Interactive learning via reductions

Daniel Hsu
Columbia University

May 9, 2019

Simons Symposium on New Directions in Theoretical Machine Learning

Interactive learning via reductions

("How can we build on the recent success in supervised learning?")

Daniel Hsu
Columbia University

May 9, 2019

Simons Symposium on New Directions in Theoretical Machine Learning

Interactive learning: Contextual bandits

Website operator

Interactive learning: Contextual bandits

Website operator

Loop:

1. User visits website with profile, browsing history . . .
2. Choose content to display on website.
3. Observe user reaction to content (*e.g.*, click, "like").

Interactive learning: Contextual bandits

Website operator
Loop:

1. User visits website with profile, browsing history . . .
2. Choose content to display on website.
3. Observe user reaction to content (*e.g.*, click, "like").

Goal: choose content that yield desired user behavior.

Interactive learning: Active learning

E-mail service provider

Interactive learning: Active learning

E-mail service provider

Loop:

1. Receive e-mail messages for users (spam or not).
2. Ask users to provide labels for some (borderline) messages.
3. Improve spam filter using newly labeled messages.

Interactive learning: Active learning

E-mail service provider

Loop:

1. Receive e-mail messages for users (spam or not).
2. Ask users to provide labels for some (borderline) messages.
3. Improve spam filter using newly labeled messages.

Goal: maximize accuracy of spam filter, minimize queries to users.

Interactive learning

Interactive learning

1. Learning agent (a.k.a. "learner") interacts with the world (e.g., patients, users) to achieve goals and gather data.

Interactive learning

1. Learning agent (a.k.a. "learner") interacts with the world (e.g., patients, users) to achieve goals and gather data.
2. Learner's performance based on chosen actions.

Interactive learning

1. Learning agent (a.k.a. "learner") interacts with the world (e.g., patients, users) to achieve goals and gather data.
2. Learner's performance based on chosen actions.
3. Data available to learner depends on chosen actions.

Interactive learning

1. Learning agent (a.k.a. "learner") interacts with the world (e.g., patients, users) to achieve goals and gather data.
2. Learner's performance based on chosen actions.
3. Data available to learner depends on chosen actions.

Efficient solutions to **exploration/exploitation dilemma** via reductions to supervised learning

Interactive learning

1. Learning agent (a.k.a. "learner") interacts with the world (e.g., patients, users) to achieve goals and gather data.
2. Learner's performance based on chosen actions.
3. Data available to learner depends on chosen actions.

Efficient solutions to **exploration/exploitation dilemma** via reductions to supervised learning

---

Rest of this talk:

1. Reductions for contextual bandits
2. Some challenges with this approach
   (an excuse to talk about generalization?)

1. Contextual bandit learning

Contextual bandit problem

For $t = 1, 2, \ldots, T$:

Contextual bandit problem

For $t = 1, 2, \ldots, T$:

1. Observe context $x_t \in \mathcal{X}$. [e.g., user profile, search query]

Contextual bandit problem

For $t = 1, 2, \ldots, T$:

1. Observe context $x_t \in \mathcal{X}$. [e.g., user profile, search query]
2. Choose action $a_t \in \mathcal{A}$. [e.g., ad to display]

Contextual bandit problem

For $t = 1, 2, \ldots, T$:

1. Observe context $x_t \in \mathcal{X}$. [e.g., user profile, search query]
2. Choose action $a_t \in \mathcal{A}$. [e.g., ad to display]
3. Collect reward $r_t(a_t) \in [0, 1]$. [e.g., 1 if click, 0 otherwise]

Contextual bandit problem

For $t = 1, 2, \ldots, T$:

0. Nature draws $(x_t, \boldsymbol{r}_t)$ from dist. $\mathcal{D}$ over $\mathcal{X} \times [0,1]^{\mathcal{A}}$.
1. Observe context $x_t \in \mathcal{X}$. [e.g., user profile, search query]
2. Choose action $a_t \in \mathcal{A}$. [e.g., ad to display]
3. Collect reward $r_t(a_t) \in [0,1]$. [e.g., 1 if click, 0 otherwise]

Contextual bandit problem

For $t = 1, 2, \ldots, T$:

0. Nature draws $(x_t, \boldsymbol{r}_t)$ from dist. $\mathcal{D}$ over $\mathcal{X} \times [0,1]^{\mathcal{A}}$.
1. Observe context $x_t \in \mathcal{X}$. [e.g., user profile, search query]
2. Choose action $a_t \in \mathcal{A}$. [e.g., ad to display]
3. Collect reward $r_t(a_t) \in [0,1]$. [e.g., 1 if click, 0 otherwise]

Task: choose a_t's that yield high expected reward (w.r.t. $\mathcal{D}$).

