Toward a theory of multi-objective learning

Geelon So, UC San Diego

October 12 - 16, 2025
Symposium on Mathematical Foundations of Trustworthy Learning

Tobias Wegel

Junhyung Park

Fanny Yang

Motivating example: self-driving car

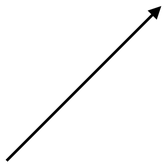
Goal: train a model for a self-driving car.

Single-objective risk minimization

$$\min_{h \in \mathcal{H}} \; R(h)$$

Single-objective risk minimization

$$\min_{h \in \mathcal{H}} R(h)$$



$h : X \to Y$ is a **model** from a model class $\mathcal{H}$

"Under situation x, the car should do y."

Single-objective risk minimization

$R(h)$ measures the **population risk** of the model h

"This is the expected fuel efficiency of h on highways."

$$\min_{h \in \mathcal{H}} R(h)$$

$h : X \to Y$ is a **model** from a model class $\mathcal{H}$

"Under situation x, the car should do y."

Single-objective risk minimization

$R(h)$ measures the **population risk** of the model h

"This is the expected fuel efficiency of h on highways."

$$\min_{h \in \mathcal{H}} R(h)$$

$h : X \to Y$ is a **model** from a model class $\mathcal{H}$

"Under situation x, the car should do y."

The learning problem

Directly optimizing R is not possible since we only have sample access to it.

Motivating example: self-driving car

Goal: train a model for a self-driving car.

What we might care about:

- Safety
- Efficiency
- Comfort
- [many more]

Multi-objective risk minimization

$$\min_{h \in \mathscr{H}} \underbrace{\mathbf{R}(h) \equiv \Big(R_1(h), \ldots, R_K(h)\Big)}$$

Now, we care about many types of risks $\mathbf{R}(h)$.

Multi-objective risk minimization

$$\min_{h \in \mathscr{H}} \underbrace{\mathbf{R}(h) \equiv \Big(R_1(h), \ldots, R_K(h)\Big)}$$

Now, we care about many types of risks $\mathbf{R}(h)$.

- It is not usually possible to minimize all risks simultaneously.

The statistical learning setting

Let $X \times Y$ be a data space.

The statistical learning setting

Let $X \times Y$ be a data space. For each objective $k \in [K]$:

- Let R_k measure the risk of a standard supervised learning task:

The statistical learning setting

Let $X \times Y$ be a data space. For each objective $k \in [K]$:

- Let R_k measure the risk of a standard supervised learning task:
  - P_k a data distribution

The statistical learning setting

Let $X \times Y$ be a data space. For each objective $k \in [K]$:

- Let R_k measure the risk of a standard supervised learning task:
  - P_k a data distribution
  - $\ell_k(y, \hat{y})$ measures the loss of predicting $\hat{y}$ when correct answer is y

$$R_k(h) = \mathbb{E}_{P_k}\Big[\ell_k\big(y, h(x)\big)\Big]$$

The statistical learning setting

Let $X \times Y$ be a data space. For each objective $k \in [K]$:

- Let R_k measure the risk of a standard supervised learning task:
  - P_k a data distribution
  - $\ell_k(y, \hat{y})$ measures the loss of predicting $\hat{y}$ when correct answer is y

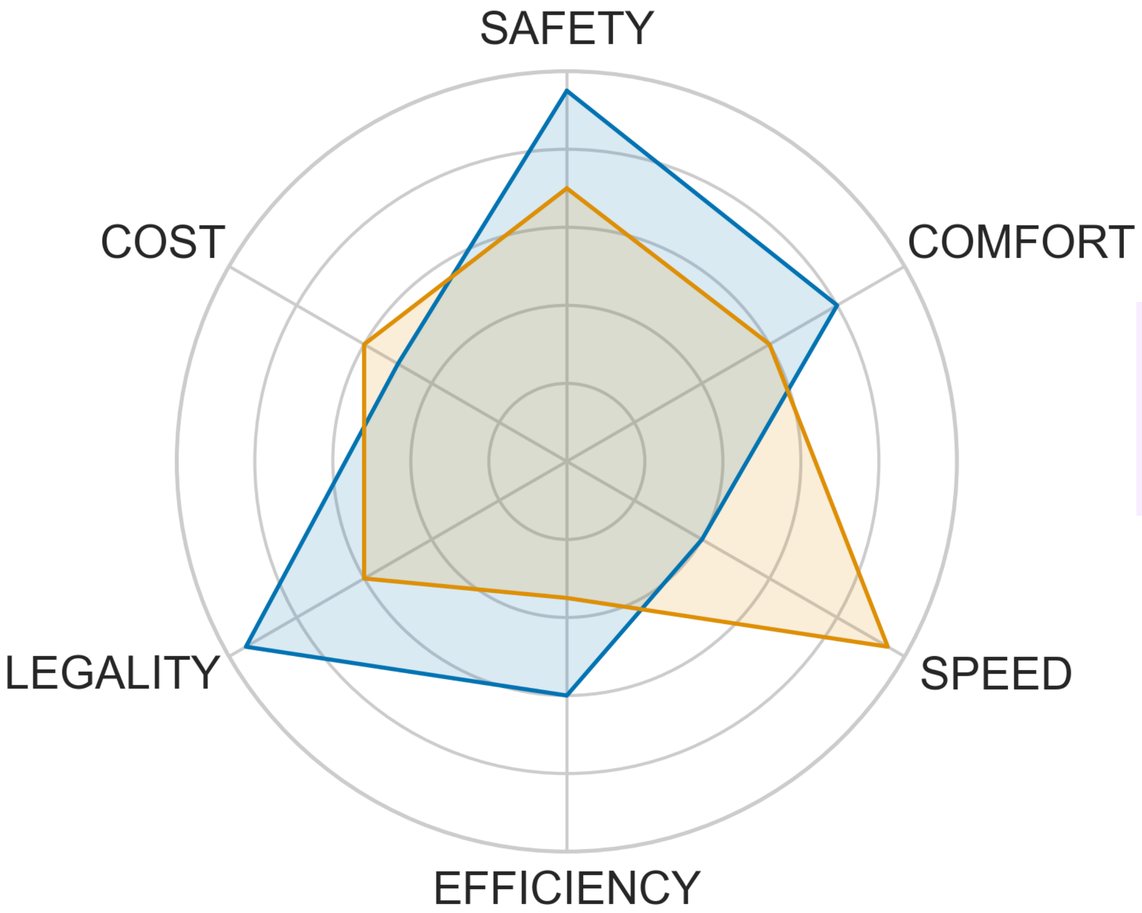
$$R_k(h) = \mathbb{E}_{P_k}\Big[\ell_k\big(y, h(x)\big)\Big]$$

