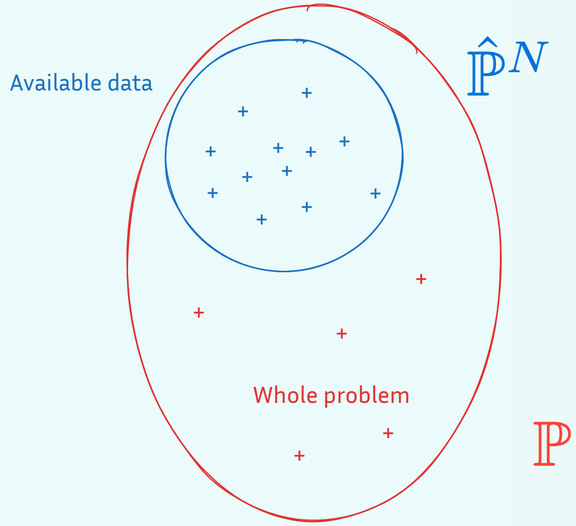
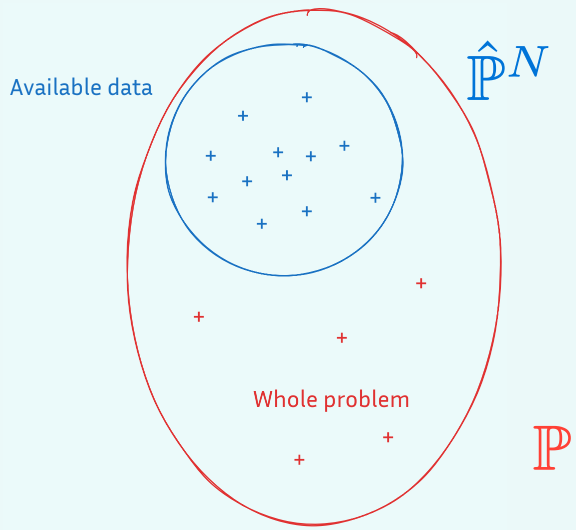
Florian Vincent – ICSP 2025

Solving Wasserstein distributionally robust classification by regularization

- Reference data: N points (finite) $\boldsymbol{\xi}_1 \ldots \boldsymbol{\xi}_N$
  $\hat{\mathbb{P}}^N = \frac{1}{N} \sum_{\boldsymbol{\xi}_i} \delta_{\boldsymbol{\xi}_i}$
- Parameterized loss: $L_\theta \rightsquigarrow$ find θ
  (e.g. θ are weights in an ANN)



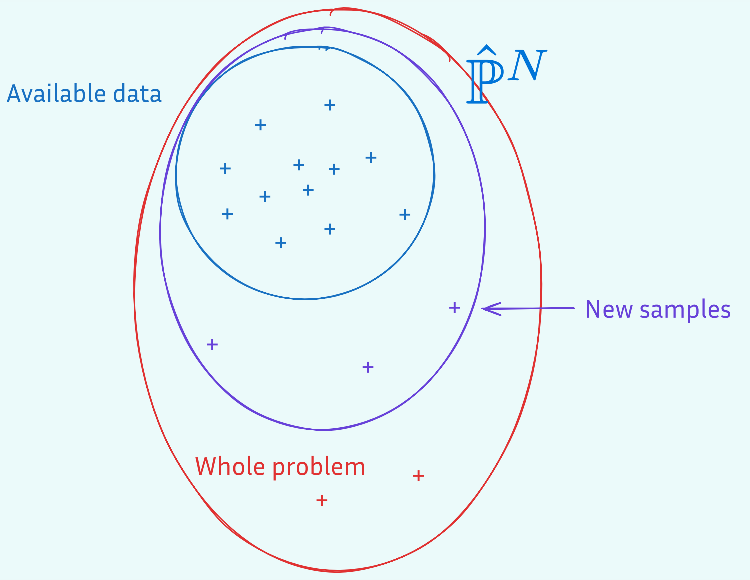
- Reference data: N points (finite) $\xi_1 \ldots \xi_N$
  $\hat{\mathbb{P}}^N = \frac{1}{N} \sum_{\xi_i} \delta_{\xi_i}$
- Parameterized loss: $L_\theta \rightsquigarrow$ find θ
  (e.g. θ are weights in an ANN)



- Learning:
  - Supervised: $(\xi_{\text{input}}, \xi_{\text{target}})$
  - Unsupervised: (ξ_{data})
- Operations research:
  - data-oriented: (ξ_{data})

Shift in the data

Various techniques exist, mainly in specific cases.



Distributional rubustness:

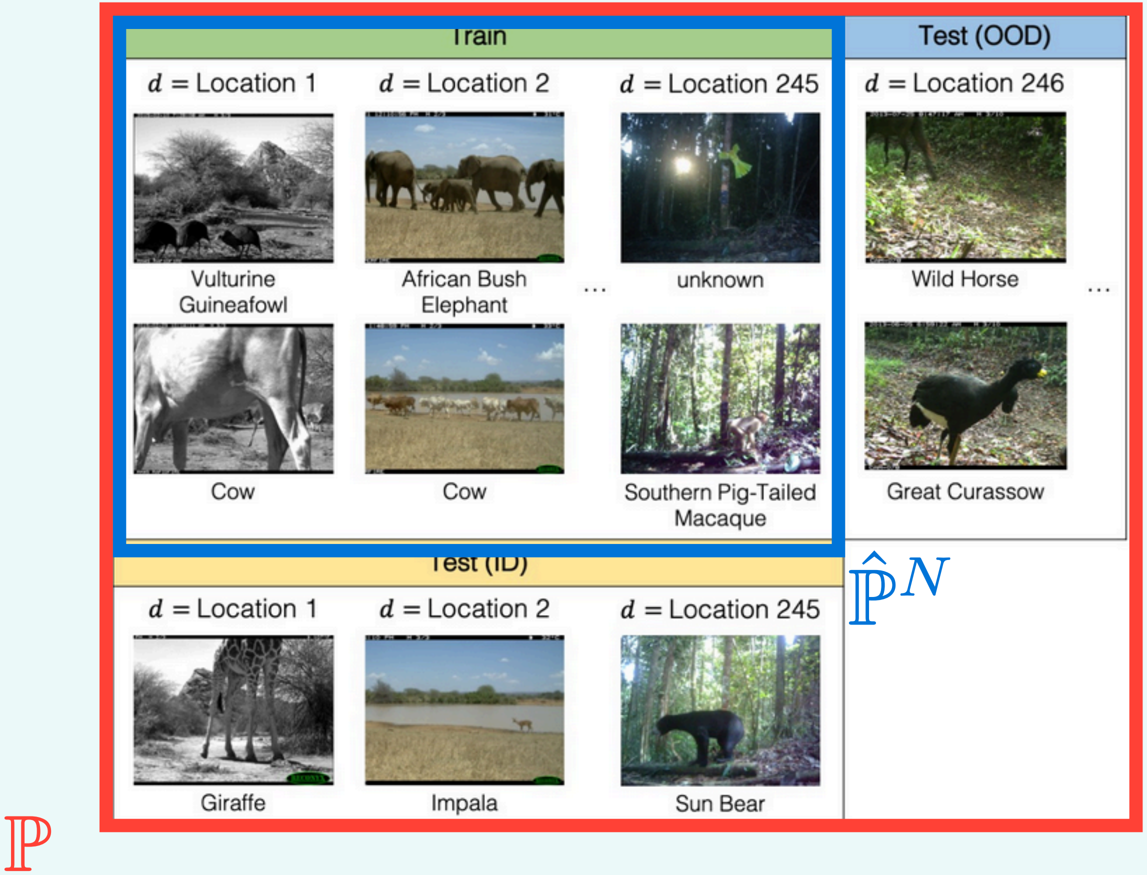
- Wide topic with rich diversity of methods cf (This Session, July 2025)

(Wasserstein) Distributional robustness:

- Entire zoos of examples formulated as convex programs (Mohajerin Esfahani & Kuhn, 2018)
- Extensive theoretical analysis (Kuhn et al., 2019)
- (New!) Sinkhorn regularisation of Wasserstein transport (Wang et al., 2023)+(Azizian et al., 2023)

Data with distribution shift

E.g.: **iWildsCam** dataset

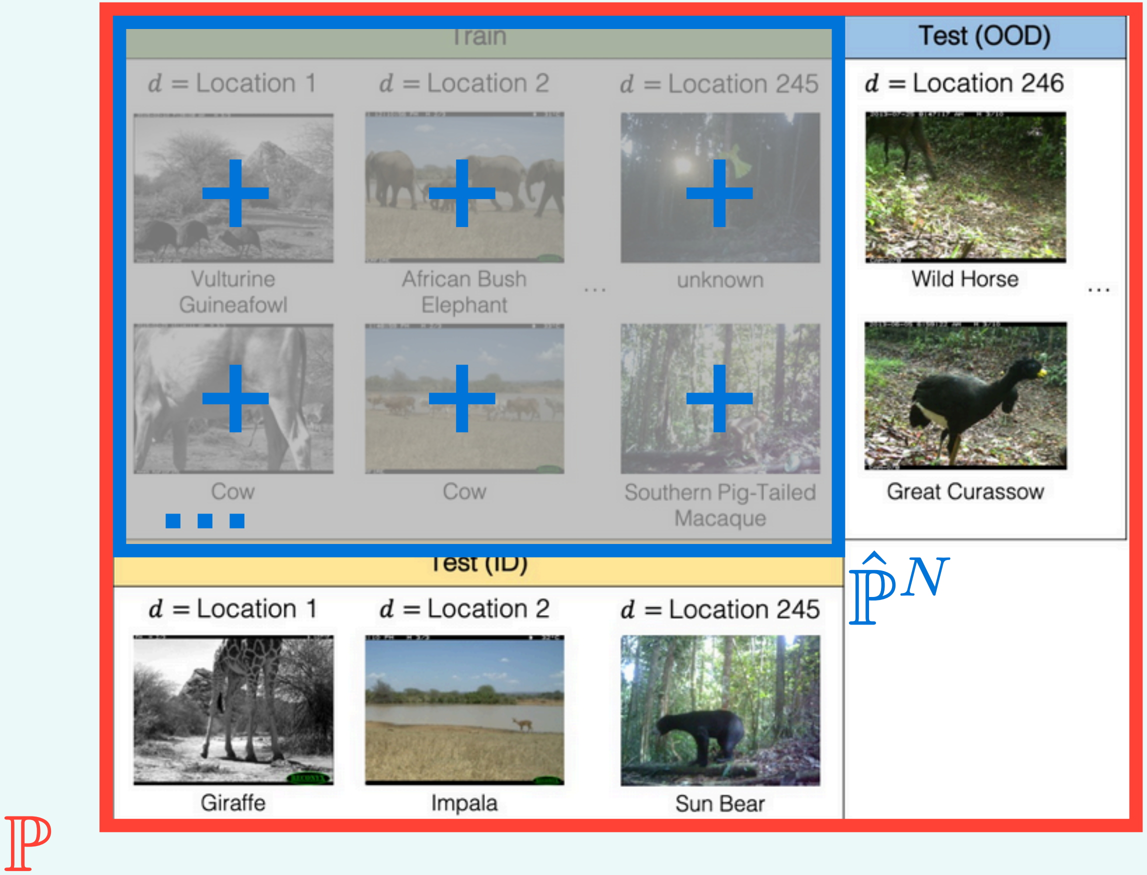


Data with distribution shift

E.g.: **iWildsCam** dataset

Empirical risk minimization (ERM):

$$\min_{\theta} \frac{1}{N} \sum_{\xi \sim \hat{\mathbb{P}}^N} [L_\theta(\xi)]$$



(Distributional) Robustness

ERM

$$\min_{\theta} \frac{1}{N}\sum_{i=1}^{N} L_{\theta}(\xi_i)$$

Fit on dataset is not enough to protect against **shifts**

(Distributional) Robustness

ERM

$$\min_{\theta} \frac{1}{N}\sum_{i=1}^{N} L_{\theta}(\xi_i)$$

Fit on dataset is not enough to protect against **shifts**

→

Robustify

$$\min_{\theta} \mathbb{E}_{\zeta\sim\mathbb{Q}} L_{\theta}(\zeta)$$

With $\mathbb{Q}$ a “wider” distribution

(Distributional) Robustness

ERM

$$\min_{\theta} \frac{1}{N}\sum_{i=1}^{N} L_{\theta}(\xi_i)$$

Fit on dataset is not enough to protect against **shifts**

→

Robustify

$$\min_{\theta} \mathbb{E}_{\zeta\sim\mathbb{Q}} L_{\theta}(\zeta)$$

With $\mathbb{Q}$ a “wider” distribution

→

Worst case

$$\min_{\theta} \sup_{\mathbb{Q}} \mathbb{E}_{\zeta\sim\mathbb{Q}} L_{\theta}(\zeta)$$

With the worst $\mathbb{Q}$! But among which?

(Distributional) Robustness

ERM

$$\min_{\theta} \frac{1}{N}\sum_{i=1}^{N} L_{\theta}(\xi_i)$$

Fit on dataset is not enough to protect against **shifts**

→

Robustify

$$\min_{\theta} \mathbb{E}_{\zeta\sim\mathbb{Q}} L_{\theta}(\zeta)$$

With $\mathbb{Q}$ a “wider” distribution

→

Worst case

$$\min_{\theta} \sup_{\mathbb{Q}} \mathbb{E}_{\zeta\sim\mathbb{Q}} L_{\theta}(\zeta)$$

With the worst $\mathbb{Q}$! But among which?

→

Define uncertainty

$$\min_{\theta} \sup_{\mathbb{Q}\approx\hat{\mathbb{P}}^N} \mathbb{E}_{\zeta\sim\mathbb{Q}} L_{\theta}(\zeta)$$

With $\mathbb{Q}$ the worst, “close to” $\hat{\mathbb{P}}^N$.

(Distributional) Robustness

ERM

$$\min_{\theta} \frac{1}{N}\sum_{i=1}^{N} L_{\theta}(\xi_i)$$

Fit on dataset is not enough to protect against **shifts**

→

Robustify

$$\min_{\theta} \mathbb{E}_{\zeta\sim\mathbb{Q}} L_{\theta}(\zeta)$$

With $\mathbb{Q}$ a “wider” distribution

→

Worst case

$$\min_{\theta} \sup_{\mathbb{Q}} \mathbb{E}_{\zeta\sim\mathbb{Q}} L_{\theta}(\zeta)$$

With the worst $\mathbb{Q}$! But among which?

→

Define uncertainty

$$\min_{\theta} \sup_{\mathbb{Q}\approx\hat{\mathbb{P}}^N} \mathbb{E}_{\zeta\sim\mathbb{Q}} L_{\theta}(\zeta)$$

With $\mathbb{Q}$ the worst, “close to” $\hat{\mathbb{P}}^N$.