Contextual bandit problem

For $t = 1, 2, \ldots, T$:

0. Nature draws $(x_t, \boldsymbol{r}_t)$ from dist. $\mathcal{D}$ over $\mathcal{X} \times [0,1]^{\mathcal{A}}$.
1. Observe context $x_t \in \mathcal{X}$. [e.g., user profile, search query]
2. Choose action $a_t \in \mathcal{A}$. [e.g., ad to display]
3. Collect reward $r_t(a_t) \in [0,1]$. [e.g., 1 if click, 0 otherwise]

Task: choose a_t's that yield high expected reward (w.r.t. $\mathcal{D}$).

Contextual: use features x_t to choose good actions a_t.

Contextual bandit problem

For $t = 1, 2, \ldots, T$:

0. Nature draws $(x_t, \boldsymbol{r}_t)$ from dist. $\mathcal{D}$ over $\mathcal{X} \times [0,1]^{\mathcal{A}}$.
1. Observe context $x_t \in \mathcal{X}$. [e.g., user profile, search query]
2. Choose action $a_t \in \mathcal{A}$. [e.g., ad to display]
3. Collect reward $r_t(a_t) \in [0,1]$. [e.g., 1 if click, 0 otherwise]

Task: choose a_t's that yield high expected reward (w.r.t. $\mathcal{D}$).

Contextual: use features x_t to choose good actions a_t.

Bandit: $r_t(a)$ for $a \neq a_t$ is not observed.

(Non-bandit setting: whole reward vector $\boldsymbol{r}_t \in [0,1]^{\mathcal{A}}$ is observed.)

Challenges

Challenges

1. Exploration vs. exploitation.
    - Use what you've already learned (exploit), but also learn about actions that could be good (explore).
    - Must balance to get good statistical performance.

Challenges

1. Exploration vs. exploitation.
   - Use what you've already learned (exploit), but also learn about actions that could be good (explore).
   - Must balance to get good statistical performance.
2. Must use context.
   - Want to do as well as the best **policy** (i.e., decision rule)

     $$\pi : \text{context } x \;\mapsto\; \text{action } a$$

     from some **policy class** Π (a set of decision rules).
   - Computationally constrained w/ large Π.

Challenges

1. Exploration vs. exploitation.
   - Use what you've already learned (exploit), but also learn about actions that could be good (explore).
   - Must balance to get good statistical performance.
2. Must use context.
   - Want to do as well as the best **policy** (i.e., decision rule)

     $$\pi : \text{context } x \;\mapsto\; \text{action } a$$

     from some **policy class** Π (a set of decision rules).
   - Computationally constrained w/ large Π.
3. Selection bias, especially while *exploiting*.

Learning objective

Regret (*i.e.*, relative performance) to a policy class Π:

$$\underbrace{\max_{\pi \in \Pi} \frac{1}{T} \sum_{t=1}^{T} r_t(\pi(x_t))}_{\text{average reward of best policy}} \quad - \quad \underbrace{\frac{1}{T} \sum_{t=1}^{T} r_t(a_t)}_{\text{average reward of learner}}$$

Strong benchmark when Π has a policy w/ high expected reward.

Goal: regret $\to 0$ as fast as possible as $T \to \infty$.

Contextual bandits via reduction to supervised learning

Let $K := |\mathcal{A}|$ and $N := |\Pi|$.

Algorithm that operates via reduction to supervised learning
(Agarwal, H., Kale, Langford, Li, & Schapire, 2014).

- Regret bound: $\tilde{O}\left(\sqrt{\frac{K \log N}{T}}\right)$.
  Near optimal statistical performance
- # calls to supervised learner for Π: $\tilde{O}\left(\sqrt{\frac{TK}{\log N}}\right)$.
  Uses supervised learner less than once per round

Hypothetical "full-information" setting

If we observed rewards for <u>all actions</u> $\boldsymbol{r}_t = (r_t(a) : a \in \mathcal{A})$ ...

