb{Q}_k^\star(y)\right]\Big|_{w=w_{k-1}} = \nabla_w \left[\max_{\mathbb{Q} \in \mathcal{U}(X_k)} \int_{\mathcal{Y}} \ell(y, m_w(X_k)\, \mathrm{d}\mathbb{Q}(y)\right]\Big|_{w=w_{k-1}}.$$

As g_k is bounded, it has a bounded variance, and thus we can conclude that

$$\mathbb{E}[g_k | w_{k-1}] = \nabla_w \mathbb{E}\left[\max_{\mathbb{Q} \in \mathcal{U}(X_k)} \int \ell(y, m_w(X_k)\, \mathrm{d}\mathbb{Q}(y)\right]\Big|_{w=w_{k-1}}.$$

This observation completes the proof. □

C. Details on Experiments

C.1. Details on the Minimum Mean-Square Estimation Experiment in Section 4.2.2

The observation is given by $X = [\widehat{Y}_1, \ldots, \widehat{Y}_{20}]$, whose components $\widehat{Y}_n$ are sampled independently from $\mathbb{P}_{Y|Z}$. Thus, Y and X are independent conditionally on Z as in Figure 1a. Note, however, that both Y and X depend on Z, and thus they are not (unconditionally) independent. We solve problem (6) with a Batch-SGD variant of Algorithm 1, where the training samples $\{(X_k, Y_k)\}_{k=1}^K$ are generated by first sampling Z from the prior $\mathcal{N}(2, 0.25)$. We then sample X_k from $\mathbb{P}_{X|Z}$ and Y_k from $\mathbb{P}_{Y|Z}$. The Batch-SGD algorithm runs over 50,000 iterations with 100 samples per batch to reduce the variance of the gradient updates. Therefore, a total of $K = 5 \times 10^6$ training samples are used for training. The neural network-based predictions $\widehat{\mu}_{\mathrm{NN}}$ are compared against the sample mean $\widehat{\mu}_{\mathrm{ERM}}$ and posterior mean $\widehat{\mu}_{\mathrm{MMSE}}$. The posterior mean can be computed in closed form. Indeed, since we use a conjugate prior for the mean of a Gaussian distribution, the solution of problem (6) can be computed analytically. This is possible because $\mathbb{P}_{Y|X} = \mathcal{N}(\mu_{\mathsf{Bayes}}, \sigma^2_{\mathsf{Bayes}})$ with

$$\mu_{\mathsf{Bayes}} = \frac{1}{\frac{1}{0.25} + \frac{20}{4}} \left(\frac{2}{0.25} + \frac{\sum_{i=1}^{20} \widehat{Y}_i}{4}\right) \quad \text{and} \quad \sigma^2_{\mathsf{Bayes}} = 4 + \left(\frac{1}{0.25} + \frac{20}{4}\right)^{-1}.$$

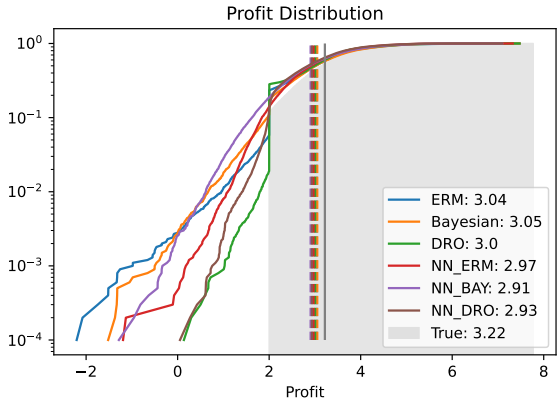


Figure 6. Cumulative distribution functions (solid lines) and expected values (dashed lines) of the out-of-sample profit generated by different data-driven strategies for the newsvendor problem (additional example with incorrect prior).

More specifically, Y can be expressed as $\mu_{\text{Bayes}} + A + B$, where A and B are both zero-mean Gaussian random variables with A having variance 4 (because $\mathbb{P}_{Y|Z} = \mathcal{N}(Z, 4)$) and B having variance $(1/0.25 + 20/4)^{-1}$, see (Murphy, 2007) for a detailed derivation of the posterior mean with conjugate prior. It follows that $\widehat{\mu}_{\text{MMSE}} = \mu_{\text{Bayes}}$. By Theorem 4.3, we expect the output $\widehat{\mu}_{\text{NN}}$ of the trained neural network to approximate the minimizer $\widehat{\mu}_{\text{MMSE}}$ of the expected posterior loss, which is biased towards 2 due to the chosen prior. On the other hand, $\widehat{\mu}_{\text{ERM}}$ is an unbiased estimator of the mean, and we thus expect it to behave differently than the other two estimators. This simple experiment empirically verifies Theorem 4.3.

