Learning with Differentiable Perturbed Optimizers

Quentin Berthet

Google AI
Brain Team

Optimization for ML - CIRM - 2020

Q. Berthet

M.Blondel

O.Teboul

M. Cuturi

J-P. Vert

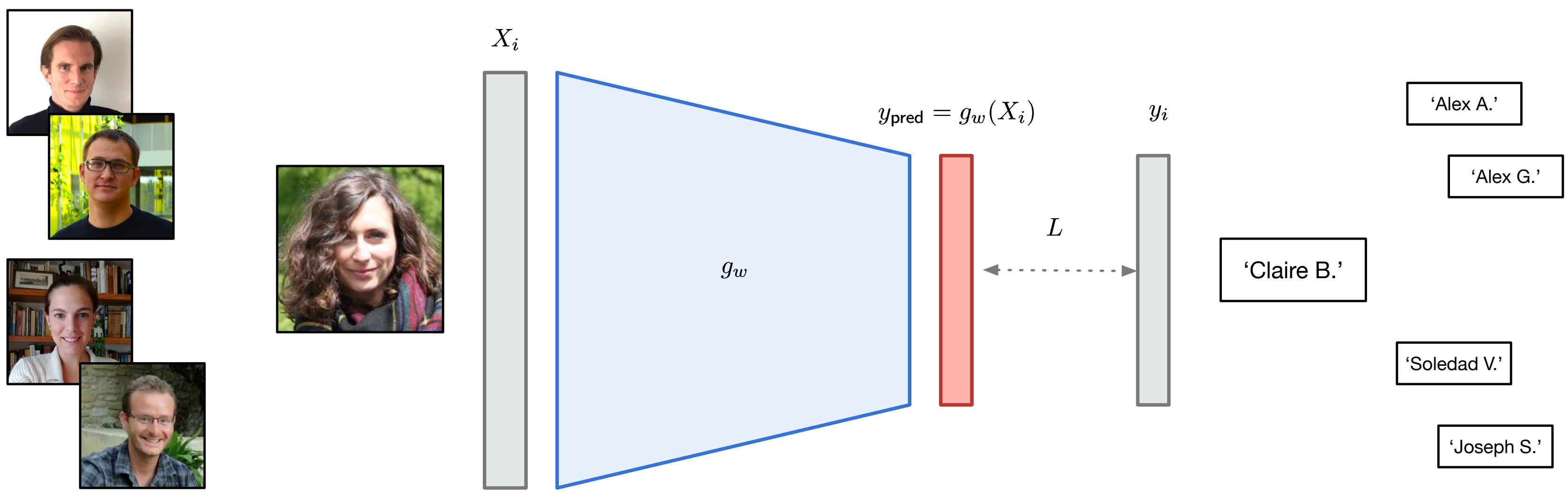
F.Bach

- **Learning with Differentiable Perturbed Optimizers**

  Preprint: **arXiv:2002.08676**

[A lot of] Machine learning these days

Supervised learning: couples of inputs/responses (X_i, y_i), a model g_w



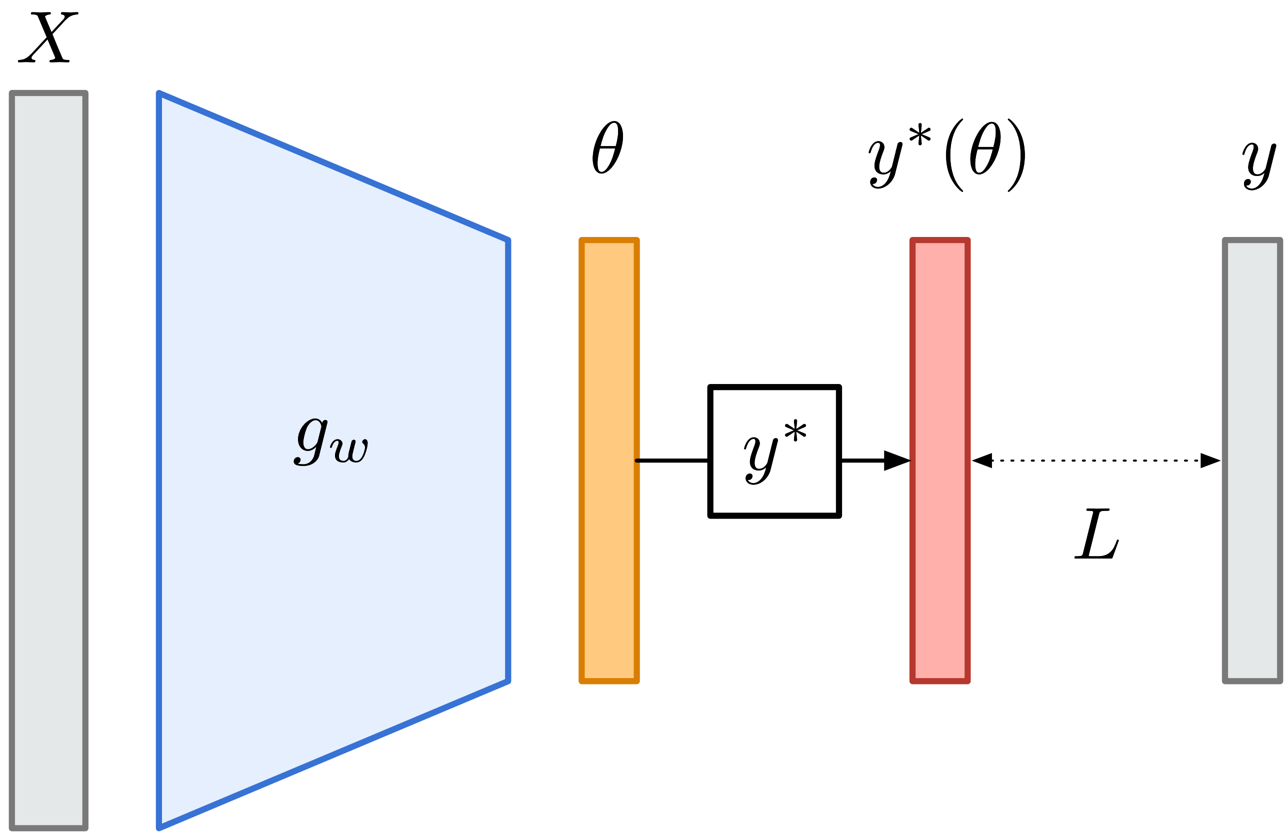
Goal: Optimize parameters $w \in \mathbf{R}^d$ of a function g_w such that $g_w(X_i) \approx y_i$

$$\min_w \sum_i L(g_w(X_i), y_i) \,.$$

Workhorse: first-order methods, based on $\nabla_w L(g_w(X_i), y_i)$, backpropagation

Problem: What if these models contain **nondifferentiable*** operations?

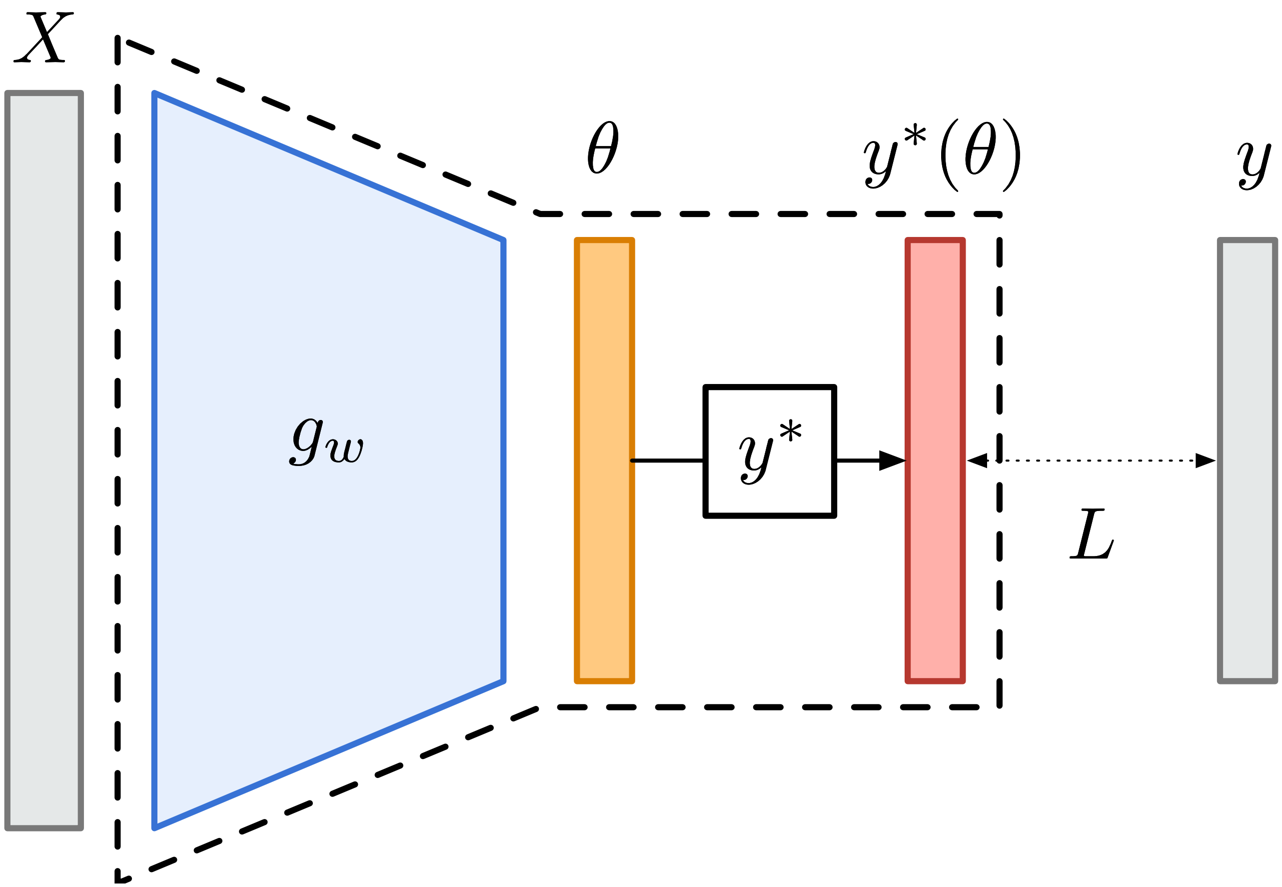
Discrete decisions in Machine learning



Examples: discrete operations (e.g. max, rankings), break autodifferentiation

- $\theta =$ scores for k products, $y^* =$ vector of ranks e.g. $[5, 2, 4, 3, 1]$
- $\theta =$ edge costs, $y^* =$ shortest path between two points
- $\theta =$ classification scores for each class, $y^* =$ one-hot vector

