Distributionally robust end-to-end portfolio construction

GIORGIO COSTA ‡§ and GARUD N. IYENGAR ‡

†Millennium Management, New York, NY, 10022, USA
‡Department of IEOR, Columbia University, New York, NY, 10027, USA

(Received 19 October 2022; accepted 10 July 2023; published online 8 August 2023)

We propose an end-to-end distributionally robust system for portfolio construction that integrates the asset return prediction model with a distributionally robust portfolio optimization model. We also show how to learn the risk-tolerance parameter and the degree of robustness directly from data. End-to-end systems have an advantage in that information can be communicated between the prediction and decision layers during training, allowing the parameters to be trained for the final task rather than solely for predictive performance. However, existing end-to-end systems are not able to quantify and correct for the impact of model risk on the decision layer. Our proposed distributionally robust end-to-end portfolio selection system explicitly accounts for the impact of model risk. The decision layer chooses portfolios by solving a minimax problem where the distribution of the asset returns is assumed to belong to an ambiguity set centered around a nominal distribution. Using convex duality, we recast the minimax problem in a form that allows for efficient training of the end-to-end system.

Keywords: Portfolio optimization; Asset allocation; Machine learning; Distributionally robust optimization; Statistical ambiguity

1. Introduction

Quantitative asset management methods typically predict the distribution of future asset returns by a parametric model, which is then used as input by the decision models that construct the portfolio of asset holdings. This two-stage ‘predict-then-optimize’ design, though intuitively appealing, is effective only under ideal conditions; i.e. when market conditions are stationary and there is sufficient data, the resulting portfolios have good performance. However, in practice, predictions are often unreliable, and consequently, the resulting portfolios have poor out-of-sample performance (Merton 1980, Best and Grauer 1991, Broadie 1993, Chopra and Ziemba 1993). There are two mitigation strategies: add a measure of risk to control variability, and allow for model robustness. However, these approaches involve unknown parameters that are hard to set in practice (Bertsimas et al. 2018).

In decision-making systems, we are concerned with decision errors instead of just prediction errors. Although it is well known that splitting the prediction and optimization task is not optimal for mitigating decision errors, it is only recently that the two steps have been combined into a ‘smart-predict-then-optimize’ framework where parameters are chosen to optimize the performance of the corresponding decision (Elmachtoub et al. 2020). This framework has been implemented as an end-to-end system where one goes directly from data to decision by combining the prediction and decision layers, and back-propagating the gradient information through both decision and prediction layers during training (Donti et al. 2017). In the context of portfolio construction, such an end-to-end system allows us to learn the parameters of a given parametric prediction model, improving the performance of a given fixed portfolio selection problem. However, these end-to-end systems cannot accommodate robust or distributionally robust decision layers that provide robustness with respect to the prediction model, nor can they accommodate optimization layers with learnable parameters.

We show that the end-to-end approach can be successfully extended to settings where the decision is chosen by solving a distributionally robust optimization problem. Introducing robustness regularizes the decision and improves its out-of-sample performance. This is a natural next step in the evolution of this approach. Although we use portfolio selection as a test case for the robust end-to-end framework.
since predictions, decisions and uncertainty are especially important in this problem, it will be clear from our model development that the approach itself can be applied to any robust decision problem.

1.1. Contributions

Our main contribution is to show how to accommodate model robustness within an end-to-end system in a tractable and intuitive fashion. The specific contributions of this paper are as follows (see also Figure 1):

(i) We propose an end-to-end portfolio construction system where the decision layer is a distributionally robust (DR) optimization problem (see the ‘Decision layer’ box in Figure 1). We show how to integrate the DR layer with any prediction layer.

(ii) The DR optimization problem requires both the point prediction as well as the prediction errors as inputs to quantify and control the model risk. Therefore, unlike standard end-to-end systems, we provide both the point prediction, as well as a set of past prediction errors as inputs to the decision layer (see the ‘Prediction layer’ box in Figure 1).

(iii) The DR layer is formulated as a minimax optimization problem where the objective function is a combination of the mean loss and the worst-case risk term that combines the impact of variability and model error. The worst-case risk is taken as a ‘distance’ $\delta$ of the empirical measure. We show that, by using convex duality, the minimax problem can be recast as a minimization problem. In this form, the gradient of the overall task loss can be back-propagated through the DR decision layer to the prediction layer. Moreover, this means the end-to-end system remains computationally tractable and can be efficiently solved using existing software. This result is of interest to embedding any minimax (or maximin) decision layer into an end-to-end system.

(iv) We show that the parameters that control the risk appetite ($\gamma$) and model robustness ($\delta$) can be learned directly from data in our proposed DR end-to-end system. Learning $\gamma$ is relatively straightforward.

However, showing that $\delta$ can be learned in the end-to-end system is non-trivial, and requires the use of convex duality. This step is very important because setting appropriate values for these parameters is, in practice, difficult and computationally expensive (Bertsimas et al. 2018).

(v) We have implemented our proposed DR end-to-end system for portfolio construction in Python. Source code is available at https://github.com/Iyengar-Lab.

1.2. Background and related work

Many different machine learning techniques have been successfully applied to many areas in quantitative finance, from time series modeling and portfolio hedging to trading strategies and derivatives pricing (e.g. see De Spiegeleer et al. 2018, Buehler et al. 2019, Wiese et al. 2020, Koshiyama et al. 2021). Recently, there has been increasing interest in end-to-end learning systems (Bengio 1997, LeCun et al. 2021). These systems come in two varieties: model-free and model-based. Model-free methods have the advantage of retaining some level of interpretability while also being more data-efficient during training. On the other hand, model-based methods rely on some predefined structure of their environment before model training can take place. Model-based methods have the advantage of retaining some level of interpretability while also being more data-efficient during training (Amos et al. 2018).

Model-based end-to-end systems have found successful applications in asset management and portfolio construction. In addition to the model-free system, Uysal et al. (2021) also propose a model-based risk budgeting portfolio construction model. Butler and Kwon (2023) propose an end-to-end mean-variance optimization model. Zhang et al. (2021) integrate
the convex optimization layer with a deep learning prediction layer for portfolio construction.

As a stand-alone tool, portfolio optimization has been criticized for its sensitivity to model and parameter errors that lead to poor out-of-sample performance (Merton 1980, Best and Grauer 1991, Broadie 1993, Chopra and Ziemba 1993). Robust optimization methods that explicitly model perturbation in the parameters of an optimization problem, and choose decisions assuming the worst-case behavior of the parameters, have been successfully employed to improve the performance of portfolio selection (e.g. see Goldfarb and Iyengar 2003, Tütüncü and Koenig 2004, Fabozzi et al. 2007, Kim et al. 2014, 2018, Costa and Kwon 2020, Xidonas et al. 2020, Yin et al. 2021). The robust optimization approach was subsequently extended to distributionally robust optimization (DRO) (Scarf 1958, Delage and Ye 2010, Ben-Tal et al. 2013), where the parameters of an optimization problem are distributed according to a probability measure that belongs to a given ambiguity set. The DRO problem can be interpreted as a game between the decision-maker who chooses an action to minimize cost, and an adversary (i.e. nature) who chooses a parameter distribution that maximizes the cost (Von Neumann 1928). The DRO approach has been implemented for portfolio optimization (e.g. see Calafiore 2007, Delage and Ye 2010, Costa and Kwon 2022).

We propose a model-based end-to-end portfolio selection framework where the decision layer is a DR portfolio selection problem. In keeping with existing end-to-end systems, we allow the prediction layer to be any differentiable supervised learning model, ranging from simple linear models to deep neural networks. However, unlike existing end-to-end frameworks, the decision layer is a minimax problem. We show how to use the errors from the prediction layer to construct this minimax problem. It is well known that a differentiable optimization problem can be embedded into the architecture of a neural network, allowing for gradient-based information to be communicated between prediction and decision layers (Amos and Kolter 2017, Dotti et al. 2017, Agrawal et al. 2019, Amos 2019). Using convex duality, we extend this result to minimax problems, and show how to communicate the gradient information from the DR decision layer back to the prediction layer.

We note that the task loss function that sets the objective for the end-to-end system need not be the same as the objective function in the decision layer. The discrepancy between these two functions stems from the fact that we want to exploit the flexibility of approximating the task loss function with a computationally tractable convex function. Convexity ensures that a local optimum is globally optimal, and there exists a dual optimal solution that can be used to drive parameter learning under an end-to-end framework. This approach has been used in the literature. In the end-to-end system proposed in Uysal et al. (2021), the decision layer is designed to diversify financial risk, whereas the task loss emphasizes financial return. In this setting, the end-to-end system is fundamentally similar to a reinforcement learning problem.

1.3. Outline

The outline of the rest of this paper is the following. In Section 2 we present the end-to-end portfolio construction system, and discuss its components and their different configurations in detail. In Section 3 we discuss the results of our numerical experiments that evaluate the multiple configurations of the end-to-end system. In Section 4 we summarize the findings and contributions of this paper.

2. End-to-end portfolio construction

In this section, we present our proposed DR end-to-end portfolio construction system and describe each individual component. To allow for a natural progression, the structure of this section follows the ‘forward pass’ of Figure 1, starting with a discussion of the prediction layer, followed by the DR decision layer, and finally the task loss function. We conclude by presenting the complete DR end-to-end algorithm.

2.1. Prediction layer

We consider portfolio selection in discrete time. Denote the present time as \( t \) and let \( x_t \in \mathbb{R}^m \) denote \( m \) financial factors (i.e. predictive features) observed at time \( t \). Using these factors, we want to predict the random return \( \tilde{y}_t \in \mathbb{R}^n \) on \( n \) assets over the period \( [t, t+1] \). Let \( \{ x_j \in \mathbb{R}^m : j = t - T, \ldots, t - 1 \} \) denote the historical time series of financial factors and let \( \{ y_j \in \mathbb{R}^n : j = t - T, \ldots, t - 1 \} \) denote the historical time series of returns on the \( n \) assets with \( T \) time steps.