In this talk: Wasserstein

W-DRO

Maximize over Wasserstein neighborhood of radius ρ (Mohajerin Esfahani & Kuhn, 2018).

$$\text{(Primal)} \qquad \min_{\theta} \sup_{W_c(\hat{\mathbb{P}}^N, \mathbb{Q}) \leq \rho} \mathbb{E}_{\zeta \sim \mathbb{Q}}[L_\theta(\zeta)]$$

$$\text{(Dual)} \qquad \min_{\theta, \lambda \geq 0} \frac{1}{N} \sum_{\xi_i \sim \hat{\mathbb{P}}^N} \sup_{\zeta} \Big\{ \underbrace{L_\theta(\zeta)}_{\text{objective}} + \lambda \underbrace{(\rho - c(\xi_i, \zeta))}_{\text{constraint (dualized)}} \Big\}$$

W-DRO

Maximize over Wasserstein neighborhood of radius ρ (Mohajerin Esfahani & Kuhn, 2018).

$$\text{(Primal)} \qquad \min_{\theta} \sup_{W_c(\hat{\mathbb{P}}^N, \mathbb{Q}) \leq \rho} \mathbb{E}_{\zeta \sim \mathbb{Q}}[L_\theta(\zeta)]$$

$$\text{(Dual)} \qquad \min_{\theta, \lambda \geq 0} \lambda \rho + \frac{1}{N} \sum_{\xi_i \sim \hat{\mathbb{P}}^N} \sup_{\zeta} \underbrace{\{L_\theta(\zeta) - \lambda\, c(\xi_i, \zeta)\}}_{c\text{-transform of } L_\theta}$$

What about the $\sup$?

$$\sup_{\zeta}\{L_\theta(\zeta) - \lambda\ \mathrm{c}(\zeta, \boldsymbol{\xi})\}$$

- $L_\theta(\zeta) - \lambda\ \mathrm{c}(\zeta, \boldsymbol{\xi})$ may be non-concave...
  $\Rightarrow$ high difficulty
- Worked out in some cases (through disciplined programming): see (Mohajerin Esfahani & Kuhn, 2018), e.g.:
  - linear/logistic regression,
  - SVM,
  - portfolio optimization,
  - piecewise affine losses

What about the $\sup$?

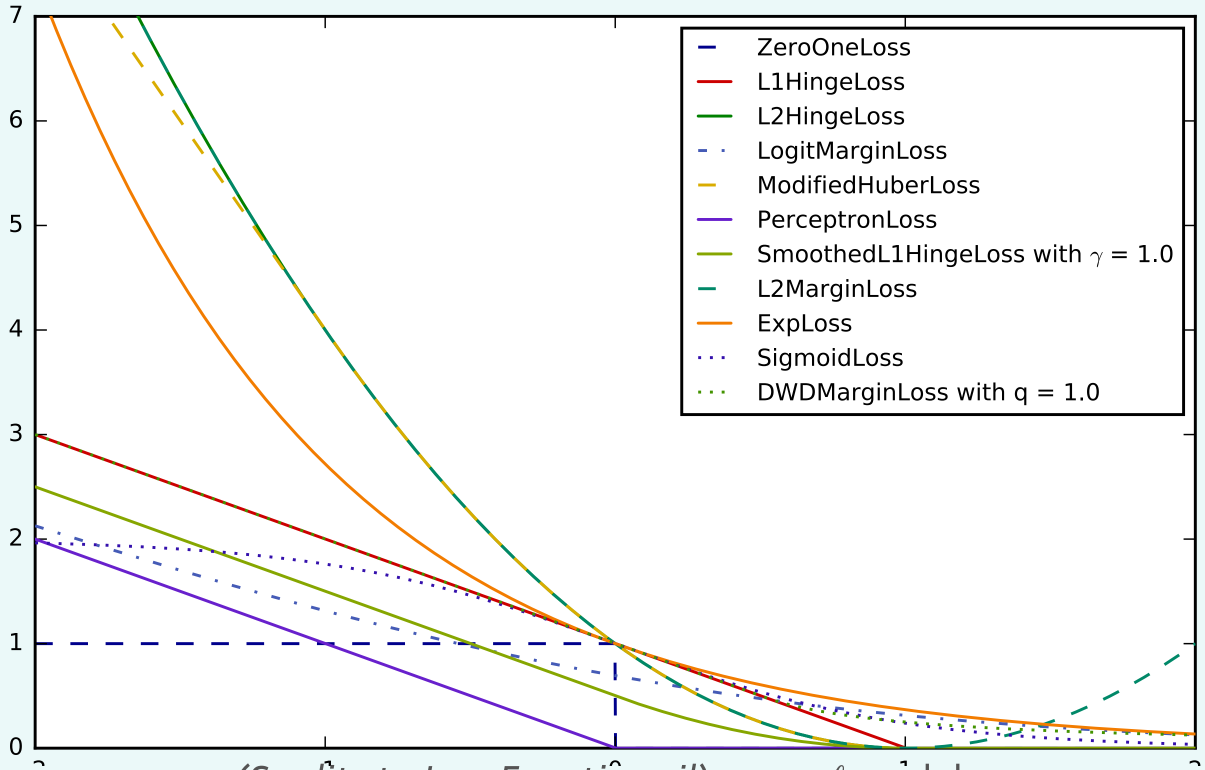
$$\sup_{\zeta}\{L_\theta(\zeta) - \lambda \, \mathrm{c}(\zeta, \boldsymbol{\xi})\}$$

- $L_\theta(\zeta) - \lambda \, \mathrm{c}(\zeta, \boldsymbol{\xi})$ may be non-concave...
  $\Rightarrow$ high difficulty
- Worked out in some cases (through disciplined programming): see (Mohajerin Esfahani & Kuhn, 2018), e.g.:
  - linear/**logistic regression**,
  - SVM,
  - portfolio optimization,
  - piecewise affine losses

The case of linear classification

Linear model:

$$\begin{cases} \boldsymbol{\xi} \in \mathbb{R}^d \\ L_{\theta}(\boldsymbol{\xi}) := \ell(\theta^T \boldsymbol{\xi}) \end{cases}$$



(Credits to LossFunctions.jl): some ℓ models

The case of linear classification

Theorem 1: [Shafieezadeh Abadeh et al. (2015)]

The WDRO problem for classification, with $c(\xi, \zeta) = \|\zeta - \xi\|_2$ *can be written:*

$$\min_{\theta} \quad \mathrm{Lip}(\ell)\rho\|\theta\|_2 + \frac{1}{N}\sum_{\xi_i \sim \hat{\mathbb{P}}^N} L_\theta(\xi_i)$$

For $c(\xi, \zeta) = \|\zeta - \xi\|$

- Soft margin: $\log\left(1 + e^{-y\theta^T x}\right)$ ✅
- Hinge: $\max\{0, 1 - y\theta^T x\}$ ✅
- ...

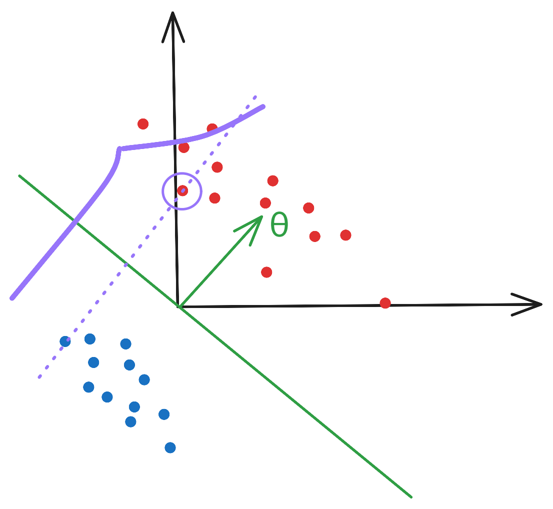
for $c(\xi, \zeta) = \|\zeta - \xi\|^2$

- L2-hinge: $\max\left\{0, \left(1 - y\theta^T x\right)^2\right\}$ ✅
- ...

The case of linear classification

This relies on a simple assumption (idea from (Gao & Kleywegt, 2023)):

The loss must behave like the costs in "worst case region"



- Well classified samples: $-\theta^T \boldsymbol{\xi} \geq 0$, region where ℓ is small
- Misclassified samples: $-\theta^T \boldsymbol{\xi} < 0$, region where ℓ grows.