Discrete decisions in Machine learning



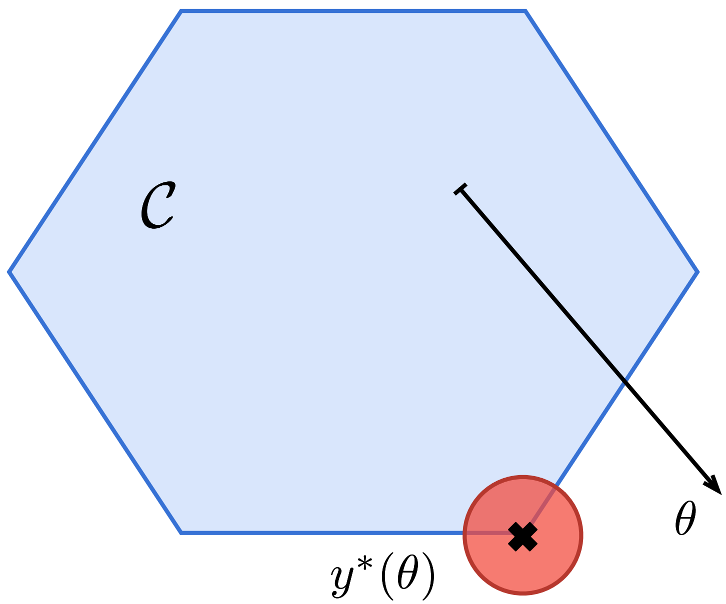
Examples: discrete operations (e.g. max, rankings), break autodifferentiation

- $\theta =$ scores for k products, $y^* =$ vector of ranks e.g. $[5, 2, 4, 3, 1]$
- $\theta =$ edge costs, $y^* =$ shortest path between two points
- $\theta =$ classification scores for each class, $y^* =$ one-hot vector

Perturbed maximizer

Discrete decisions: optimizers of linear program over $\mathcal{C}$, convex hull of $\mathcal{Y} \subseteq \mathbf{R}^d$

$$F(\theta) = \max_{y \in \mathcal{C}} \langle y, \theta \rangle\,, \quad \text{and} \quad y^*(\theta) = \operatorname*{argmax}_{y \in \mathcal{C}} \langle y, \theta \rangle = \nabla_\theta F(\theta)\,.$$

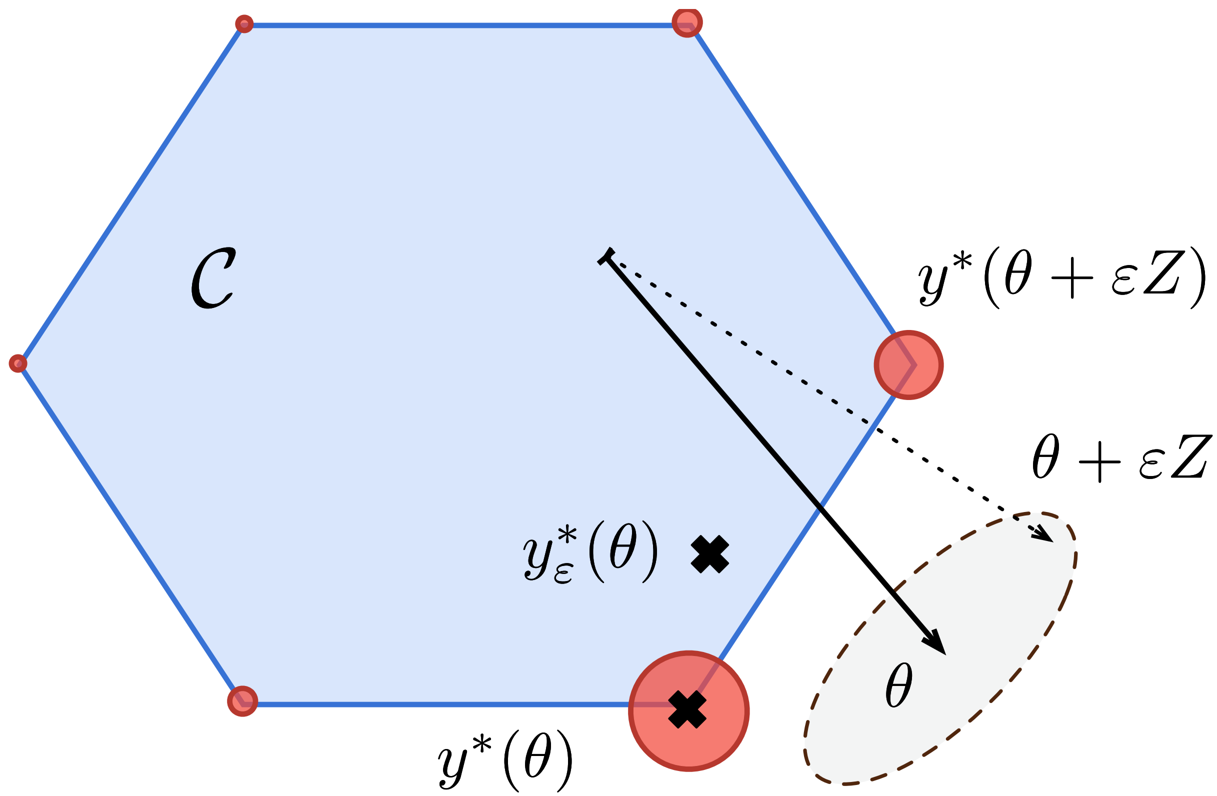


Perturbed maximizer: average of solutions for inputs with noise εZ

$$F_\varepsilon(\theta) = \mathbf{E}[\max_{y \in \mathcal{C}} \langle y, \theta \rangle]\,, \quad y^*_\varepsilon(\theta) = \mathbf{E}[y^*(\theta + \varepsilon Z)] = \mathbf{E}[\operatorname*{argmax}_{y \in \mathcal{C}} \langle y, \theta + \varepsilon Z \rangle] = \nabla_\theta F_\varepsilon(\theta)\,.$$

Perturbed maximizer

Discrete decisions: optimizers of linear program over $\mathcal{C}$, convex hull of $\mathcal{Y} \subseteq \mathbf{R}^d$



Perturbed maximizer: average of solutions for inputs with noise εZ

$$F_\varepsilon(\theta) = \mathbf{E}[\max_{y \in \mathcal{C}} \langle y, \theta \rangle]\,, \quad y^*_\varepsilon(\theta) = \mathbf{E}[y^*(\theta + \varepsilon Z)] = \mathbf{E}[\operatorname*{argmax}_{y \in \mathcal{C}} \langle y, \theta + \varepsilon Z \rangle] = \nabla_\theta F_\varepsilon(\theta)\,.$$

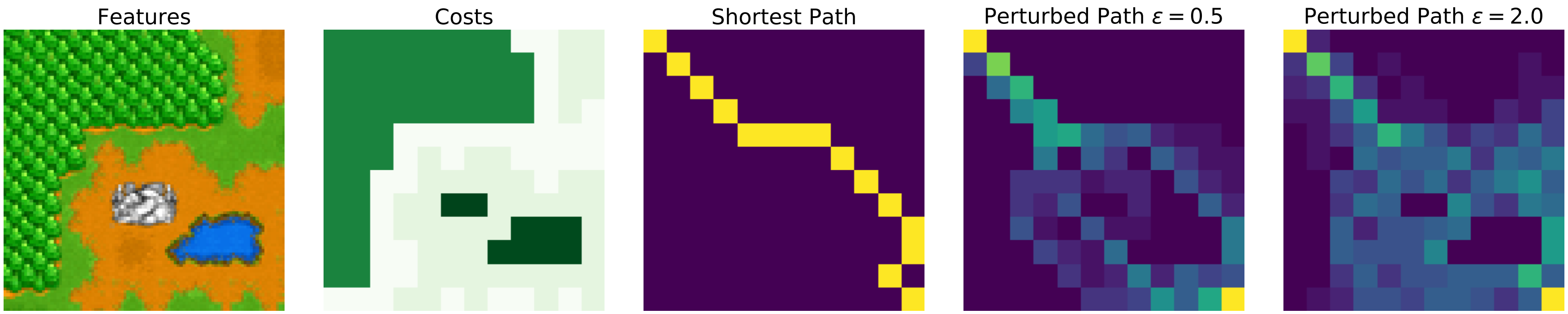
Perturbed model

Model of optimal decision under uncertainty Luce (1959), McFadden et al. (1973)

$$Y = \underset{y \in \mathcal{C}}{\mathrm{argmax}} \langle y, \theta + \varepsilon Z \rangle$$

Follows a **perturbed model** with $Y \sim p_\theta(y)$, expectation $y^*_\varepsilon(\theta) = \mathbf{E}_{p_\theta}[Y]$.