Suppose we have access to the features \( x_t \). A prediction model \( g_t : \mathbb{R}^m \rightarrow \mathbb{R}^n \) that maps \( x_t \) to the prediction \( \hat{y}_t \), of the expected return \( \mathbb{E}[y_j] \) is assumed be a differentiable function of the parameter \( \theta \); otherwise the model may be as simple or as complex as required. An illustrative example is the linear model,

\[
\hat{y}_t = g_{\theta}(x_t) = \theta^\top x_t,
\]

where \( \tilde{y}_t \in \mathbb{R}^n \) is a prediction and \( \theta \in \mathbb{R}^{m \times n} \) is the matrix of weights for this specific model. Note that the dimensions of \( \theta \) will change depending on the structure of the prediction model.

Let \( \tilde{\epsilon}_t, \epsilon_t = \tilde{y}_t - \hat{y}_t = \tilde{y}_t - g_{\theta}(x_t) \in \mathbb{R}^n \) denote the prediction error. Note that the prediction error is a combination of stochastic noise (i.e. variance) and model risk. We assume that the set of past prediction errors \( \{ \epsilon_j = y_j - g_{\theta}(x_j) : j = t - T, \ldots, t - 1 \} \) are \( T \) IID samples of \( \tilde{\epsilon}_t \). This sample set of prediction errors will be used to introduce distributional robustness in the decision layer.

Traditionally, prediction models are trained by minimizing a prediction loss function to improve predictive accuracy. However, an end-to-end system is concerned with minimizing the task loss rather than the prediction loss. In our case, the task loss corresponds to some measure of out-of-sample portfolio performance, which we will discuss in Section 2.3. For now, we will focus on how to use the set of prediction
errors to introduce distributional robustness into the decision layer.

2.2. Decision layer

A feed-forward neural network is trained by iterating over ‘forward’ and ‘backward’ passes. During the ‘forward pass’, the network is evaluated using the current set of weights. This is followed by the ‘backward pass’, where the gradient of the loss function is computed and propagated backwards through the layers of the neural network in order to update the weights.

In existing end-to-end systems, during the forward pass, the decision layer is treated as a standard convex minimization problem. On the other hand, the backward pass requires that we differentiate through the ‘argmin’ operator (Donti et al. 2017). In general, the solutions of optimization problems cannot be written as explicit functions of the input parameters, i.e. they generally do not admit a closed-form solution that cannot be written as explicit functions of the input parameters, et al. 2019.

First, we adapt the existing end-to-end system to solve a portfolio selection problem involving risk measures. This extension requires us to work with the set of prediction errors \( \epsilon \) approximating the expected return by the output \( \hat{y} \). Next, we show how to extend the methodology to distributionally robust portfolio selection.

A portfolio is a vector of asset weights \( z \in \mathbb{Z} \). In order to keep the exposition simple, the set of admissible portfolios is

\[
\mathcal{Z} \triangleq \{ z \in \mathbb{R}^n : z \succeq 0, \ 1^\top z = 1 \}.
\]

The equality constraint in \( \mathcal{Z} \) is the budget constraint and it ensures that the entirety of our available budget is invested, while the non-negativity constraint disallows the short selling of financial assets. Our methodology extends to sets defined by limits on industry/sector exposures and leverage constraints.

Suppose at time \( t \) we have access to the factors \( x_t \) but not to the realized asset returns \( y_t \) over the period \( [t, t+1] \). We approximate the expected return by the output \( \hat{y}_t = g_\theta(x_t) \) of the prediction layer. It follows that the portfolio expected return is \( \hat{y}_t^\top z \). Next, we characterize the variability in the portfolio return both due to the stochastic noise (i.e. variance) and the model risk. Given that we allow the prediction layer to have any general form, we avoid attempting to measure the parametric uncertainty associated with the prediction layer weights. Instead, we take a data-driven approach and estimate the combined effect of variance and model risk directly from a sample set of past prediction errors. We quantify the ‘risk’ associated with the portfolio \( z \) by a deviation risk measure (Rockafellar et al. 2006) defined below.

Proposition 2.1 Deviation risk measure Let \( \epsilon = \{ \epsilon_j \in \mathbb{R}^n : j = 1, \ldots, T \} \) denote the finite set of prediction error outcomes and let \( z \in \mathbb{Z} \subseteq \mathbb{R}^n \) denote a fixed portfolio. Suppose \( R : \mathbb{R} \to \mathbb{R}_+ \cup +\infty \) is a closed convex function where \( R(0) = 0 \) and \( R(X) = R(-X) \). Let \( p \) denote any probability mass function (PMF) in the probability simplex \( \mathcal{Q} \triangleq \{ p \in \mathbb{R}^T : p \succeq 0, \ 1^\top p = 1 \} \).

Then, the deviation risk measure \( f_\epsilon(z, p) \) associated with the set of outcomes \( \epsilon \), the portfolio \( z \) and PMF \( p \) is given by

\[
f_\epsilon(z, p) = \min_{f} \sum_{t=1}^{T} p_t \cdot R(\epsilon_j^\top z - c). \tag{1}
\]

The deviation risk measure \( f_\epsilon(z, p) \) has the following properties.

(i) \( f_\epsilon(z, p) : \mathcal{Z} \to \mathbb{R}_+ \) is convex for any fixed \( p \in \mathcal{Q} \).

(ii) \( f_\epsilon(z, p) \geq 0 \) for all \( z \in \mathcal{Z} \) and \( p \in \mathcal{Q} \).

(iii) \( f_\epsilon(z, p) \) is shift-invariant with respect to \( \epsilon \).

(iv) \( f_\epsilon(z, p) \) is symmetric with respect to \( \epsilon \).

Proof See Appendix 1.

Since our ‘deviation risk measure’ pertains to the prediction error rather than financial risk, we use a broader definition of the risk measure as compared to traditional financial risk measures (e.g. see Rockafellar et al. 2006).

The centering parameter \( c \) plays an important role. It is crucial for the shift invariant property 3, and this property in turn implies that the deviation risk associated with the outcomes \( \epsilon \) is the same as that associated with the mean adjusted outcomes \( \{ \epsilon_j - \sum_{k=1}^{K} p_k \cdot \epsilon_k : j = 1, \ldots, T \} \). Thus, the deviation risk measure is really a function of the deviations around the mean. When \( R(X) = X^2 \), the associated deviation risk measure

\[
f_\epsilon(z, p) = \sum_{t=1}^{T} p_t \cdot \left( \epsilon_j^\top z - \sum_{k=1}^{K} p_k \cdot \epsilon_k^\top z \right)^2
\]

is the variance, where the optimal \( c^* = \sum_{k=1}^{K} p_k \cdot c_k^\top z \) (see Appendix 2 for details). Later, we introduce the worst-case deviation risk by taking the maximum over \( p \in \mathcal{P}(\delta) \). In that setting, the centering parameter \( c \) ensures that the adversary cannot increase risk by putting all the weight on the worst \( \epsilon_j \).

2.2.1. Nominal layer. We start with the assumption that every outcome in the set \( \epsilon = \{ \epsilon_j : j = t - T, \ldots, t - 1 \} \) has equal probability. We refer to this as the nominal decision problem. Let \( q \in \mathcal{Q} \) denote a uniform probability distribution (i.e. \( q_j = 1/T \forall j \)). Then, the nominal decision layer computes

\[
z^*_\gamma(\gamma) = \arg\min_{z \in \mathcal{Z}} f_\epsilon(z, q) - \gamma \cdot \hat{y}_t^\top z \tag{2}
\]

where \( \gamma \in \mathbb{R}_+ \) is the risk appetite parameter and \( f_\epsilon \) is a deviation risk measure as defined by Proposition 2.1. The risk appetite \( \gamma \) balances the risk and return of the optimal portfolio \( z^*_\gamma \). As \( \gamma \) is increased, the expected return \( \hat{y}_t^\top z \) increases. The set of optimal portfolios \( \mathcal{Z}^\gamma \triangleq \{ z^*_\gamma(\gamma) \in \mathbb{R}^n : \gamma \geq 0 \} \) generated by different values of \( \gamma \) is a Pareto optimal set of portfolios for twin objectives of risk and return. Amos and
Koller (2017) and Agrawal et al. (2019) show that the partial derivative of the task loss function with respect to \( \gamma \) can be computed; thus, \( \gamma \) can be optimized via gradient descent like any other parameter in the network. In our end-to-end system, the risk appetite \( \gamma \) is learnable from data, meaning we implicitly search for a Pareto optimal portfolio \( \mathbf{z}^* \in \mathbb{Z}^* \) that optimizes the out-of-sample task loss. Even in the setting where the task loss is a risk–return utility function with a user-specified \( \gamma \), the in-sample parameter \( \gamma \) that maximizes the out-of-sample performance corresponding to \( \gamma \) is likely to be different. Thus, even in this special case, allowing the flexibility of learning \( \gamma \) is likely to improve performance.

