ALGORITHMS WITH PREDICTIONS

WARMING UP AND RENTING SKIS

Adam Polak

SOFSEM PhD School, Kraków, February 9th, 2026

MOTIVATION

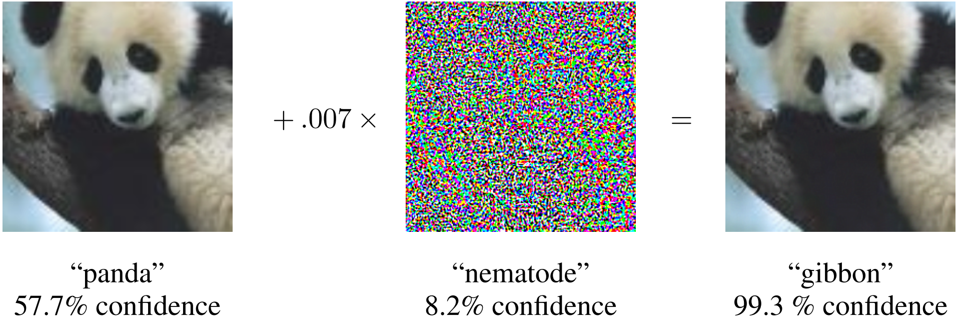
Classical algorithms

- Worst-case guarantees
- Overly pessimistic on easy instances

Machine learning

- Powerful most of the time
- No guarantees, can go crazy

RECALL: ADVERSARIAL EXAMPLES



Source: arxiv.org/abs/1412.6572

MOTIVATION

Classical algorithms

- Worst-case guarantees
- Overly pessimistic on easy instances

Machine learning

- Powerful most of the time
- No guarantees, can go crazy

MOTIVATION

Classical algorithms

- Worst-case guarantees
- Overly pessimistic on easy instances

Machine learning

- Powerful most of the time
- No guarantees, can go crazy

Best of both worlds: learning-augmented algorithms

Input + black-box predictions (e.g., coming from ML model)

ALGORITHMS WITH PREDICTIONS = ALPS

MOTIVATION

Classical algorithms

- Worst-case guarantees
- Overly pessimistic on easy instances

Machine learning

- Powerful most of the time
- No guarantees, can go crazy

Best of both worlds: learning-augmented algorithms

Input + black-box predictions (e.g., coming from ML model)

MOTIVATION

Classical algorithms

- Worst-case guarantees
- Overly pessimistic on easy instances

Machine learning

- Powerful most of the time
- No guarantees, can go crazy

Best of both worlds: learning-augmented algorithms

Input + black-box predictions (e.g., coming from ML model)

- **Consistency:** close-to-optimal performance when predictions accurate

MOTIVATION

Classical algorithms

- Worst-case guarantees
- Overly pessimistic on easy instances

Machine learning

- Powerful most of the time
- No guarantees, can go crazy

Best of both worlds: learning-augmented algorithms

Input + black-box predictions (e.g., coming from ML model)

- **Consistency:** close-to-optimal performance when predictions accurate
- **Robustness:** worst-case guarantees, even when predictions adversarial

MOTIVATION

Classical algorithms

- Worst-case guarantees
- Overly pessimistic on easy instances

Machine learning

- Powerful most of the time
- No guarantees, can go crazy

Best of both worlds: learning-augmented algorithms

Input + black-box predictions (e.g., coming from ML model)

- **Consistency:** close-to-optimal performance when predictions accurate
- **Robustness:** worst-case guarantees, even when predictions adversarial
- **Smoothness:** performance degrades slowly in the prediction error

MOTIVATION

Classical algorithms

- Worst-case guarantees
- Overly pessimistic on easy instances

Machine learning

- Powerful most of the time
- No guarantees, can go crazy

Best of both worlds: learning-augmented algorithms

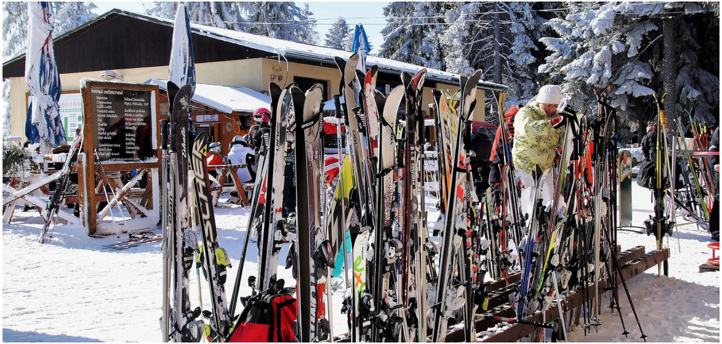
Input + black-box predictions (e.g., coming from ML model)

- **Consistency:** close-to-optimal performance when predictions accurate
- **Robustness:** worst-case guarantees, even when predictions adversarial
- **Smoothness:** performance degrades slowly in the prediction error
- **Learnability:** good predictions can be learned efficiently

WARM UP:

SKI RENTAL

SKI RENTAL



Each day of a ski season of unknown length decide whether to:

- rent the skis for one more day paying 1, or
- buy the skis paying cost B.