Perturb and map Papandreou & Yuille (2011), FT Perturbed L Kalai & Vempala (2003)



Example. Over the unit simplex $\mathcal{C} = \Delta^d$ with Gumbel noise Z, $F(\theta) = \max_i \theta_i$.

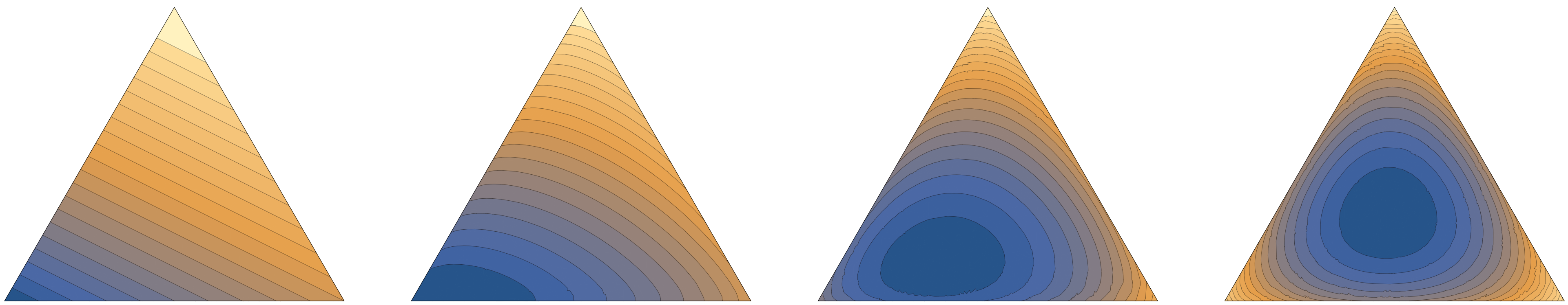
$$F_\varepsilon(\theta) = \varepsilon \log \sum_{i \in [d]} e^{\frac{\theta_i}{\varepsilon}}, \qquad p_\theta(e_i) \propto \exp(\langle \theta, e_i \rangle / \varepsilon), \qquad [y^*_\varepsilon(\theta)]_i = \frac{e^{\frac{\theta_i}{\varepsilon}}}{\sum e^{\frac{\theta_j}{\varepsilon}}}$$

Properties

Link with regularization: $\varepsilon\,\Omega = \big(F_\varepsilon\big)^*$ is a convex function with domain $\mathcal{C}$

$$y^*_\varepsilon(\theta) = \underset{y\in\mathcal{C}}{\operatorname{argmax}}\left\{\langle y,\theta\rangle - \varepsilon\Omega(y)\right\}.$$

Consequence of duality and $y^*_\varepsilon(\theta) = \nabla_\varepsilon F_\varepsilon(\theta)$. Generalization of entropy



$\varepsilon = 0$ | tiny ε | small ε | large ε

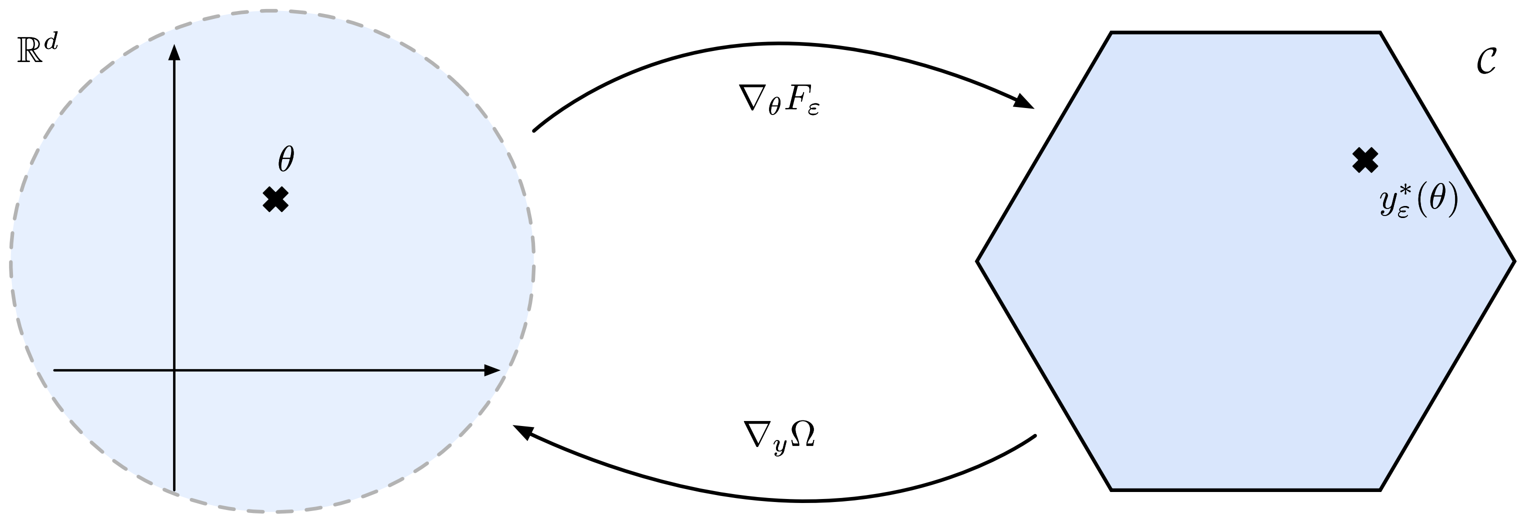
Extreme temperatures. When $\varepsilon \to 0$, $y^*_\varepsilon(\theta) \to y^*(\theta)$ for unique max.

When $\varepsilon \to \infty$, $y^*_\varepsilon(\theta) \to \operatorname{argmin}_y \Omega(y)$. Nonasymptotic results.

Properties

Mirror maps: For $\mathcal{C}$ with full interior, Z with smooth density μ, full support

F_ε strictly convex, gradient Lipschitz. Ω strongly convex, Legendre type.



Differentiability. Functions are smooth in the inputs. For $\mu(z) \propto \exp(-\nu(z))$

$$y^*_\varepsilon(\theta) = \nabla_\theta F_\varepsilon(\theta) = \mathbf{E}[y^*(\theta + \varepsilon Z)] = \mathbf{E}[F(\theta + \varepsilon Z)\nabla_z \nu(Z)/\varepsilon]\,,$$

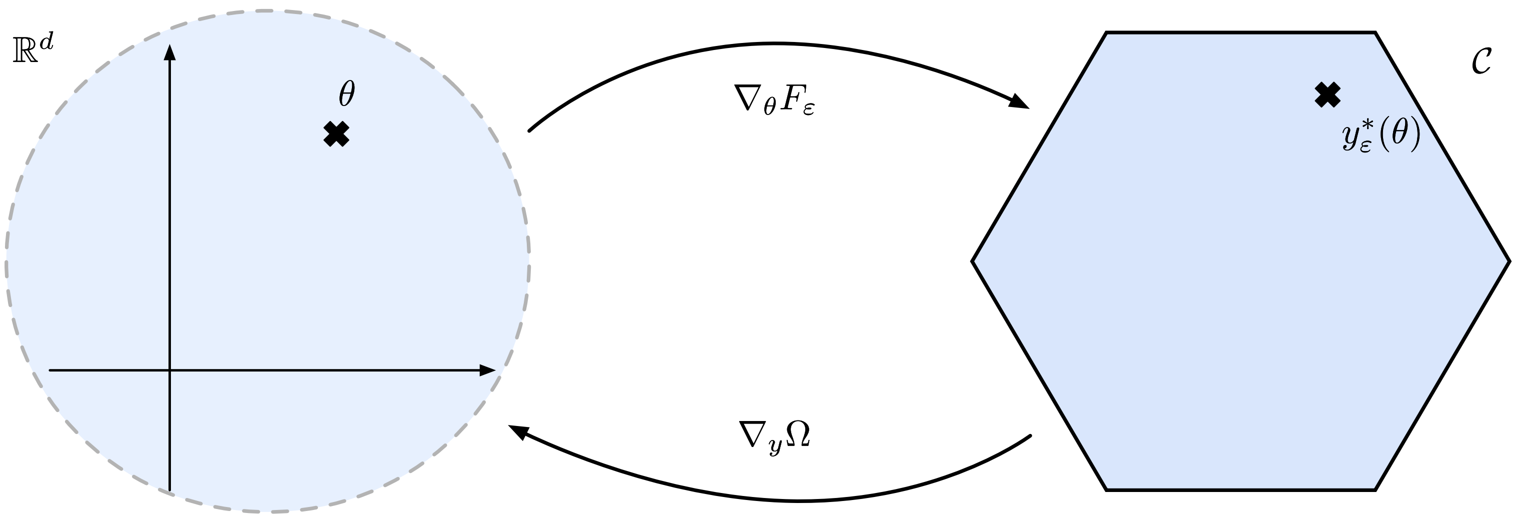
$$J_\theta\, y^*_\varepsilon(\theta) = \nabla^2 F_\varepsilon(\theta) = \mathbf{E}[y^*(\theta + \varepsilon Z)\nu(Z)^\top/\varepsilon]\,.$$

Perturbed maximizer y^*_ε never locally constant in θ. Abernethy et al. (2014)

Properties

Mirror maps: For $\mathcal{C}$ with full interior, Z with smooth density μ, full support

F_ε strictly convex, gradient Lipschitz. Ω strongly convex, Legendre type.