2.2.2. DR layer. The nominal problem puts equal weight on every sample in the set \( \epsilon \), i.e. the PMF \( \mathbf{p} \) defining the risk measure is the uniform distribution \( q \). We introduce distributional robustness by allowing \( \mathbf{p} \) to deviate from \( q \) to within a certain ‘distance’ (Calafiore 2007, Ben-Tal et al. 2013, Kannan et al. 2020, Costa and Kwon 2022). Let \( I_\delta \triangleq \sum_{j=1}^J \delta_j \cdot \phi(p_j/q_j) : Q \times Q \to \mathbb{R} \), denote a statistical distance function based on a \( \phi \)-divergence (e.g. Kullback–Leibler, Hellinger, chi-squared). Then, the ambiguity set \( \mathcal{P}(\delta) \) for the distribution \( \mathbf{p} \) is given by

\[
\mathcal{P}(\delta) \triangleq \{ \mathbf{p} \in \mathbb{R}^T : \mathbf{p} \succeq 0, \mathbf{1}^\top \mathbf{p} = 1, I_\delta(\mathbf{p}, q) \leq \delta \}.
\]

The size parameter \( \delta \) defines the maximum permissible distance between \( \mathbf{p} \) and the nominal distribution \( q \). The DR decision layer chooses the portfolio \( \mathbf{z}^* \) assuming the worst case behavior of \( \mathbf{p} \in \mathcal{P}(\delta) \), i.e. it solves the following minimax problem:

\[
\mathbf{z}^* = \arg\min_{\mathbf{z} \in \mathbb{Z}} \max_{\mathbf{p} \in \mathcal{P}(\delta)} f_\epsilon(\mathbf{z}, \mathbf{p}) - \gamma \cdot \hat{y}_j^\top \mathbf{z}.
\]

In general, solving minimax problems can be computationally expensive and can also lead to local optimal solutions rather than global solutions. It is straightforward to check that the objective in (3) is convex in \( \mathbf{z} \); however, the concavity property of the deviation risk measure \( f_\epsilon(\mathbf{z}, \mathbf{p}) \) as a function of \( \mathbf{p} \) is not immediately obvious. Nevertheless, we show in Appendix 3 that we can still use convex duality to reformulate the minimax problem (3) into the following minimization problem,

\[
\mathbf{z}^*_\epsilon = \arg\min_{\mathbf{z} \in \mathbb{Z}, \lambda \geq 0, \xi, \epsilon} f^\lambda_\epsilon(\mathbf{z}, \epsilon, \lambda, \xi) - \gamma \cdot \hat{y}_j^\top \mathbf{z},
\]

where

\[
f^\lambda_\epsilon(\mathbf{z}, \epsilon, \lambda, \xi) \triangleq \xi + \delta \cdot \lambda + \frac{\lambda}{T} \sum_{j=1}^T \phi^\epsilon \left( \frac{R(\epsilon_j^\top \mathbf{z} - c) - \xi}{\lambda} \right),
\]

\( \phi^\epsilon(\cdot) \) is the convex conjugate of the \( \phi \)-divergence defining the ambiguity set \( \mathcal{P}(\delta) \), and \( \lambda \geq 0, \xi \in \mathbb{R} \) are auxiliary variables arising from constructing the Lagrangian dual. Note that we have abused our notation of the ‘argmin’ operator in (4) since \( \mathbf{z}^*_\epsilon \) is the only pertinent output.

Tractable reformulations of the DR layer exist for many choices of \( \phi \)-divergence. In Appendix 4, we show that the DR layer can be formulated as a second-order cone problem when \( \phi \) is the Hellinger distance and as a linear optimization problem when \( \phi \) is the Variational distance. Ben-Tal et al. (2013) provide tractable reformulations for other choices of \( \phi \)-divergence.

Recasting the minimax problem into a convex minimization problem allows us to differentiate through the DR layer during training of the end-to-end system (Amos and Koller 2017, Amos 2019). Another benefit of dualizing the inner maximization problem is that the ambiguity sizing parameter \( \delta \) becomes an explicit part of the DR layer’s objective function, and can, therefore, be treated as a learnable parameter of the end-to-end system. Determining the size of an ambiguity set a priori is often a subjective exercise, with many users resorting to a probabilistic confidence level. By treating \( \delta \) as a learnable parameter, we relieve the user from the responsibility of having to assign a value of \( \delta \) a priori.

2.3. Task loss

In standard supervised learning models, the loss function is a measure of predictive accuracy. For example, a popular prediction loss function is the mean squared error (MSE). For a prediction at time \( t \), the loss is

\[
l_{\text{mse}}(\hat{y}_t, y_t) \triangleq \frac{1}{n} \| y_t - \hat{y}_t \|^2_2.
\]

In an end-to-end system, predictive loss measures the performance of only the the prediction layer. However, using the predictive loss to measure the performance of the entire system fails to consider our main objective: the out-of-sample performance of the decision.

Therefore, in contrast to standard supervised learning models, end-to-end systems measure their performance using a ‘task loss’ function, which is chosen in order to train the system based on the out-of-sample performance of the optimal decision. For example, the task loss in Butler and Kwon (2023) has the same form as the objective function of the decision layer, except the predictions are replaced with the corresponding realizations in order to calculate the out-of-sample performance. We allow for the possibility that the task loss is different from the objective function of the decision layer. This allows us to approximate a task loss function that is hard to optimize with a more tractable surrogate. In such cases, the end-to-end system is fundamentally similar to a reinforcement learning problem.

We define the task loss as the financial performance of our optimal portfolio \( \mathbf{z}^*_\epsilon \) measured over some out-of-sample period of length \( v + 1 \) with the realized asset returns \( \{ y_j : j = \)}
Algorithm 1: DR end-to-end system training

\textbf{Input:} \{x_j\}_{j=1}^{T-v} \cdot \{y_j\}_{j=1}^{T_0}

\textbf{Initialize:} system parameters \( \theta, \gamma, \delta \); learning rate \( \eta \); number of epochs \( K \)

1. \textbf{for} \( k = 1, \ldots, K \) \textbf{do}
2. \hspace{1em} \( L \leftarrow 0 \)
3. \hspace{1em} \textbf{for} \( t = T + 1, \ldots, T_0 - v \) \textbf{do}
4. \hspace{2em} \( \hat{y} = \{g_{\theta}(x_j) : j = t - T, \ldots, t\} \)
5. \hspace{2em} \( \epsilon = \{y_j - \hat{y} : j = t - T, \ldots, t - 1\} \)
6. \hspace{2em} \( z^* \leftarrow \arg\min_{z \in Z, \alpha \geq 0, \xi} f_j^z(z, \alpha, \xi) - \gamma \cdot \hat{y}^\top z \)
7. \hspace{2em} \( L \leftarrow L + \frac{1}{T_0 - T - v} \sum_{i=1}^{T} \log (z^*_i, [y_j])_{j=m}^{T} \)
8. \textbf{Update} \( \theta, \gamma, \delta \) with \( \nabla_{\theta}L, \nabla_{\gamma}L, \nabla_{\delta}L \)

\textbf{Output:} \( \theta, \gamma, \delta \)

\[ l_{SR}(z^*_i, [y_j])_{j=m}^{T} \triangleq \frac{\text{mean}(y_j^\top z^*_i)_{j=m}^{T}}{\text{std}(y_j^\top z^*_i)_{j=m}^{T}}, \quad (7) \]

where the operator \( \text{mean}(\cdot) \) calculates the mean of a set, while \( \text{std}(\cdot) \) calculates the standard deviation.

In general, the task loss is any differentiable function of the optimal portfolio \( z^*_i \). In turn, \( z^*_i \) is an implicit function of the parameters \( \theta, \gamma \) and \( \delta \). Therefore, during the backward training pass, we can differentiate the task loss with respect to \( \gamma \) and \( \delta \) in the decision layer, and with respect to \( \theta \) in the prediction layer.

2.4. Training the DR end-to-end system

Our proposed DR end-to-end system for portfolio construction is detailed in Algorithm 1. Recall that \( T \) denotes the number of prediction error samples from which to estimate the portfolio’s prediction error, while \( v \) is the length of the out-of-sample performance window. Additionally, let us define \( T_0 \) as the total number of observations in the full training data set.

In certain settings, a user may be unable or unwilling to integrate the prediction layer with the rest of the system. Alternatively, they may be using a prediction layer that cannot be trained via gradient descent, e.g. a tree-based predictor. In such cases, we assume the prediction layer is fixed, which in turn means that the prediction errors are constant during training.

Nevertheless, we can still pass a sample set of \( T \) prediction errors as an input to the DR layer during training. Since the DR layer is a differentiable convex optimization problem, we are still able to learn values of \( \gamma \) and \( \delta \) that minimize the task loss. In practice, setting the risk appetite parameter \( \gamma \) and model robustness parameter \( \delta \) is difficult, and requires significant effort. For example, Bertsimas et al. (2018) propose using cross-validation techniques to set the level of robustness. Instead, our end-to-end system learns these parameters directly from data as part of the overall training in a much more efficient manner.

3. Numerical experiments

We present the results of five numerical experiments. Each experiment evaluates different characteristics of our proposed DR end-to-end system. The first four experiments are conducted using historical data from the U.S. stock market. The fifth experiment uses synthetic data generated from a controlled stochastic process.

The first experiment provides a holistic measure of financial performance. The second, third and fourth experiments isolate the out-of-sample effect of allowing the end-to-end system to learn the parameters \( \gamma, \delta \) and \( \theta \), respectively. Finally, the fifth experiment evaluates the effect of robustness when working with complex prediction layers.

The numerical experiments were conducted using a code written in Python (version 3.8.5), with PyTorch (version 1.10.0) (Paszke et al. 2019) and Cvxpylayers (version 0.1.4) (Agrawal et al. 2019) used to compute the end-to-end systems. The neural network is trained using the ‘Adam’ optimizer (Kingma and Ba 2014) and the ‘ECOS’ convex optimization solver (Domahidi et al. 2013).

3.1. Competing investment systems

The numerical experiments involve seven different investment systems. The individual experiments compare these systems against each other. Although many of the systems are designed to learn the parameters \( \theta, \gamma \) and \( \delta \), some experiments purposely keep these parameters constant in order to isolate the effect of learning the remaining parameters. The seven investment systems are described below.

(i) Equal weight (EW): Simple portfolio where all assets have equal weight – no prediction or optimization is required and no parameters need to be learned. Equal weight portfolios promote diversity and have been empirically shown to have a good out-of-sample Sharpe ratio (DeMiguel et al. 2009).

(ii) Predict-then-optimize (PO): Two-stage system with a linear prediction layer. The decision layer is given by the nominal problem defined in (2) with \( \gamma \) held constant. No parameters are learned (i.e. once the parameters are initialized, they are held constant).

(iii) Base: End-to-end system that does not incorporate the risk function \( f_\epsilon(z, p) \) and chooses portfolios by solving the optimization problem

\[ z^*_i = \arg\max_{z \in Z} \hat{y}_i^{\top} z. \]
The prediction layer is linear and the only learnable parameter is $\theta$. Note that the base system is equivalent to a system where the variability of the outcome is not impacted by the decision $z_t$ – as was the case in Donti et al. (2017).

(iv) Nominal: End-to-end system with a linear prediction layer and a decision layer corresponds to the nominal problem (2). The learnable parameters are $\theta$ and $\gamma$.