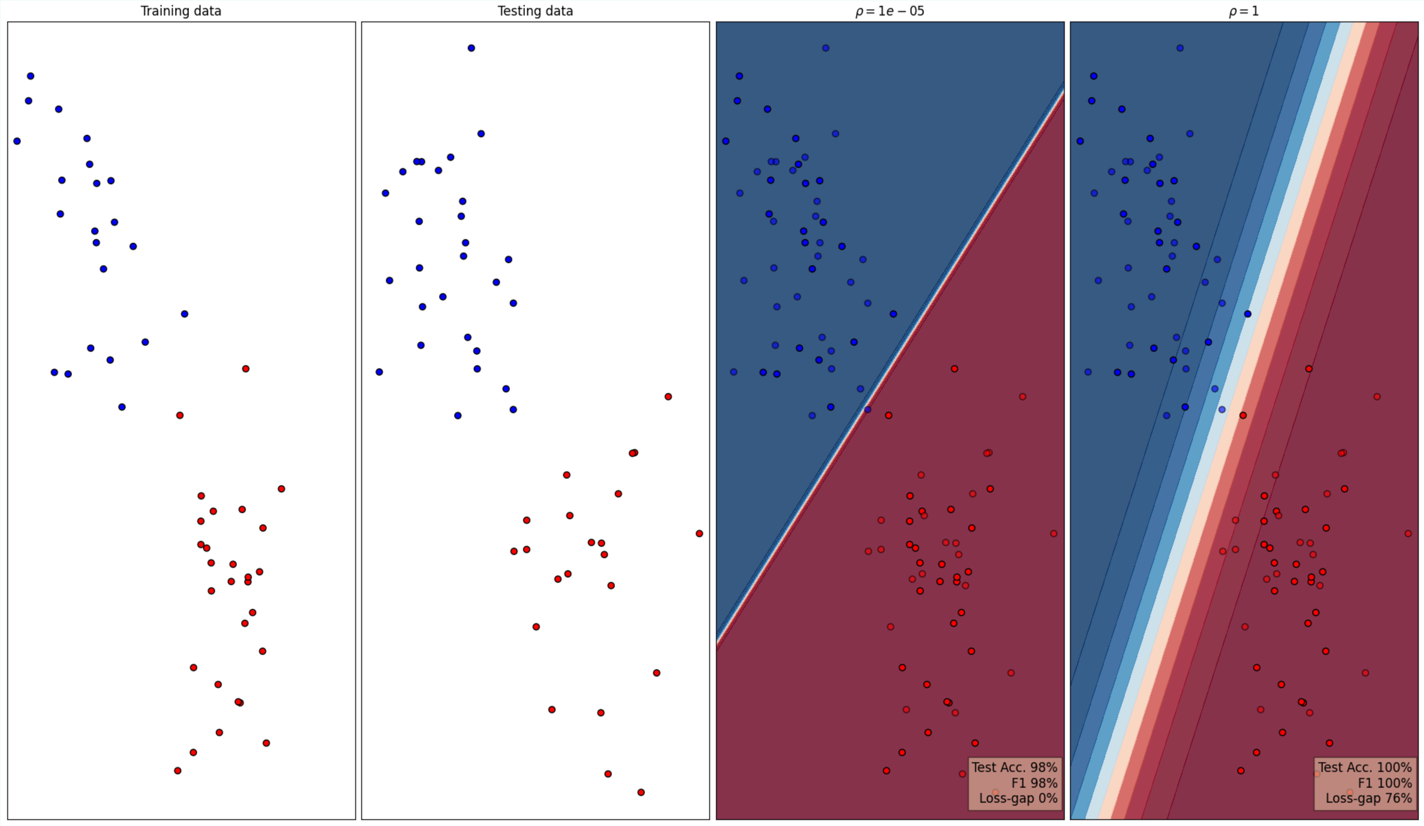
We know where the $\sup_{\zeta}\{\ell(\theta^T \zeta) - \lambda\|\zeta - \xi\|_2\}$ is in this case, so we can test numerically.

Toy example: linear classification #1

Simple balanced separable dataset,

for $\rho = 10^{-5}$ vs 1:

- gained accuracy
- lower test loss (−78%)
- wider margin.