C.2. Details on the Newsvendor Experiment in Section 5.1

We set the number of possible demand levels to $d = 11$, the wholesale price to $p = 5$ and and the retail price to $q = 7$. We further assume that an observation X consists of $N = 20$ historical demand samples (i.e., $X = [\widehat{Y}_1, \ldots, \widehat{Y}_{20}]$). Finally, we define the ambiguity set $\mathcal{U}(X)$ in the DRO problem (4) as the family of all demand distributions whose Kullback-Leibler divergence with respect to the empirical distribution on the samples in X does not exceed 0.25. In this case problem (4) can be solved efficiently via the convex optimization techniques developed in (Ben-Tal et al., 2013). Instead of using Algorithms 1, 2 and 3 directly, we train the neural networks via the Adam optimizer (Kingma & Ba, 2015). Training proceeds over 1,000 iterations using batches of 1,000 samples of $Z \sim \mathbb{P}_Z$ and 5 samples of $(X, Y) \sim \mathbb{P}_{(X,Y)|Z}$ per iteration. This corresponds to $K = 5 \times 10^6$ samples of (X, Y, Z) in total as described in Section 5.1. When generating training samples, we assume that $\mathbb{P}_Z$ represents the uniform distribution on the 11-dimensional probability simplex. Note that this uniform distribution coincides with the Dirichlet distribution of order $d = 11$ whose 11 parameters are all equal to 1. As the prior $\mathbb{P}_Z$ is a Dirichlet distribution, the posterior $\mathbb{P}_{Z|X}$ is also a Dirichlet distribution with new parameters updated by the observation X (Berger, 2013). Thus, the posterior Bayes action map (the "True" strategy) can be computed in closed form. The out-of-sample profit $-\mathbb{E}[\ell(Y, m(X))]$ of any decision strategy $m(X)$ is evaluated empirically on 10,000 test samples of (X, Y, Z). To assess the advantages and disadvantages of the different decision strategies, we consider several test distributions. These test distributions are constructed exactly like the training distribution but use different Dirichlet parameters for $\mathbb{P}_Z$. We expect that the performance of the ERM and DRO strategies is immune to misspecifications of the prior $\mathbb{P}_Z$. The Bayesian strategy and its neural network approximation, however, are expected to suffer under a biased prior. The test performance shown in Figure 5a is evaluated under the correct prior (i.e., all parameters of the Dirichlet distribution $\mathbb{P}_Z$ are equal to 1). Figures 5b and 6 show the impact of a distribution shift. Specifically, the Dirichlet parameters of the test distribution underlying Figure 5b are set to 0.1 for the five lowest demand levels and to 2 for the 6 highest demand levels. Thus, the prior used for training gives too much weight to low demand levels. Similarly, the Dirichlet parameters of the test distribution underlying Figure 6 are set to 2 for the 6 lowest demand levels and to 0.1 for the 5 highest demand levels. Thus, the prior used for training gives too much weight to high demand levels.

Figure 7 compares the decisions obtained with the different methods. In this experiment, the training and testing distributions match, and we set the Dirichlet parameters of the prior $\mathbb{P}_Z$ to 0.5 for the 7 lowest demand levels and to 2 for the 4 highest demand levels. This particular prior is chosen because it clearly exposes the differences between different strategies. We generate 10^5 random observations $X \sim \mathbb{P}_X$ and compute the corresponding decisions. Each chart in Figure 7 compares one

Table 3. Generation costs and capacities of the generators

Generator j	1	2	3	4	5	6
Generation Cost c_j	15	20	15	20	30	25
Capacity $\bar{a}_j$	1	0.5	1	1	1	0.5

of the exactly computed decisions (vertical axis) against a neural network-based approximation (horizontal axis) trained with Algorithms 1, 2 and 3. The resulting point clouds visualized in Figure 7 are consistent with our theoretical results. That is, the neural network-based approximations align best with the exactly computed decisions in the three charts on the diagonal. Additionally, we observe that the Bayesian strategies order more than the ERM strategies because demand distributions with a large expected value are more likely under the chosen prior. In contrast, the DRO strategies order less than the ERM strategies due to the embedded ambiguity aversion, which favors conservative decisions.