(v) DR: Proposed end-to-end system with a linear prediction layer and a decision layer corresponds to the DR problem (4). We choose the Hellinger distance as the $\phi$-divergence to define the ambiguity set $\mathcal{P}(\delta)$. The learnable parameters are $\theta$, $\gamma$ and $\delta$.

(vi) NN-nominal: End-to-end system with a non-linear prediction layer. The prediction layer is composed of a neural network with either two or three hidden layers. The decision layer corresponds to the nominal problem (2). The learnable parameters are $\theta$, $\gamma$ and $\delta$.

(vii) NN-DR: End-to-end system with a non-linear prediction layer. The prediction layer is composed of a neural network with either two or three hidden layers. The decision layer corresponds to the DR problem (4). The learnable parameters are $\theta$, $\gamma$ and $\delta$.

Additionally, since retaining some degree of predictive accuracy in the prediction layer is often desirable (Donti et al. 2017), we define the task loss as a linear combination of the Sharpe ratio loss in (7) and the MSE loss in (6),

$$l_{\text{task}}(z_t^*, \hat{y}_t, \{y_j\}_{j=1}^{T_{\text{ten}}}) = 0.5 \cdot l_{\text{mse}}(\hat{y}_t, y_t) + l_{\text{sr}}(z_t^*, \{y_j\}_{j=1}^{T_{\text{ten}}}).$$

The look-ahead evaluation period of the task loss from $t$ to $t+v$ consists of one financial quarter (i.e. 13 weeks), which means $v=12$. The weight of 0.5 on the MSE loss was chosen to ensure a reasonable trade-off between out-of-sample performance and prediction error. We expect similar performance with other weights.

3.2. Experiments with historical data

The historical data consisted of weekly asset and feature returns from 07–Jan–2000 to 01–Oct–2021. For the features, we chose to use the Fama-French factors that are standard in the financial literature and practice, and sourced the data from Kenneth French’s database (French 2021) which consist of the weekly returns of eight features. The asset data were sourced from AlphaVantage (www.alphavantage.co) and consist of the weekly returns of 20 U.S. stocks belonging to the S&P 500 index. The selected features and assets are listed in Table 1. To avoid prediction biases, the input–output pairs are lagged by a period of one week, e.g. asset returns $y$ for 14–Jan–2000 are predicted using the feature vector $x$ observed on 07–Jan–2000.

We assessed the explanatory power of the selected features by regressing the assets against each of the individual features. The regressions are performed over the complete time period and incorporate the one-week lag between features and assets. Table 2 presents a heat map of the $p$-values and shows that all features have at least some explanatory power with 90% confidence. Note that this exercise does not measure the amount of variability explained by each feature, nor does it account for any potential collinearity between features. Since these are standard features used in industry, we use all of them in our experiments.

The experimental design and system training are summarized below.

- Experimental setup. For each experiment, the setup of the participating investment systems is outlined in a corresponding table (see Table 3 as an example). These tables indicate the initial values for the parameters $\theta$, $\gamma$ and $\delta$, and whether these parameters were learned during training or they remained constant during the experiment. Some experiments were designed to isolate the effect of learning a specific parameter; in these experiments, other parameters were kept constant even when the investment system could potentially learn them.

- Risk measure. The portfolio error variance is the deviation risk measure $f_e$ for all investment systems. Appendix 2 shows how the portfolio error variance can be cast as a deviation risk measure $f_e$.

- Initialization. For consistency, all linear prediction layers are initialized to the ordinary least squares (OLS) weights. The choice of using linear prediction layers aligns with standard industry practices for the basket of features in Table 1. Additionally, the initial values of the risk appetite parameter $\gamma$ and robustness parameter $\delta$ were sampled uniformly from appropriately defined intervals (see Appendix 5 for the initialization methodology).

- System training. The data is separated by a 60:40 ratio into training and testing sets, respectively. The training set ranges from 07–Jan–2000 to 18–Jan–2013 and serves to train and validate the end-to-end systems. For each point prediction, two years of weekly observations are passed to the system as a representative sample of prediction errors ($T = 104$).

- Validation. The learning rate $\eta$ and the number of training epochs $K$ are tuned via cross-validation. These hyperparameters are tuned through a time series split cross-validation process (Hyndman and Athanasopoulos 2018). Tuning is carried out by training and validating the end-to-end systems over

| Market | Profitability | Investment |
|--------|--------------|------------|
| Size   | ST reversal  | LT reversal|
| Value  | Momentum     |            |

Table 1. List of features and assets.
Table 2. Heat map of $p$-values from regressing assets against individual features.

| Market | Size | Value | Profit. | Invest. | Mom. | ST rev. | LT rev. |
|--------|------|-------|---------|---------|------|---------|---------|
| AAPL   | 0.59 | 0.90  | 0.64    | 0.83    | 0.49 | 0.28    | 0.18    | 0.32    |
| MSFT   | 0.27 | 0.41  | 0.84    | 0.98    | 0.25 | 0.22    | 0.42    | 0.35    |
| AMZN   | 0.31 | 0.07  | 0.71    | 0.19    | 0.53 | 0.10    | 0.11    | 0.28    |
| C      | 0.17 | 0.79  | 0.02    | 0.01    | 0.30 | 0.20    | 0.01    | 0.45    |
| JPM    | 0.25 | 1.00  | 0.00    | 0.47    | 0.31 | 0.15    | 0.08    | 0.03    |
| BAC    | 0.13 | 0.43  | 0.00    | 0.14    | 0.77 | 0.21    | 0.05    | 0.29    |
| XOM    | 0.05 | 0.06  | 0.96    | 0.06    | 0.07 | 0.68    | 0.51    | 0.58    |
| HAL    | 0.54 | 0.24  | 0.61    | 0.19    | 0.40 | 0.15    | 0.13    | 0.85    |
| MCD    | 0.81 | 0.43  | 0.89    | 0.05    | 0.63 | 0.23    | 1.00    | 0.17    |
| WMT    | 0.00 | 0.01  | 0.57    | 0.00    | 0.75 | 0.78    | 0.87    | 0.01    |
| COST   | 0.00 | 0.32  | 0.43    | 0.00    | 0.54 | 0.40    | 0.97    | 0.01    |
| CAT    | 0.73 | 0.59  | 0.96    | 0.14    | 0.29 | 0.35    | 0.62    | 0.73    |
| LMT    | 0.05 | 0.29  | 0.06    | 0.03    | 0.95 | 0.36    | 0.85    | 0.57    |
| INJ    | 0.05 | 0.28  | 0.35    | 0.09    | 0.76 | 0.65    | 0.42    | 0.15    |
| PFE    | 0.08 | 0.71  | 0.81    | 0.96    | 0.55 | 0.62    | 0.52    | 0.37    |
| DIS    | 0.20 | 0.81  | 0.42    | 0.03    | 0.49 | 0.42    | 0.31    | 0.13    |
| VZ     | 0.53 | 0.90  | 0.05    | 0.05    | 0.33 | 0.13    | 0.52    | 0.02    |
| T      | 0.25 | 0.27  | 0.01    | 0.84    | 0.03 | 0.26    | 0.70    | 0.03    |
| ED     | 0.17 | 0.22  | 0.69    | 0.01    | 0.43 | 0.78    | 0.84    | 0.79    |
| NEM    | 0.03 | 0.11  | 0.23    | 0.00    | 0.16 | 0.28    | 0.82    | 0.71    |

Note: Gray shading indicates level of statistical significance.

four separate folds. For each fold, the original training set is divided into a training subset and a validation subset. The first fold uses the first 20% of the training set as the training subset, and the subsequent 20% as the validation set. For each subsequent fold, this is increased to a ratio of 40:20, 60:20 and, finally, 80:20. Tuning is performed over all possible combinations between three possible learning rates, $\eta \in \{0.005, 0.0125, 0.02\}$, and number of epochs $K \in \{30, 40, 50, 60, 80, 100\}$. Once all four folds are completed, the average validation loss is calculated and used to select the optimal hyperparameters that yield the lowest validation loss for each end-to-end system. After selecting the optimal hyperparameters, these are kept constant during the out-of-sample test. The average validation losses of the end-to-end systems from Experiments 1–4 are presented in Table A1 in Appendix 6. The table also highlights the optimal hyperparameters selected for each system.

- **Out-of-sample testing.** The out-of-sample test ranges from 25–Jan–2013 to 01–Oct–2021, with the first prediction taking place on 25–Jan–2013. All systems are retrained using the full training data before the beginning of the out-of-sample test set in order to take advantage of the most recently available data (Kuhn and Johnson 2013). Note that the hyperparameters remain constant with the values selected during the validation stage.

- **Periodic retraining.** Following the best practice for time series models, we retrained the investment systems approximately every two years using all past data available at the time of training. Therefore, the investment systems were trained a total of four times. Before retraining takes place, the prediction layer weights are reset to the OLS weights computed from the corresponding training data set. In addition, the parameters $\gamma$ and $\delta$ are reset to their initial values before retraining takes place.

- **Results.** For each experiment, the out-of-sample results are presented in the form of a wealth evolution plot, as well as a table that summarizes the financial performance of the competing portfolios. The summary statistics include the annualized return, annualized volatility and the Sharpe ratio calculated over the complete investment horizon. Additionally, the average inverse Herfindahl–Hirschman index (IHHI) is included as a measure of portfolio diversification. For our experiments with 20 assets, the EW portfolio has an IHHI equal to 20. Conversely, a perfectly concentrated portfolio would have an IHHI equal to one.

3.2.1. **Experiment 1: general evaluation.** The first experiment is a complete financial ‘backtest’ to evaluate the performance of the DR end-to-end learning system as an asset management tool. To do so, the DR system is compared against four other competing systems presented in Table 3. In this set of experiments, the systems were able to learn all available parameters within their purview.
The out-of-sample financial performance of the five competing investment systems is compared as follows: Figure 2 shows the wealth evolution of the five corresponding portfolios, Figure 3 shows the cumulative Sharpe ratio, and Table 4 presents a summary of the results over the complete investment horizon. The experimental results lead to the following observations.