Hypothetical "full-information" setting

If we observed rewards for <u>all actions</u> $\boldsymbol{r}_t = (r_t(a) : a \in \mathcal{A}) \ldots$

- Like **supervised learning**, have *labeled data* after t rounds:

$$(x_1, \boldsymbol{r}_1), \ldots, (x_t, \boldsymbol{r}_t) \in \mathcal{X} \times \mathbb{R}^{\mathcal{A}}.$$

context	$\longrightarrow$	features
actions	$\longrightarrow$	classes
rewards	$\longrightarrow$	$-$costs
policy	$\longrightarrow$	classifier

Hypothetical "full-information" setting

If we observed rewards for <u>all actions</u> $\boldsymbol{r}_t = (r_t(a) : a \in \mathcal{A})$ ...

- Like **supervised learning**, have *labeled data* after t rounds:

$$(x_1, \boldsymbol{r}_1), \ldots, (x_t, \boldsymbol{r}_t) \in \mathcal{X} \times \mathbb{R}^{\mathcal{A}} .$$

context	$\longrightarrow$	features
actions	$\longrightarrow$	classes
rewards	$\longrightarrow$	$-$costs
policy	$\longrightarrow$	classifier

- Can often exploit structure of Π to get tractable algorithms. **Abstraction for supervised learning**: arg max oracle (AMO)

$$\mathrm{AMO}\left(\{(x_i, \boldsymbol{r}_i)\}_{i=1}^{t}\right) := \operatorname*{arg\,max}_{\pi \in \Pi} \sum_{i=1}^{t} r_i(\pi(x_i)) .$$

Hypothetical "full-information" setting

If we observed rewards for <u>all actions</u> $\boldsymbol{r}_t = (r_t(a) : a \in \mathcal{A})$ ...

- Like **supervised learning**, have *labeled data* after t rounds:

$$(x_1, \boldsymbol{r}_1), \ldots, (x_t, \boldsymbol{r}_t) \in \mathcal{X} \times \mathbb{R}^{\mathcal{A}} \,.$$

context	$\longrightarrow$	features
actions	$\longrightarrow$	classes
rewards	$\longrightarrow$	$-$costs
policy	$\longrightarrow$	classifier

- Can often exploit structure of Π to get tractable algorithms.
  Abstraction for supervised learning: arg max oracle (AMO)

$$\mathrm{AMO}\big(\{(x_i, \boldsymbol{r}_i)\}_{i=1}^t\big) \;:=\; \arg\max_{\pi \in \Pi} \sum_{i=1}^t r_i(\pi(x_i)) \,.$$

In bandit setting: *use randomization + importance weighting*.
Draw $a_t \sim \boldsymbol{P}_t$ for some pre-specified prob. dist. $\boldsymbol{P}_t$.

Inverse propensity weighting (Horvitz & Thompson, 1952)

Importance-weighted estimate of reward from round t:

$$\forall a \in \mathcal{A} \,.\quad \hat{r}_t(a) \;:=\; \begin{cases} \dfrac{r_t(a_t)}{P_t(a)} & \text{if } a = a_t \,, \\ 0 & \text{otherwise} \,. \end{cases}$$

Estimate avg. reward of policy: $\widehat{\mathrm{Rew}}_t(\pi) := \frac{1}{t}\sum_{i=1}^{t} \hat{r}_i(\pi(x_i))$.

How should we choose action distribution P_t?

Hedging over policies

Get action distributions via policy distributions.

$$\underbrace{(\boldsymbol{Q}, x)}_{(\text{policy distribution}, \text{context})} \longmapsto \underbrace{\boldsymbol{P}}_{\text{action distribution}}$$

Hedging over policies

Get action distributions via policy distributions.

$$\underbrace{(\boldsymbol{Q}, x)}_{\text{(policy distribution, context)}} \longmapsto \underbrace{\boldsymbol{P}}_{\text{action distribution}}$$

Policy distribution: $\boldsymbol{Q} = (Q(\pi) : \pi \in \Pi)$
probability dist. over policies π in the policy class Π

Hedging over policies

Get action distributions via policy distributions.

$$\underbrace{(\boldsymbol{Q}, x)}_{\text{(policy distribution, context)}} \quad \longmapsto \quad \underbrace{\boldsymbol{P}}_{\text{action distribution}}$$

Pick initial distribution $\boldsymbol{Q}_1$ over policies Π.
for round $t = 1, 2, \ldots$ **do**
    Nature draws $(x_t, \boldsymbol{r}_t)$ from dist. $\mathcal{D}$ over $\mathcal{X} \times [0,1]^{\mathcal{A}}$.
    Observe context x_t.
    Compute distribution $\boldsymbol{P}_t$ over $\mathcal{A}$ (using $\boldsymbol{Q}_t$ and x_t).
    Pick action $a_t \sim \boldsymbol{P}_t$.
    Collect reward $r_t(a_t)$.
    Compute new distribution $\boldsymbol{Q}_{t+1}$ over policies Π.
end for

