Distributionally Robust End-to-End Portfolio Construction

8th Annual Bloomberg-Columbia
Machine Learning in Finance Workshop

September 2022

Giorgio Costa and Garud N. Iyengar

Industrial Engineering and Operations Research
COLUMBIA | ENGINEERING

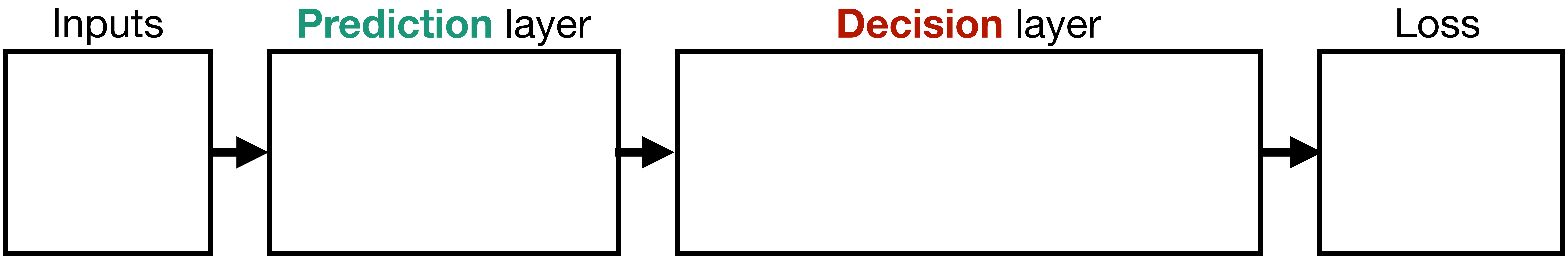
Distributional robustness in end-to-end learning systems

Distributional robustness in end-to-end learning systems

- **End-to-end**: Integrate the prediction and optimization steps.

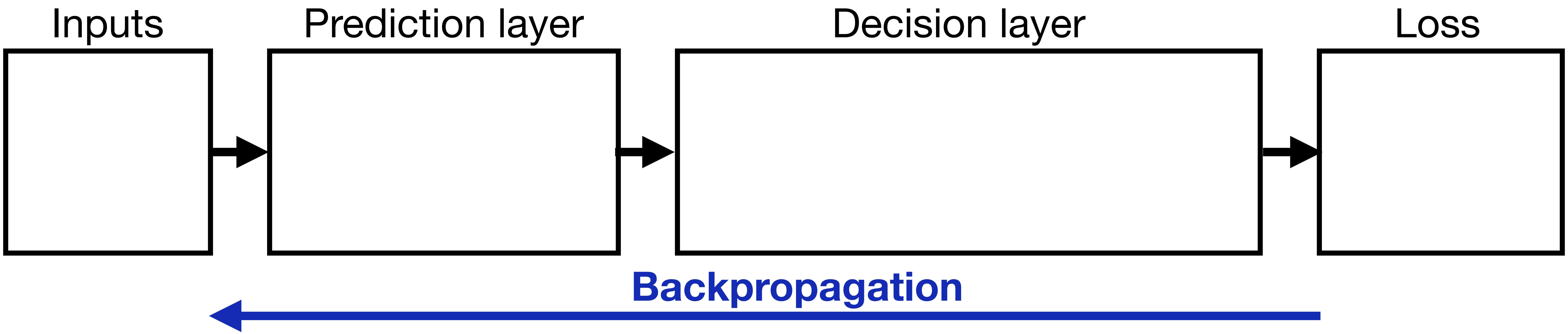
Distributional robustness in end-to-end learning systems

- **End-to-end**: Integrate the prediction and optimization steps.



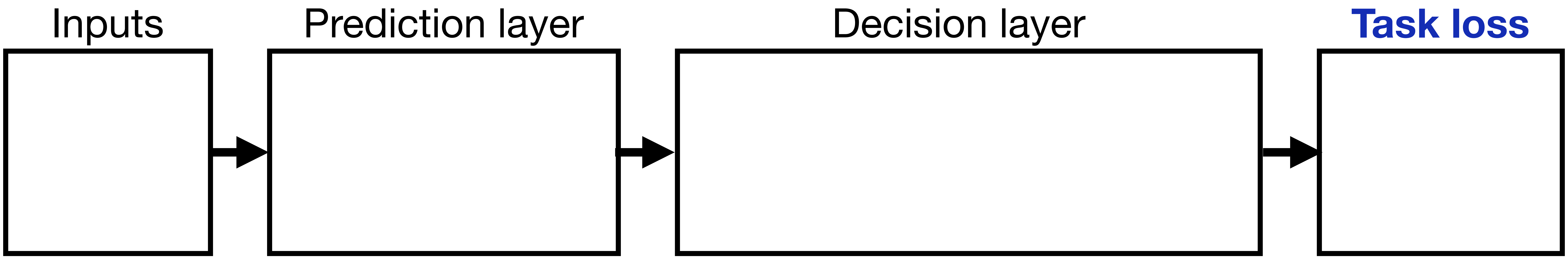
Distributional robustness in end-to-end learning systems

- **End-to-end**: Integrate the prediction and optimization steps.
  - Information is passed between prediction and decision layers during training.

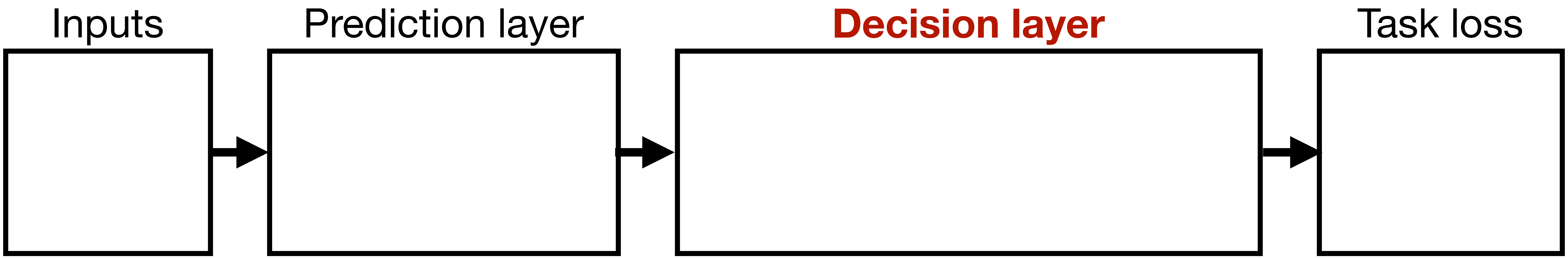


Distributional robustness in end-to-end learning systems

- **End-to-end**: Integrate the prediction and optimization steps.
  - Information is passed between prediction and decision layers during training.
  - Training is based on the final task rather than predicted performance.

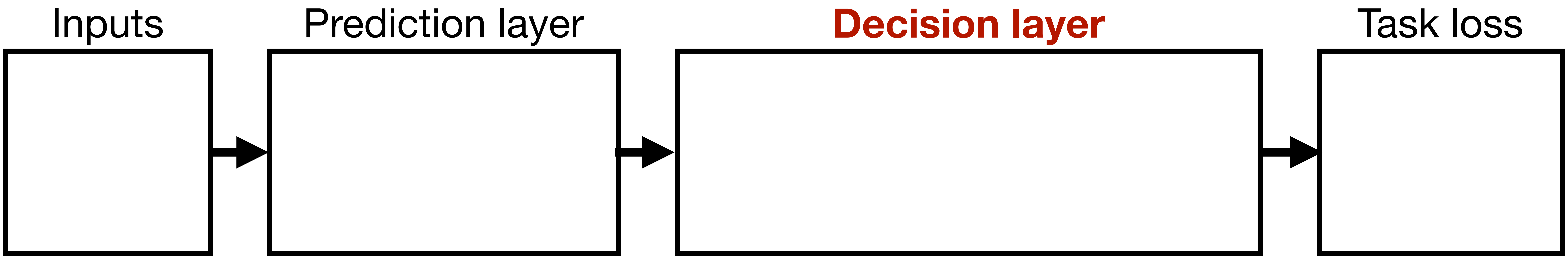


Distributional robustness in end-to-end learning systems



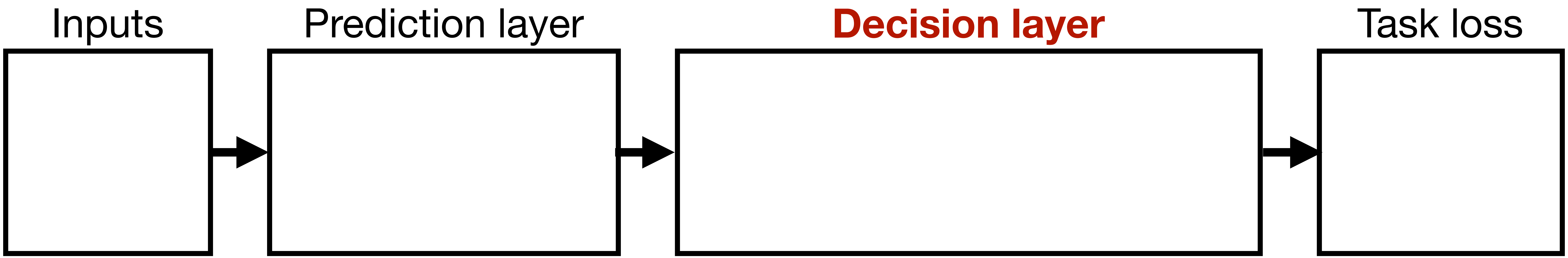
Distributional robustness in end-to-end learning systems

- The decision layer is a portfolio optimization problem.