C.3. Details on the Economic Dispatch Experiment in Section 5.2

We assume that the constant energy demand $d = 4$ must be covered by the uncertain output Y of the wind turbine and by the outputs a_j, $j = 1, \ldots, 6$, of the six controllable generators. The capacity of the wind turbine equals 2. Thus, at least 2 units of energy must be produced by conventional generators. The capacities and generation costs of these generators are listed in Table 3. The wind turbine produces energy for free but cannot be controlled. A dataset of historical wind power production and weather records with a 10 minute resolution is available from Kaggle.[3] The dataset covers the period from 1 January 2018 to 30 March 2020. After removing corrupted samples, the period from 1 January 2018 to 31 December 2019 comprises 59,532 records, which we use as the training set. The remaining records are used for testing. We traverse the test set in steps of 10, 30 and 60 minutes to simulate different sampling frequencies. For each interval between two consecutive time steps we solve the economic dispatch problem described in Section 2.

Maximum Likelihood Estimation (MLE) The MLE approach first uses least squares regression on the training data to construct a prediction $\widehat{Y}$ of the wind energy production Y. This prediction is then used as an input for the deterministic prescription problem $\min_{a \in \mathcal{A}} \ell(\widehat{Y}, a)$, which outputs the MLE decision. If Y can be expressed as a linear function of the observation X with an additive Gaussian error, then least squares regression is indeed equivalent to MLE. While MLE outputs an unbiased prediction $\widehat{Y}$, the task loss caused by a prediction error $Y - \widehat{Y}$ is misaligned with the regression loss. Indeed, if $\widehat{Y}$ overestimates Y, then the MLE decision produces too little energy, which incurs high costs of $p = 100$ per unit of unmet demand. Conversely, if $\widehat{Y}$ underestimates Y, then the MLE decision produces too much energy. However, this incurs a cost of at most 30 per unit of surplus, that is, the unit production cost of generator 5.

End-to-End (E2E) We compare multiple neural network architectures.

(OPL): The OPL architecture consists of a feature extractor that maps the observation X to a prediction $\widehat{Y}$ of Y and a prescriptor that maps $\widehat{Y}$ to a decision. The feature extractor involves one hidden layer with 64 neurons and ReLU activation functions and an output layer with 1 neuron and a Softplus activation function. The prescriptor subsequently solves the deterministic economic dispatch problem $\min_{a \in \mathcal{A}} \ell(\widehat{Y}, a)$, which outputs the E2E decision.

(CAL): The CAL architecture consists of a feature extractor that maps the observation X to a 6-dimensional feature R and a prescriptor that maps R into the feasible set $\mathcal{A}$. The feature extractor involves one hidden layer with 64 neurons and ReLU activation functions and an output layer with 6 neurons and Sigmoid activation functions, which determine the output of each generator as a percentage of its capacity. A simple rescaling with the generator capacities then yields a decision in $\mathcal{A}$.

Table 4 reports the out-of-sample costs of all data-driven decision strategies corresponding to different observations and sampling frequencies. It repeats the results of Table 2 but also shows the out-of-sample costs that can be earned by observing X only every 60 minutes. These costs are uncertain for two reasons: (1) the neural network weights are randomly initialized, and (2) the training dataset is shuffled before training. We report the mean as well as the standard deviation of the average cost on the test data over 5 replications of the experiment.