  - $f_k^\star$ is the Bayes-optimal model minimizing R_k

I. Background

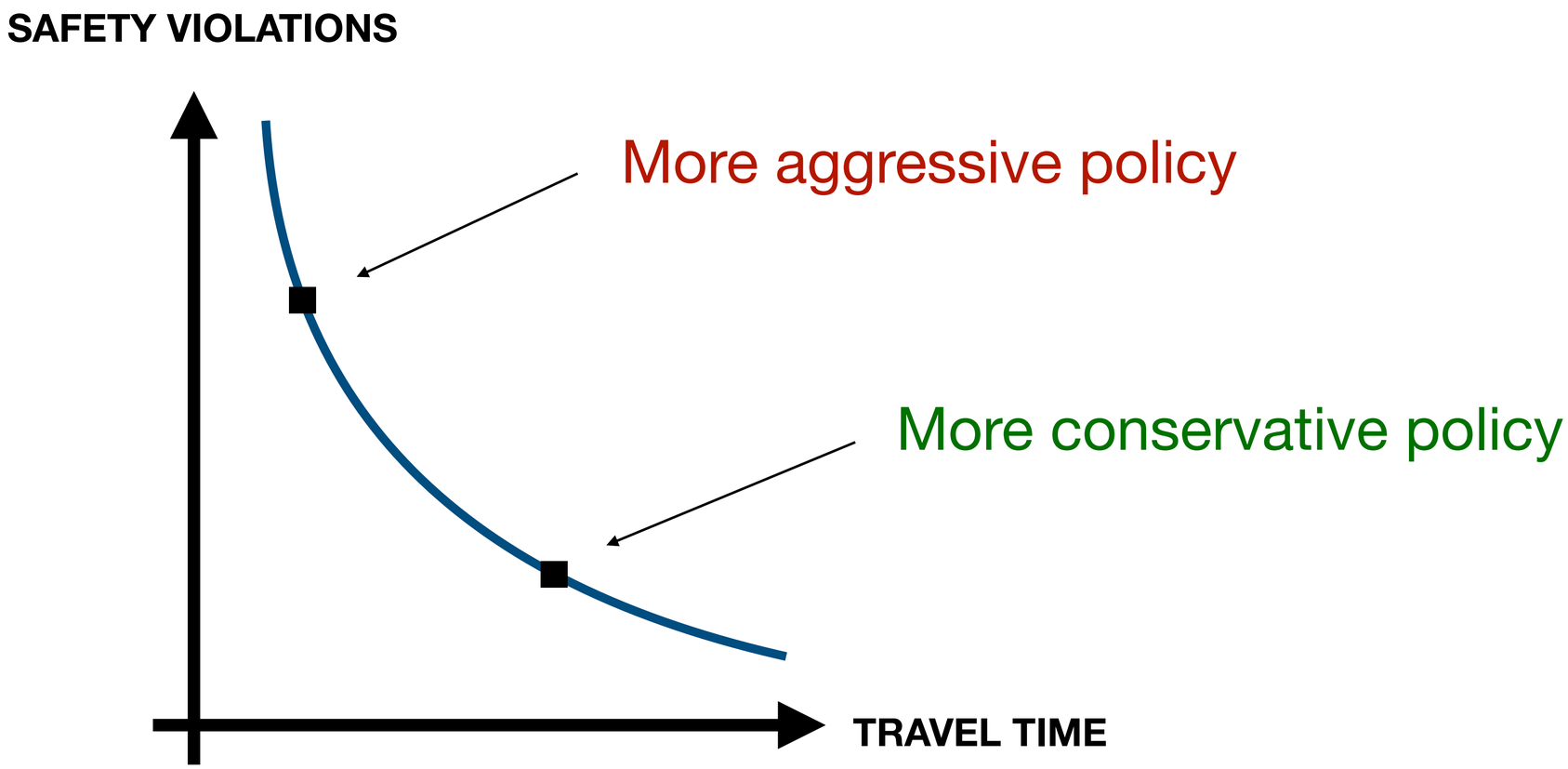
Question: what does it even mean to minimize $\mathbf{R}(h)$?

Solution concept: Pareto optimality

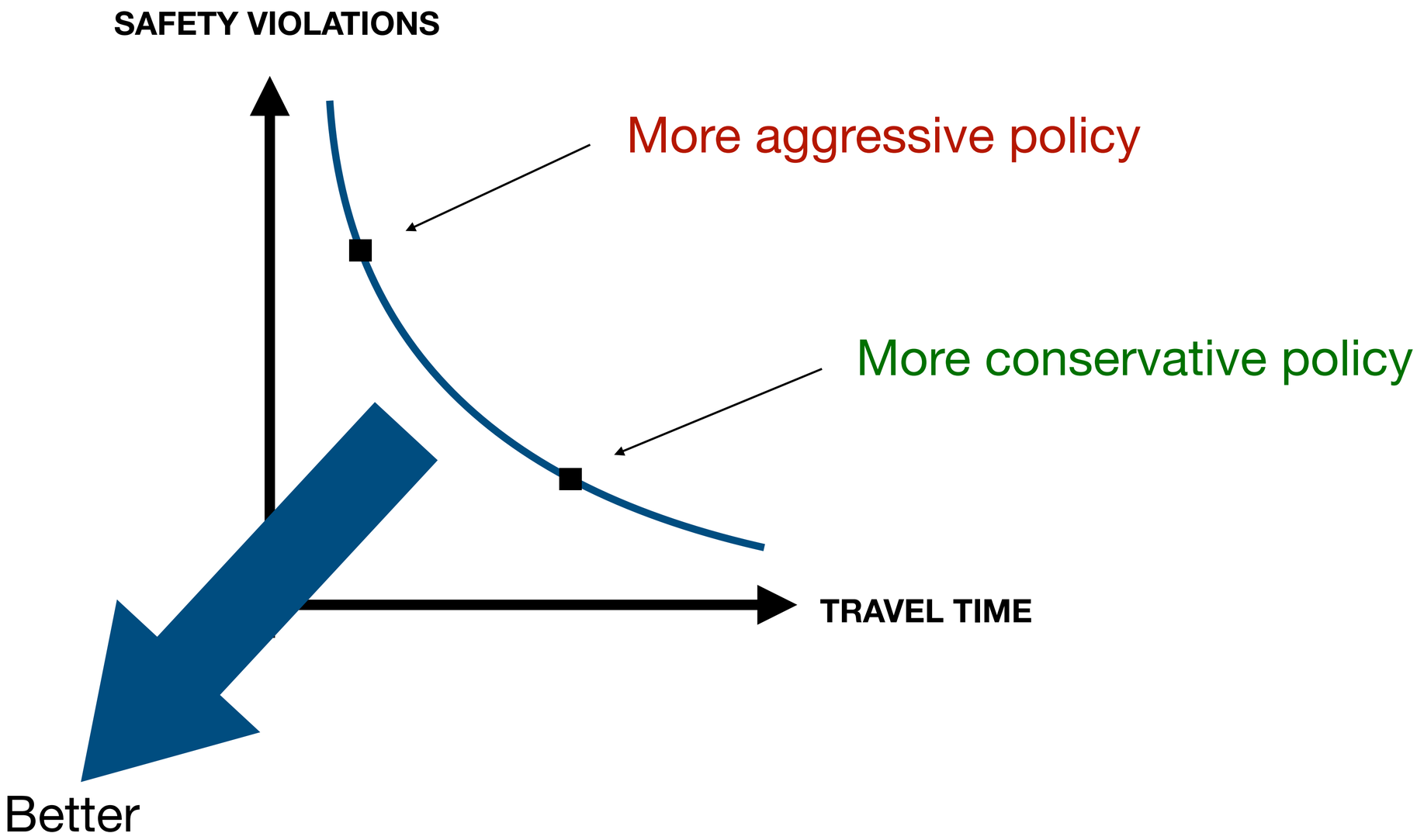


A model is **Pareto optimal** if improving one objective must come at a cost of another.