Differentiability. Functions are smooth in the inputs. For $\mu(z) \propto \exp(-\nu(z))$

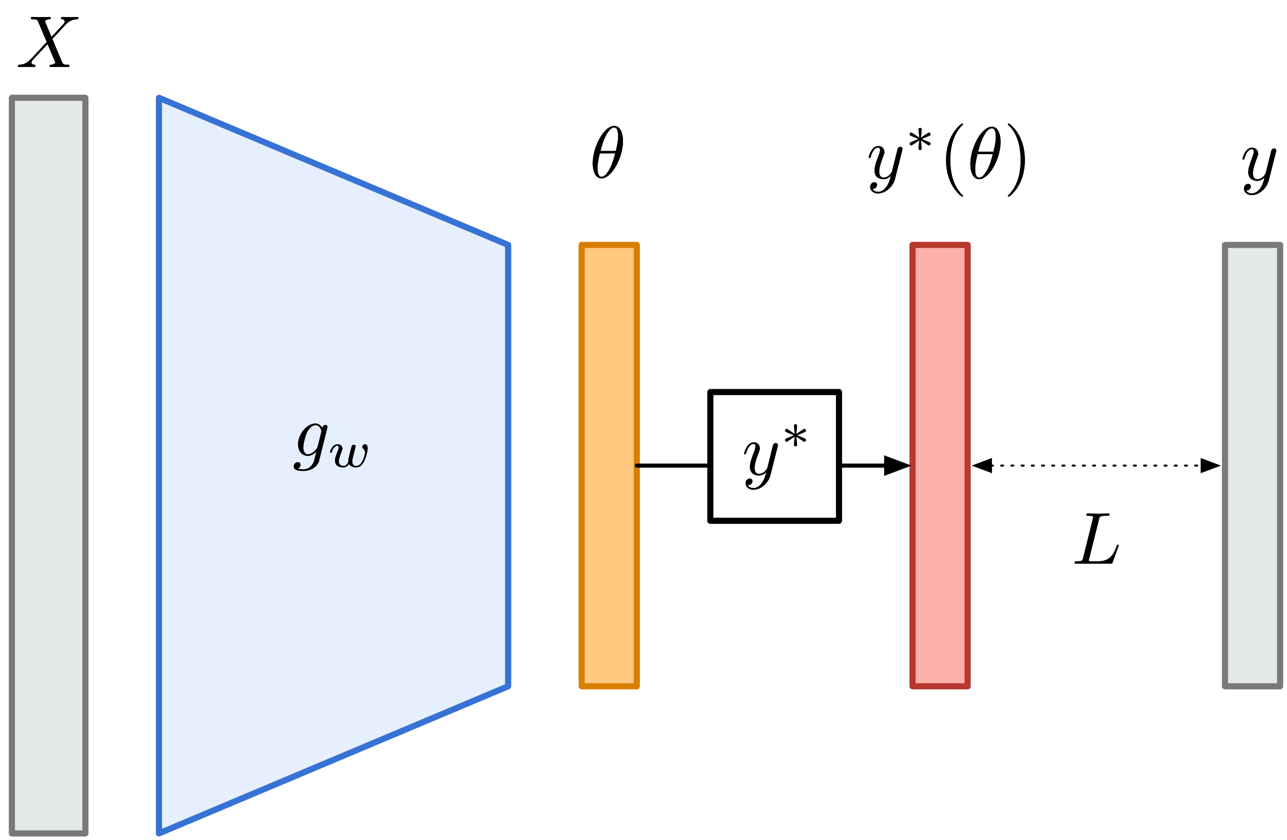
$$y^*_\varepsilon(\theta) = \nabla_\theta F_\varepsilon(\theta) = \mathbf{E}[y^*(\theta + \varepsilon Z)] = \mathbf{E}[F(\theta + \varepsilon Z)\nabla_z \nu(Z)/\varepsilon]\,,$$

$$J_\theta\, y^*_\varepsilon(\theta) = \nabla^2 F_\varepsilon(\theta) = \mathbf{E}[y^*(\theta + \varepsilon Z)\nu(Z)^\top/\varepsilon]\,.$$

Perturbed maximizer y^*_ε never locally constant in θ. Abernethy et al. (2014)

Learning with perturbed optimizers

Machine learning pipeline: variable X, discrete label y, model outputs $\theta = g_w(X)$

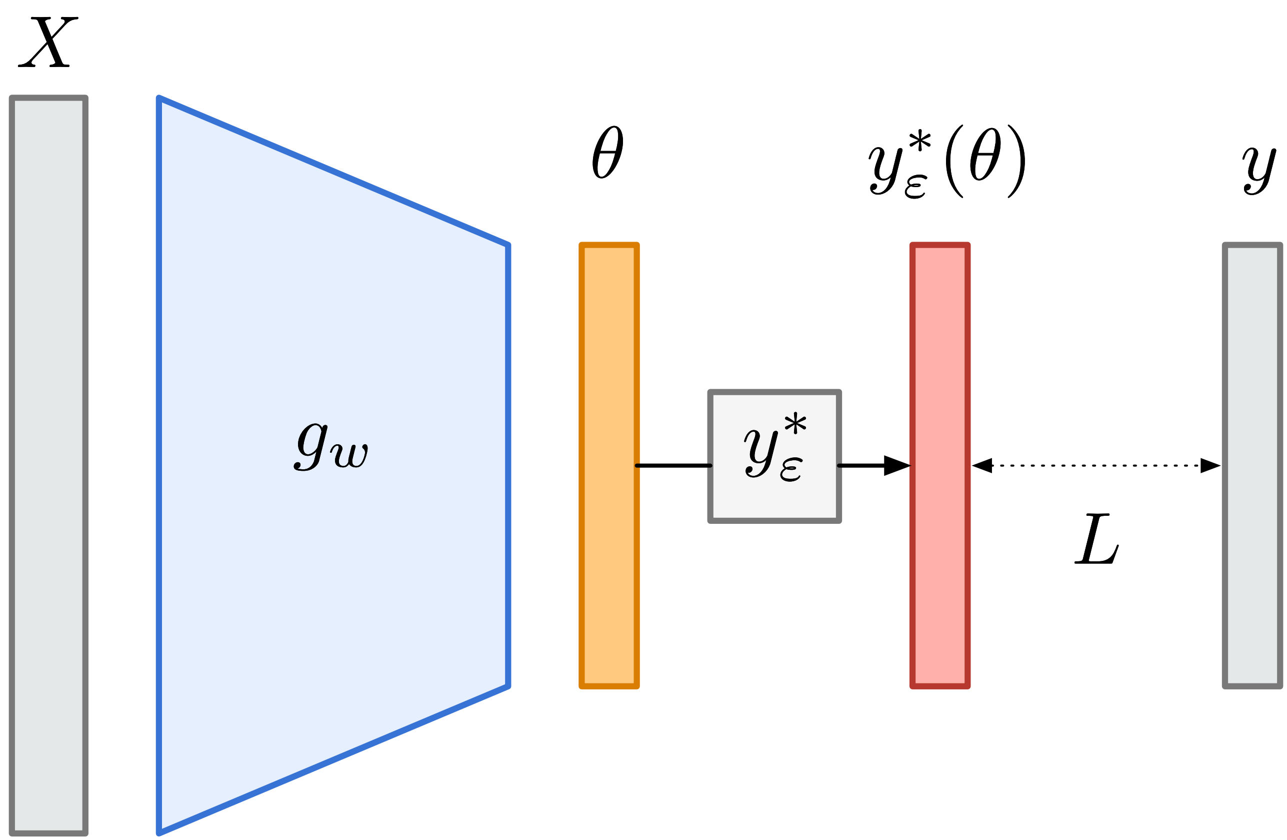


Labels are solutions of optimization problems (one-hots, ranks, shortest paths)

Small modification of the model: end-to-end differentiable

Learning with perturbed optimizers

Machine learning pipeline: variable X, discrete label y, model outputs $\theta = g_w(X)$