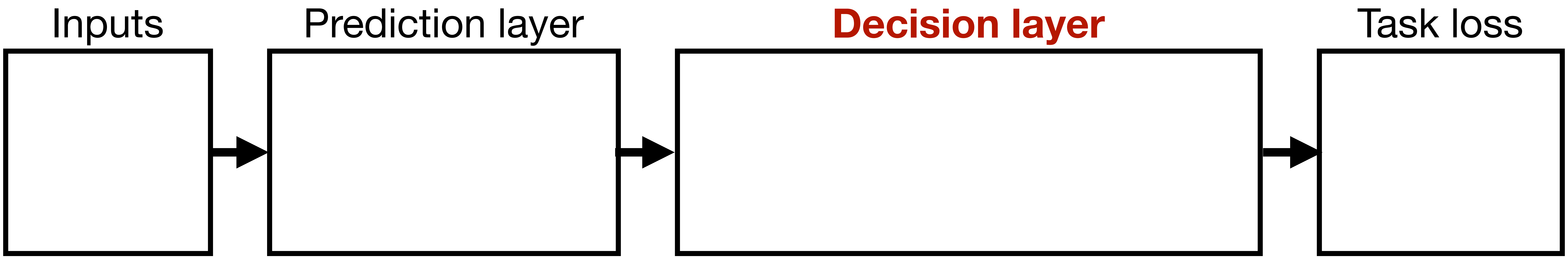
Distributional robustness in end-to-end learning systems

- The decision layer is a portfolio optimization problem.
  - Suffers from sensitivity to prediction errors.

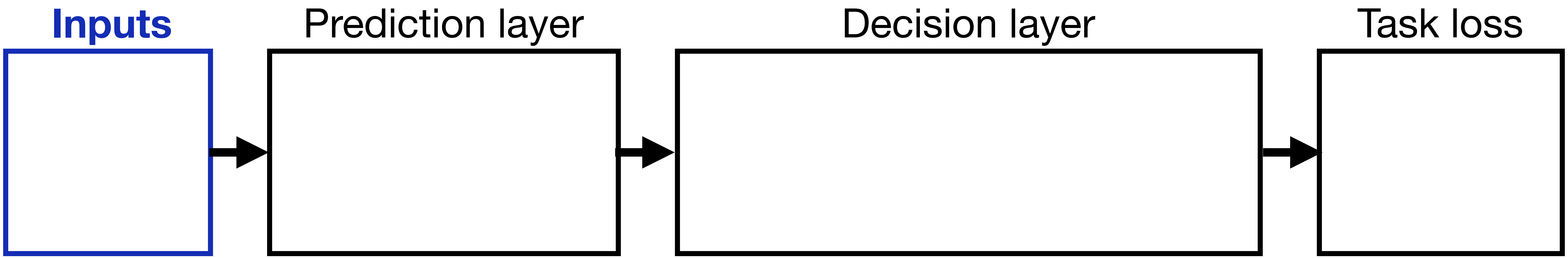


Distributional robustness in end-to-end learning systems

- The decision layer is a portfolio optimization problem.
  - Suffers from sensitivity to prediction errors.
  - We will use robustness to mitigate model error

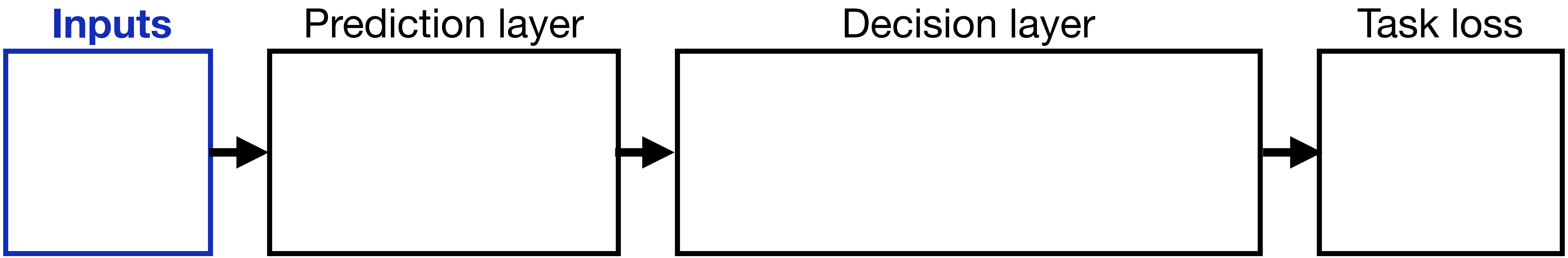


Data: features and realizations



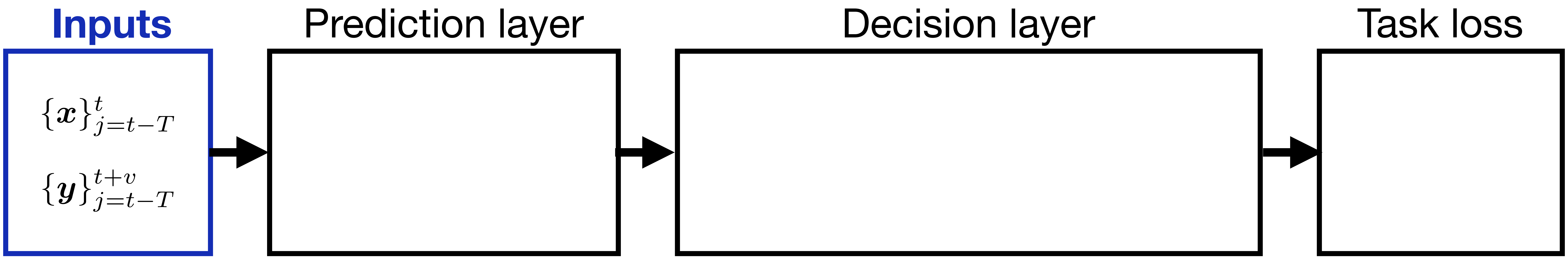
Data: features and realizations

- $\{\boldsymbol{x}\}_{j=t-T}^{t}$: Time series of financial factors
- $\{\boldsymbol{y}\}_{j=t-T}^{t+v}$: Time series of asset returns

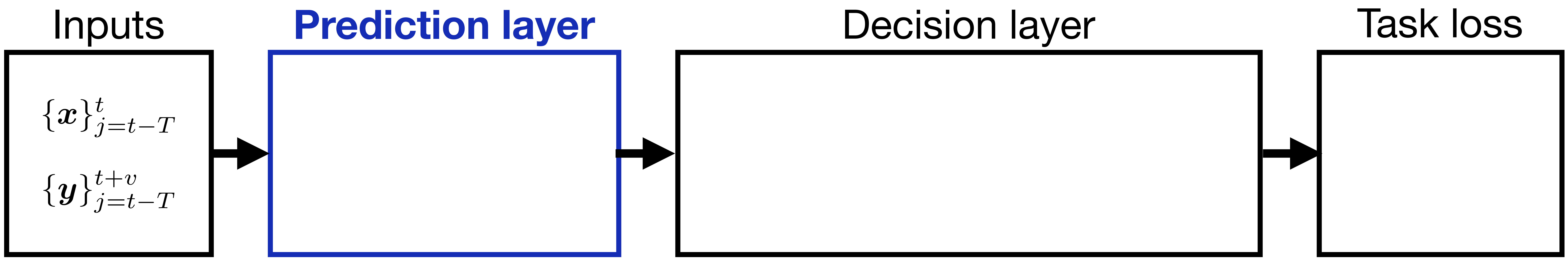


Data: features and realizations

- $\{\boldsymbol{x}\}_{j=t-T}^{t}$: Time series of financial factors
- $\{\boldsymbol{y}\}_{j=t-T}^{t+v}$: Time series of asset returns