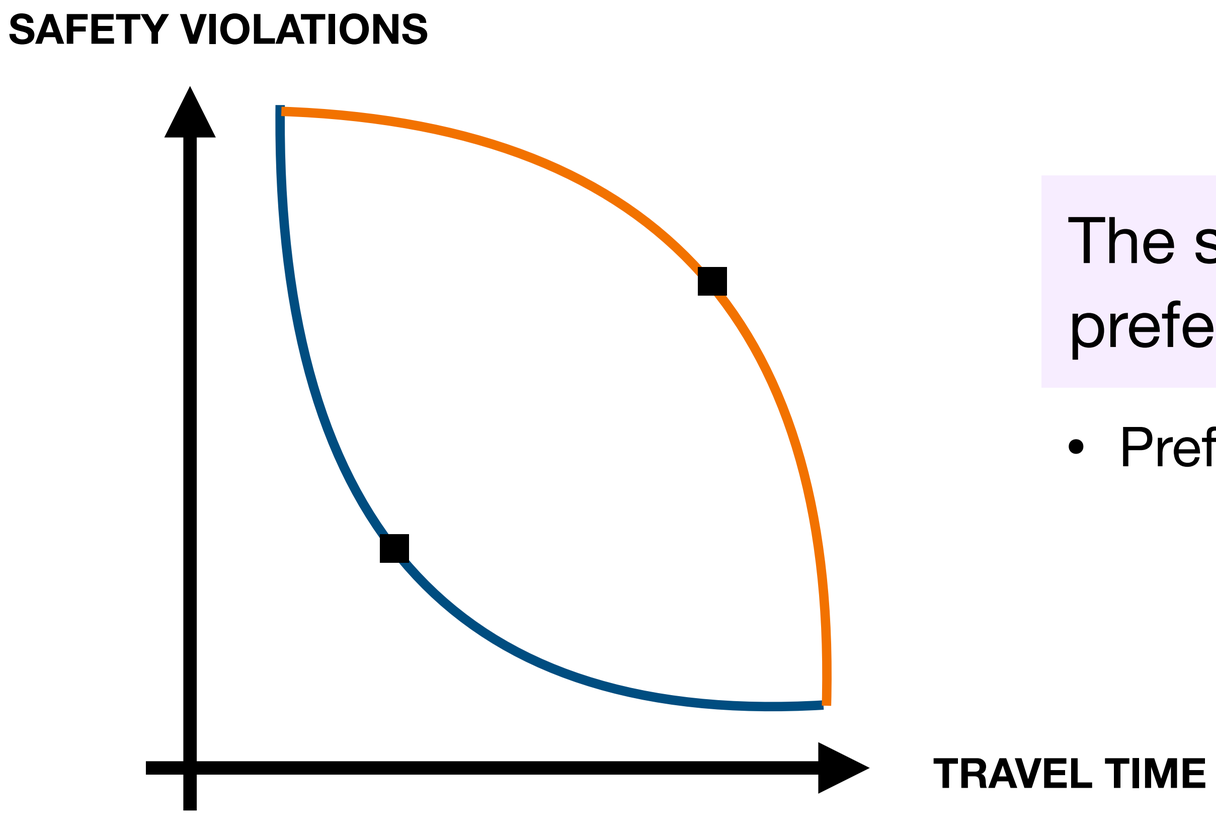
Visualization: Pareto front



Visualization: Pareto front



Which trade-off to pick?



The specific type of *trade-off* that is preferred is not usually known beforehand.

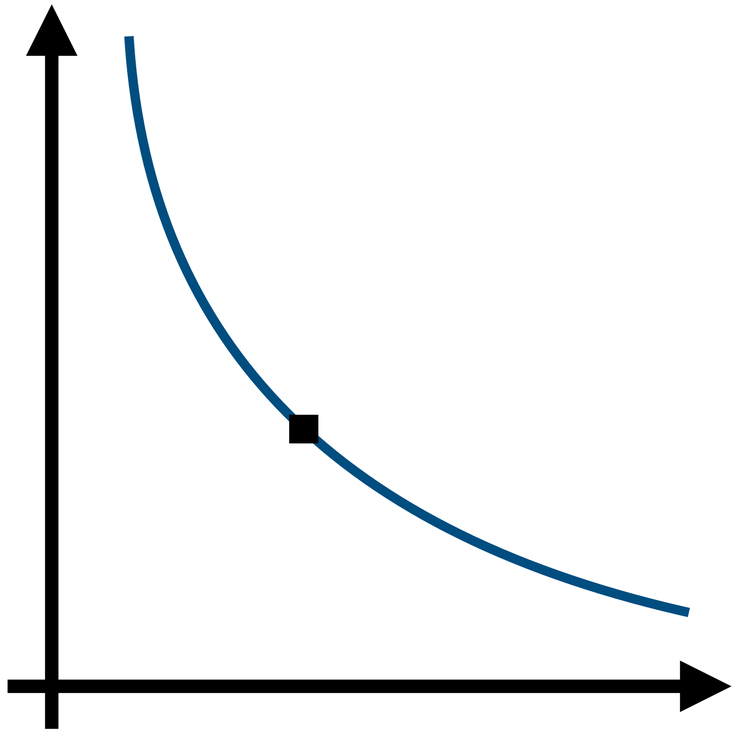
- Preference may depend on the Pareto set itself.

Main goal: learn all Pareto-optimal models

(Allows for downstream user preferences to dictate which one to deploy).

Question: how do you find Pareto-optimal models?

Scalarization



Pareto-optimal models often corresponds to solutions of *scalarized* objectives.

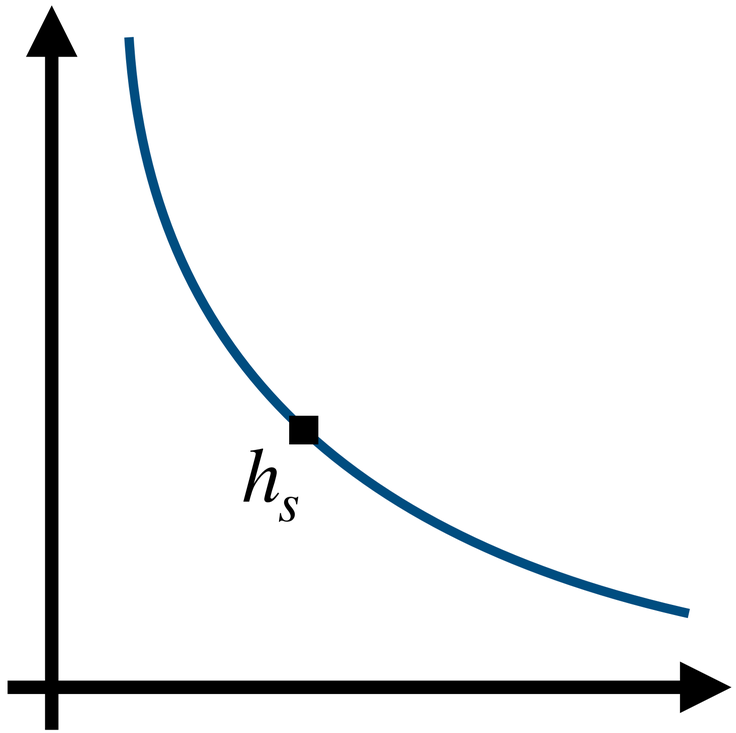
Examples:

$$\min_{h \in \mathscr{H}} \frac{1}{K} \sum_{k \in [K]} R_k(h) \quad \text{or} \quad \min_{h \in \mathscr{H}} \max_{k \in [K]} R_k(h) \,.$$