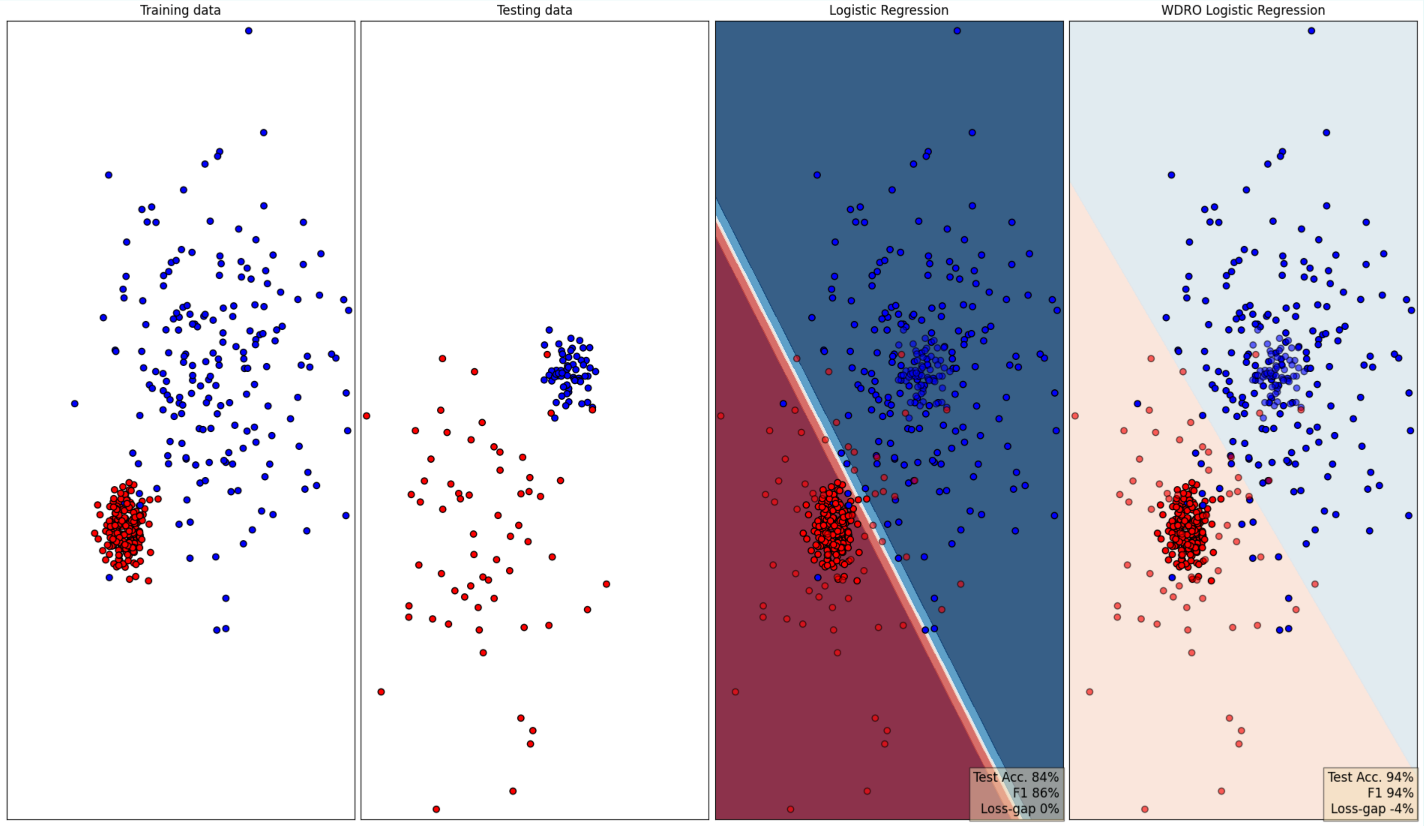


Toy example: linear classification #1

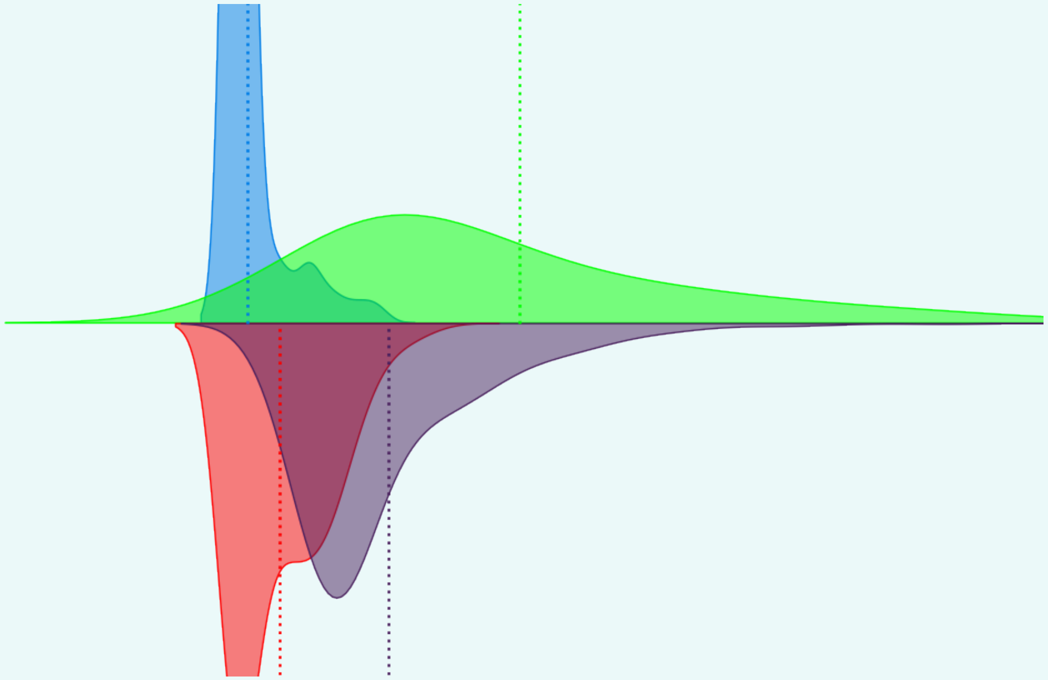
<u>Swap variances</u>

Hence distribution **shift** !

⇒ ~ 10% accuracy gain w.r.t. non-robust (ERM)



Toy example: linear classification #2



Train 1000 models on random data:

- ERM training losses
- ERM test losses
- DRO training losses
- DRO test losses

x-axis: average loss on the dataset, y-axis: frequency

General case

$$\sup_{\zeta}\{L_{\theta}(\zeta) - \lambda\ \mathrm{c}(\boldsymbol{\xi}, \zeta)\}$$

When the supremum:

- does not admit any closed form 🖋️
- and is intractable numerically 📈

💡 Turn to smoothing the $\sup$

Dual of regularized WDRO problem – easier to solve for:

$$\min_{\theta, \lambda \geq 0} \lambda \rho + \frac{1}{N} \sum_{\xi_i \sim \hat{\mathbb{P}}^N} \varepsilon \log \mathbb{E}_{\zeta \sim \mathcal{N}(\xi_i, \sigma^2)} \left[e^{\frac{L_\theta(\zeta) - \lambda\, c(\zeta, \xi_i)}{\varepsilon}} \right]$$

💡 (Azizian et al., 2023)

Close to other ideas:

- (Blanchet & Kang, 2020) : Smooth explicitly with unlabeled data points
- (Wang et al., 2023) : Sinkhorn regularization of the Wasserstein distance

E.g.: Smoothing the linear classification

Theorem 2: [Vincent et al. 2025]

With $c(\boldsymbol{\xi}, \zeta) = \|\zeta - \boldsymbol{\xi}\|_2^2$ *and* $K := \frac{1}{\varepsilon} \in \mathbb{N}$ *the regularized WDRO can be written:*

$$\min_{\theta, \lambda \geq 0} \frac{1}{K} \left\{ \lambda \boldsymbol{\rho} - \frac{1}{2} \log\left(1 + \lambda^d \sigma^{2d}\right) + \frac{1}{N} \sum_{\boldsymbol{\xi}_i \sim \hat{\mathbb{P}}^N} \log\left[\sum_{k=1}^{K} \binom{K}{k} e^{Q_{\theta,\lambda}^{(k)}(\boldsymbol{\xi}_i)} \right] \right\}$$

for a quadratic form: $Q_{\theta,\lambda}^{(k)}(\boldsymbol{\xi}) = k\ \theta^T \boldsymbol{\xi} - k^2 \frac{\sigma^2}{2(\lambda\sigma^2+1)} \|\theta\|_2^2$

$\rightarrow$ Not simple ! **Tractable**, but less explicit than vanilla WDRO.

In general: smooth your own loss function

$$\varepsilon \log \mathbb{E}_{\zeta\sim\mathcal{N}(\boldsymbol{\xi},\sigma^2)}\left[e^{\frac{L_\theta(\zeta)-\lambda\ \mathrm{c}(\zeta,\boldsymbol{\xi})}{\varepsilon}}\right]$$

- Smooth $\Rightarrow$ you can optimize it, as you would L_θ
- Pick the hyperparameter ε depending on σ^2

Is it better?

	Pros	Cons
$\sup_{\zeta} L_\theta(\zeta) - \lambda\ c(\zeta, \boldsymbol{\xi})$	- No hyperparameter	- No closed form (in general)
$\varepsilon \log \mathbb{E}_{\zeta \sim \mathcal{N}(\boldsymbol{\xi}, \sigma^2)} \left[e^{\frac{L_\theta(\zeta) - \lambda\ c(\zeta, \boldsymbol{\xi})}{\varepsilon}} \right]$	- $\mathbb{E}_{\zeta \sim \mathcal{N}(\cdot, \sigma^2)}$ tractable by sampling (e.g. MC)	- Pick ε and σ^2

Is it better?

$$\sup_{\zeta} L_\theta(\zeta) - \lambda \; c(\zeta, \boldsymbol{\xi})$$

$$\varepsilon \log \mathbb{E}_{\zeta \sim \mathcal{N}(\boldsymbol{\xi}, \sigma^2)} \left[e^{\frac{L_\theta(\zeta) - \lambda \; c(\zeta, \boldsymbol{\xi})}{\varepsilon}} \right]$$

Pros	Cons
• No hyperparameter	• No closed form (in general)
• $\mathbb{E}_{\zeta \sim \mathcal{N}(\cdot, \sigma^2)}$ tractable by sampling (e.g. MC)	• Pick ε and σ^2

Our work: the SkWDRO toolbox

Plan

A library for your own case: `SkWDRO`	Experiments (and real world data)	Insights on what is behind the scene

Numerics

- Tackle with the smoothed $\sup$
- Relevant default hyperparameters proposed

SkWDRO [Vincent et al. 2024]

- A python package:

```
$ pip install skwdro
$ uv pip install skwdro
$ mamba install flvincen:skwdro
```

- `PyTorch` interface to solve smoothed problem for given ρ.

SkWDRO - PyTorch interface

(Try defaults)

```
def dualize_primal_loss(

  loss_: nn.Module,
  transform_: Optional[nn.Module],
  rho: pt.Tensor,
  xi_batchinit: pt.Tensor,
  xi_labels_batchinit: Optional[pt.Tensor],


) -> DualLoss:
```

This `DualLoss` torch `Module` can be optimized in the same way as the original loss.