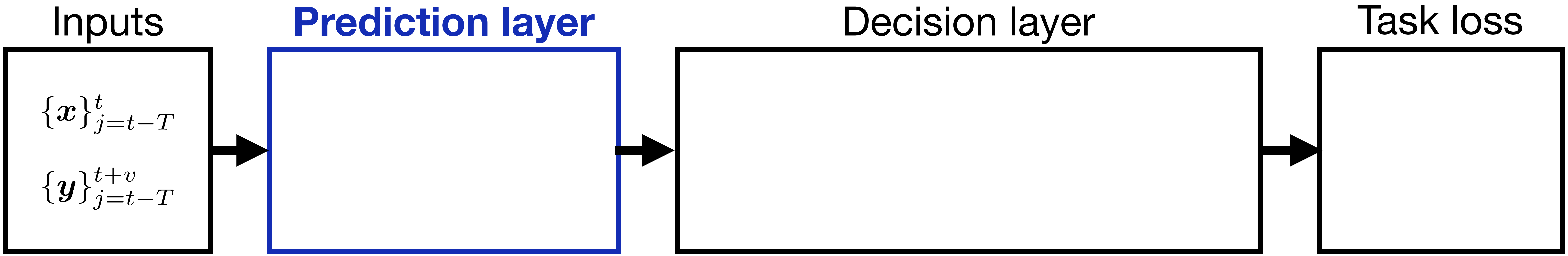


Prediction layer



Prediction layer

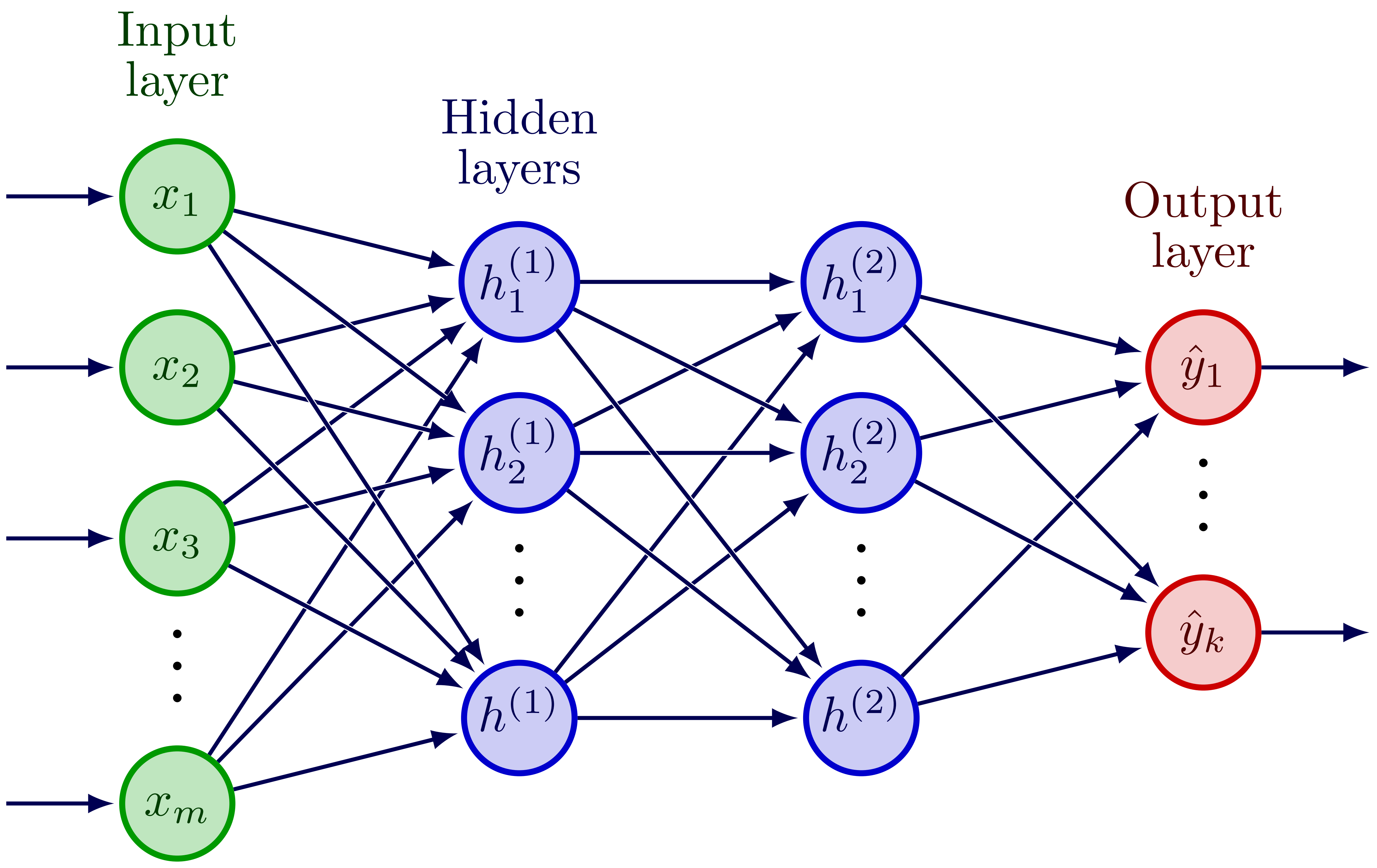
- $g_{\boldsymbol{\theta}} : \mathbb{R}^m \to \mathbb{R}^n$: Prediction model that maps features $\boldsymbol{x}_j$ to predictions $\hat{\boldsymbol{y}}_j$



Prediction layer

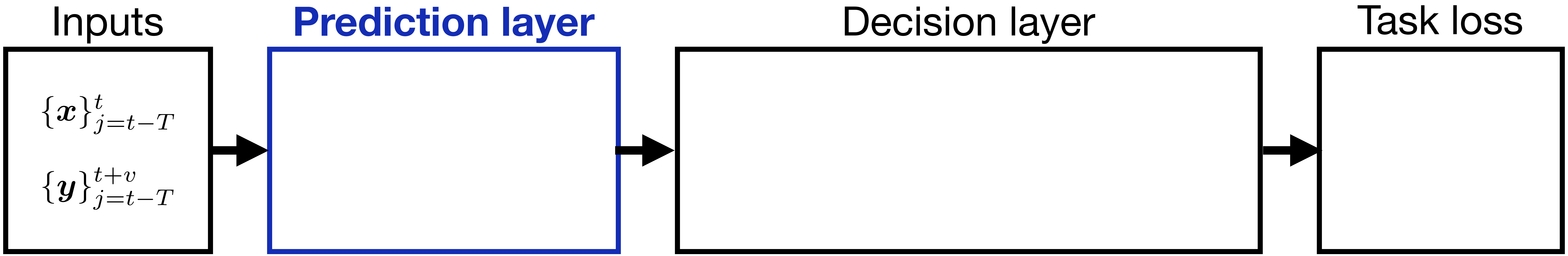
- $g_{\boldsymbol{\theta}} : \mathbb{R}^m \to \mathbb{R}^n$: Prediction model that maps features $\boldsymbol{x}_j$ to predictions $\hat{\boldsymbol{y}}_j$

The prediction model can have any form that allows for gradient-based learning



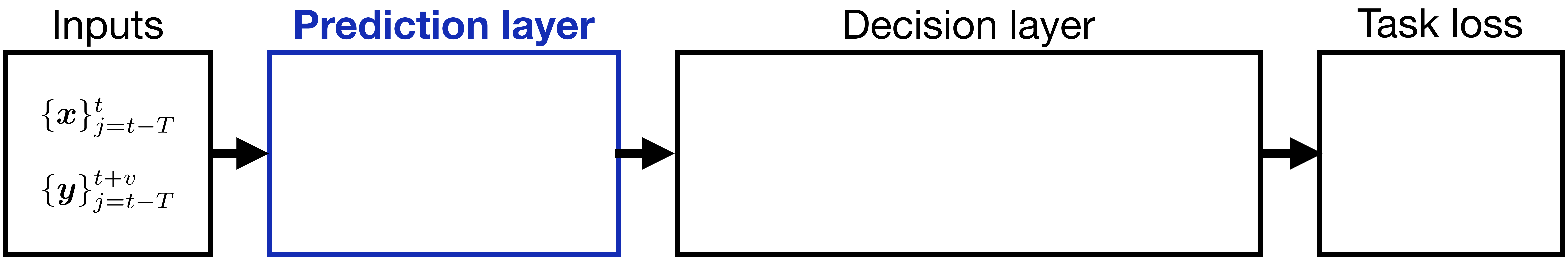
Prediction layer

- $g_{\boldsymbol{\theta}} : \mathbb{R}^m \rightarrow \mathbb{R}^n$: Prediction model that maps features $\boldsymbol{x}_j$ to predictions $\hat{\boldsymbol{y}}_j$
- $\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$: Predicted asset returns