Minimize task-averaged risk

Minimize worst risk

The s-trade-off solutions



- Let $s : \mathbb{R}^K \to \mathbb{R}$ be a scalarizer.
- The s-**trade-off solution** minimizes:

$$h_s = \arg\min_{h \in \mathcal{H}} s\Big(\mathbf{R}(h)\Big).$$

- The Pareto set can be **parameterized** by a family $\{h_s : s \in \mathcal{S}\}$

What learning the Pareto set means

$$s\left(\mathbf{R}(\hat{h}_s)\right) < \min_{h \in \mathscr{H}} s\left(\mathbf{R}(h)\right) + \varepsilon\,.$$

- Return ε-optimal model for each s-trade-off where $s \in \mathcal{S}$.

What learning the Pareto set means

$$\Pr\left(\forall s \in \mathcal{S}, \quad s\left(\mathbf{R}(\hat{h}_s)\right) < \min_{h \in \mathcal{H}} s\left(\mathbf{R}(h)\right) + \varepsilon \right) > 1 - \delta \,.$$

- Return ε-optimal model for each s-trade-off where $s \in \mathcal{S}$.
- Succeed on the whole Pareto set with high probability.

Question: How much data is needed to learn all Pareto optimal models?

Sample Complexity Bounds

Supervised multi-objective learning

C. Cortes, M. Mohri, J. Gonzalvo, and D. Storcheus. *Agnostic learning with multiple objectives*. NeurIPS 2020.

P. Súkeník and C. Lampert. *Generalization in multi-objective machine learning*. Neural Computing and Applications 2024.

Sample Complexity Bounds

Informal Theorem (Súkeník & Lampert 2024). Let $\mathscr{H}$ be a model class. To ε-learn all Pareto optimal models in $\mathscr{H}$, we need:

$$\tilde{O}\left(\frac{\mathrm{VC}(\mathscr{H}) \cdot K}{\varepsilon^2}\right) \text{ samples,}$$

where K is the number of tasks.

Supervised multi-objective learning

C. Cortes, M. Mohri, J. Gonzalvo, and D. Storcheus. *Agnostic learning with multiple objectives*. NeurIPS 2020.

P. Súkeník and C. Lampert. *Generalization in multi-objective machine learning*. Neural Computing and Applications 2024.

Sample Complexity Bounds

Informal Theorem (Súkeník & Lampert 2024). Let $\mathscr{H}$ be a model class. To ε-learn all Pareto optimal models in $\mathscr{H}$, we need:

$$\tilde{\Theta}\left(\frac{\mathrm{VC}(\mathscr{H}) \cdot K}{\varepsilon^2}\right) \text{ samples,}$$

where K is the number of tasks.

Tight because solving each task alone costs: $\Omega\left(\frac{\mathrm{VC}(\mathscr{H})}{\varepsilon^2}\right)$ samples.

Sample Complexity Bounds

Informal Theorem (Súkeník & Lampert 2024). Let $\mathscr{H}$ be a model class. To ε-learn all Pareto optimal models in $\mathscr{H}$, we need:

$$\tilde{\Theta}\left(\frac{\mathrm{VC}(\mathscr{H}) \cdot K}{\varepsilon^2}\right) \text{ samples,}$$

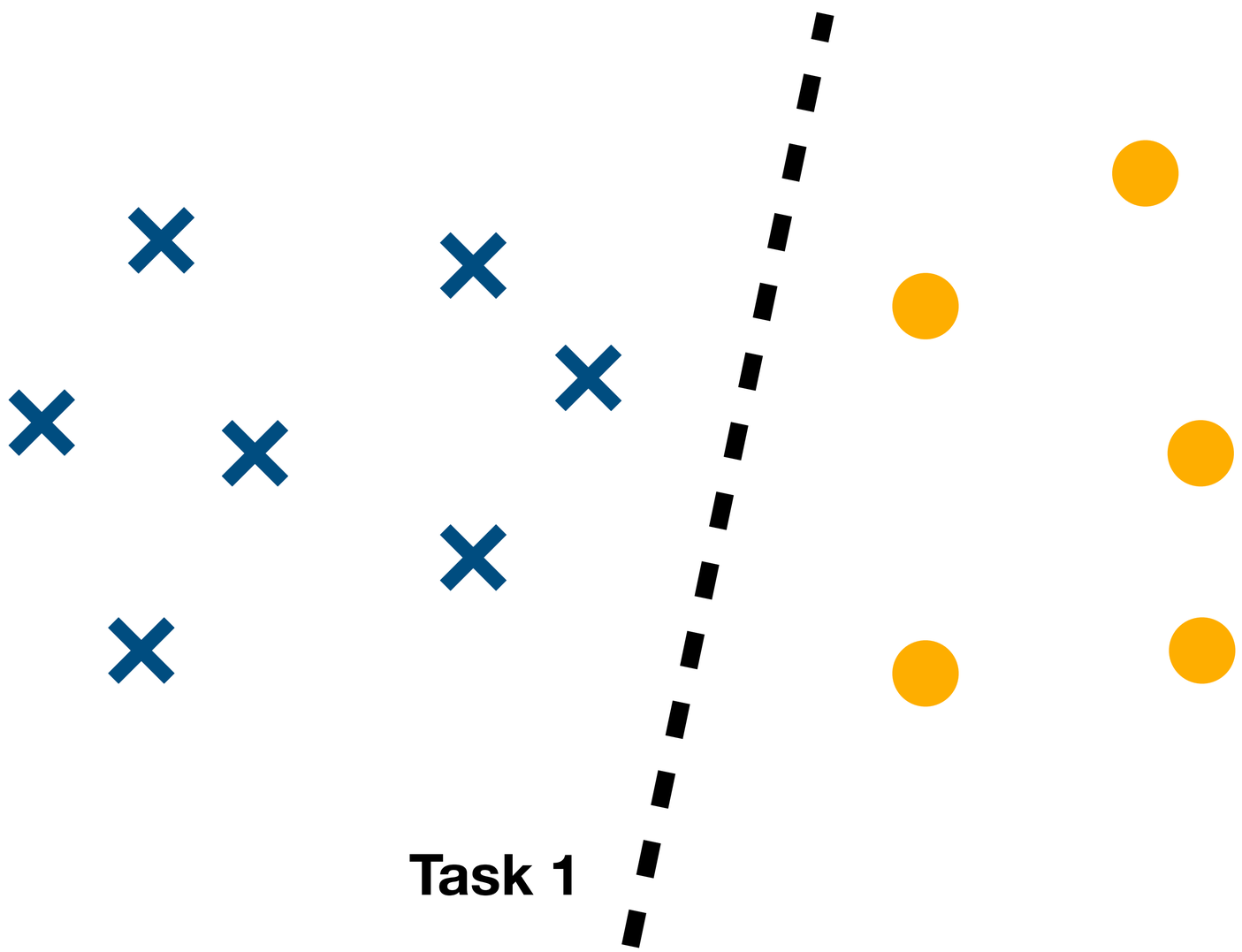
where K is the number of tasks.