- **Benefit of using a sample-based model risk measure**: Unlike the base model, the nominal and DR systems integrate a sample-based prediction error into the decision layer. The results in Table 4 clearly show the significance of incorporating prediction errors into the decision layer – both the nominal and DR systems have a higher return and lower volatility than the base system.

- **Impact of end-to-end learning**: When compared against the straightforward predict-then-optimize and equal weight systems, the nominal and DR systems have higher Sharpe ratios on average over the entire investment horizon, highlighting the advantage of end-to-end systems of being able to learn the prediction and decision parameters. We note that the nominal and DR portfolios have a higher volatility due to their pronounced growth in wealth, but nevertheless maintain a high cumulative Sharpe ratio as shown in Figure 3.

- **Distributional robustness**: Comparing the nominal and DR systems, we clearly see the benefit of incorporating robustness into our portfolio selection system. The DR system has a higher Sharpe...
Table 4. Experiment 1 – Results.

| EW | PO | Base | Nom. | DR |
|----|----|------|------|----|
| Return (%) | 15.6 | 13.4 | 16.1 | 23.4 | 20.1 |
| Volatility (%) | 14.9 | 15.3 | 25.0 | 18.8 | 15.5 |
| Sharpe ratio | 1.05 | 0.88 | 0.64 | 1.24 | 1.30 |
| IHHI | 20.0 | 4.18 | 1.00 | 2.21 | 3.37 |

Note: Values are annualized.

Table 5. Experiment 2 – List of models.

| System | Val. | Lrn | Val. | Lrn | Val. | Lrn |
|--------|------|-----|------|-----|------|-----|
| PO OLS | – | – | 0.046 | – | – | – |
| DR OLS | × | – | 0.046 | × | 0.312 | × |

Note: Val, Initial value; Lrn, Learnable.

Table 6. Experiment 2 – Results.

| Return (%) | 13.4 | 12.9 | 12.6 |
| Volatility (%) | 15.3 | 12.8 | 12.8 |
| Sharpe ratio | 0.88 | 1.01 | 0.99 |
| IHHI | 4.18 | 3.49 | 3.59 |

Note: Values are annualized.

3.2.2. Experiment 2: learning $\delta$. The second experiment focused on understanding the improvement in performance by learning $\delta$; therefore, $\theta$ and $\gamma$ were fixed. We focused on three investment systems: the predict-then-optimize (PO) system (since $\theta$ and $\gamma$ are fixed, this is the same as the nominal system), the DR system with constant $\theta$, $\gamma$ and $\delta$; and the DR system with constant $\theta$ and $\gamma$, but with a learnable $\delta$. The difference in performance between the PO system and the DR system with all parameters fixed will highlight the benefit of adding some (not optimized) robustness, and the difference in performance between the DR systems with optimized $\delta$ and fixed $\delta$ is a measure of the impact of size of the uncertainty set. Details of the three systems are presented in Table 5. The out-of-sample financial performance of the three investment systems is summarized in Figure 4 and Table 6. The experimental results lead to the following observations.

- **Impact of robustness without learning**: The results for the PO system and the DR with no parameter learning show that, even when learning is not allowed, robustness has a positive impact on out-of-sample performance.

- **Isolated learning of $\delta$**: The results in Table 6 show that, although adding robustness improves the Sharpe ratio, solely optimizing $\delta$ can be detrimental to the system’s out-of-sample performance.

3.2.3. Experiment 3: learning $\gamma$. The third experiment assessed the out-of-sample impact of learning $\gamma$. Therefore, here we experimented with only those systems involving $\gamma$, i.e. nominal system with constant $\theta$, and a DR system with constant $\theta$ and $\delta$ but with a learnable $\gamma$, with the PO system added as a baseline control. Details of the three systems are given in Table 7.

The out-of-sample financial performance of the three investment systems is summarized in Figure 5 and Table 8. The experimental results lead to the following observations.

- **Impact of robustness**: The results of this experiment once again highlight the out-of-sample benefits of incorporating robustness into the system: the Sharpe ratio of the DR system is the highest. In fact, in this experiment, the DR system had the highest return and lowest volatility.

- **Isolated learning of $\gamma$**: Comparing the PO system with the nominal system, we observe that only learning $\gamma$ may not be beneficial to the out-of-sample performance of a system. However, when robustness is incorporated into the system, the out-of-sample performance is greatly enhanced.

3.2.4. Experiment 4: learning $\theta$. The fourth experiment assesses the out-of-sample impact of learning the prediction layer weights $\theta$. We tested four investment systems: the PO system, the base system, the nominal system with constant $\gamma$, and the DR system with constant $\gamma$ and $\delta$. Details of the four systems are presented in Table 9.
The out-of-sample financial performance of the four investment systems is presented in Figure 6 and Table 10. The experimental results lead to the following observations.

- **Impact of incorporating a deviation risk measure**: The difference in the out-of-sample performance between the base system and the PO system clearly highlights the benefit of the deviation risk measure – even though the base system learns $\theta$, this is not enough to combat the inherent uncertainty of the portfolio returns.

- **Learning $\theta$**: Learning $\theta$ becomes advantageous once a risk measure is added to the decision layer. The PO and nominal systems differ only in that the nominal system is able to learn values of $\theta$ that differ from the OLS weights. Thus, as suggested by Donti et al. (2017), learning $\theta$ enhances the mapping of the prediction layer from the feature space to the asset space to extract a higher quality decision, but only if the impact of prediction error in the decision layer is properly modeled.

- **Impact of robustness**: Comparing the nominal and DR systems, we can see that incorporating robustness may not always be advantageous, in particular when the robustness sizing parameter $\delta$ is not optimally calibrated. The results in Table 10 indicate that $\delta$ was set too conservative – even though the volatility of the DR system is lower than the
Figure 6. Experiment 4 – Wealth evolution.

Table 10. Experiment 4 – Results.

|                  | PO  | Base | Nom. | DR  |
|------------------|-----|------|------|-----|
| Return (%)       | 13.4| 16.1 | 23.0 | 22.3|
| Volatility (%)   | 15.3| 25.0 | 19.1 | 18.5|
| Sharpe ratio     | 0.88| 0.64 | 1.21 | 1.21|
| IHHI             | 4.18| 1.00 | 2.18 | 2.81|

Note: Values are annualized.

nominal, it incurs a large opportunity cost with regards to the portfolio return.

3.3. Experiment with synthetic data

We simulate an investment universe with 10 assets and 5 features. The features were assumed to follow an uncorrelated zero-mean Brownian motion. The asset returns are generated from a linear model of the features,

\[ y_t = \alpha + \beta^\top x_t + \xi_t + \kappa_t \cdot \omega_t, \]

where \( \alpha \sim U(0, 0.015) \in \mathbb{R}^n \) is the vector of biases, \( \beta \in \mathbb{R}^{m \times n} \) where \( \beta_{ij} \sim \mathcal{N}(0, 0.015) \forall \ i, j \) is the matrix of weights (i.e. loadings), \( \xi_i \in \mathbb{R}^n \) where \( \xi_i \sim \mathcal{N}(0, 0.015) \forall \ i \) is a Gaussian noise vector, \( \omega_i \in \mathbb{R}^n \) where \( \omega_i \sim \text{Exp}(0.015) \forall \ i \) is an exponential noise vector, and \( \kappa_t \) is a discrete random variable that takes the value \(-1, 0, 1\) with probability \(0.15, 0.7, 0.15\), respectively, and serves to periodically perturb the asset return process bidirectionally with the magnitude of \( \omega_i \) in order to simulate ‘jumps’ in the asset returns.

The experimental data set is composed of 1200 observations, with the first 840 reserved for training and the remaining 360 reserved for testing. However, unlike previous experiments, the synthetic data allows us to use a single fold for validation. Here, the complete training set is separated into a single training subset and single validation subset. The validation results are shown in Table A2 in Appendix 6. The table also highlights the optimal hyperparameters selected for each system.

The fifth experiment explored the advantage of robustness when the prediction layer is more complex, e.g. a neural network with multiple hidden layers. Specifically, this experiment compares nominal and DR systems when the prediction layer is either a linear model, or neural network with two or three hidden layers. The neural networks had fully connected hidden layers, and the activation functions were rectified linear units (ReLUs).

The prediction layer in this experiment is initialized to random weights using the standard PyTorch (Paszke et al. 2019) initialization mechanism. This applies to all three prediction layer designs tested in this experiment. The initial values of the risk appetite parameter \( \gamma \) and robustness parameter \( \delta \) were sampled uniformly from the same intervals as the previous experiments (see Appendix 5 for the initialization methodology).

The objective of the experiment is to investigate whether robustness enhances the out-of-sample performance of a system with a more complex prediction layer, i.e. with a prediction layer that is more difficult to train accurately. To avoid biases pertaining to the design of the prediction layer, our assessment is based on a pairwise comparison between the two and three-layer systems. Details of the four investment systems are presented in Table 11.

The out-of-sample financial performance of the four investment systems is summarized in Figure 7 and Table 12. As previously noted, our assessment is based on a pairwise comparison between the linear, and two- and three-layer systems. The experimental results lead to the following observation.

• Impact of robustness: As indicated by the Sharpe ratios in Table 12, introducing distributional robustness greatly enhances the portfolio out-of-sample performance in all three cases. Recall that, unlike
previous experiments where the prediction layers were initialized to the naturally intuitive OLS weights, the prediction layer in the current systems are initialized to fully randomized weights. Thus, through this experiment we can appreciate how robustness protects the portfolios from model error, particularly as the complexity of the prediction layer increases.

4. Conclusion

This paper introduces a novel DR end-to-end learning system for portfolio construction. Specifically, the system integrates a prediction layer of asset returns with a decision layer that selects portfolios by solving a DR optimization problem that explicitly incorporates model risk.

The decision layer in our end-to-end system consists of a DR portfolio selection problem that gets as input both the point prediction and the prediction errors. The prediction errors are used to define a deviation risk measure, and we penalize the performance of a portfolio with the worst-case risk over a set of probability measures. We show that, even though the deviation risk measure is not a linear function of the probability measure, one can still use convex duality to reformulate the minimax DR optimization problem into an equivalent minimization problem. This reformulation is critical to ensure that the gradients with respect to the task loss can be back-propagated to the prediction layer. Numerical experiments clearly show that incorporating model robustness via a DR layer leads to enhanced financial performance.