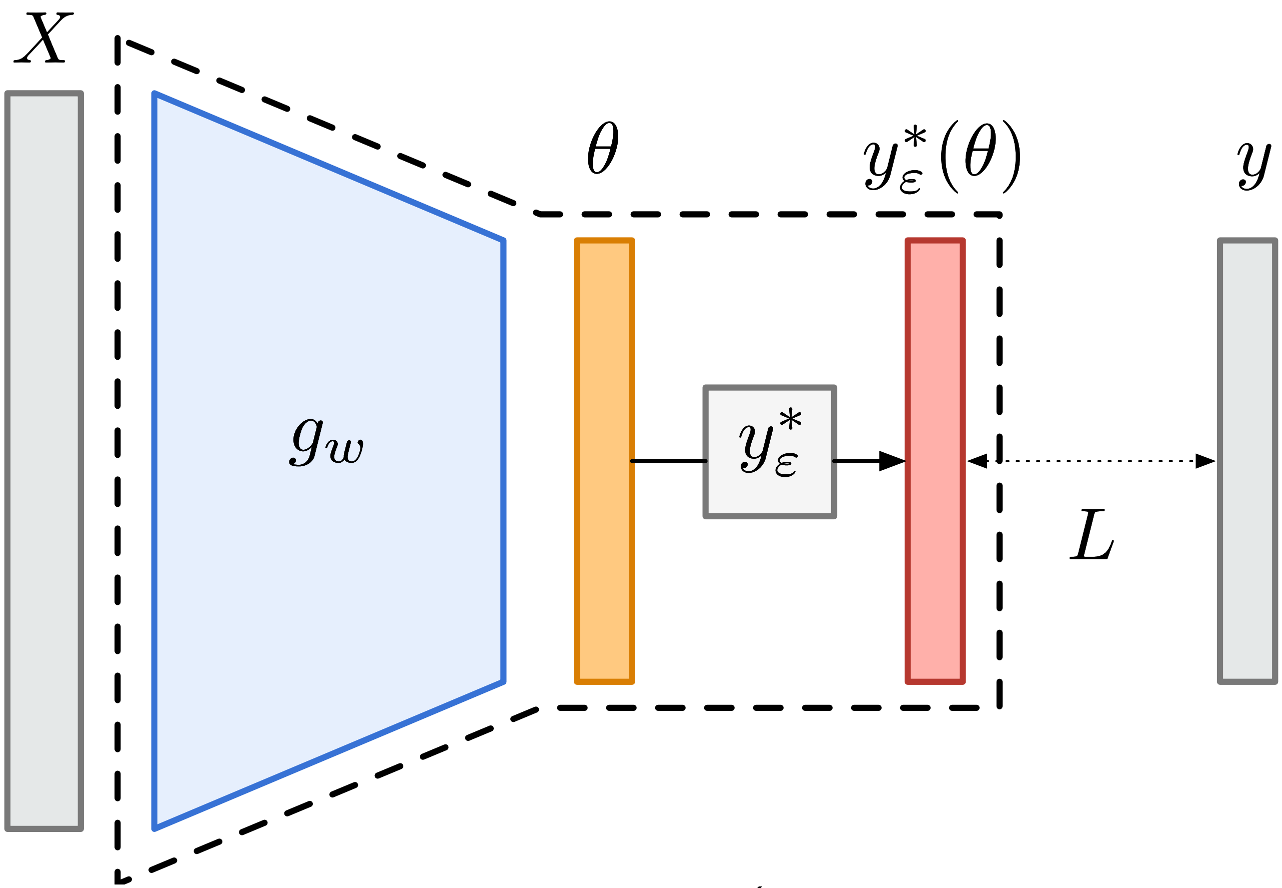


Labels are solutions of optimization problems (one-hots, ranks, shortest paths)

Small modification of the model: end-to-end differentiable

Learning with perturbed optimizers

Machine learning pipeline: variable X, discrete label y, model outputs $\theta = g_w(X)$



Labels are solutions of optimization problems (one-hots, ranks, shortest paths)

Small modification of the model: end-to-end differentiable

Why? and How?

Learning problems:

Features X_i, model output $\theta_w = g_w(X_i)$, prediction $y_{\text{pred}} = y^*_\varepsilon(\theta_w)$, loss L

$$F(w) = L\big(y^*_\varepsilon(\theta_w)\,, y_i\big)\,, \quad \text{gradients require } J_\theta\, y^*_\varepsilon(\theta_w)\,.$$

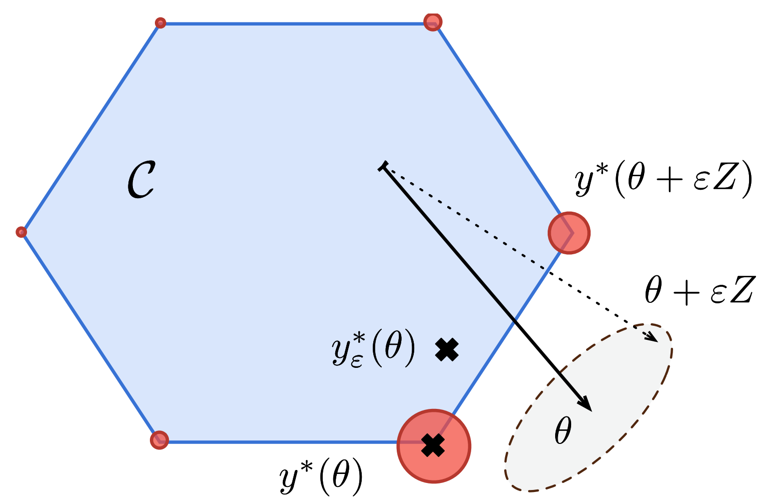
Monte Carlo estimates. Perturbed maximizer and derivatives as expectations.

For $\theta \in \mathbf{R}^d$, $Z^{(1)}, \ldots, Z^{(M)}$ i.i.d. copies

$$y^{(\ell)} = y^*(\theta + \varepsilon Z^{(\ell)})$$

Unbiased estimate of $y^*_\varepsilon(\theta)$ given by

$$\bar{y}_{\varepsilon,M}(\theta) = \frac{1}{M}\sum_{\ell=1}^{M} y^{(\ell)}\,.$$



Fenchel-Young losses

Natural loss to introduce, directly on θ, motivated by duality. Blondel et al. (2019)

$$L_\varepsilon(\theta; y) = F_\varepsilon(\theta) + \varepsilon\Omega(y) - \langle \theta, y \rangle \, .$$

Interesting **properties** in a learning framework:

- Convex in θ, minimized at θ s.t. $y^*_\varepsilon(\theta) = y$, with value 0.
- Equal to Bregman divergence $D_{\varepsilon\Omega}(y^*_\varepsilon(\theta) \,|\, y)$
- For random Y, $\mathbf{E}[L_\varepsilon(\theta; Y)] = L_\varepsilon(\theta; \mathbf{E}[Y]) + C$

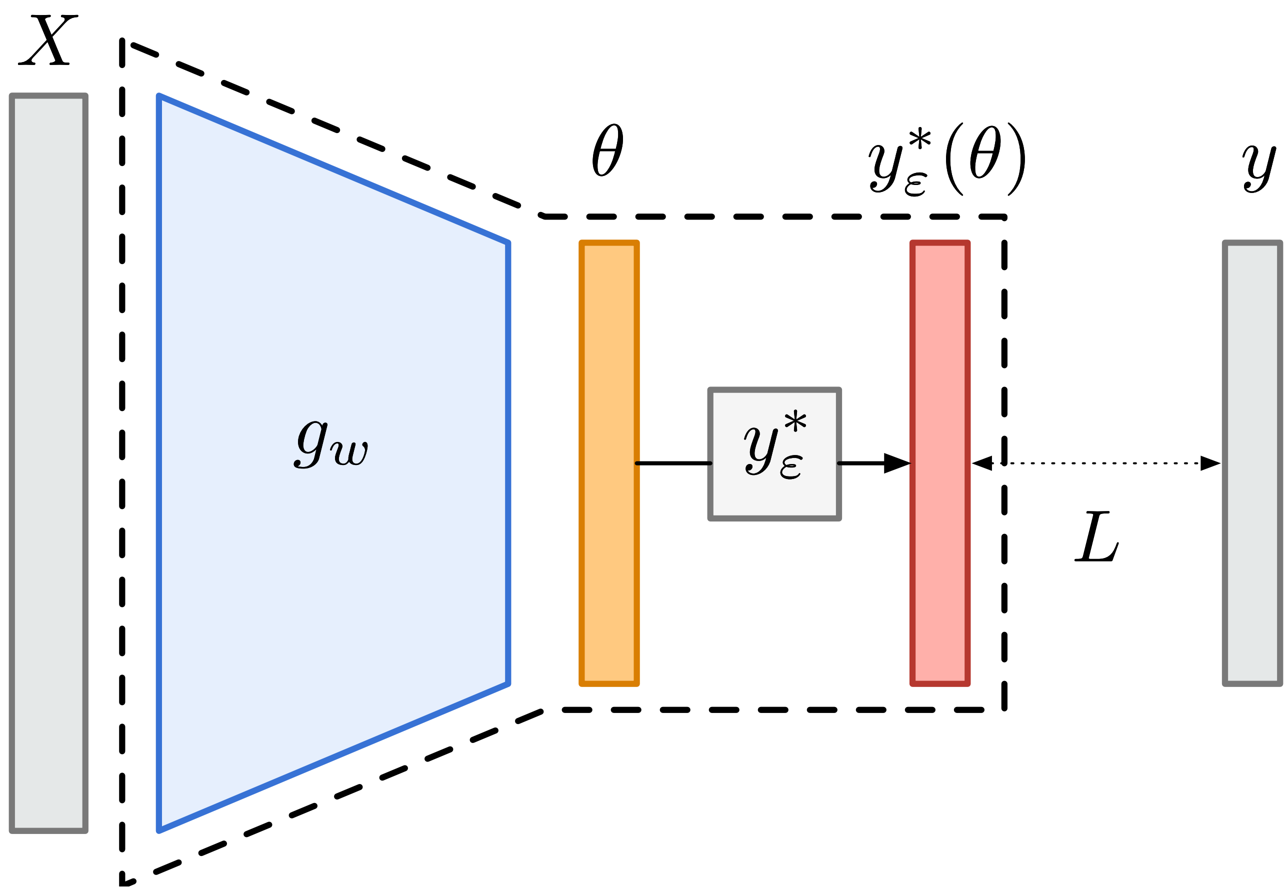
  e.g. for $Y = \mathrm{argmax}_{y \in \mathcal{C}} \langle \theta_0 + \varepsilon Z, y \rangle$

  $$\mathbf{E}[L_\varepsilon(\theta; Y)] = L_\varepsilon(\theta; y^*_\varepsilon(\theta_0)) + C \, ,$$

  population loss minimized at θ_0.
- Convenient gradients: $\nabla_\theta L_\varepsilon(\theta; y) = y^*_\varepsilon(\theta) - y$.