The "good policy distribution" problem

Convex feasibility problem for policy distribution Q

The "good policy distribution" problem

Convex feasibility problem for policy distribution Q

$$\sum_{\pi \in \Pi} Q(\pi) \cdot \widehat{\mathrm{Reg}}_t(\pi) \leq \sqrt{\frac{K \log N}{t}} \qquad \text{(Low regret)}$$

The "good policy distribution" problem

Convex feasibility problem for policy distribution *Q*

$$\sum_{\pi \in \Pi} Q(\pi) \cdot \widehat{\text{Reg}}_t(\pi) \leq \sqrt{\frac{K \log N}{t}} \qquad \text{(Low regret)}$$

$$\widehat{\text{var}}_{\boldsymbol{Q}}\left(\widehat{\text{Rew}}_t(\pi)\right) \leq K\left(1 + \frac{\widehat{\text{Reg}}_t(\pi)}{\sqrt{\frac{K \log N}{t}}}\right) \quad \forall \pi \in \Pi \qquad \text{(Low variance)}$$

The "good policy distribution" problem

Convex feasibility problem for policy distribution Q

$$\sum_{\pi \in \Pi} Q(\pi) \cdot \widehat{\mathrm{Reg}}_t(\pi) \leq \sqrt{\frac{K \log N}{t}} \qquad \text{(Low regret)}$$

$$\widehat{\mathrm{var}}_Q\left(\widehat{\mathrm{Rew}}_t(\pi)\right) \leq K\left(1 + \frac{\widehat{\mathrm{Reg}}_t(\pi)}{\sqrt{\frac{K \log N}{t}}}\right) \quad \forall \pi \in \Pi \qquad \text{(Low variance)}$$

Theorem: Using feasible Q_t in round $t \Rightarrow$ near-optimal regret.

The "good policy distribution" problem

Convex feasibility problem for policy distribution Q

$$\sum_{\pi \in \Pi} Q(\pi) \cdot \widehat{\mathrm{Reg}}_t(\pi) \leq \sqrt{\frac{K \log N}{t}} \qquad \text{(Low regret)}$$

$$\widehat{\mathrm{var}}_{\boldsymbol{Q}}\left(\widehat{\mathrm{Rew}}_t(\pi)\right) \leq K \left(1 + \frac{\widehat{\mathrm{Reg}}_t(\pi)}{\sqrt{\frac{K \log N}{t}}}\right) \quad \forall \pi \in \Pi \qquad \text{(Low variance)}$$

Theorem: Using feasible $\boldsymbol{Q}_t$ in round $t \Rightarrow$ near-optimal regret.

$\star$ Can implement efficient "coordinate descent" solver via AMO.

Implementation via AMO

Finding "low variance" constraint violation for $\boldsymbol{Q}$:

$$\widehat{\mathrm{var}}_{\boldsymbol{Q}}\left(\widehat{\mathrm{Rew}}_t(\pi)\right) \leq K\left(1 + \frac{\widehat{\mathrm{Reg}}_t(\pi)}{\sqrt{\frac{K \log N}{t}}}\right) \quad \forall \pi \in \Pi \quad \text{(Low variance)}$$

1. Create fictitious rewards for each $i = 1, 2, \ldots, t$:

$$\widetilde{r}_i(a) \;:=\; K \cdot \frac{\hat{r}_i(a)}{\sqrt{\frac{K \log N}{t}}} + \frac{1}{Q(a|x_i)} \quad \forall a \in \mathcal{A}\,.$$

2. Obtain $\widetilde{\pi} := \mathrm{AMO}\big(\{(x_i, \widetilde{\boldsymbol{r}}_i)\}_{i=1}^t\big)$.

   Fact: $\widetilde{\mathrm{Rew}}_t(\widetilde{\pi}) >$ threshold iff $\widetilde{\pi}$'s constraint is violated.

Recap

Statistically optimal and efficient algorithm for contextual bandits by **reduction to supervised learning**.

Recap

Statistically optimal and efficient algorithm for contextual bandits by **reduction to supervised learning**.

- Take advantage of advances in supervised learning technology (e.g., deep learning)!

Recap

Statistically optimal and efficient algorithm for contextual bandits by **reduction to supervised learning**.