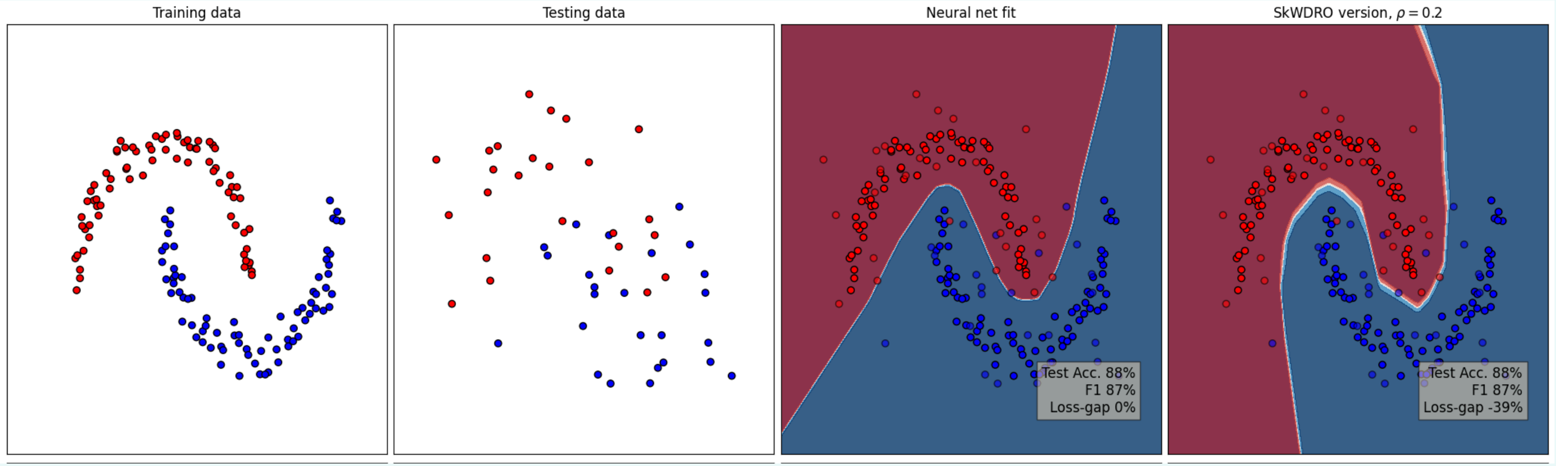
SkWDRO - PyTorch interface

```
def dualize_primal_loss(

  loss_: nn.Module,
  transform_: Optional[nn.Module],
  rho: pt.Tensor,
  xi_batchinit: pt.Tensor,
  xi_labels_batchinit: Optional[pt.Tensor],
  post_sample: bool=True,
  cost_spec: Optional[str]=None,
  n_samples: int=10,
  seed: int=42,

  *,
  epsilon: Optional[float]=None,
  sigma: Optional[float]=None,
  l2reg: Optional[float]=None,
  adapt: str="prodigy",
  imp_samp: bool=True

) -> DualLoss:
```

Lot of control
on the
hyperparameters

Toy example: classification #2

Non-convex setting: moons+noise



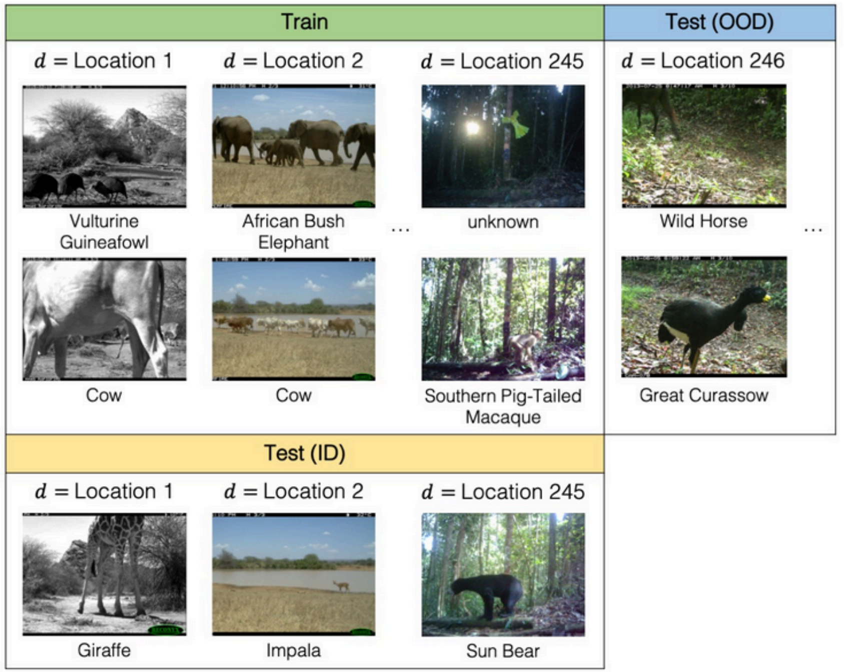
Train $\mathcal{N}\left(0, \begin{pmatrix} 2.5.10^{-3} & 0 \\ 0 & 10^{-2} \end{pmatrix}\right)$

$\rightarrow$

Test $\mathcal{N}\left(0, \begin{pmatrix} 10^{-2} & 0 \\ 0 & 2.5.10^{-3} \end{pmatrix}\right)$

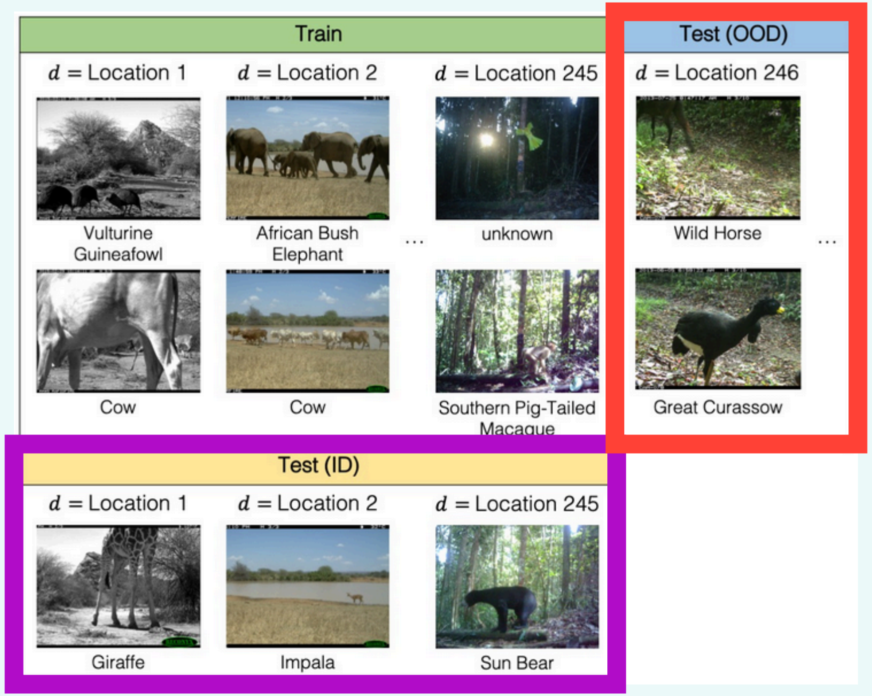
Test loss gets 39% down, as the model gets conservative.

Back to the Wilds 🦁.