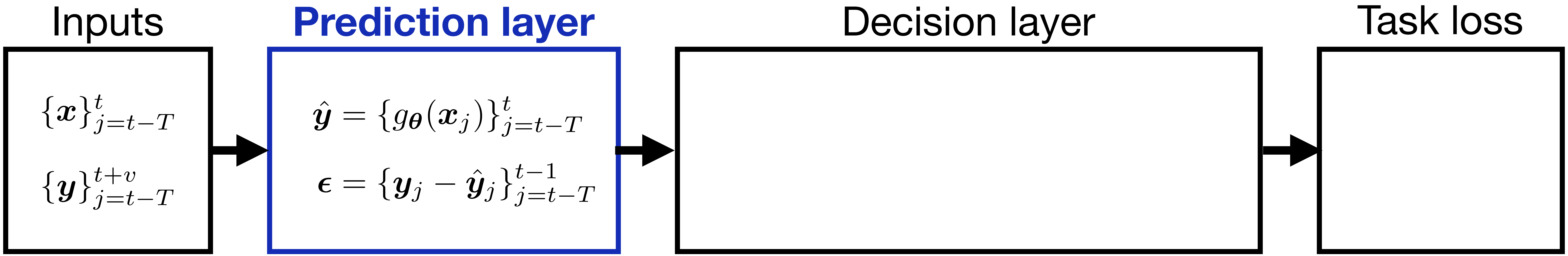
Prediction layer

- $g_{\boldsymbol{\theta}} : \mathbb{R}^m \to \mathbb{R}^n$: Prediction model that maps features $\boldsymbol{x}_j$ to predictions $\hat{\boldsymbol{y}}_j$
- $\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$: Predicted asset returns
- $\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$: Prediction errors



Prediction layer

- $g_{\boldsymbol{\theta}} : \mathbb{R}^m \to \mathbb{R}^n$: Prediction model that maps features $\boldsymbol{x}_j$ to predictions $\hat{\boldsymbol{y}}_j$
- $\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$: Predicted asset returns
- $\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$: Prediction errors



Decision layer

Inputs

$\{\boldsymbol{x}\}_{j=t-T}^{t}$

$\{\boldsymbol{y}\}_{j=t-T}^{t+v}$

Prediction layer

$\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$

$\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$

Decision layer

Task loss

Decision layer

Optimization problem

$$\underbrace{z_t^*}_{\text{Optimal portfolio}} = \underset{z \in \mathcal{Z}}{\operatorname{argmin}} \; f_\epsilon(z) - \gamma \cdot \hat{y}_t^\top z$$

Inputs	Prediction layer	Decision layer	Task loss
$\{x\}_{j=t-T}^{t}$ $\{y\}_{j=t-T}^{t+v}$	$\hat{y} = \{g_\theta(x_j)\}_{j=t-T}^{t}$ $\epsilon = \{y_j - \hat{y}_j\}_{j=t-T}^{t-1}$		

Decision layer

Optimization problem

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z}\in\mathcal{Z}}{\operatorname{argmin}} \ \underbrace{f_{\epsilon}(\boldsymbol{z})}_{\text{Deviation risk measure}} - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$

Inputs

$\{\boldsymbol{x}\}_{j=t-T}^{t}$

$\{\boldsymbol{y}\}_{j=t-T}^{t+v}$

Prediction layer

$\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$

$\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$

Decision layer

Task loss

Decision layer

Optimization problem

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z} \in \mathcal{Z}}{\operatorname{argmin}} \; f_{\epsilon}(\boldsymbol{z}) - \gamma \cdot \underbrace{\hat{\boldsymbol{y}}_t^\top \boldsymbol{z}}_{\text{Predicted portfolio return}}$$

Inputs

$\{\boldsymbol{x}\}_{j=t-T}^{t}$

$\{\boldsymbol{y}\}_{j=t-T}^{t+v}$

Prediction layer

$\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$

$\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$

Decision layer

Task loss

Decision layer

Optimization problem

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z} \in \mathcal{Z}}{\operatorname{argmin}} \; f_{\boldsymbol{\epsilon}}(\boldsymbol{z}) - \underbrace{\gamma}_{\text{Risk aversion parameter}} \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$

Inputs

$\{\boldsymbol{x}\}_{j=t-T}^{t}$

$\{\boldsymbol{y}\}_{j=t-T}^{t+v}$

Prediction layer

$\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$

$\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$

Decision layer

Task loss

Decision layer

Optimization problem

$$z_t^* = \operatorname*{argmin}_{z \in \mathcal{Z}} \underbrace{f_\epsilon(z)}_{} - \gamma \cdot \hat{y}_t^\top z$$

Deviation risk measure

Let $R : \mathbb{R} \to \mathbb{R}_+ \cup +\infty$ is a closed convex function where $R(0) = 0$ and $R(X) = R(-X)$

Then, $f_\epsilon(z) = \min_c \frac{1}{T} \sum_{j=1}^{T} R(\epsilon_j^\top z - c)$

is a deviation risk measure

Inputs	Prediction layer	Decision layer	Task loss
$\{x\}_{j=t-T}^{t}$ $\{y\}_{j=t-T}^{t+v}$	$\hat{y} = \{g_\theta(x_j)\}_{j=t-T}^{t}$ $\epsilon = \{y_j - \hat{y}_j\}_{j=t-T}^{t-1}$		

Decision layer

Optimization problem

$$z_t^* = \underset{z \in \mathcal{Z}}{\operatorname{argmin}} \; \underbrace{f_\epsilon(z)} - \gamma \cdot \hat{y}_t^\top z$$

Deviation risk measure

Example: $R(X) = X^2$

Then, $f_\epsilon(z) = \min_c \frac{1}{T} \sum_{j=1}^{T} (\epsilon_j^\top z - c)^2 = \operatorname{var}(\epsilon^\top z)$

is the portfolio error variance

Inputs	Prediction layer	Decision layer	Task loss
$\{x\}_{j=t-T}^{t}$ $\{y\}_{j=t-T}^{t+v}$	$\hat{y} = \{g_\theta(x_j)\}_{j=t-T}^{t}$ $\epsilon = \{y_j - \hat{y}_j\}_{j=t-T}^{t-1}$		

Decision layer

Optimization problem

$$z_t^* = \underset{z \in \mathcal{Z}}{\operatorname{argmin}} \; f_\epsilon(z) - \gamma \cdot \hat{y}_t^\top z$$

- z_t^* : Optimal portfolio
- $f_\epsilon(z)$: Convex deviation risk measure
- $\hat{y}_t^\top z$: Predicted portfolio return
- γ : Risk aversion parameter

Inputs

$\{x\}_{j=t-T}^{t}$

$\{y\}_{j=t-T}^{t+v}$

Prediction layer

$\hat{y} = \{g_\theta(x_j)\}_{j=t-T}^{t}$

$\epsilon = \{y_j - \hat{y}_j\}_{j=t-T}^{t-1}$

Decision layer

Task loss

Decision layer

Optimization problem

$$z_t^* = \underset{z \in \mathcal{Z}}{\operatorname{argmin}} \ f_\epsilon(z) - \gamma \cdot \hat{y}_t^\top z$$

- z_t^* : Optimal portfolio
- $f_\epsilon(z)$: Convex deviation risk measure
- $\hat{y}_t^\top z$: Predicted portfolio return
- γ : Risk appetite *- Learnable parameter*

Inputs

$\{x\}_{j=t-T}^{t}$

$\{y\}_{j=t-T}^{t+v}$

Prediction layer

$\hat{y} = \{g_\theta(x_j)\}_{j=t-T}^{t}$

$\epsilon = \{y_j - \hat{y}_j\}_{j=t-T}^{t-1}$

Decision layer

Task loss

Decision layer

Optimization problem

$$z_t^* = \underset{z \in \mathcal{Z}}{\operatorname{argmin}} \; f_\epsilon(z) - \gamma \cdot \hat{y}_t^\top z$$