Tight because **solving each task alone** costs: $\Omega\left(\frac{\mathrm{VC}(\mathscr{H})}{\varepsilon^2}\right)$ samples.

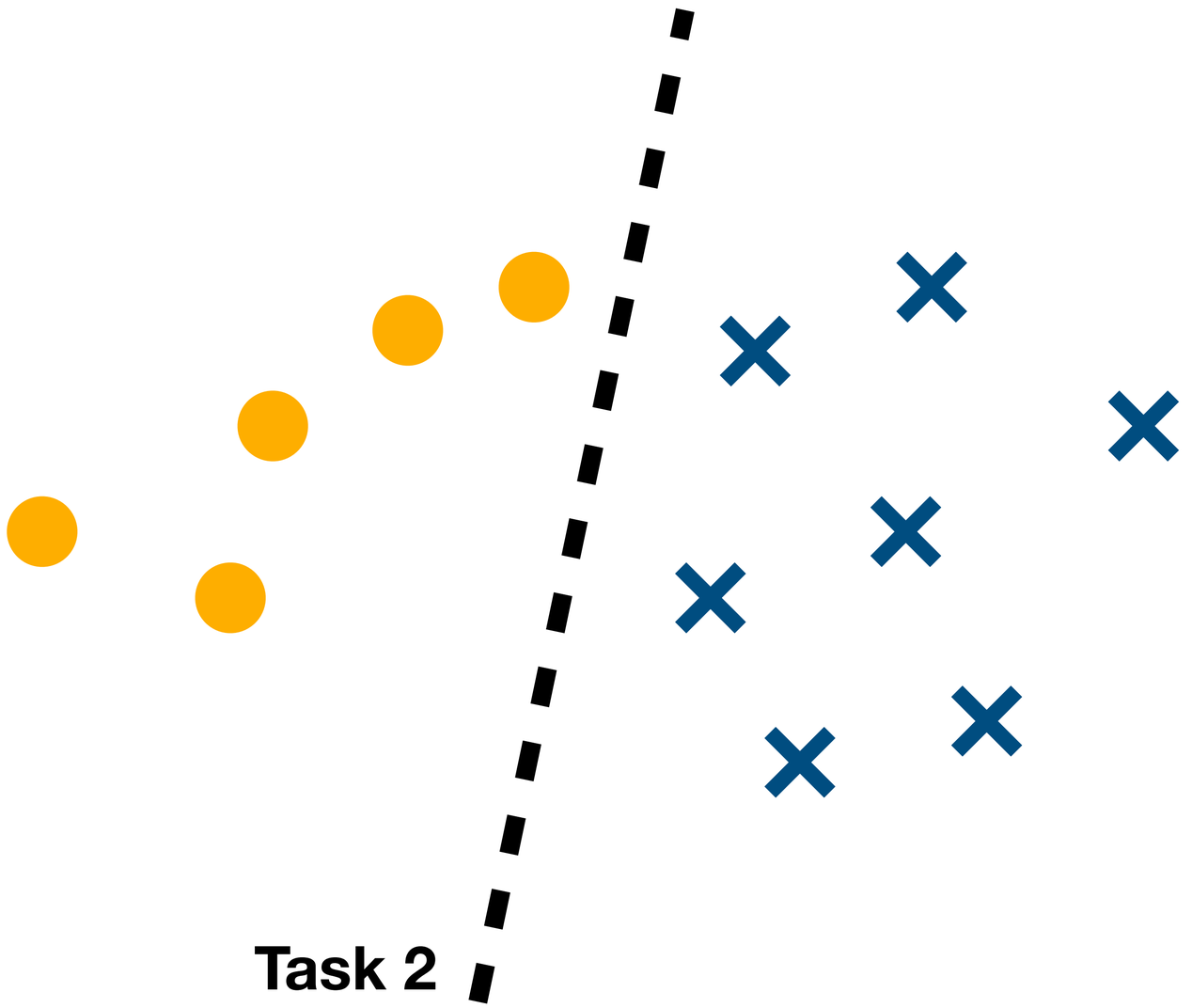
Is this really where the statistical hardness is coming from?

II. Multi-objective learning for simple tasks?

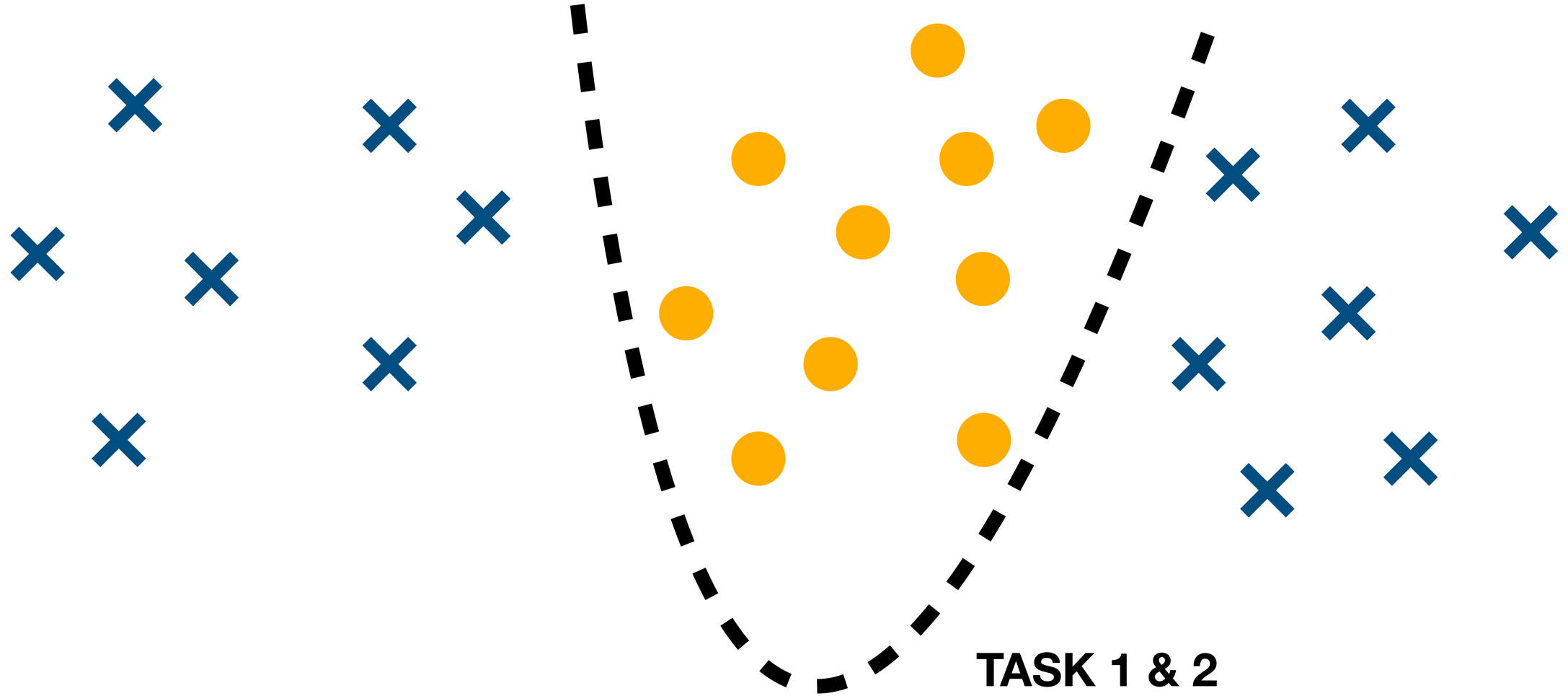
Even if individual tasks are easy to learn