Learning with perturbations and F-Y losses

Within the same framework, possible to virtually bypass the optimization block

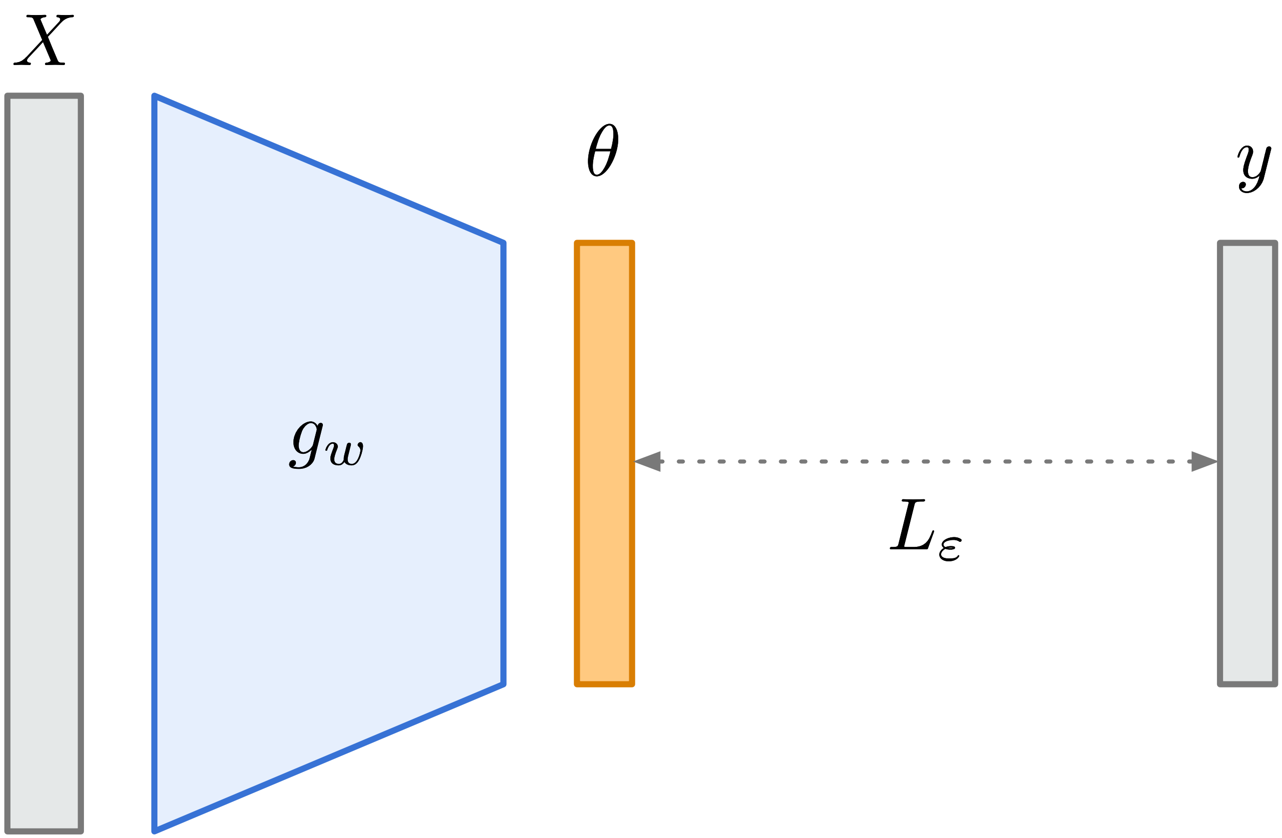


Easier to implement, no Jacobian of y^*_ε

Population loss minimized at ground truth for perturbed generative model.

Learning with perturbations and F-Y losses

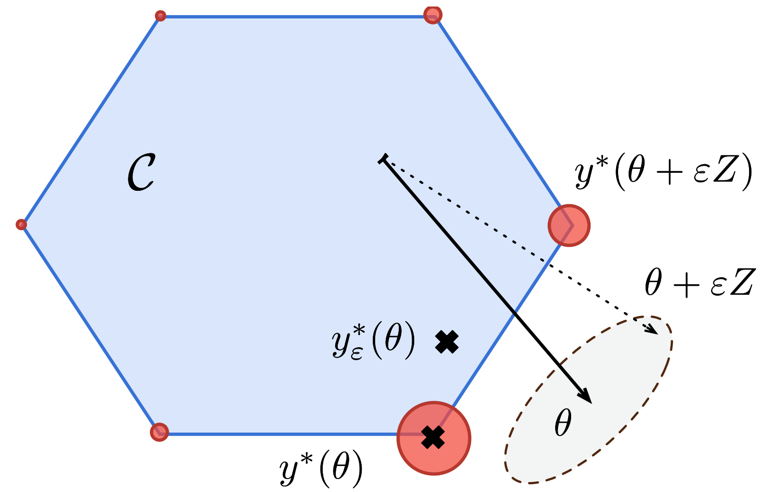
Within the same framework, possible to virtually bypass the optimization block



Easier to implement, no Jacobian of y^*_ε

Population loss minimized at ground truth for perturbed generative model.

Unsupervised learning - parameter estimation



Observation: $Y_1, \dots, Y_n$ i.i.d. copies of

$$Y_i = \underset{y \in \mathcal{C}}{\operatorname{argmax}} \langle \theta_0 + \varepsilon Z_i, y \rangle$$

Estimating unknown θ_0

Minimization of empirical loss - related to inference in Gibbs models

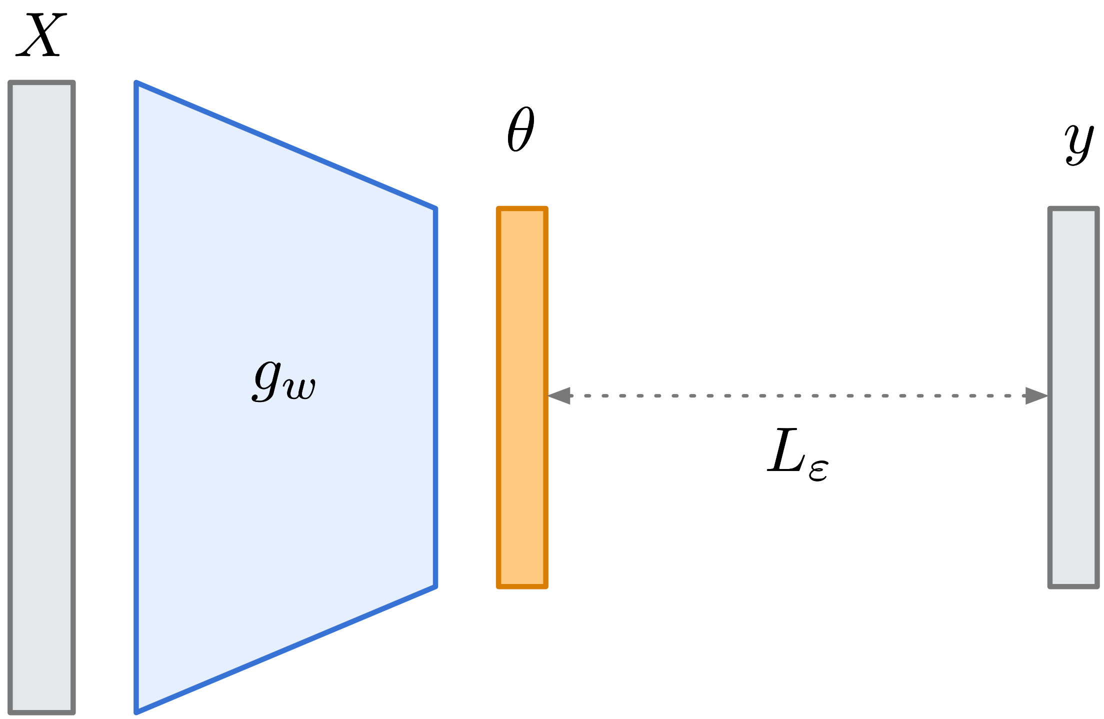
$$\bar{L}_{\varepsilon,n}(\theta) = \frac{1}{n} \sum_{i=1}^{n} L(\theta; Y_i)\,, \quad \text{stochastic grad. } \nabla_\theta L_\varepsilon(\theta, Y_i) = y^*_\varepsilon(\theta) - Y_i$$

Equal up to an additive constant to $L_\varepsilon(\theta; \bar{Y}_n)$, in expectation to $L_\varepsilon(\theta; y^*_\varepsilon(\theta_0))$

Asymptotic normality for minimizer $\hat{\theta}_n$ around θ_0

Supervised learning

Motivated by model where $y_i = \mathrm{argmax}_{y \in \mathcal{C}} \langle g_{w_0}(X_i) + \varepsilon Z_i, y \rangle$