- z_t^* : Optimal portfolio
- $f_\epsilon(z)$: Convex deviation risk measure
- $\hat{y}_t^\top z$: Predicted portfolio return
- γ : Risk appetite *- Learnable parameter*

Inputs → Prediction layer → **Decision layer** → Task loss

Inputs

$$\{x\}_{j=t-T}^{t}$$
$$\{y\}_{j=t-T}^{t+v}$$

Prediction layer

$$\hat{y} = \{g_\theta(x_j)\}_{j=t-T}^{t}$$
$$\epsilon = \{y_j - \hat{y}_j\}_{j=t-T}^{t-1}$$

Decision layer

$$z_t^* = \underset{z \in \mathcal{Z}}{\operatorname{argmin}} \; f_\epsilon(z) - \gamma \cdot \hat{y}_t^\top z$$

Task loss

Distributionally robust decision layer

Inputs

$$\{\boldsymbol{x}\}_{j=t-T}^{t}$$

$$\{\boldsymbol{y}\}_{j=t-T}^{t+v}$$

Prediction layer

$$\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$$

$$\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$$

DR decision layer

Task loss

Distributionally robust decision layer

Nominal decision layer

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z} \in \mathcal{Z}}{\operatorname{argmin}} \; f_{\boldsymbol{\epsilon}}(\boldsymbol{z}) - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$

Inputs

$$\{\boldsymbol{x}\}_{j=t-T}^{t}$$

$$\{\boldsymbol{y}\}_{j=t-T}^{t+v}$$

Prediction layer

$$\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$$

$$\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$$

DR decision layer

Task loss

Distributionally robust decision layer

Nominal decision layer

$$z_t^* = \underset{z \in \mathcal{Z}}{\operatorname{argmin}} \; f_{\epsilon}(z) - \gamma \cdot \hat{y}_t^\top z$$

Derived from a set of past prediction errors

Distributionally robust decision layer

Nominal decision layer

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z} \in \mathcal{Z}}{\operatorname{argmin}} \; f_{\boldsymbol{\epsilon}}(\boldsymbol{z}, \boldsymbol{q}) - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$

Deviation risk measure

$$f_{\boldsymbol{\epsilon}}(\boldsymbol{z}, \boldsymbol{q}) = \min_c \sum_{j=1}^{T} q_j \cdot R\big(\boldsymbol{\epsilon}_j^\top \boldsymbol{z} - c\big)$$

Derived from a set of past prediction errors

- Nominal assumption: All scenarios are equally likely, $q_j = 1/T$ for $j = 1, \ldots, T$

Distributionally robust decision layer

Nominal decision layer

$$z_t^* = \underset{z \in \mathcal{Z}}{\operatorname{argmin}} \; f_\epsilon(z, q) - \gamma \cdot \hat{y}_t^\top z$$

- Nominal assumption: All scenarios are equally likely, $q_j = 1/T$ for $j = 1, \ldots, T$
- Can we protect against scenario probabilities changing in the future?

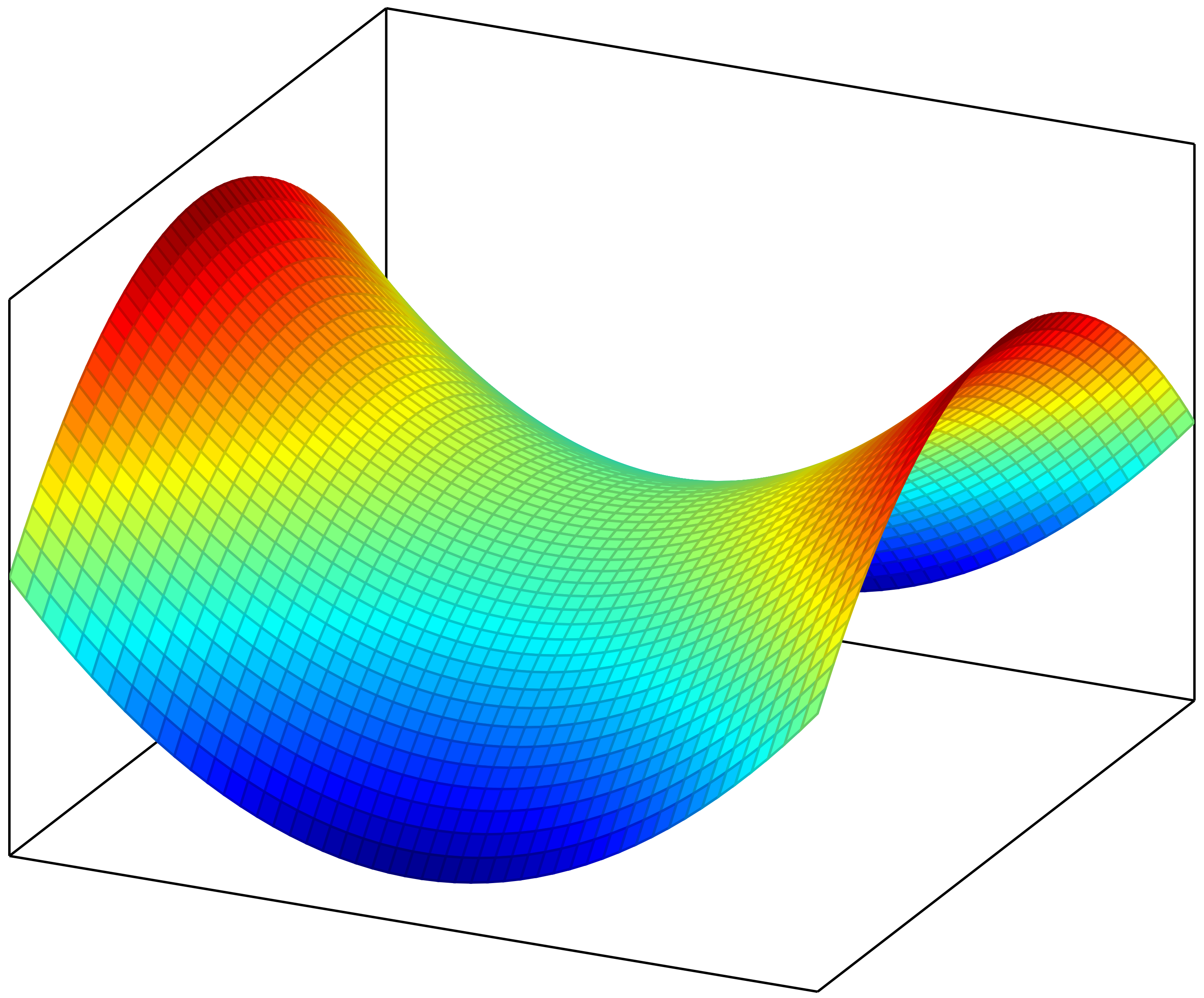
Distributionally robust decision layer

Nominal decision layer

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z} \in \mathcal{Z}}{\operatorname{argmin}} \; f_{\epsilon}(\boldsymbol{z}, \boldsymbol{q}) - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$

DR decision layer

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z} \in \mathcal{Z}}{\operatorname{argmin}} \; \max_{\boldsymbol{p} \in \mathcal{P}(\delta)} \; f_{\epsilon}(\boldsymbol{z}, \boldsymbol{p}) - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$



Distributionally robust decision layer

Nominal decision layer

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z} \in \mathcal{Z}}{\operatorname{argmin}} \; f_{\epsilon}(\boldsymbol{z}, \boldsymbol{q}) - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$

DR decision layer

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z} \in \mathcal{Z}}{\operatorname{argmin}} \; \max_{\boldsymbol{p} \in \mathcal{P}(\delta)} \; f_{\epsilon}(\boldsymbol{z}, \boldsymbol{p}) - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$

$$\underbrace{\mathcal{P}(\delta)}_{\text{Ambiguity set}} \triangleq \{\boldsymbol{p} \in \mathbb{R}^T : \boldsymbol{p} \geq \boldsymbol{0}, \; \boldsymbol{1}^\top \boldsymbol{p} = 1, \; I_\phi(\boldsymbol{p}, \boldsymbol{q}) \leq \delta\}$$

- $\boldsymbol{p}$: probability mass function

Distributionally robust decision layer

Nominal decision layer

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z} \in \mathcal{Z}}{\operatorname{argmin}} \; f_{\boldsymbol{\epsilon}}(\boldsymbol{z}, \boldsymbol{q}) - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$