[3] https://www.kaggle.com/datasets/theforcecoder/wind-power-forecasting

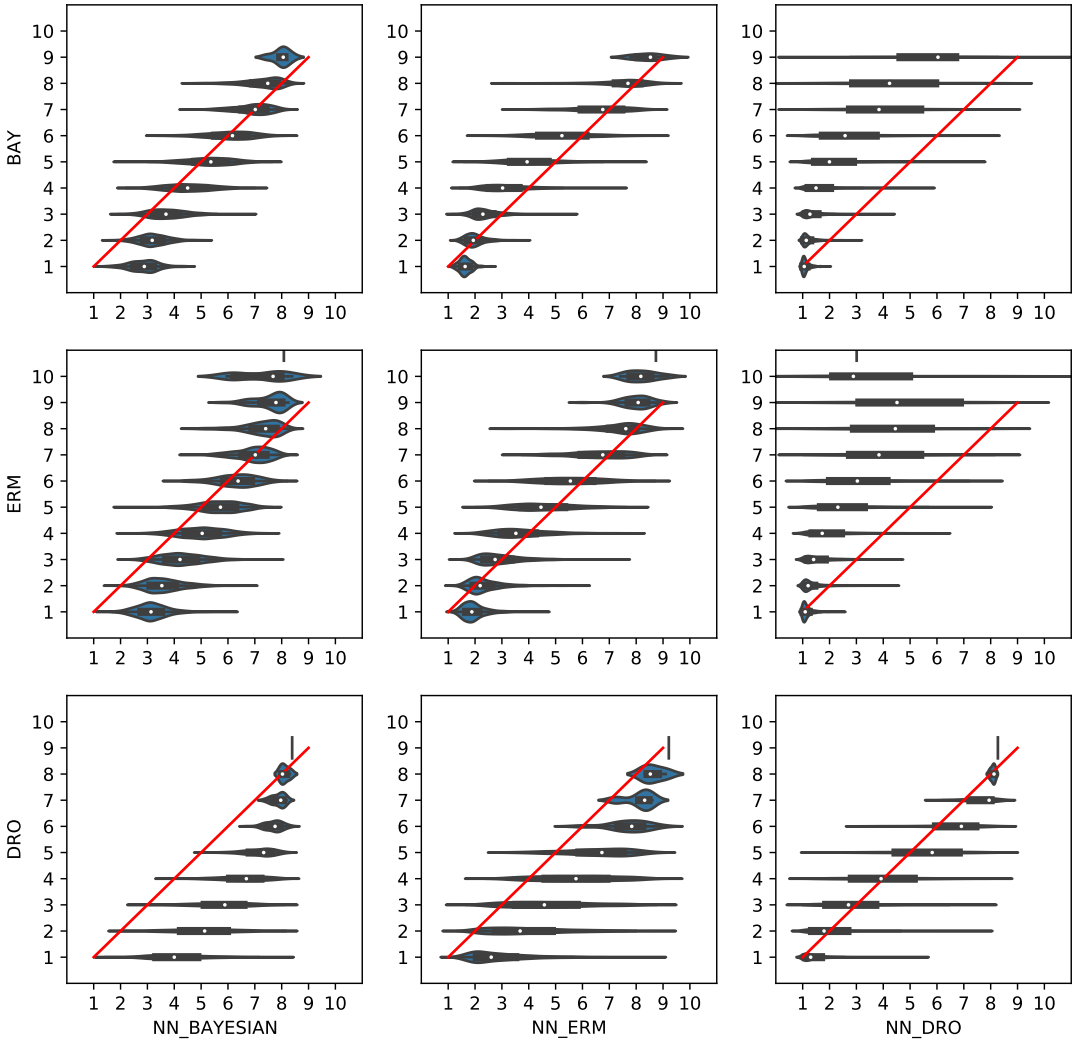


Figure 7. Comparison of the Bayesian, ERM, and DRO strategies against the corresponding neural network-based approximations.

Table 4. Mean and standard deviation (in parentheses) of the average test costs generated by different data-driven strategies for the economic dispatch problem.

Approach	Observation	10 minute frequency	30 minute frequency	60 minute frequency
Baseline	Oracle	60.688	60.691	60.703
	Lag-1	64.959	67.705	70.714
MLE	Myopic	281.436(1.652)	280.103(0.75)	278.686(0.504)
	Myopic Incomp.	348.414(0.0)	348.425(0.0)	348.488(0.0)
	Historical	304.558(11.472)	277.736(2.378)	277.541(5.637)
E2E-CAL	Myopic	68.251(3.73)	66.566(3.068)	69.565(3.727)
	Myopic Incomp.	72.616(0.003)	74.107(2.997)	74.611(3.999)
	Historical	67.088(4.502)	77.06(5.585)	79.827(3.879)
E2E-OPL-Relu	Myopic	72.601(0.0)	72.601(0.0)	72.603(0.0)
	Myopic Incomp.	72.601(0.0)	72.601(0.0)	72.603(0.0)
	Historical	72.601(0.0)	72.601(0.0)	72.603(0.0)
E2E-OPL-Softplus	Myopic	71.326(1.872)	72.527(0.129)	69.312(2.673)
	Myopic Incomp.	72.604(0.006)	72.602(0.002)	72.606(0.005)
	Historical	72.601(0.0)	72.601(0.0)	72.603(0.0)