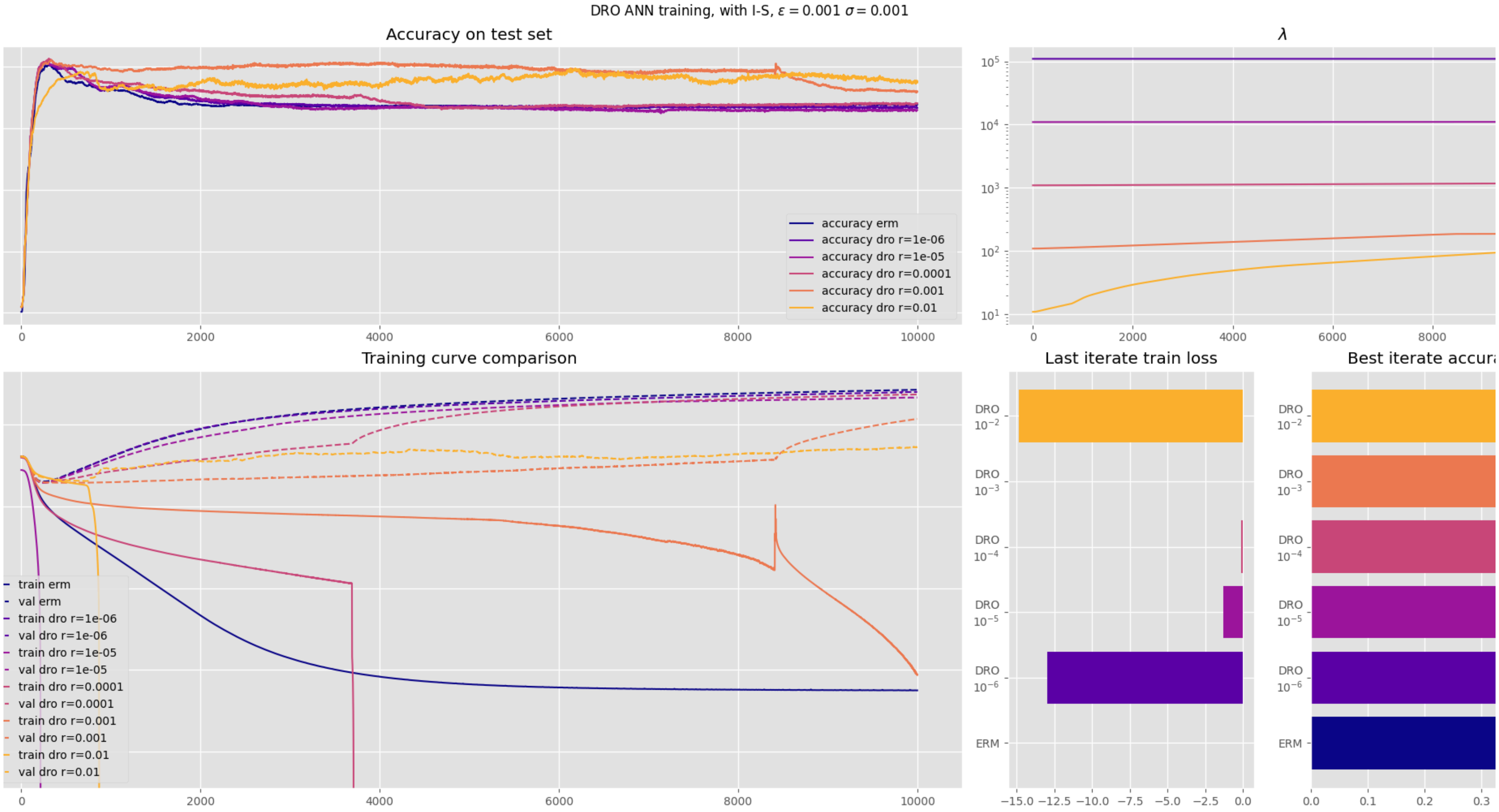
Back to the Wilds 🦁.

Track the validation loss during training, with test accuracy



Setup:

- Data pre-treated with a resnet ($\Rightarrow d = 157$)
- 1 hidden layer neural-net ($d_h = 64$)
- |Train|=20000 |Val|=|Test|=5000


DRO ANN training, with I-S, ε = 0.001 σ = 0.001
Accuracy on test set
accuracy erm
accuracy dro r=1e-06
accuracy dro r=1e-05
accuracy dro r=0.0001
accuracy dro r=0.001
accuracy dro r=0.01
λ
Training curve comparison
train erm
val erm
train dro r=1e-06
val dro r=1e-06
train dro r=1e-05
val dro r=1e-05
train dro r=0.0001
val dro r=0.0001
train dro r=0.001
val dro r=0.001
train dro r=0.01
val dro r=0.01
Last iterate train loss
Best iterate accur
DRO 10^-2
DRO 10^-3
DRO 10^-4
DRO 10^-5
DRO 10^-6
ERM

Behind the scenes

- Clever sampling for the inner `logsumexp`
- Heuristics to set ε, Σ
- Numerically stable backward pass on the loss
- Heuristic to set starting λ
- User-friendly interfaces
- All-in-one API, no troubles to define the problem
- Test suite & documentation

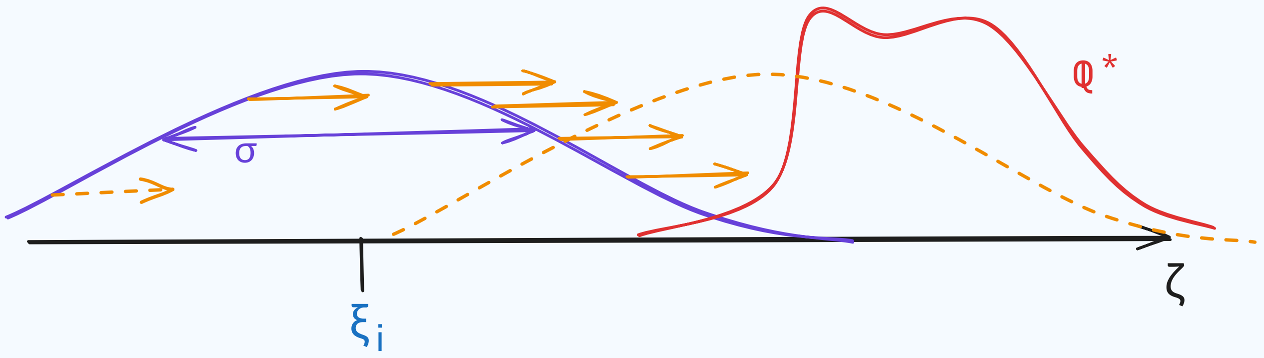
Behind the scenes

- **Clever sampling for the inner `logsumexp`**
- Heuristics to set ε, Σ
- Numerically stable backward pass on the loss
- Heuristic to set starting λ
- User-friendly interfaces
- All-in-one API, no troubles to define the problem
- Test suite & documentation

Clever sampling strategy

$$\varepsilon \log\left(\mathbb{E}_{\zeta|\boldsymbol{\xi}}\left[e^{\frac{1}{\varepsilon}(L_\theta - \lambda\ c(\boldsymbol{\xi},\zeta))}\right]\right)$$

Importance sampling: push samples where they have more weights in $\mathbb{E}_{\zeta\sim\mathcal{N}(\boldsymbol{\xi},\sigma^2)}$.



Conclusion

Skwdro

A library that allows you to robustify your decision model written in `PyTorch`.

Take-away message

Big non-convex models are amenable to WDRO

Perspectives:

- Add constraints
- Scale up

Try it out!

[1] P. Mohajerin Esfahani and D. Kuhn, “Data-driven distributionally robust optimization using the Wasserstein metric: performance guarantees and tractable reformulations,” *Mathematical Programming*, vol. 171, 2018.

[2] D. Kuhn, P. M. Esfahani, V. A. Nguyen, and S. Shafieezadeh-Abadeh, “Wasserstein distributionally robust optimization: Theory and applications in machine learning,” *Operations research & management science in the age of analytics*. Informs, 2019.

[3] J. Wang, R. Gao, and Y. Xie, “Sinkhorn Distributionally Robust Optimization.” 2023.

[4] W. Azizian, F. Iutzeler, and J. Malick, “Regularization for Wasserstein distributionally robust optimization,” *ESAIM: COCV*, 2023, doi: 10.1051/cocv/2023019.

[5] S. Shafieezadeh Abadeh, P. M. Mohajerin Esfahani, and D. Kuhn, “Distributionally Robust Logistic Regression,” in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., Curran Associates, Inc., 2015. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2015/file/cc1aa436277138f61cda703991069eaf-Paper.pdf

[6] R. Gao and A. Kleywegt, "Distributionally Robust Stochastic Optimization with Wasserstein Distance," *Mathematics of Operations Research*, vol. 48, no. 2, pp. 603–655, 2023, doi: 10.1287/moor.2022.1275.