DR decision layer

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z} \in \mathcal{Z}}{\operatorname{argmin}} \; \max_{\boldsymbol{p} \in \mathcal{P}(\delta)} \; f_{\boldsymbol{\epsilon}}(\boldsymbol{z}, \boldsymbol{p}) - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$

$$\mathcal{P}(\delta) \triangleq \{\boldsymbol{p} \in \mathbb{R}^T : \underbrace{\boldsymbol{p} \geq \boldsymbol{0}, \; \boldsymbol{1}^\top \boldsymbol{p} = 1}_{\text{Probability simplex}}, \; I_\phi(\boldsymbol{p}, \boldsymbol{q}) \leq \delta\}$$

- $\boldsymbol{p}$: probability mass function

Distributionally robust decision layer

Nominal decision layer

$$z_t^* = \underset{z \in \mathcal{Z}}{\operatorname{argmin}} \; f_\epsilon(z, q) - \gamma \cdot \hat{y}_t^\top z$$

DR decision layer

$$z_t^* = \underset{z \in \mathcal{Z}}{\operatorname{argmin}} \; \max_{p \in \mathcal{P}(\delta)} \; f_\epsilon(z, p) - \gamma \cdot \hat{y}_t^\top z$$

$$\mathcal{P}(\delta) \triangleq \{ p \in \mathbb{R}^T : p \geq \mathbf{0}, \; \mathbf{1}^\top p = 1, \; \underbrace{I_\phi(p, q) \leq \delta}_{\delta\text{-constrained distance measure}} \}$$

- p : probability mass function
- Distance measure: ϕ-divergence (e.g., Kullback-Leibler, Hellinger)

Distributionally robust decision layer

Nominal decision layer

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z}\in\mathcal{Z}}{\operatorname{argmin}} \; f_{\boldsymbol{\epsilon}}(\boldsymbol{z}, \boldsymbol{q}) - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$

DR decision layer

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z}\in\mathcal{Z}}{\operatorname{argmin}} \; \max_{\boldsymbol{p}\in\mathcal{P}(\delta)} \; f_{\boldsymbol{\epsilon}}(\boldsymbol{z}, \boldsymbol{p}) - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$

Inputs

$$\{\boldsymbol{x}\}_{j=t-T}^{t}$$

$$\{\boldsymbol{y}\}_{j=t-T}^{t+v}$$

Prediction layer

$$\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$$

$$\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$$

DR decision layer

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z}\in\mathcal{Z}}{\operatorname{argmin}} \; \max_{\boldsymbol{p}\in\mathcal{P}(\delta)} \; f_{\boldsymbol{\epsilon}}(\boldsymbol{z}, \boldsymbol{p}) - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$

Task loss

Distributionally robust decision layer

Minimax problem

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z}\in\mathcal{Z}}{\operatorname{argmin}} \max_{\boldsymbol{p}\in\mathcal{P}(\delta)} \; f_{\boldsymbol{\epsilon}}(\boldsymbol{z},\boldsymbol{p}) - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$

Inputs

$$\{\boldsymbol{x}\}_{j=t-T}^{t}$$

$$\{\boldsymbol{y}\}_{j=t-T}^{t+v}$$

Prediction layer

$$\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$$

$$\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$$

DR decision layer

Task loss

Distributionally robust decision layer

Minimax problem

$$z_t^* = \underset{z \in \mathcal{Z}}{\operatorname{argmin}} \max_{p \in \mathcal{P}(\delta)} f_{\epsilon}(z, p) - \gamma \cdot \hat{y}_t^{\top} z$$

Duality $\Rightarrow$

Convex minimization problem

$$z_t^* = \underset{z \in \mathcal{Z},\ \lambda \geq 0,\ \xi,\ c}{\operatorname{argmin}} f_{\epsilon}^{\delta}(z, c, \lambda, \xi) - \gamma \cdot \hat{y}_t^{\top} z$$

Inputs → Prediction layer → **DR decision layer** → Task loss

Inputs:

$$\{x\}_{j=t-T}^{t}$$

$$\{y\}_{j=t-T}^{t+v}$$

Prediction layer:

$$\hat{y} = \{g_{\theta}(x_j)\}_{j=t-T}^{t}$$

$$\epsilon = \{y_j - \hat{y}_j\}_{j=t-T}^{t-1}$$

Distributionally robust decision layer

Duality

Minimax problem

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z}\in\mathcal{Z}}{\operatorname{argmin}} \max_{\boldsymbol{p}\in\mathcal{P}(\delta)} f_{\boldsymbol{\epsilon}}(\boldsymbol{z}, \boldsymbol{p}) - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$

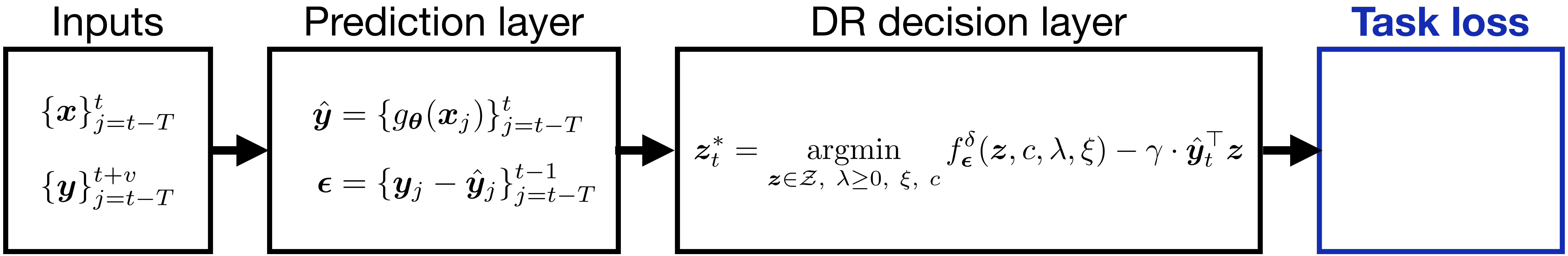
Convex minimization problem

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z}\in\mathcal{Z},\ \lambda\geq 0,\ \xi,\ c}{\operatorname{argmin}} f_{\boldsymbol{\epsilon}}^{\delta}(\boldsymbol{z}, c, \lambda, \xi) - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$

δ *is now learnable*

Inputs	Prediction layer	DR decision layer	Task loss
$\{\boldsymbol{x}\}_{j=t-T}^{t}$ $\{\boldsymbol{y}\}_{j=t-T}^{t+v}$	$\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$ $\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$	$\boldsymbol{z}_t^* = \underset{\boldsymbol{z}\in\mathcal{Z},\ \lambda\geq 0,\ \xi,\ c}{\operatorname{argmin}} f_{\boldsymbol{\epsilon}}^{\delta}(\boldsymbol{z}, c, \lambda, \xi) - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$	

Task loss


Inputs
$\{\boldsymbol{x}\}_{j=t-T}^{t}$
$\{\boldsymbol{y}\}_{j=t-T}^{t+v}$
Prediction layer
$\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$
$\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$
DR decision layer
$\boldsymbol{z}_t^* = \underset{\boldsymbol{z}\in\mathcal{Z},\ \lambda\geq 0,\ \xi,\ c}{\operatorname{argmin}} \; f_{\boldsymbol{\epsilon}}^{\delta}(\boldsymbol{z}, c, \lambda, \xi) - \gamma \cdot \hat{\boldsymbol{y}}_t^{\top} \boldsymbol{z}$
Task loss

Task loss

- Standard supervised learning: loss function = prediction error.

Inputs

$$\{\boldsymbol{x}\}_{j=t-T}^{t}$$

$$\{\boldsymbol{y}\}_{j=t-T}^{t+v}$$

→

Prediction layer

$$\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$$

$$\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$$

→

DR decision layer

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z}\in\mathcal{Z},\ \lambda\geq 0,\ \xi,\ c}{\operatorname{argmin}} \quad f_{\boldsymbol{\epsilon}}^{\delta}(\boldsymbol{z}, c, \lambda, \xi) - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$

→

Task loss

Task loss

- Standard supervised learning: loss function = prediction error.
- End-to-end system: Task loss = out-of-sample performance of the decision.

Inputs

$$\{\boldsymbol{x}\}_{j=t-T}^{t}$$

$$\{\boldsymbol{y}\}_{j=t-T}^{t+v}$$

→

Prediction layer

$$\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$$

$$\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$$

→

DR decision layer

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z}\in\mathcal{Z},\ \lambda\geq 0,\ \xi,\ c}{\operatorname{argmin}} \quad f_{\boldsymbol{\epsilon}}^{\delta}(\boldsymbol{z}, c, \lambda, \xi) - \gamma \cdot \hat{\boldsymbol{y}}_t^{\top} \boldsymbol{z}$$

→

Task loss

Task loss

- Standard supervised learning: loss function = prediction error.
- End-to-end system: Task loss = out-of-sample performance of the decision.
- Task loss function ≠ objective function of the decision layer.