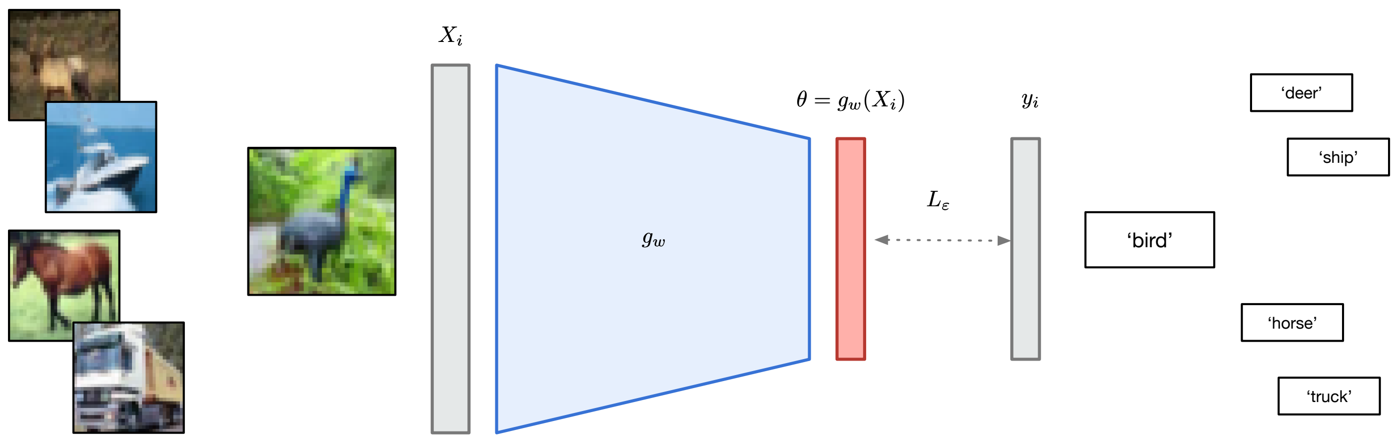
Stochastic gradients for empirical loss only require

$$\nabla_\theta L(\theta = g_w(X_i); y_i) = y^*_\varepsilon(g_w(X_i)) - y_i \, .$$

Simulated by a doubly stochastic scheme.

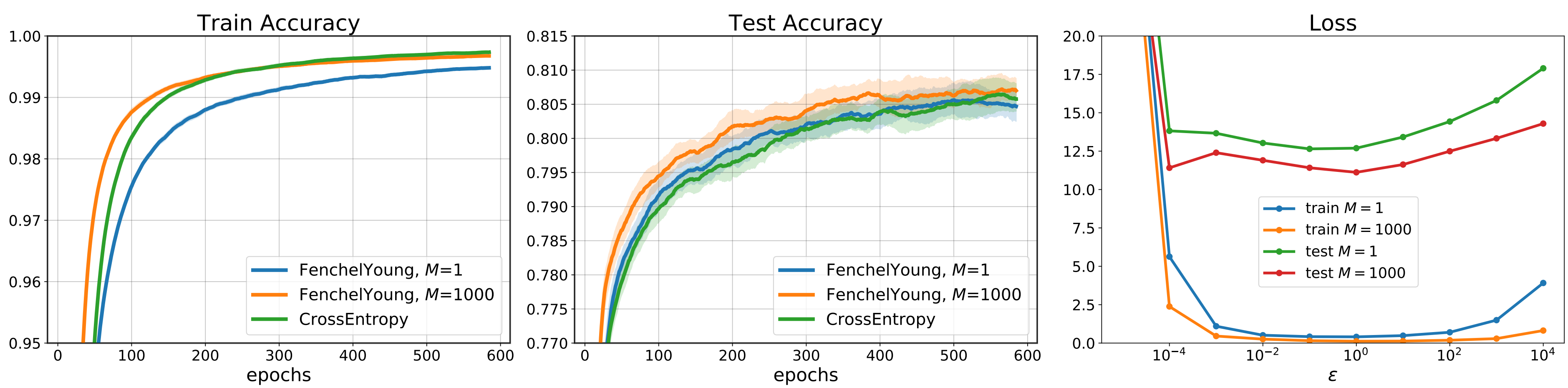
Experiments

Classification: CIFAR-10 dataset of images with 10 classes - Toy comparison



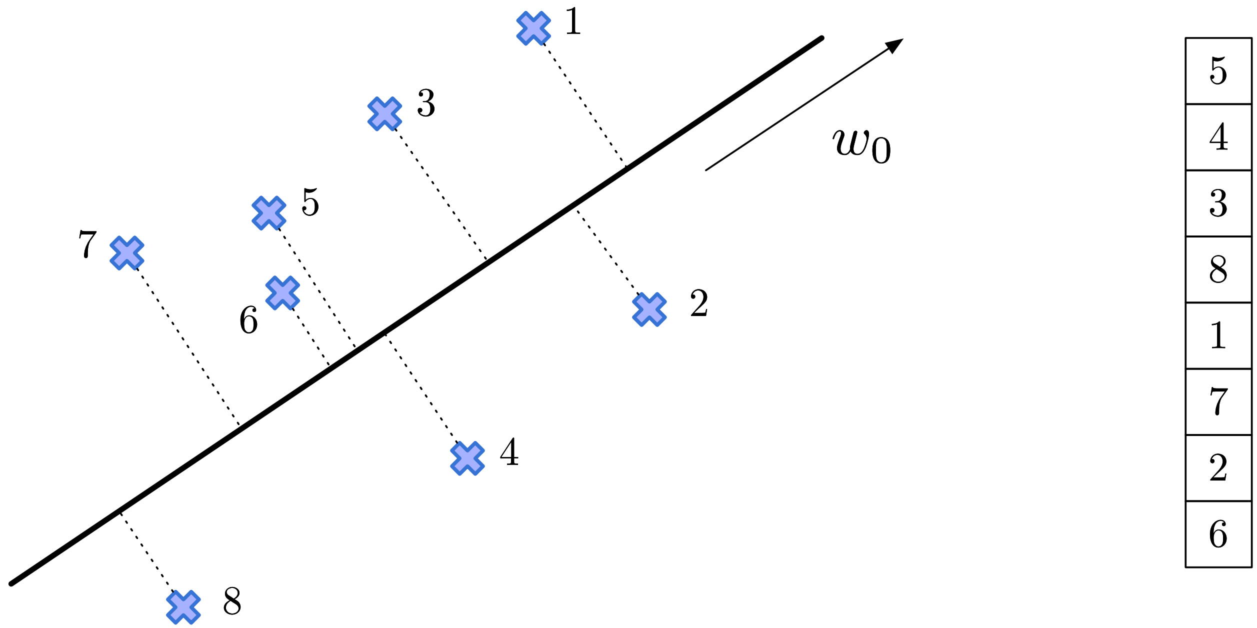
Architecture: vanilla-CNN made of 4 convolutional and 2 fully connected layers.

Training: 600 epochs with minibatches of size 32 - influence of M and ε



Experiments

Learning from rankings: Created dataset - ranked projection along unknown w_0.

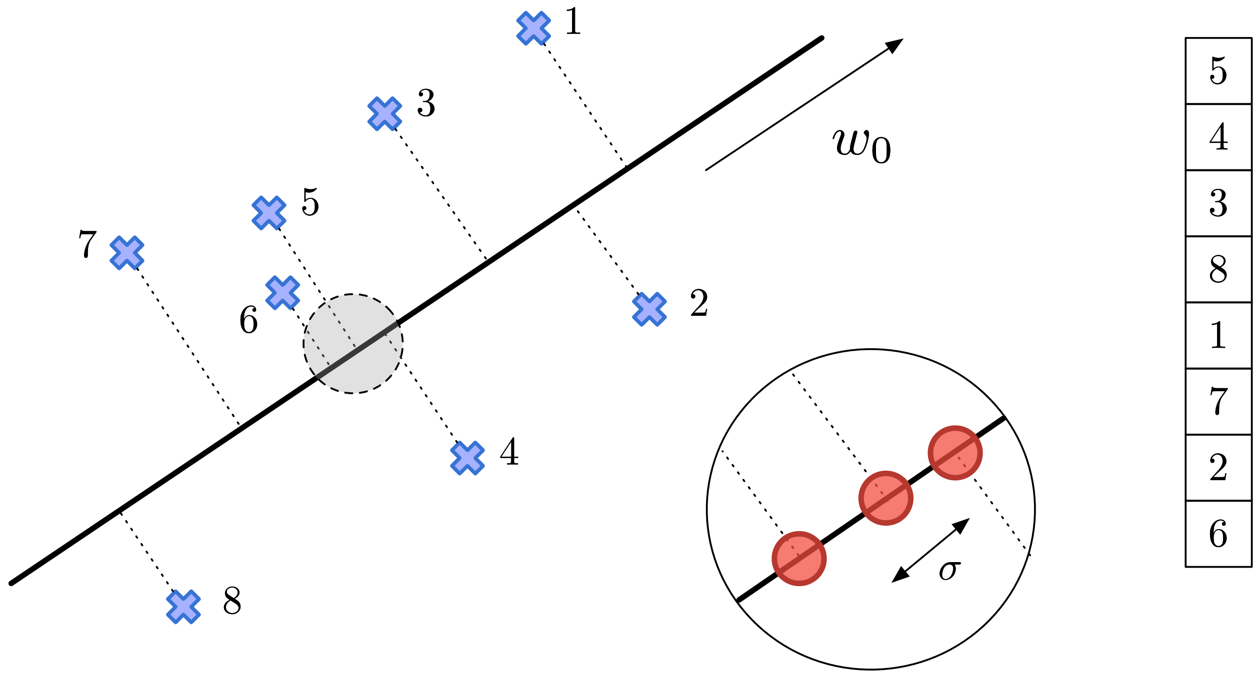


From data, predict ranks on future instances (simulated learning to rank).