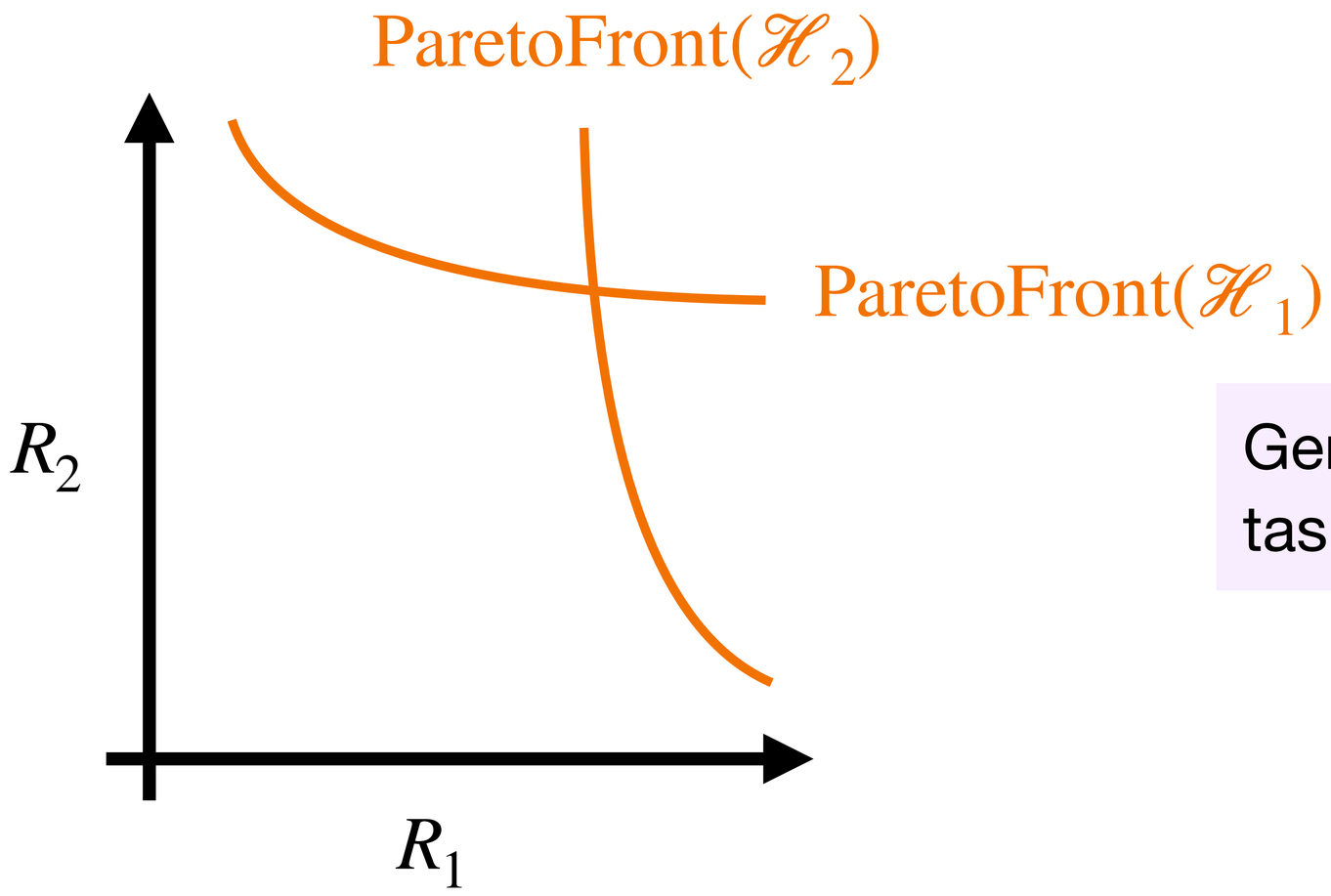
Even if individual tasks are easy to learn