Inputs

$$\{\boldsymbol{x}\}_{j=t-T}^{t}$$

$$\{\boldsymbol{y}\}_{j=t-T}^{t+v}$$

Prediction layer

$$\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$$

$$\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$$

DR decision layer

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z}\in\mathcal{Z},\ \lambda\geq 0,\ \xi,\ c}{\operatorname{argmin}} \quad f_{\boldsymbol{\epsilon}}^{\delta}(\boldsymbol{z}, c, \lambda, \xi) - \gamma \cdot \hat{\boldsymbol{y}}_t^{\top} \boldsymbol{z}$$

Task loss

Task loss

- Define the task loss as the financial performance over the next v time steps.

Inputs

$$\{\boldsymbol{x}\}_{j=t-T}^{t}$$

$$\{\boldsymbol{y}\}_{j=t-T}^{t+v}$$

Prediction layer

$$\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$$

$$\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$$

DR decision layer

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z}\in\mathcal{Z},\ \lambda\geq 0,\ \xi,\ c}{\operatorname{argmin}} \; f_{\boldsymbol{\epsilon}}^{\delta}(\boldsymbol{z}, c, \lambda, \xi) - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$

Task loss

Task loss

- Define the task loss as the financial performance over the next v time steps.
- For example, the task loss may be defined as the Sharpe ratio:

$$l\left(\boldsymbol{z}_t^*, \{\boldsymbol{y}_j\}_{j=t}^{t+v}\right) = -\frac{\mathrm{mean}\left(\{\boldsymbol{y}_j^\top \boldsymbol{z}_t^*\}_{j=t}^{t+v}\right)}{\mathrm{std}\left(\{\boldsymbol{y}_j^\top \boldsymbol{z}_t^*\}_{j=t}^{t+v}\right)}$$

Inputs

$$\{\boldsymbol{x}\}_{j=t-T}^{t}$$

$$\{\boldsymbol{y}\}_{j=t-T}^{t+v}$$

→ **Prediction layer**

$$\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$$

$$\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$$

→ **DR decision layer**

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z}\in\mathcal{Z},\ \lambda\geq 0,\ \xi,\ c}{\mathrm{argmin}}\ f_{\boldsymbol{\epsilon}}^{\delta}(\boldsymbol{z}, c, \lambda, \xi) - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$

→ **Task loss**

Task loss

- Define the task loss as the financial performance over the next ν time steps.
- For example, the task loss may be defined as the Sharpe ratio:

$$l\left(\boldsymbol{z}_t^*, \{\boldsymbol{y}_j\}_{j=t}^{t+v}\right) = -\frac{\operatorname{mean}\left(\{\boldsymbol{y}_j^\top \boldsymbol{z}_t^*\}_{j=t}^{t+v}\right)}{\operatorname{std}\left(\{\boldsymbol{y}_j^\top \boldsymbol{z}_t^*\}_{j=t}^{t+v}\right)}$$

Inputs

$$\{\boldsymbol{x}\}_{j=t-T}^{t}$$
$$\{\boldsymbol{y}\}_{j=t-T}^{t+v}$$

→ **Prediction layer**

$$\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$$
$$\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$$

→ **DR decision layer**

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z}\in\mathcal{Z},\ \lambda\geq 0,\ \xi,\ c}{\operatorname{argmin}} \quad f_{\boldsymbol{\epsilon}}^{\delta}(\boldsymbol{z}, c, \lambda, \xi) - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$

→ **Task loss**

$$l\left(\boldsymbol{z}_t^*, \{\boldsymbol{y}_j\}_{j=t}^{t+v}\right)$$

Training

Training

- The model is trained through gradient descent.

Training

Forward Pass:

Inputs

$$\{\boldsymbol{x}\}_{j=t-T}^{t}$$

$$\{\boldsymbol{y}\}_{j=t-T}^{t+v}$$

$\rightarrow$

Prediction layer

$$\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$$

$$\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$$

$\rightarrow$

DR decision layer

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z}\in\mathcal{Z},\ \lambda\geq 0,\ \xi,\ c}{\operatorname{argmin}} \quad f_{\boldsymbol{\epsilon}}^{\delta}(\boldsymbol{z}, c, \lambda, \xi) - \gamma \cdot \hat{\boldsymbol{y}}_t^{\top} \boldsymbol{z}$$

$\rightarrow$

Task loss

$$l\left(\boldsymbol{z}_t^*, \{\boldsymbol{y}_j\}_{j=t}^{t+v}\right)$$

Training

Forward Pass:

Inputs

$\{\boldsymbol{x}\}_{j=t-T}^{t}$

$\{\boldsymbol{y}\}_{j=t-T}^{t+v}$

Prediction layer

$\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$

$\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$

DR decision layer

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z}\in\mathcal{Z},\ \lambda\geq 0,\ \xi,\ c}{\operatorname{argmin}} \ f_{\boldsymbol{\epsilon}}^{\delta}(\boldsymbol{z}, c, \lambda, \xi) - \gamma \cdot \hat{\boldsymbol{y}}_t^\top \boldsymbol{z}$$

Task loss

$l\left(\boldsymbol{z}_t^*, \{\boldsymbol{y}_j\}_{j=t}^{t+v}\right)$

Backward Pass:

Update

θ, γ, δ

$\frac{\partial \hat{y}_t}{\partial \theta}, \frac{\partial \epsilon}{\partial \theta}$

$\frac{\partial z_t^*}{\partial \hat{y}_t}, \frac{\partial z_t^*}{\partial \delta}, \frac{\partial z_t^*}{\partial \epsilon}, \frac{\partial z_t^*}{\partial \gamma}$

$\frac{\partial l}{\partial z_t^*}$

Training

- Backpropagation through an optimization problem is possible by differentiating the system of equations arising from the KKT optimality conditions.

$$\text{Update } \theta, \gamma, \delta \leftarrow \frac{\partial \hat{y}_t}{\partial \theta}, \frac{\partial \epsilon}{\partial \theta} \leftarrow \frac{\partial z_t^*}{\partial \hat{y}_t}, \frac{\partial z_t^*}{\partial \delta}, \frac{\partial z_t^*}{\partial \epsilon}, \frac{\partial z_t^*}{\partial \gamma} \leftarrow \frac{\partial l}{\partial z_t^*}$$

Training

- Backpropagation through an optimization problem is possible by differentiating the system of equations arising from the KKT optimality conditions.
- Both γ and δ are learned parameters to enhance out-of-sample performance.

$$\boxed{\text{Update } \theta, \gamma, \delta} \leftarrow \boxed{\frac{\partial \hat{y}_t}{\partial \theta}, \frac{\partial \epsilon}{\partial \theta}} \leftarrow \boxed{\frac{\partial z_t^*}{\partial \hat{y}_t}, \frac{\partial z_t^*}{\partial \delta}, \frac{\partial z_t^*}{\partial \epsilon}, \frac{\partial z_t^*}{\partial \gamma}} \leftarrow \boxed{\frac{\partial l}{\partial z_t^*}}$$

Numerical experiment

Numerical experiment

- **Data**: Weekly, 07–Jan–2000 to 01-Oct-2021
  - *Assets*: 20 stocks from the S&P500
  - *Features*: 8 Fama-French factors
- **System:**
  - *Prediction layer*: Linear
  - *Task loss*: Sharpe ratio + prediction MSE

Numerical experiment

- **Data**: Weekly, 07–Jan–2000 to 01-Oct-2021
  - *Assets*: 20 stocks from the S&P500
  - *Features*: 8 Fama-French factors
- **System:**
  - *Prediction layer*: Linear
  - *Task loss*: Sharpe ratio + prediction MSE
- **Training**: 07–Jan–2000 to 18–Jan–2013
  - Prediction layer initialized to OLS weights
  - For each point prediction, prediction errors computed from $T = 104$ observations
  - The Sharpe ratio is computed over the subsequent $v = 13$ weeks
  - Time series split cross-validation is used to calibrate the learning rate and number of epochs
- **Testing**: 25–Jan–2013 to 01–Oct–2021
  - Systems are retrained every 2 years

Numerical experiment: Competing models

- **Equal weight:** $z_t = 1/n$

Numerical experiment: Competing models

- **Equal weight**
- **Predict-then-optimize:** Prediction layer ➜ Fixed OLS weights