Robustness to noise σ before ranking - uncertainty of user.

Experiments

Learning from rankings: Created dataset - ranked projection along unknown w_0.

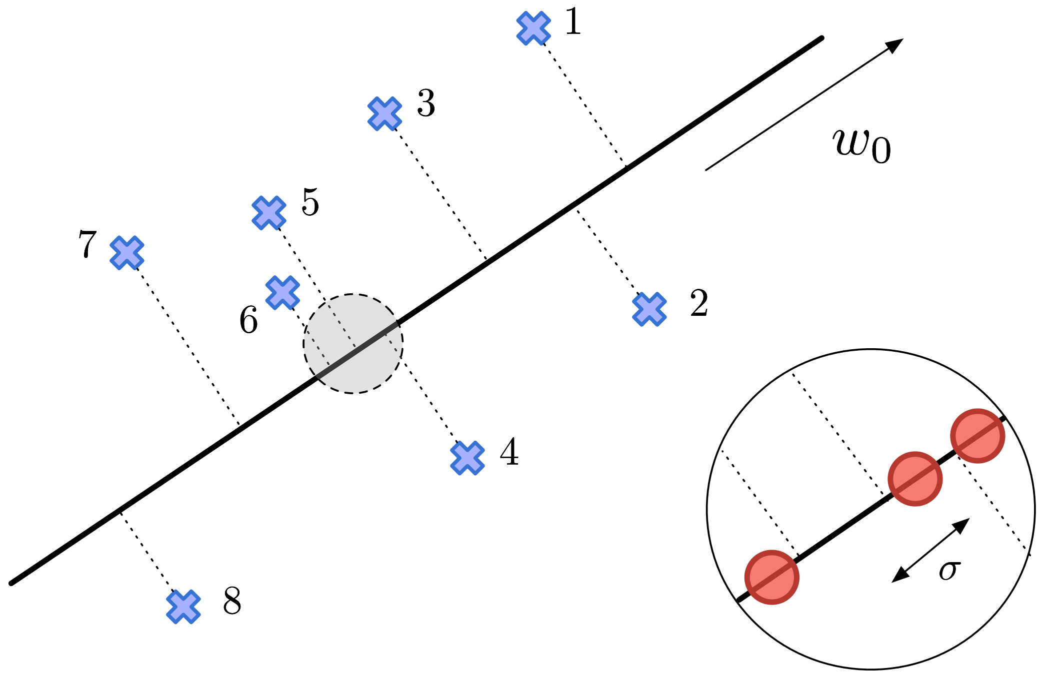


From data, predict ranks on future instances (simulated learning to rank).

Robustness to noise σ before ranking - uncertainty of user.

Experiments

Learning from rankings: Created dataset - ranked projection along unknown w_0.



6
5
3
8
1
7
2
4

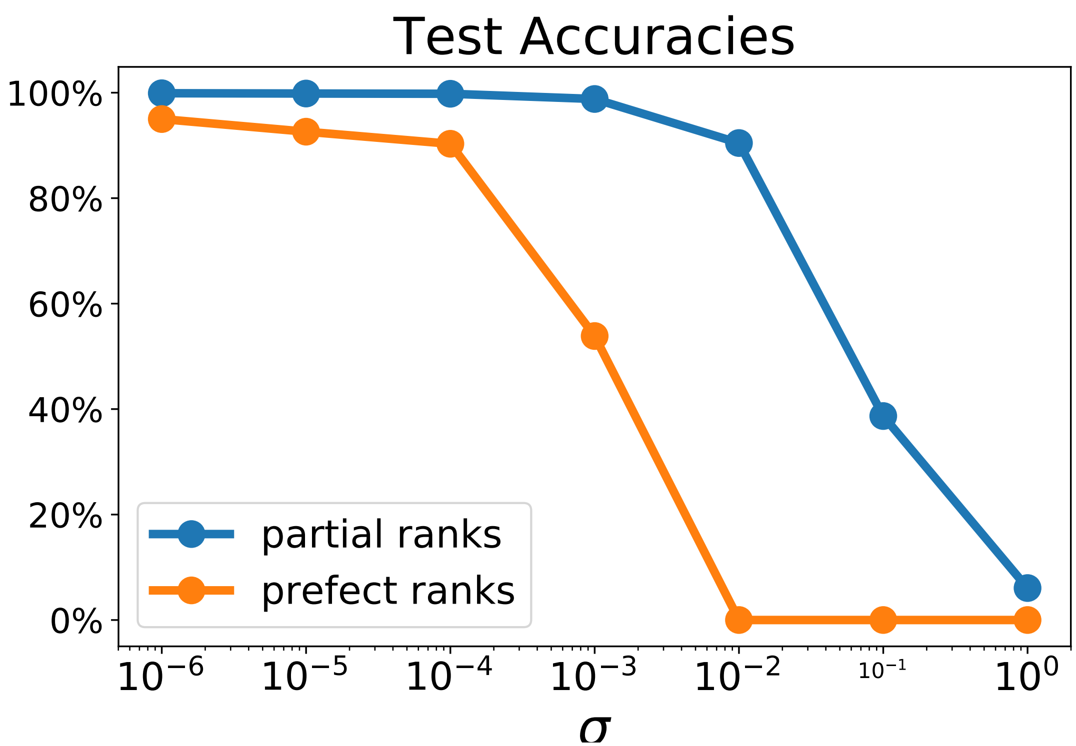
From data, predict ranks on future instances (simulated learning to rank).

Robustness to noise σ before ranking - uncertainty of user.

Experiments

Experiments on 4k instances of 100 vectors to rank, in dimension 9.

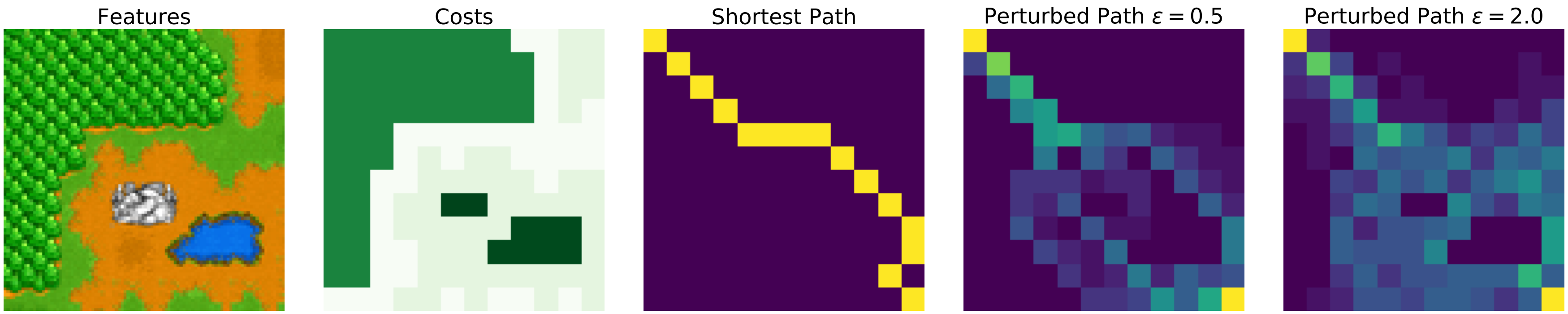
Robustness to noise observed for some tolerated variance



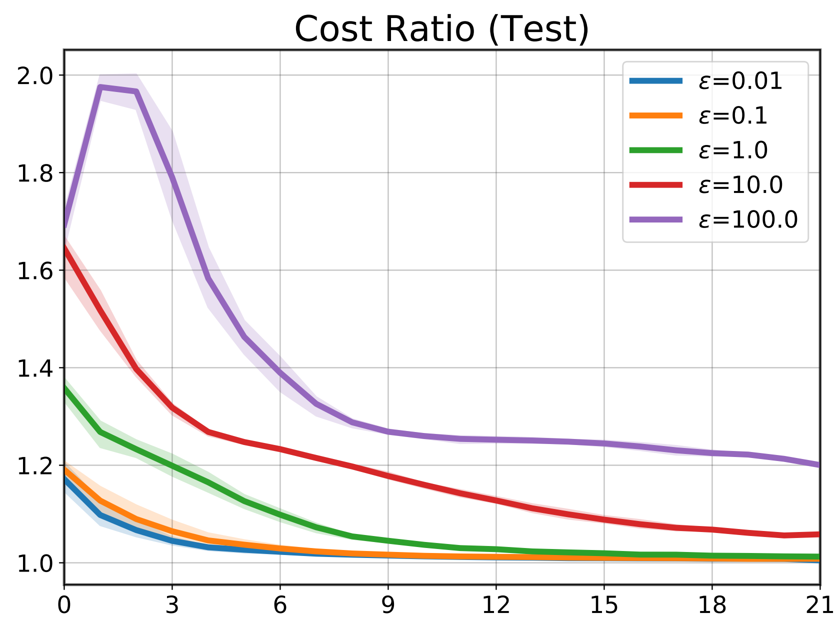
Fenchel-Young loss is convex in w: linear model, possible theoretical analysis.

Experiments

Learning from shortest paths: From 10k examples of Warcraft 96×96 RGB images, representing 12×12 costs, and matrix of shortest paths. (Vlastelica et al. 19)



Train a CNN for 20 epochs, to learn costs recovery of optimal paths.



MERCI