The parameters that control risk and robustness are learnable as part of the end-to-end system, meaning these two parameters are optimized directly from data based on the system’s out-of-sample performance. The learnability of the robustness parameter is a consequence of the convex duality that we use to reformulate the DR layer. Setting these two parameters in practice is a challenge, and our end-to-end system relieves the user from having to set them. Furthermore, our numerical experiments show that these parameters do significantly impact performance.

We have implemented our proposed robust end-to-end system in Python, and have made the code available on GitHub. We anticipate the DR approach to be very impactful for any application where model risk is an important consideration. Portfolio construction is only the beginning.
Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was supported by Natural Sciences and Engineering Research Council of Canada [PDF - 557467 - 2021].

Data availability

The data that support the numerical experiments in this study are available online from the following two sources below.

- Feature data: These data are openly available through Kenneth French’s Data Library at https://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html.
- Asset data: AlphaVantage at www.alphavantage.co. Restrictions apply to the availability of these data, which were used under a free academic license for this study.

ORCID

Giorgio Costa @ http://orcid.org/0000-0002-2119-5295
Garud N. Iyengar @ http://orcid.org/0000-0001-6546-4154

References

Agrawal, A., Amos, B., Barratt, S., Boyd, S., Diamond, S. and Kolter, J.Z., Differentiable convex optimization layers. Adv. Neural Inf. Process. Syst., 2019, 32, 9562–9574.
Amos, B., Differentiable optimization-based modeling for machine learning. PhD Thesis, Carnegie Mellon University, 2019.
Amos, B. and Kolter, J.Z., Optnet: Differentiable optimization as a layer in neural networks. In International Conference on Machine Learning, pp. 136–145, 2017 (PMLR).
Amos, B., Rodriguez, I.D.J., Sacks, J., Boots, B. and Kolter, J.Z., Differentiable MPC for end-to-end planning and control. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, pp. 8299–8310, 2018.
Ben-Tal, A., Den Hertog, D., De Waegenaere, A., Melenberg, B. and Rennen, G., Robust solutions of optimization problems affected by uncertain probabilities. Manag. Sci., 2013, 59(2), 341–357.
Bengio, Y. Using a financial training criterion rather than a prediction criterion. Int. J. Neural Syst., 1997, 8(4), 433–443.
Bertsimas, D., Gupta, V. and Kallus, N., Data-driven robust optimization. Math. Program., 2018, 167(2), 235–292.
Best, M.J. and Grauer, R.R., On the sensitivity of mean-variance-efficient portfolios to changes in asset means: Some analytical and computational results. Rev. Financ. Stud., 1991, 4(2), 315–342.
Boyd, S. and Vandenberghe, L., Convex Optimization, 2004 (Cambridge University Press: New York, NY).
Broadie, M., Computing efficient frontiers using estimated parameters. Ann. Oper. Res., 1993, 45(1), 21–58.
Buchler, H., Gonon, L., Teichmann, J. and Wood, B., Deep hedging. Quant Finance, 2019, 19(8), 1271–1291.
Butler, A. and Kwon, R.H., Integrating prediction in mean-variance portfolio optimization. Quant Finance, 2023, 23(3), 429–452.
Calhoun, G.C., Ambiguous risk measures and optimal robust portfolios. SIAM J. Optim., 2007, 18(3), 853–877.
Chopra, V.K. and Ziemba, W.T., The effect of errors in means, variances, and covariances on optimal portfolio choice. J Portf Manag, 1993, 19, 6–11.
Costa, G. and Kwon, R.H., A robust framework for risk parity portfolios. J Asset Manag, 2020, 21, 447–466.
Costa, G. and Kwon, R.H., Data-driven distributionally robust risk parity portfolio optimization. Optim Methods Softw, 2022, 37(5), 1876–1911.
De Spiegeleer, J., Madan, D.B., Reyners, S. and Schoutens, W., Machine learning for quantitative finance: Fast derivative pricing, hedging and fitting. Quant Finance, 2018, 18(10), 1635–1643.
Delage, E. and Ye, Y., Distributionally robust optimization under moment uncertainty with application to data-driven problems. Oper. Res., 2010, 58(3), 595–612.
DeMiguel, V., Garlappi, L. and Uppal, R., Optimal versus naive diversification: How inefficient is the 1/n portfolio strategy?. Rev. Financ. Stud., 2009, 22(5), 1915–1953.
Domahidi, A., Chu, E. and Boyd, S., ECOS: An SOCP solver for embedded systems. In European Control Conference (ECC), pp. 3071–3076, 2013.
Donti, P.L., Amos, B. and Kolter, J.Z., Task-based end-to-end model learning in stochastic optimization. In Proceedings of the 31st International Conference on Neural Information Processing Systems, pp. 5490–5500, 2017.
Elmachtoub, A., Liang, J.C.N. and McNeiIls, R., Decision trees for decision-making under the predict-then-optimize framework. In International Conference on Machine Learning, pp. 2858–2867, 2020 (PMLR).
Elmachtoub, A.N. and Grigas, P., Smart “predict, then optimize”. Manage. Sci., 2022, 68(1), 9–26.
Fabozzi, F.J., Kolm, P.N., Pachamanova, D.A. and Focardi, S.M., Robust portfolio optimization. J. Portf. Manag., 2007, 33(3), 40–48.
French, K.R., Data library, 2021 (Accessed 10 December 2021).
Goldfarb, D. and Iyengar, G., Robust portfolio selection problems. Math. Oper. Res., 2003, 28(1), 1–38.
Hyndman, R.J. and Athanasopoulos, G., Forecasting: Principles and Practice, 2018 (OTexts: Melbourne).
Kannan, R., Bayraksan, G. and Luedtke, J.R., Residuals-based distributionally robust optimization with covariate information, 2020. arXiv preprint arXiv:2012.01088.
Kim, J.H., Kim, W.C. and Fabozzi, F.J., Recent developments in robust portfolios with a worst-case approach. J. Optim. Theory Appl., 2014, 161, 103–121.
Kim, J.H., Kim, W.C. and Fabozzi, F.J., Recent advancements in robust optimization for investment management. Ann. Oper. Res., 2018, 266, 183–198.
Kingma, D.P. and Ba, J., Adam: A method for stochastic optimization, 2014. arXiv preprint arXiv:1412.6980.
Koshiyama, A., Firoozye, N. and Treleaven, P., Generative adversarial networks for financial trading strategies fine-tuning and combination. Quant. Finance, 2021, 21(5), 797–813.
Kuhn, M. and Johnson, K., Applied Predictive Modeling, volume 26. 2013 (Springer: New York, NY).
LeCun, Y., Muller, U., Ben, J., Cosatto, E. and Flepp, B., Off-road obstacle avoidance through end-to-end learning. In Proceedings of the 18th International Conference on Neural Information Processing Systems, pp. 739–746, 2005.
Merton, R.C., On estimating the expected return on the market: An exploratory investigation. J. Financ. Econ., 1980, 8(4), 323–361.
Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J. and Chintala, S., Pytorch: An imperative style, high-performance deep learning library. In Advances in Neural Information Processing Systems 32, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché
Buc, E. Fox, R. Garnett, pp. 8024–8035, 2019 (Curran Associates, Inc.).
Rockafellar, R.T., Uryasev, S. and Zabarankin, M. Generalized deviations in risk analysis. Finance Stoch., 2006, 10(1), 51–74.
Scaf, H. A min-max solution of an inventory problem. In Studies in the mathematical theory of inventory and production, pp. 201–209, 1958.
Sharpe, W.F., The sharpe ratio. J. Portf. Manag., 1994, 21(1), 49–58.
Thomas, R.W., Friend, D.H., DaSilva, L.A. and MacKenzie, A.B., Cognitive networks: Adaptation and learning to achieve end-to-end performance objectives. IEEE Commun. Mag., 2006, 44(12), break 51–57.
Tütüncü, R.H. and Koenig, M., Robust asset allocation. Ann. Oper. Res., 2004, 132(1–4), 157–187.
Uysal, A.S., Li, X. and Mulvey, J.M., End-to-end risk budgeting portfolio optimization with neural networks, 2021. arXiv preprint arXiv:2107.04636.
Vandenberghe, L., The mathemtical theory of investment and production, 2004.

Appendix 1. Proof of Proposition 2.1

Proof By definition, we have that
\[ f_{\epsilon+a}(z,p) = \min_{c} \sum_{j=1}^{T} p_{j} \cdot R\left( (\epsilon_{j} + a)^{\top} z - c \right), \]
\[ = \min_{c} \sum_{j=1}^{T} p_{j} \cdot R\left( \epsilon_{j}^{\top} z - (c - a^{\top} z) \right), \]
\[ = \min_{c} \sum_{j=1}^{T} p_{j} \cdot R\left( \epsilon_{j}^{\top} z - c' \right), \]
where \( c' = c - a^{\top} z \). The result follows from the last expression.

(iv) \( f_{\epsilon}(z,p) \) is symmetric with respect to \( \epsilon \), i.e. \( f_{\epsilon}(z,p) = f_{-\epsilon}(z,p) \).

Proof By definition, we have that
\[ f_{-\epsilon}(z,p) = \min_{c} \sum_{j=1}^{T} p_{j} \cdot R\left( -\epsilon_{j}^{\top} z - c \right), \]
\[ = \min_{c} \sum_{j=1}^{T} p_{j} \cdot R\left( \epsilon_{j}^{\top} z + c \right), \]
\[ = \min_{c} \sum_{j=1}^{T} p_{j} \cdot R\left( \epsilon_{j}^{\top} z - c' \right), \]
where \( c' = -c \), and (A2) follows from the fact that \( R(X) = R(-X) \). The result follows from the last expression.