SKI RENTAL

Each day of a ski season of unknown length x decide whether to:

- rent the skis for one more day paying 1, or
- buy the skis paying cost B.

Competitive ratio:

$$\max \frac{\text{cost}(ALG)}{\text{cost}(OPT)}$$

Each day of a ski season of unknown length x decide whether to:

- rent the skis for one more day paying 1, or
- buy the skis paying cost B.

Competitive ratio:

$$\max \frac{\text{cost}(ALG)}{\text{cost}(OPT)}$$

Without predictions: (folklore)

- 2-competitive deterministic algorithm: wait B days and buy
- no deterministic algorithm can do better

SKI RENTAL

Each day of a ski season of unknown length x decide whether to:

- rent the skis for one more day paying 1, or
- buy the skis paying cost B.

Without predictions: (folklore)

- 2-competitive deterministic algorithm: wait B days and buy
- no deterministic algorithm can do better

Learning-augmented: (Purohit, Svitkina, Kumar, NeurIPS 2018)

- predicted length of ski season: y days

SKI RENTAL

Each day of a ski season of unknown length x decide whether to:

- rent the skis for one more day paying 1, or
- buy the skis paying cost B.

Without predictions: (folklore)

- 2-competitive deterministic algorithm: wait B days and buy
- no deterministic algorithm can do better

Learning-augmented: (Purohit, Svitkina, Kumar, NeurIPS 2018)

- predicted length of ski season: y days
- blindly following the prediction: if $y > B$, buy on day 1; if $y \leq B$, never buy
  $\text{cost}(ALG) \leq \text{cost}(OPT) + |x - y|$ (consistent, smooth, but not robust)

SKI RENTAL

Each day of a ski season of unknown length x decide whether to:

- rent the skis for one more day paying 1, or
- buy the skis paying cost B.

Without predictions: (folklore)

- 2-competitive deterministic algorithm: wait B days and buy
- no deterministic algorithm can do better

Learning-augmented: (Purohit, Svitkina, Kumar, NeurIPS 2018)

- predicted length of ski season: y days
- deterministic algorithm with a trade-off parameter $\lambda \in (0, 1)$
  - if $y > B$, buy on day λB; if $y \leq B$, buy on day B/λ

SKI RENTAL

Each day of a ski season of unknown length x decide whether to:

- rent the skis for one more day paying 1, or
- buy the skis paying cost B.

Without predictions: (folklore)

- 2-competitive deterministic algorithm: wait B days and buy
- no deterministic algorithm can do better

Learning-augmented: (Purohit, Svitkina, Kumar, NeurIPS 2018)

- predicted length of ski season: y days
- deterministic algorithm with a trade-off parameter $\lambda \in (0, 1)$
  - if $y > B$, buy on day λB; if $y \leq B$, buy on day B/λ
  - correct prediction: $(1 + \lambda)$-competitive **(consistency)**
  - wrong prediction: $(1 + \frac{1}{\lambda})$-competitive **(robustness)**

SKI RENTAL

Each day of a ski season of unknown length x decide whether to:

- rent the skis for one more day paying 1, or
- buy the skis paying cost B.

Without predictions: (folklore)

- 2-competitive deterministic algorithm: wait B days and buy
- no deterministic algorithm can do better

Learning-augmented: (Purohit, Svitkina, Kumar, NeurIPS 2018)

- predicted length of ski season: y days
- deterministic algorithm with a trade-off parameter $\lambda \in (0, 1)$
  - correct prediction: $(1 + \lambda)$-competitive **(consistency)**
  - wrong prediction: $(1 + \frac{1}{\lambda})$-competitive **(robustness)**
- this is Pareto-optimal (Angelopoulos, Durr, Jin, Kamali, ITCS 2020)

RANDOMIZATION HELPS

Without predictions:

- 2-competitive deterministic algorithm: wait *B* days and buy (folklore)
- no deterministic algorithm can do better

RANDOMIZATION HELPS

Without predictions:

- 2-competitive deterministic algorithm: wait B days and buy (folklore)
- no deterministic algorithm can do better
- simple 1.875-competitive algorithm: wait $\frac{3}{4}B$ days with $p = 0.5$, else wait B

RANDOMIZATION HELPS

Without predictions:

- 2-competitive deterministic algorithm: wait B days and buy (folklore)
- no deterministic algorithm can do better
- simple 1.875-competitive algorithm: wait $\frac{3}{4}B$ days with $p = 0.5$, else wait B
- $\frac{e}{e-1}$-competitive randomized algorithm (Karlin et al. '90)

RANDOMIZATION HELPS

Without predictions:

- 2-competitive deterministic algorithm: wait B days and buy (folklore)
- no deterministic algorithm can do better
- simple 1.875-competitive algorithm: wait $\frac{3}{4}B$ days with $p = 0.5$, else wait B
- $\frac{e}{e-1}$-competitive randomized algorithm (Karlin et al. ’90)
- no algorithm can do better

RANDOMIZATION HELPS

Without predictions:

- 2-competitive deterministic algorithm: wait B days and buy (folklore)
- no deterministic algorithm can do better
- simple 1.875-competitive algorithm: wait $\frac{3}{4}B$ days with $p = 0.5$, else wait B
- $\frac{e}{e-1}$-competitive randomized algorithm (Karlin et al. ’90)
- no algorithm can do better

Learning-augmented: (Purohit, Svitkina, Kumar, NeurIPS 2018)

- deterministic: $(1+\lambda)$-consistent, $(1+\frac{1}{\lambda})$-robust

RANDOMIZATION HELPS

Without predictions:

- 2-competitive deterministic algorithm: wait B days and buy (folklore)
- no deterministic algorithm can do better
- simple 1.875-competitive algorithm: wait $\frac{3}{4}B$ days with $p = 0.5$, else wait B
- $\frac{e}{e-1}$-competitive randomized algorithm (Karlin et al. ’90)
- no algorithm can do better

Learning-augmented: (Purohit, Svitkina, Kumar, NeurIPS 2018)

- deterministic: $(1 + \lambda)$-consistent, $(1 + \frac{1}{\lambda})$-robust
  i.e.: $(\frac{w}{w-1})$-consistent, w-robust

RANDOMIZATION HELPS

Without predictions:

- 2-competitive deterministic algorithm: wait B days and buy (folklore)
- no deterministic algorithm can do better
- simple 1.875-competitive algorithm: wait $\frac{3}{4}B$ days with $p = 0.5$, else wait B
- $\frac{e}{e-1}$-competitive randomized algorithm (Karlin et al. '90)
- no algorithm can do better

Learning-augmented: (Purohit, Svitkina, Kumar, NeurIPS 2018)

- deterministic: $(1 + \lambda)$-consistent, $(1 + \frac{1}{\lambda})$-robust
  i.e.: $(\frac{w}{w-1})$-consistent, w-robust
- randomized: $(w \ln \frac{w}{w-1})$-consistent, w-robust

RANDOMIZATION HELPS

Without predictions:

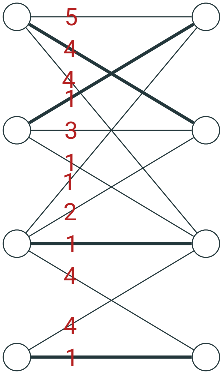
- 2-competitive deterministic algorithm: wait B days and buy (folklore)
- no deterministic algorithm can do better
- simple 1.875-competitive algorithm: wait $\frac{3}{4}B$ days with $p = 0.5$, else wait B
- $\frac{e}{e-1}$-competitive randomized algorithm (Karlin et al. ’90)
- no algorithm can do better

Learning-augmented: (Purohit, Svitkina, Kumar, NeurIPS 2018)

- deterministic: $(1 + \lambda)$-consistent, $(1 + \frac{1}{\lambda})$-robust
  i.e.: $(\frac{w}{w-1})$-consistent, w-robust
- randomized: $(w \ln \frac{w}{w-1})$-consistent, w-robust
- this is Pareto-optimal (Wei, Zhang, NeurIPS 2020)

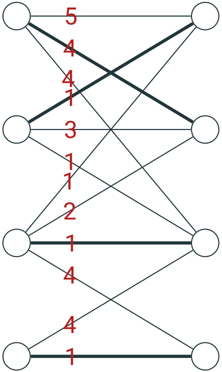
MINIMUM WEIGHT BIPARTITE MATCHING

MINIMUM WEIGHT BIPARTITE MATCHING



Hungarian algorithm: $O(nm)$ time

MINIMUM WEIGHT BIPARTITE MATCHING



Hungarian algorithm: $O(nm)$ time

What if we solve many similar instances? E.g.:

- instances sampled from a distribution,
- one instance slowly changing over time.