  Decision layer ➜ $z_t^* = \underset{z \in \mathcal{Z}}{\operatorname{argmin}} \; f_{\epsilon}(z, q) - \gamma \cdot \hat{y}_t^\top z$

Constants

Numerical experiment: Competing models

- Equal weight
- Predict-then-optimize
- Nominal:

Inputs

$$\{\boldsymbol{x}\}_{j=t-T}^{t}$$

$$\{\boldsymbol{y}\}_{j=t-T}^{t+v}$$

Prediction layer

$$\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$$

$$\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$$

Decision layer

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z}\in\mathcal{Z}}{\operatorname{argmin}} \;\; f_{\boldsymbol{\epsilon}}(\boldsymbol{z}, \boldsymbol{q}) - \gamma \cdot \hat{\boldsymbol{y}}_t^{\top} \boldsymbol{z}$$

Task loss

$$l\left(\boldsymbol{z}_t^*, \{\boldsymbol{y}_j\}_{j=t}^{t+v}\right)$$

Numerical experiment: Competing models

- Equal weight
- Predict-then-optimize
- Nominal
- DR (Hellinger-based):

Inputs

$$\{\boldsymbol{x}\}_{j=t-T}^{t}$$

$$\{\boldsymbol{y}\}_{j=t-T}^{t+v}$$

Prediction layer

$$\hat{\boldsymbol{y}} = \{g_{\boldsymbol{\theta}}(\boldsymbol{x}_j)\}_{j=t-T}^{t}$$

$$\boldsymbol{\epsilon} = \{\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j\}_{j=t-T}^{t-1}$$

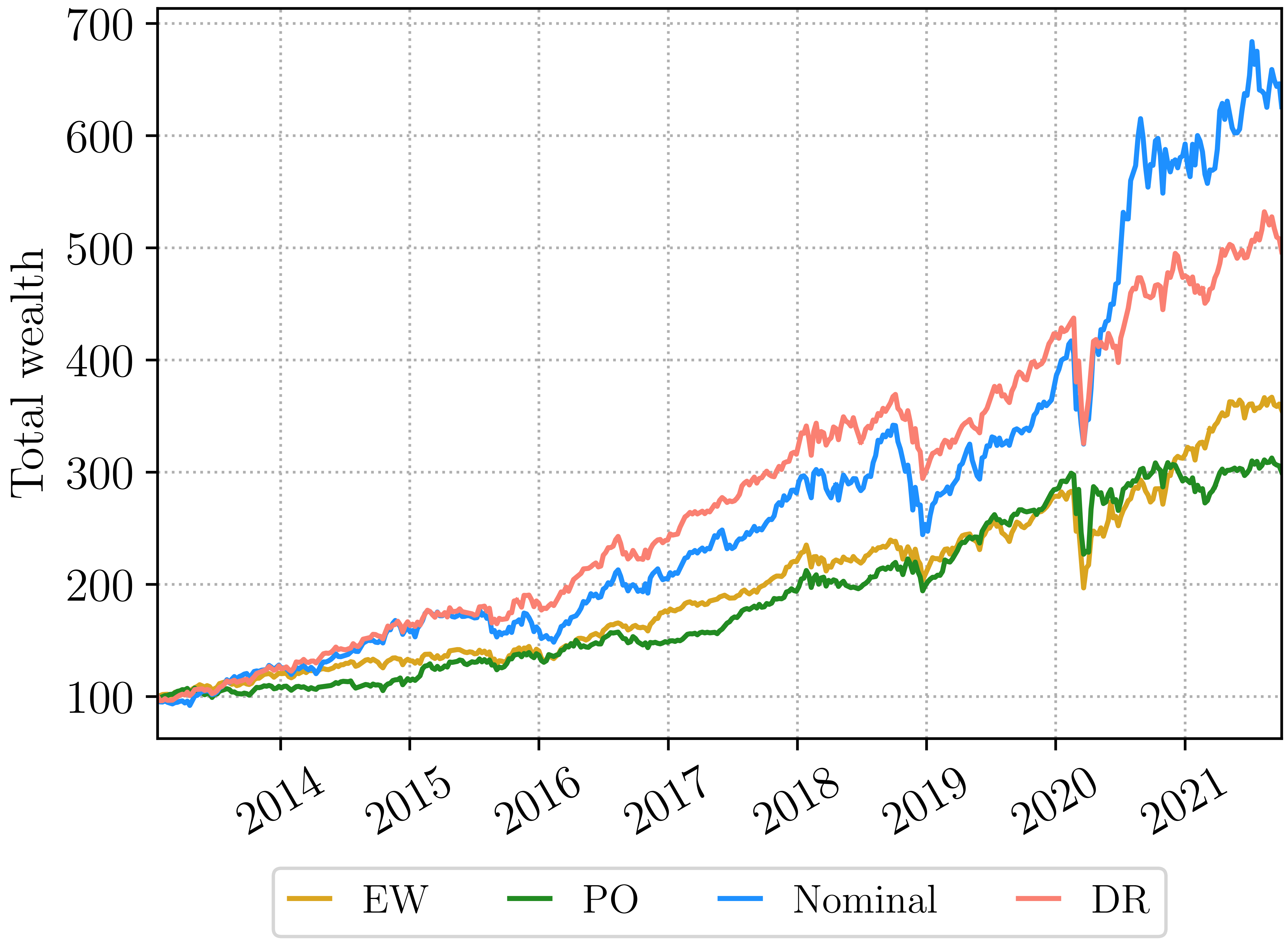
DR Decision layer

$$\boldsymbol{z}_t^* = \underset{\boldsymbol{z}\in\mathcal{Z},\ \lambda\geq 0,\ \xi,\ c}{\operatorname{argmin}} \quad f_{\boldsymbol{\epsilon}}^{\delta}(\boldsymbol{z}, c, \lambda, \xi) - \gamma \cdot \hat{\boldsymbol{y}}_t^{\top} \boldsymbol{z}$$

Task loss

$$l\left(\boldsymbol{z}_t^*, \{\boldsymbol{y}_j\}_{j=t}^{t+v}\right)$$

Out-of-sample financial performance



Sharpe ratio	
Equal weight:	1.05
Pred-then-Opt:	0.88
Nominal:	1.24
DR:	1.30

Conclusion

Conclusion

- End-to-end system with a **robust** decision layer that explicitly incorporates prediction model risk.

Conclusion

- End-to-end system with a **robust** decision layer that explicitly incorporates prediction model risk.
  - By design, we pass both the prediction and a set of prediction errors to the decision layer.
  - Furthermore, we introduce robustness by taking the worst-case risk over a set of probability measures.

Conclusion

- End-to-end system with a **robust** decision layer that explicitly incorporates prediction model risk.
  - By design, pass both the prediction and a set of prediction errors to the decision layer.
  - Furthermore, introduce robustness by taking the worst-case risk over a set of probability measures.
- Use convex duality to show that the DR decision layer is computationally tractable.

Conclusion

- End-to-end system with a **robust** decision layer that explicitly incorporates prediction model risk.
  - By design, pass both the prediction and a set of prediction errors to the decision layer.
  - Furthermore, introduce robustness by taking the worst-case risk over a set of probability measures.
- Use convex duality to show that the DR decision layer is computationally tractable.
- Risk and robustness parameters are ***learned*** directly from data.

Conclusion

- End-to-end system with a **robust** decision layer that explicitly incorporates prediction model risk.
  - By design, we pass both the prediction and a set of prediction errors to the decision layer.
  - Furthermore, we introduce robustness by taking the worst-case risk over a set of probability measures.
- Use convex duality to show that the DR decision layer is computationally tractable.
- Risk and robustness parameters are ***learned*** directly from data.
  - Being able to learn these parameters relieves practitioners from the challenge of determining them a priori.

References

- Amos, B. and Kolter, J. Z. Optnet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning*, pp. 136–145. PMLR, 2017.
- Ben-Tal, A., Den Hertog, D., De Waegenaere, A., Melen- berg, B., and Rennen, G. Robust solutions of optimization problems affected by uncertain probabilities. *Management Science*, 59(2):341–357, 2013.
- Calafiore, G. C. Ambiguous risk measures and optimal robust portfolios. *SIAM Journal on Optimization*, 18(3): 853–877, 2007.
- 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.

A draft version our paper is now available

- Costa, G., & Iyengar, G. N. (2022). Distributionally Robust End-to-End Portfolio Construction. arXiv preprint arXiv:2206.05134.

Thank you!