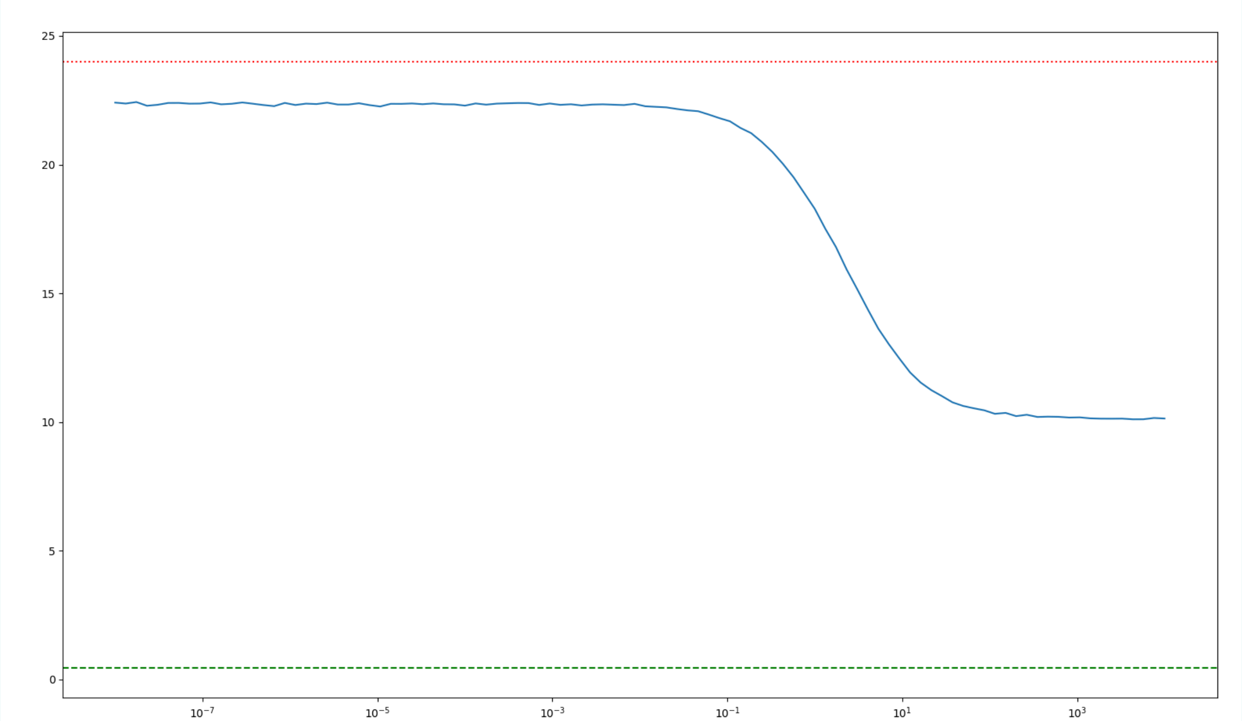
[7] J. Blanchet and Y. Kang, "Semi-supervised Learning Based on Distributionally Robust Optimization," Wiley, 2020, pp. 1–33. doi: 10.1002/9781119721871.ch1.

[8] T. Le, "Unregularized limit of stochastic gradient method for Wasserstein distributionally robust optimization." [Online]. Available: https://arxiv.org/abs/2506.04948

Hyperparameters

Numerical difficulties:

- SkWDRO is above ERM
- But it does not reach true WDRO



Regularized loss dependency on ε (logscale)

The Wasserstein distance

Wasserstein distance is inspired from Optimal Transport:

$$W_c(\mathbb{A}, \mathbb{B}) := \inf_{\pi \mid \begin{cases} [\pi]_1 = \mathbb{A} \\ [\pi]_2 = \mathbb{B} \end{cases}} \{\mathbb{E}_\pi[c]\}.$$

One may regularize this with entropy:

$$W_{\varepsilon,c}(\mathbb{A}, \mathbb{B}) := \inf_{\pi \mid \begin{cases} [\pi]_1 = \mathbb{A} \\ [\pi]_2 = \mathbb{B} \end{cases}} \{\mathbb{E}_\pi[c] + \varepsilon \operatorname{KL}(\pi \,\|\, \pi_0)\}.$$

$\Rightarrow$ need to pick ε and π_0

Robustness

Maximize over “close” distribution.

$$\min_{\theta} \sup_{\mathbb{Q} \text{ "close to" } \hat{\mathbb{P}}^N} \mathbb{E}_{\zeta\sim\mathbb{Q}}[L_\theta(\zeta)]$$

Robustness

Maximize over “close” distribution.

$$\min_{\theta} \sup_{\mathbb{Q} \text{ "close to" } \hat{\mathbb{P}}^N} \mathbb{E}_{\zeta\sim\mathbb{Q}}[L_\theta(\zeta)]$$

Multiple possible notions of neighborhood:

- KL divergence
- χ^2-distance
- TV-distance
- ...

Focus on the Wasserstein distance:

$$W_c\left(\hat{\mathbb{P}}^N, \mathbb{Q}\right) := \inf_{\pi}\left\{\mathbb{E}_{(\xi,\zeta)\sim\pi}[c(\xi,\zeta)] \,\middle|\, [\pi]_1 = \hat{\mathbb{P}}^N, [\pi]_2 = \mathbb{Q}\right\}$$

- Today mostly $c(\zeta,\xi) = \|\zeta - \xi\|_2^2$ and $\|\zeta - \xi\|_2$

Convergence result:

Theorem (Le, 2025):

Gradients of (ε-Regularized WDRO) *converge with* $\varepsilon \to 0$ *and* $M \to \infty$ *to the Clarke differential of* (WDRO).

$$\varepsilon \frac{1}{N} \sum_{i=1}^{N} \log \frac{1}{M} \sum_{j=1}^{M} e^{\frac{L_\theta(\zeta_j) - \lambda\, c(\zeta_j, \xi_i)}{\varepsilon}} \underset{\substack{\varepsilon \to 0 \\ M \to \infty}}{\longrightarrow} \frac{1}{N} \sum_{i=1}^{N} \sup_{\zeta} \{ L_\theta(\zeta) - \lambda\, c(\zeta, \xi_i) \}$$

SkWDRO [Vincent et al. 2024]

CI	Test	TESTS PASSING
	Style	STYLE PASSING
	Doc	Workflow Doc
Doc	Readthedocs	READTHEDOCS
Checks	Code style	RUFF
	Types	MYPY CHECKED
	Build	BUILT BY RATTLE-BUILD
Install	Pip	PYTHON 2.7 | 3.5 | 3.6 | 3.7
	Conda	Anaconda.org 1.0.1
	Github	GITHUB
Cite		ARXIV 2410.21231

SkWDRO - Wasserstein Distributionaly Robust Optimization

Model robustification with thin interface

"You can make pigs fly", [Kolter&Madry, 2018]

PYTHON PYTORCH SCIKITLEARN LICENSE BSD 3-CLAUSE

`skwdro` is a Python package that offers **WDRO versions** for a large range of estimators, either by extending `scikit-learn` **estimator** or by providing a wrapper for `pytorch` **modules**.

skwdro: a library for Wasserstein distributionally robust machine learning

Florian Vincent, Waïss Azizian, Jérôme Malick
FIRSTNAME.NAME@UNIV-GRENOBLE-ALPES.FR
Univ. Grenoble Alpes, CNRS, Grenoble INP, LJK, F-38000 Grenoble, France

Franck Iutzeler FRANCK.IUTZELER@MATH.UNIV-TOULOUSE.FR
Université de Toulouse, CNRS, UPS, F-31062 Toulouse, France

Abstract

We present `skwdro`, a Python library for training robust machine learning models. The library is based on distributionally robust optimization using optimal transport distances. For ease of use, it features both `scikit-learn` compatible estimators for popular objectives, as well as a wrapper for `PyTorch` modules, enabling researchers and practitioners to use it in a wide range of models with minimal code changes. Its implementation relies on an entropic smoothing of the original robust objective in order to ensure maximal model flexibility. The library is available at `https://github.com/iutzeler/skwdro`.

Keywords: Distributionally robust optim., distribution shifts, entropic regularization

A library *and a preprint*

Our work: the SkWDRO toolbox

Numerics

- Tackle with the smoothed $\sup$
- Sound default hyperparameters proposed

Engineering

- Maintained library
- Documentation and examples
- **Friendly+idiomatic interfaces**