RECALL: LP FORMULATION OF MATCHING

Primal:

$$\begin{aligned} \text{minimize} \quad & \sum_{e \in E} c_e x_e \\ \text{subject to} \quad & \sum_{e \in N(v)} x_e = 1 \qquad \forall\, v \in V \\ & x_e \geqslant 0 \qquad \forall\, e \in E \end{aligned}$$

Dual:

$$\begin{aligned} \text{maximize} \quad & \sum_{v \in V} y_v \\ \text{subject to} \quad & y_u + y_v \leqslant c_{u,v} \qquad \forall\, (u, v) \in E \end{aligned}$$

Theorem: (Dinitz et al., NeurIPS 2021)

Suppose input comes with **predicted dual** $\hat{y}$

There is an $O(m\sqrt{n} \cdot \min\{||\hat{y} - y||_1, \sqrt{n}\})$-time algorithm

LEARNING-AUGMENTED MATCHING

Theorem: (Dinitz et al., NeurIPS 2021)

Suppose input comes with **predicted dual** $\hat{y}$

There is an $O(m\sqrt{n} \cdot \min\{||\hat{y} - y||_1, \sqrt{n}\})$-time algorithm

“Can I learn to predict duals?” (you, maybe)

LEARNING-AUGMENTED MATCHING

Theorem: (Dinitz et al., NeurIPS 2021)

Suppose input comes with **predicted dual** $\hat{y}$

There is an $O(m\sqrt{n} \cdot \min\{||\hat{y} - y||_1, \sqrt{n}\})$-time algorithm

"Can I learn to predict duals?" (you, maybe)

Theorem: (Dinitz et al., NeurIPS 2021)

Suppose input (edge costs) comes from distribution $\mathcal{D}$, maximum cost $\leqslant C$

Find optimal duals $\hat{y}$ for $\tilde{O}(n^3 C^2)$ samples

W.h.p., $\hat{y}$ approximately (up to $+\varepsilon$) minimizes $\mathbb{E}_{c \sim \mathcal{D}} ||\hat{y} - y_{\mathrm{OPT}}(c)||_1$

LEARNING-AUGMENTED MATCHING

Theorem: (Dinitz et al., NeurIPS 2021)

Suppose input comes with **predicted dual** $\hat{y}$

There is an $O(m\sqrt{n} \cdot \min\{||\hat{y} - y||_1, \sqrt{n}\})$-time algorithm

"Can I learn to predict duals?" (you, maybe)

Theorem: (Dinitz et al., NeurIPS 2021)

Suppose input (edge costs) comes from distribution $\mathcal{D}$, maximum cost $\leqslant C$

Find optimal duals $\hat{y}$ for $\tilde{O}(n^3 C^2)$ samples

W.h.p., $\hat{y}$ approximately (up to $+\varepsilon$) minimizes $\mathbb{E}_{c \sim \mathcal{D}} ||\hat{y} - y_{\mathrm{OPT}}(c)||_1$

Proof: bound VC-dim of $\{c \mapsto ||\hat{y} - y_{\mathrm{OPT}}(c)||_1 \mid c \in \mathbb{R}^n\}$; use PAC-learning toolbox

LEARNING-AUGMENTED MATCHING

Theorem: (Dinitz et al., NeurIPS 2021)

Suppose input comes with **predicted dual** $\hat{y}$

There is an $O(m\sqrt{n} \cdot \min\{||\hat{y} - y||_1, \sqrt{n}\})$-time algorithm

"Can I learn to predict duals?" (you, maybe)

Theorem: (Dinitz et al., NeurIPS 2021)

Suppose input (edge costs) comes from distribution $\mathcal{D}$, maximum cost $\leqslant C$

Find optimal duals $\hat{y}$ for $\tilde{O}(n^3C^2)$ samples (i.e., point-wise median of dual solutions)

W.h.p., $\hat{y}$ approximately (up to $+\varepsilon$) minimizes $\mathbb{E}_{c\sim\mathcal{D}}||\hat{y} - y_{\mathrm{OPT}}(c)||_1$

Proof: bound VC-dim of $\{c \mapsto ||\hat{y} - y_{\mathrm{OPT}}(c)||_1 \mid c \in \mathbb{R}^n\}$; use PAC-learning toolbox

THANK YOU!