Joint tasks may require more complex classes

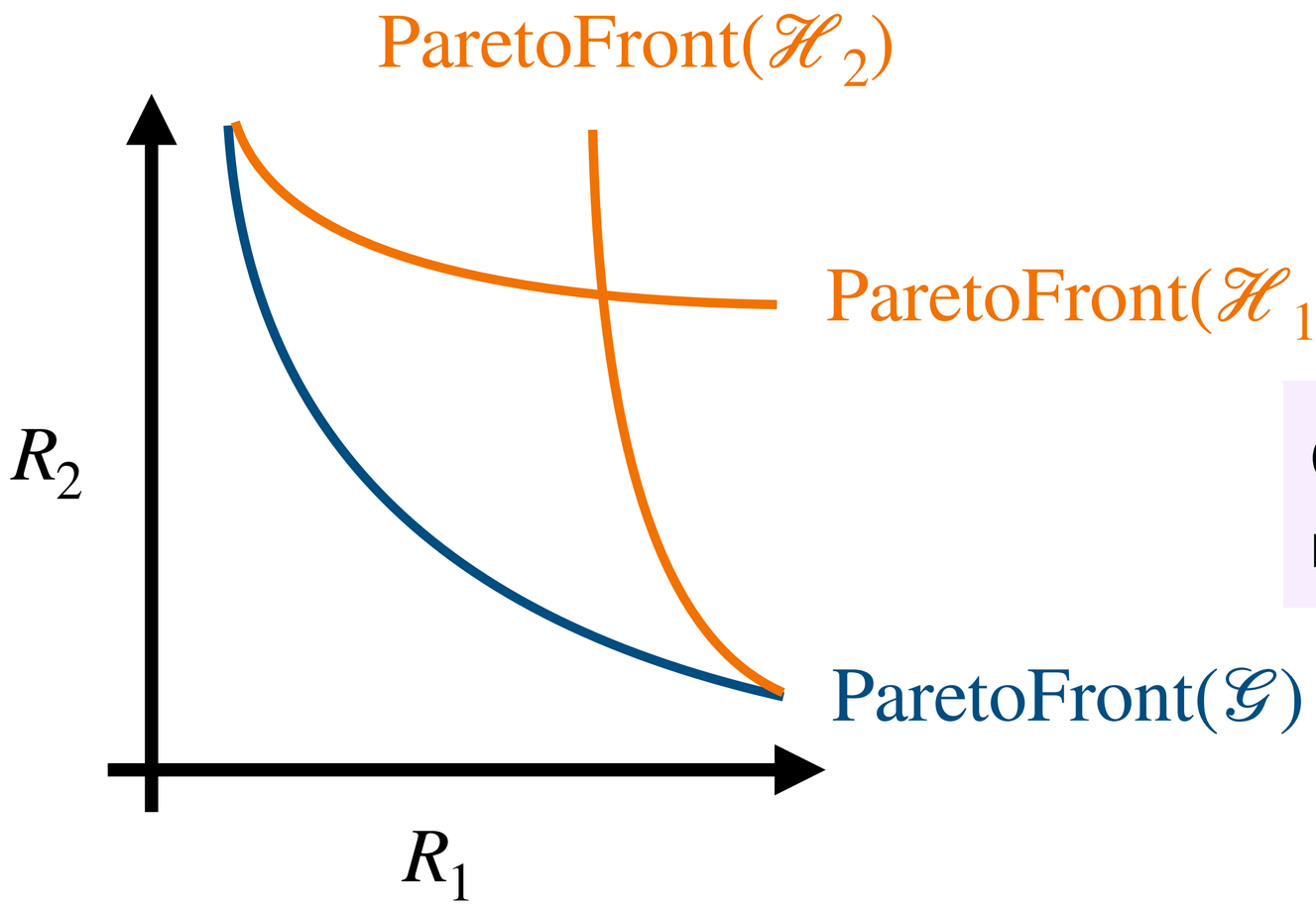


From individual to joint model classes



Generally, models good for one task may not work well for others.

From individual to joint model classes



Classes with good trade-offs $\mathscr{G}$ may be much harder to learn.

Question: is paying the full supervised cost in the more complex class needed?

Question: is paying the full supervised cost in the more complex class needed?

or, can we break up a complex multi-objective task into many simple ones?

Question: is paying the full supervised cost in the more complex class needed?

or, can we break up a complex multi-objective task into many simple ones?

We want the best of both worlds:

- low statistical cost to learn
- efficient trade-offs across risks

Semi-supervised setting

Assumptions:

Semi-supervised setting

Assumptions:

- We have access to simple model classes $\mathscr{H}_k$ for each task $k \in [K]$.

Semi-supervised setting

Assumptions:

- We have access to simple model classes $\mathscr{H}_k$ for each task $k \in [K]$.
- These classes are guaranteed to perform well individually. In fact,

$$f_k^\star \in \mathscr{H}_k$$

Semi-supervised setting

Assumptions:

- We have access to simple model classes $\mathscr{H}_k$ for each task $k \in [K]$.
- These classes are guaranteed to perform well individually. In fact,

$$f_k^\star \in \mathscr{H}_k$$

- We also have access to unlabeled data from P_X^k.

A negative result

Theorem. Let $\mathscr{G}$ be a model class. To ε-learn all Pareto optimal models, we need:

$$\tilde{\Theta}\left(\frac{\mathrm{VC}(\mathscr{G}) \cdot K}{\varepsilon^2}\right) \text{ samples,}$$

A negative result

Theorem. Let $\mathcal{G}$ be a model class. To ε-learn all Pareto optimal models, we need:

$$\tilde{\Theta}\left(\frac{\mathrm{VC}(\mathcal{G}) \cdot K}{\varepsilon^2}\right) \text{ samples,}$$

even if we know $f_k^\star$ and have infinite unlabeled data from P_k for each $k \in [K]$.

A negative result

Theorem. Let $\mathscr{G}$ be a model class. To ε-learn all Pareto optimal models, we need:

$$\tilde{\Theta}\left(\frac{\mathrm{VC}(\mathscr{G}) \cdot K}{\varepsilon^2}\right) \text{ samples,}$$

even if we know $f_k^\star$ and have infinite unlabeled data from P_k for each $k \in [K]$.

Intuition: ability to drive fast + ability to be safe $\not\Rightarrow$ ability to drive fast safely.

A negative result

Theorem. Let $\mathcal{G}$ be a model class. To ε-learn all Pareto optimal models, we need:

$$\tilde{\Theta}\left(\frac{\mathrm{VC}(\mathcal{G}) \cdot K}{\varepsilon^2}\right) \text{ samples,}$$

even if we know $f_k^\star$ and have infinite unlabeled data from P_k for each $k \in [K]$.

Intuition: ability to drive fast + ability to be safe $\not\Rightarrow$ ability to drive fast safely.

Implication: learning trade-offs can be arbitrarily harder than solving individual tasks.

Proof idea

Task 1: predict outcome of flip of Coin 1

Task 2: predict outcome of flip of Coin 2

Multi-objective task: predict the majority vote

Proof idea

Task 1: predict outcome of flip of Coin 1

Task 2: predict outcome of flip of Coin 2

Multi-objective task: predict the majority vote

Promise:

- The Bernoulli parameters (p_1, p_2) of the coins satisfy:

$$p_1 \in \left\{ \frac{1}{2}, \frac{1}{2} + \varepsilon \right\} \qquad \text{and} \qquad p_2 \in \left\{ \frac{1}{2} - \varepsilon, \frac{1}{2} \right\}.$$

Proof idea

Task 1: predict outcome of flip of Coin 1 (optimal prediction = H)

Task 2: predict outcome of flip of Coin 2

Multi-objective task: predict the majority vote

Promise:

- The Bernoulli parameters (p_1, p_2) of the coins satisfy:

$$p_1 \in \left\{ \frac{1}{2}, \frac{1}{2} + \varepsilon \right\} \qquad \text{and} \qquad p_2 \in \left\{ \frac{1}{2} - \varepsilon, \frac{1}{2} \right\}.$$

Proof idea

Task 1: predict outcome of flip of Coin 1 (optimal prediction = H)

Task 2: predict outcome of flip of Coin 2 (optimal prediction = T)

Multi-objective task: predict the majority vote

Promise:

- The Bernoulli parameters (p_1, p_2) of the coins satisfy:

$$p_1 \in \left\{ \frac{1}{2}, \frac{1}{2} + \varepsilon \right\} \qquad \text{and} \qquad p_2 \in \left\{ \frac{1}{2} - \varepsilon, \frac{1}{2} \right\}.$$

Proof idea

Task 1: predict outcome of flip of Coin 1 (optimal prediction = H)

Task 2: predict outcome of flip of Coin 2 (optimal prediction = T)

Multi-objective task: predict the majority vote (requires learning bias of coins)

Promise:

- The Bernoulli parameters (p_1, p_2) of the coins satisfy:

$$p_1 \in \left\{ \frac{1}{2}, \frac{1}{2} + \varepsilon \right\} \qquad \text{and} \qquad p_2 \in \left\{ \frac{1}{2} - \varepsilon, \frac{1}{2} \right\}.$$

Problem: loss functions such as the zero-one loss can be “uninformative”.