Appendix 2. Portfolio error variance

Recall that \( \epsilon = [\epsilon_{j} \in \mathbb{R}^{n}: j = 1, \ldots, T] \) denotes the finite set of prediction error outcomes. For a fixed distribution \( p \in \mathcal{Q} \), the expected error and its corresponding covariance matrix are
\[ \mu_{\epsilon}(p) \triangleq \sum_{j=1}^{T} p_{j} \cdot \epsilon_{j}, \] (A3)
\[ \Sigma_{\epsilon}(p) \triangleq \sum_{j=1}^{T} p_{j} \cdot (\epsilon_{j} - \mu_{\epsilon}(p)) (\epsilon_{j} - \mu_{\epsilon}(p))^{\top}, \] (A4)
where \( \mu_{\epsilon}(p) \in \mathbb{R}^{n} \) and \( \Sigma_{\epsilon}(p) \in \mathbb{R}^{n \times n} \). The matrix \( \Sigma_{\epsilon}(p) \) results from the weighted sum of \( T \) rank-1 symmetric matrices, meaning it is guaranteed to be positive semidefinite.

Note that \( \Sigma_{\epsilon}(p) \) is a non-linear function of \( p \); consequently, the portfolio variance \( z^{\top} \Sigma_{\epsilon}(p) z \) is also non-linear in \( p \). Thus, in this form, the worst-case variance \( \max_{p \in \mathcal{P}} z^{\top} \Sigma_{\epsilon}(p) z \) does not have the correct convexity properties that allow one to reformulate the problem using duality. We resolve this issue by recasting the portfolio error variance into the form prescribed by Proposition 2.1, which yields the following corollary.

**Corollary A.1** For a fixed \( z \in \mathbb{Z} \) and \( p \in \mathcal{Q} \), the portfolio variance is
\[ z^{\top} \Sigma_{\epsilon}(p) z = \min_{c} \sum_{j=1}^{T} p_{j} \cdot (\epsilon_{j}^{\top} z - c)^{2}, \]
where \( c \in \mathbb{R} \) is an unrestricted auxiliary variable.
Proof. Let $h(c) = \sum_{j=1}^{T} p_j \cdot (\epsilon_j^T z - c)^2$ and recall that $\sum_{j=1}^{T} p_j = 1$. Then

$$\frac{dh}{dc} = 2 \sum_{j=1}^{T} (p_j \cdot c - p_j \cdot \epsilon_j^T z)$$

$$= 2c - 2 \sum_{j=1}^{T} p_j \cdot \epsilon_j^T z,$$

$$\frac{d^2 h}{dc^2} = 2.$$

Since $d^2 h/dc^2 > 0 \forall c \in \mathbb{R}$, then $h(c)$ is strictly convex in $c$. Therefore, solving for $c$ such that $dh/dc = 0$ suffices to find the unique global minimizer $c^*$,

$$\frac{dh}{dc} |_{c=c^*} = 2c^* - 2 \sum_{j=1}^{T} p_j \cdot \epsilon_j^T z = 0,$$

$$c^* = \sum_{j=1}^{T} p_j \cdot \epsilon_j^T z,$$

for all $p \in \mathcal{Q}$ and $\{\epsilon_j : j = 1, \ldots, T\}$. Then, by (A3), we have

$$c^* = \sum_{j=1}^{T} p_j \cdot \epsilon_j^T z = z^T \sum_{j=1}^{T} p_j \cdot \epsilon_j = z^T \mu_c(p).$$

Now, using (A4), the portfolio error variance is

$$z^T \Sigma_e(p) z = \sum_{j=1}^{T} p_j \cdot z^T (\epsilon_j - \mu_e(p)) (\epsilon_j - \mu_e(p))^T z$$

$$= \sum_{j=1}^{T} p_j \cdot \left(z^T (\epsilon_j - \mu_e(p))\right)^2$$

$$= \sum_{j=1}^{T} p_j \cdot \left(z^T \epsilon_j - z^T \mu_e(p)\right)^2$$

$$= \sum_{j=1}^{T} p_j \cdot (\epsilon_j^T z - c)^2$$

$$= \min_{c} \sum_{j=1}^{T} p_j \cdot (\epsilon_j^T z - c)^2,$$

as desired.

\[ \square \]

Appendix 3. Dualizing the DR layer

Recall that $h_1(c,z,p)$ in (A1) is a convex function in $(c,z)$ for any fixed $p \in \mathcal{Q}$. Moreover, $h_2(c,z,p)$ is linear in $p$ for any fixed $(c,z) \in \mathbb{R} \times \mathcal{Z}$. Thus, $h_4(c,z,p)$ is convex–linear in $(c,z)$ and $p$, respectively.

Fix $z \in \mathcal{Z}$. Then, the minimax theorem for convex duality (Von Neumann 1928) implies that

$$\max_{p \in \mathcal{P}(\delta)} f_\delta(z,p) = \min_{c \in \mathbb{R}} \max_{p \in \mathcal{P}(\delta)} h_4(c,z,p) = \min_{c \in \mathbb{R}} \max_{p \in \mathcal{P}(\delta)} \sum_{j=1}^{T} p_j \cdot R(\epsilon_j^T z - c).$$

Next, we adapt the results in Ben-Tal et al. (2013) to write the maximization over $p$ as a dual minimization problem. Note that this transformation is straightforward, and is only included for completeness.

Fix $(c,z) \in \mathbb{R} \times \mathcal{Z}$. From the definition of the ambiguity set $\mathcal{P}(\delta)$, it follows that the maximization problem in $p$ is

$$\max_{p} \sum_{j=1}^{T} p_j \cdot R(\epsilon_j^T z - c),$$

s.t. $\sum_{j=1}^{T} p_j = 1,$

$$I_\delta(p,q) = \sum_{j=1}^{T} q_j \cdot \phi(p_j/q_j) \leq \delta,$$

$p \geq 0.$

Associate a dual variable $\xi$ with the constraint $\sum_{j=1}^{T} p_j = 1$ and a dual variable $\lambda \geq 0$ with the constraint $I_\delta(p,q) \leq \delta$. Then, the Lagrangian dual function is

$$f_\delta^*(z,c,\lambda,\xi)$$

$$\Delta \max_{p \geq 0} \sum_{j=1}^{T} p_j \cdot R(\epsilon_j^T z - c) + \xi \cdot (1 - 1^T p)$$

$$+ \lambda \cdot \left( \delta - \sum_{j=1}^{T} q_j \cdot \phi(p_j/q_j) \right)$$

$$= \xi + \delta \cdot \lambda$$

$$+ \max_{p \geq 0} \sum_{j=1}^{T} p_j \cdot \left(R(\epsilon_j^T z - c) - \xi \right) - \lambda \cdot q_j \cdot \phi(p_j/q_j)$$

$$= \xi + \delta \cdot \lambda$$

$$+ \max_{p \geq 0} \sum_{j=1}^{T} p_j \cdot \left(R(\epsilon_j^T z - c) - \xi \right) - \lambda \cdot q_j \cdot \phi(s_j)$$

$$= \xi + \delta \cdot \lambda$$

$$+ \sum_{j=1}^{T} q_j \cdot \min_{s \geq 0} \left( s \cdot \left(R(\epsilon_j^T z - c) - \xi \right) - \lambda \cdot \phi(s) \right)$$

$$= \xi + \delta \cdot \lambda$$

$$+ \sum_{j=1}^{T} q_j \cdot (\lambda \phi^*) \left(R(\epsilon_j^T z - c) - \xi \right)$$

$$= \xi + \delta \cdot \lambda$$

$$+ \sum_{j=1}^{T} q_j \cdot (\lambda \phi^*) \left(R(\epsilon_j^T z - c) - \xi \right) \overline{\lambda}.$$ 

Note that we arrive at (A5) by taking the convex conjugate of $\phi$ and by using the identity $(\lambda \phi^*)^*(w) = \lambda \cdot \phi^*(w/\lambda)$ for $\lambda \geq 0$. Additionally, recall that our nominal assumption states that $q_j = 1/T \forall j$.

Since the conjugate $\phi^*$ of a $\phi$-divergence is a convex function, it follows that the Lagrangian function $f_\delta^*(z,c,\lambda,\xi)$ is jointly convex in $(z,c,\lambda,\xi) \in \mathbb{Z} \times \mathbb{R} \times \mathbb{R}_+ \times \mathbb{R}$ (see Ben-Tal et al. 2013 for details). Finally, note that the Lagrangian dual in (A5) above is the same as the distributionally robust deviation risk measure in (5). The above steps demonstrate the equivalence between the minimax problem in (3) and the minimization problem in (4), i.e.

$$\min \max_{c \in \mathbb{R}, \xi \geq 0} \sum_{j=1}^{T} p_j \cdot R(\epsilon_j^T z - c) - \gamma \cdot \tilde{y}_j^T z \iff \min_{z \in \mathbb{Z}, \xi \geq 0} \sum_{j=1}^{T} q_j \cdot (\lambda \phi^*) \left(R(\epsilon_j^T z - c) - \xi \right) \overline{\lambda}.$$
Appendix 4. Tractable reformulations

Recall that the DR layer in (4) corresponds to the following minimization problem,
\[
\min_{\xi, \lambda, \gamma, \zeta} \xi + \delta \cdot \lambda + \beta \sum_{j=1}^{T} \phi^{*} \left( \frac{R(e_j^T z - c) - \xi}{\lambda} \right) - \gamma \cdot \hat{y}_j^T z
\]
s.t. \( \lambda \geq 0 \)
\( \zeta \in Z \)

The computational tractability of this optimization problem depends on the complexity of the \( \phi \)-divergence selected to construct the ambiguity set \( P(\delta) \).

A.1. Hellinger distance

The Hellinger distance is defined as
\[
I_{\phi_H}^H(p, q) \triangleq \sum_{j=1}^{T} q_j \cdot \phi_H(p_j/q_j) = \sum_{j=1}^{T} (\sqrt{p_j} - \sqrt{q_j})^2
\]
where \( \phi_H(w) \triangleq (\sqrt{w} - 1)^2 \) for \( w \geq 0 \). The convex conjugate
\[
\phi_H^*(s) \triangleq \frac{s}{1 - s} \text{ for } s < 1.
\]

If we construct the probability ambiguity set \( P(\delta) \) using the Hellinger distance, then the DR layer becomes a highly non-linear convex optimization problem. Nevertheless, the problem can be reformulated into a tractable problem as follows. Let \( h(s) \triangleq 1/(1 - s) = \phi_H^*(s) + 1 \). Then \( h^{-1}(s) = 1 - 1/s \) and the objective function is
\[
\xi + (\delta - 1) \cdot \lambda + \beta \sum_{j=1}^{T} \lambda \cdot h \left( \frac{R(e_j^T z - c) - \xi}{\lambda} \right) - \gamma \cdot \hat{y}_j^T z.
\]

Next, introduce the auxiliary variable \( \beta \in \mathbb{R}_+^T \) and let
\[
\beta_j \geq \lambda \cdot h \left( \frac{R(e_j^T z - c) - \xi}{\lambda} \right)
\]
\[
\iff h^{-1} \left( \frac{\beta_j}{\lambda} \right) \geq \frac{R(e_j^T z - c) - \xi}{\lambda}
\]
\[
\iff \lambda - \frac{\lambda^2}{\beta_j} \geq R(e_j^T z - c) - \xi.
\]

Note that, since \( \phi_H^*(s) \) requires that \( s < 1 \), then \( \beta \geq 0 \).

Finally, introduce the auxiliary variable \( \tau \in \mathbb{R}_+^T \) to arrive at the tractable reformulation of the Hellinger-based DR layer,
\[
\min_{\xi, \lambda, \gamma, \tau, \beta} \xi + (\delta - 1) \cdot \lambda + \beta \sum_{j=1}^{T} \beta_j - \gamma \cdot \hat{y}_j^T z
\]
s.t. \( \xi + \lambda \geq R(e_j^T z - c) + \tau_j, \quad j = 1, \ldots, T \)
\( \beta_j \tau_j \geq \lambda^2, \quad j = 1, \ldots, T \)
\( \beta, \tau \geq 0, \)
\( \lambda \geq 0, \)
\( \zeta \in Z. \)

Note that the hyperbolic constraint, \( \beta_j \tau_j \geq \lambda^2 \), can be equivalently written as a rotated second-order cone constraint.

A.2. Variation distance

The Variation distance is defined as
\[
I_{\phi_V}^V(p, q) \triangleq \sum_{j=1}^{T} q_j \cdot \phi_V(p_j/q_j) = \sum_{j=1}^{T} |p_j - q_j|
\]
where \( \phi_V(w) \triangleq |w - 1| \) for \( w \geq 0 \). The conjugate
\[
\phi_V^*(s) \triangleq \begin{cases} 
-1, & s \leq -1 \\
0, & -1 \leq s \leq 1 \\
1, & s \geq 1 \end{cases}.
\]

The tractable reformulation presented here is adapted from Ben-Tal et al. (2013). Introduce the auxiliary variable \( \beta \in \mathbb{R}_+^T \). Then,
\[
\min_{\xi, \lambda, \gamma, \beta} \xi + \delta \cdot \lambda + \beta \sum_{j=1}^{T} \beta_j - \gamma \cdot \hat{y}_j^T z
\]
s.t. \( \beta_j \geq -\lambda, \quad j = 1, \ldots, T \)
\( \beta_j \geq R(e_j^T z - c) - \xi, \quad j = 1, \ldots, T \)
\( \lambda \geq R(e_j^T z - c) - \xi, \quad j = 1, \ldots, T \)
\( \lambda \geq 0, \)
\( \zeta \in Z. \)

Appendix 5. Initializing \( \gamma \) and \( \delta \) during experiments

The initial value for the parameter \( \gamma \) was randomly sampled from a uniform distribution over a finite interval. The upper and lower boundaries of this interval are set such that the nominal value of the deviation risk measure, i.e. portfolio error variance, and the nominal return are comparable for the equal weight portfolio, i.e. \( \tau_i = 1/n \) for \( i = 1, \ldots, n \). Note that the error variance was computed using errors corresponding to a linear prediction model with OLS weights.

Using the same training set as Section 3 with a sample set of \( T = 104 \) prediction errors, we obtained samples of plausible \( \gamma \) values as follows,
\[
\hat{\gamma}_j \leq \frac{\tau_j}{\sum_{j=1}^{T} \tau_j} \text{ for } j = T + 1, \ldots, T_0,
\]
where \( \Sigma_j(q) \) is calculated as in (A4). To ensure reasonable emphasis is given to the deviation risk measure, we set the lower and upper bounds for the interval for \( \gamma \) to the 1st and 25th percentiles of the sample set, respectively. This gave us the interval \([0.02, 0.10] \).

The initial value for \( \delta \) was also sampled from a uniform distribution over a finite interval. For the Hellinger distance, which we use for our experiments, the maximum distance is \( \delta_{\text{max}} = \max_p I_{\phi_H}^H(p, q) = 2(1 - 1/\sqrt{T}) \), where the nominal distribution \( q = 1/T \). The theoretical upper bound is a useful benchmark, but we note that this is an extreme and implausible value. Thus, we set the upper bound \( \delta_{\text{up}} = 0.25 \cdot \delta_{\text{max}}, \) while we set the lower bound \( \delta_{\text{lo}} = 0.05 \cdot \delta_{\text{max}} \). In Experiments 1–4, where \( T = 104 \), the sampling interval is \( \delta \in [0.09, 0.45]. \) Initializing \( \delta \) from this interval ensures that the distance between \( p \) and \( q \) is not too extreme at the start of the training process.
Appendix 6. Validation and hyperparameter selection

Table A1. Average validation loss in Experiments 1–4.

| $\eta$ | Epochs | Base | $\theta, \gamma$ | $\theta$ |
|--------|--------|------|-----------------|--------|
| 0.0050 | 30     | $-0.1522$ | $-0.1769$ | $-0.1682$ | $-0.1747$ |
| 0.0125 | 30     | $-0.1130$ | $-0.1784$ | $-0.1672$ | $-0.1854$ |
| 0.0200 | 30     | $-0.1099$ | $-0.1809$ | $-0.1675$ | $\bf{0.1874}$ |
| 0.0050 | 40     | $-0.1372$ | $-0.1793$ | $-0.1680$ | $-0.1733$ |
| 0.0125 | 40     | $-0.1099$ | $-0.1767$ | $-0.1675$ | $-0.1828$ |
| 0.0200 | 40     | $-0.1044$ | $-0.1819$ | $-0.1675$ | $-0.1848$ |
| 0.0050 | 50     | $-0.1278$ | $-0.1820$ | $-0.1679$ | $-0.1727$ |
| 0.0125 | 50     | $-0.1091$ | $-0.1772$ | $-0.1674$ | $-0.1839$ |
| 0.0200 | 50     | $-0.1079$ | $-0.1887$ | $-0.1683$ | $-0.1854$ |
| 0.0050 | 60     | $-0.1267$ | $-0.1853$ | $-0.1679$ | $-0.1754$ |
| 0.0125 | 60     | $-0.1094$ | $-0.1807$ | $-0.1677$ | $-0.1846$ |
| 0.0200 | 60     | $-0.0978$ | $\bf{-0.1950}$ | $\bf{-0.1684}$ | $-0.1838$ |
| 0.0050 | 80     | $-0.1069$ | $-0.1888$ | $-0.1679$ | $-0.1792$ |
| 0.0125 | 80     | $-0.1036$ | $-0.1804$ | $-0.1682$ | $-0.1846$ |
| 0.0200 | 80     | $-0.0858$ | $-0.1907$ | $-0.1679$ | $-0.1703$ |
| 0.0050 | 100    | $-0.1032$ | $-0.1914$ | $-0.1679$ | $-0.1815$ |
| 0.0125 | 100    | $-0.1045$ | $-0.1740$ | $-0.1679$ | $-0.1791$ |
| 0.0200 | 100    | $-0.0932$ | $-0.1909$ | $-0.1680$ | $-0.1523$ |

Table A2. Validation loss in Experiment 5.

| $\eta$ | Epochs | Linear | $\theta, \gamma$ | $\theta$ |
|--------|--------|--------|-----------------|--------|
| 0.0050 | 20     | $-0.2255$ | $-0.1167$ | $-0.1376$ |
| 0.0125 | 20     | $-0.2391$ | $-0.0725$ | $-0.1966$ |
| 0.0200 | 20     | $-0.2677$ | $\bf{-0.2475}$ | $\bf{-0.3014}$ |
| 0.0050 | 40     | $-0.2483$ | $-0.2237$ | $-0.1377$ |
| 0.0125 | 40     | $-0.2740$ | $-0.2270$ | $-0.1965$ |
| 0.0200 | 40     | $-0.2564$ | $-0.2140$ | $-0.2481$ |
| 0.0050 | 60     | $\bf{-0.2796}$ | $-0.2346$ | $-0.1377$ |
| 0.0125 | 60     | $-0.2135$ | $-0.1729$ | $-0.1965$ |
| 0.0200 | 60     | $-0.2628$ | $-0.1796$ | $-0.2545$ |

Notes: (1) Average validation loss of end-to-end systems over four training folds. (2) The lowest validation score of each system is **bolded**.

| $\eta$ | Epochs | Linear | $\theta, \gamma$ | $\theta$ |
|--------|--------|--------|-----------------|--------|
| 0.0050 | 20     | $-0.0603$ | $-0.1373$ | $-0.0601$ |
| 0.0125 | 20     | $-0.0677$ | $-0.1968$ | $-0.0601$ |
| 0.0200 | 20     | $-0.0408$ | $-0.2576$ | $-0.0645$ |
| 0.0050 | 40     | $-0.0603$ | $-0.0645$ | $-0.0601$ |
| 0.0125 | 40     | $-0.0408$ | $-0.1758$ | $-0.0404$ |
| 0.0200 | 40     | $-0.0602$ | $-0.0832$ | $-0.0404$ |
| 0.0050 | 60     | $-0.0603$ | $-0.1777$ | $-0.0603$ |
| 0.0125 | 60     | $-0.0602$ | $-0.0832$ | $-0.0404$ |
| 0.0200 | 60     | $-0.0405$ | $-0.1353$ | $-0.0404$ |

Notes: (1) Validation loss of end-to-end systems over a single training fold. (2) The lowest validation score of each system is **bolded**.