II. Multi-objective learning with nice losses

Semi-supervised setting

Assumptions:

- We have access to simple model classes $\mathscr{H}_k$ for each task $k \in [K]$.
- These classes are guaranteed to perform well individually. In fact,

$$f_k^\star \in \mathscr{H}_k$$

- We also have access to unlabeled data from P_X^k.
- **We have nice losses ℓ_k.**

Bregman losses

There are standard losses that are much nicer than the zero-one loss:

- Square loss
- Logistic loss/cross-entropy loss
- etc.

Bregman losses

There are standard losses that are much nicer than the zero-one loss:

- Square loss
- Logistic loss/cross-entropy loss
- etc.

Risk Decomposition for Bregman Losses

$$R(h) = \mathbb{E}\left[\ell(Y, f^\star(X))\right] + \mathbb{E}\left[\ell(f^\star(X), h(X))\right]$$

Bregman losses

There are standard losses that are much nicer than the zero-one loss:

- Square loss
- Logistic loss/cross-entropy loss
- etc.

Risk Decomposition for Bregman Losses

$$R(h) = \mathbb{E}\left[\ell(Y, f^\star(X))\right] + \mathbb{E}\left[\ell(f^\star(X), h(X))\right]$$

This expectation is only over unlabeled data P_X

Risk functional estimation

Risk Decomposition for Bregman Losses

$$R(h) = \mathbb{E}\left[\ell(Y, f^\star(X))\right] + \mathbb{E}\left[\ell(f^\star(X), h(X))\right]$$

$\Longrightarrow$ knowledge of the risk minimizer $f^\star$ tells us a lot about the risk R.

Risk functional estimation

Risk Decomposition for Bregman Losses

$$R(h) = \mathbb{E}\left[\ell(Y, f^\star(X))\right] + \mathbb{E}\left[\ell(f^\star(X), h(X))\right]$$

A plug-in estimator

- $\hat{f} \leftarrow \widehat{\arg\min_{h \in \mathscr{H}}} \hat{\mathbb{E}}\left[\ell(Y, h(X))\right]$

Risk functional estimation

Risk Decomposition for Bregman Losses

$$R(h) = \mathbb{E}\Big[\ell(Y, f^\star(X))\Big] + \mathbb{E}\Big[\ell(f^\star(X), h(X))\Big]$$

A plug-in estimator

- $\hat{f} \leftarrow \widehat{\arg\min_{h \in \mathscr{H}}} \hat{\mathbb{E}}\Big[\ell(Y, h(X))\Big]$
- $\hat{R}(h) = \hat{\mathbb{E}}\Big[\ell(Y, \hat{f}(X))\Big] + \hat{\mathbb{E}}\Big[\ell(\hat{f}(X), h(X))\Big]$

A pseudo-labeling algorithm

For each task $k \in [K]$:

- Use labeled data to approximately recover $f_k^\star$ from $\mathscr{H}_k$.

A pseudo-labeling algorithm

For each task $k \in [K]$:

- Use labeled data to approximately recover $f_k^\star$ from $\mathscr{H}_k$.
- Generate a lot of pseudo-label data $(X, \hat{f}_k(X))$ to estimate $\hat{R}_k$.

A pseudo-labeling algorithm

For each task $k \in [K]$:

- Use labeled data to approximately recover $f_k^\star$ from $\mathscr{H}_k$.
- Generate a lot of pseudo-label data $(X, \hat{f}_k(X))$ to estimate $\hat{R}_k$.

This only requires additional unlabeled data from P_X^k.

A pseudo-labeling algorithm

For each task $k \in [K]$:

- Use labeled data to approximately recover $f_k^\star$ from $\mathscr{H}_k$.
- Generate a lot of pseudo-label data $(X, \hat{f}_k(X))$ to estimate $\hat{R}_k$.

For each $s \in \mathcal{S}$ (parametrizing the Pareto set):

A pseudo-labeling algorithm

For each task $k \in [K]$:

- Use labeled data to approximately recover $f_k^\star$ from $\mathscr{H}_k$.
- Generate a lot of pseudo-label data $(X, \hat{f}_k(X))$ to estimate $\hat{R}_k$.

For each $s \in \mathcal{S}$ (parametrizing the Pareto set):

- Solve the optimization problem:

$$\hat{g}_s \leftarrow \underset{g \in \mathscr{G}}{\arg\min}\ s\left(\hat{\mathbf{R}}(g)\right).$$

A positive result

Theorem. Let $\mathscr{G}$ be a joint model class,

A positive result

Theorem. Let $\mathscr{G}$ be a joint model class, and let $\mathscr{H}_k \ni f_k^\star$ be model class that contains the Bayes-optimal model.

A positive result

Theorem. Let $\mathscr{G}$ be a joint model class, and let $\mathscr{H}_k \ni f_k^\star$ be model class that contains the Bayes-optimal model. If the losses ℓ_k are Bregman losses:

A positive result

Theorem. Let $\mathscr{G}$ be a joint model class, and let $\mathscr{H}_k \ni f_k^\star$ be model class that contains the Bayes-optimal model. If the losses ℓ_k are Bregman losses:

$$O\left(\frac{\sum_k \mathrm{VC}(\mathscr{H}_k)}{\varepsilon^4}\right) \text{ labeled samples,} \qquad O\left(\frac{\mathrm{VC}(\mathscr{G})}{\varepsilon^2}\right) \text{unlabeled samples}$$

are enough to ε-learn all Pareto-optimal models.

A positive result

Theorem. Let $\mathscr{G}$ be a joint model class, and let $\mathscr{H}_k \ni f_k^\star$ be model class that contains the Bayes-optimal model. If the losses ℓ_k are Bregman losses:

$$O\left(\frac{\sum_k \mathrm{VC}(\mathscr{H}_k)}{\varepsilon^4}\right) \text{ labeled samples,} \quad O\left(\frac{\mathrm{VC}(\mathscr{G})}{\varepsilon^2}\right) \text{unlabeled samples}$$

are enough to ε-learn all Pareto-optimal models.

Importantly, the label sample complexity does not the complexity of the joint class $\mathscr{G}$ in which the good trade-offs are possible.

A positive result

Theorem. Let $\mathscr{G}$ be a joint model class, and let $\mathscr{H}_k \ni f_k^\star$ be model class that contains the Bayes-optimal model. If the losses ℓ_k are Bregman losses:

$$O\left(\frac{\sum_k \mathrm{VC}(\mathscr{H}_k)}{\varepsilon^4}\right) \text{ labeled samples,} \quad O\left(\frac{\mathrm{VC}(\mathscr{G})}{\varepsilon^2}\right) \text{unlabeled samples}$$

are enough to ε-learn all Pareto-optimal models.

Importantly, the label sample complexity does not the complexity of the joint class $\mathscr{G}$ in which the good trade-offs are possible.

Tighter, data-dependent bounds are also possible. See paper/poster.

Takeaways

- Multi-objective learning (MOL) problems are ubiquitous in practice, but subtle.
- Learning good trade-offs can be much harder than solving the individual tasks.
- Structure in loss/feedback important for efficient multi-objective generalization.

Thanks!

On the sample complexity of semi-supervised multi-objective learning (NeurIPS 2025)

https://arxiv.org/abs/2508.17152