- Take advantage of advances in supervised learning technology (e.g., deep learning)!
- Similar algorithm design strategy works for **active learning**

  (Balcan, Beygelzimer, & Langford, 2006; Dasgupta, H., and Monteleoni, 2007; Beygelzimer, H., Langford, & Zhang, 2010; Zhang & Chaudhuri, 2014; Huang, Agarwal, H., Langford, and Schapire, 2015; Krishnamurthy, Agarwal, Huang, Daumé, & Langford, 2017; . . .)

Recap

Statistically optimal and efficient algorithm for contextual bandits by **reduction to supervised learning**.

- Take advantage of advances in supervised learning technology (e.g., deep learning)!
- Similar algorithm design strategy works for **active learning**

  (Balcan, Beygelzimer, & Langford, 2006; Dasgupta, H., and Monteleoni, 2007; Beygelzimer, H., Langford, & Zhang, 2010; Zhang & Chaudhuri, 2014; Huang, Agarwal, H., Langford, and Schapire, 2015; Krishnamurthy, Agarwal, Huang, Daumé, & Langford, 2017; . . .)

So what is the catch?

2. Problems

Major impediments with current reductions

Convex feasibility problem for policy distribution Q

$$\sum_{\pi \in \Pi} Q(\pi) \cdot \widehat{\mathrm{Reg}}_t(\pi) \;\le\; \sqrt{\frac{K \log N}{t}} \qquad \text{(Low regret)}$$

$$\widehat{\mathrm{var}}_Q\left(\widehat{\mathrm{Rew}}_t(\pi)\right) \le K\left(1 + \frac{\widehat{\mathrm{Reg}}_t(\pi)}{\sqrt{\frac{K \log N}{t}}}\right) \quad \forall \pi \in \Pi \qquad \text{(Low variance)}$$

Major impediments with current reductions

Convex feasibility problem for policy distribution Q

$$\sum_{\pi \in \Pi} Q(\pi) \cdot \widehat{\mathrm{Reg}}_t(\pi) \;\leq\; \sqrt{\frac{K \log N}{t}} \qquad \text{(Low regret)}$$

$$\widehat{\mathrm{var}}_Q\left(\widehat{\mathrm{Rew}}_t(\pi)\right) \leq K\left(1 + \frac{\widehat{\mathrm{Reg}}_t(\pi)}{\sqrt{\frac{K \log N}{t}}}\right) \quad \forall \pi \in \Pi \qquad \text{(Low variance)}$$

Algorithm parameters depends critically on uniform generalization bound for policy class Π.

Major impediments with current reductions

Convex feasibility problem for policy distribution Q

$$\sum_{\pi \in \Pi} Q(\pi) \cdot \widehat{\mathrm{Reg}}_t(\pi) \;\leq\; \sqrt{\frac{K \log N}{t}} \qquad \text{(Low regret)}$$

$$\widehat{\mathrm{var}}_Q\left(\widehat{\mathrm{Rew}}_t(\pi)\right) \leq K\left(1 + \frac{\widehat{\mathrm{Reg}}_t(\pi)}{\sqrt{\frac{K \log N}{t}}}\right) \quad \forall \pi \in \Pi \qquad \text{(Low variance)}$$

Algorithm parameters depends critically on uniform generalization bound for policy class Π.

- Used for balancing exploration & exploitation.

Major impediments with current reductions

Convex feasibility problem for policy distribution Q

$$\sum_{\pi \in \Pi} Q(\pi) \cdot \widehat{\text{Reg}}_t(\pi) \;\leq\; \sqrt{\frac{K \log N}{t}} \qquad \text{(Low regret)}$$

$$\widehat{\text{var}}_Q\left(\widehat{\text{Rew}}_t(\pi)\right) \leq K\left(1 + \frac{\widehat{\text{Reg}}_t(\pi)}{\sqrt{\frac{K \log N}{t}}}\right) \quad \forall \pi \in \Pi \qquad \text{(Low variance)}$$

Algorithm parameters depends critically on uniform generalization bound for policy class Π.

- Used for balancing exploration & exploitation.
- **Similar issue with active learning**: generalization bounds are crucially used to measure prediction "confidence".

Problems with uniform / *a posteriori* generalization bounds

- **Uniform convergence bounds** (Vapnik & Chervonenkis, 1971):
  Never use (except maybe when $\log N = O(1)$).

Problems with uniform / *a posteriori* generalization bounds

- **Uniform convergence bounds** (Vapnik & Chervonenkis, 1971):
  Never use (except maybe when $\log N = O(1)$).
- **Margin/norm-based generalization bounds** (e.g., Schapire, Freund, Bartlett, & Lee, 1998; Bartlett, Foster, & Telgarsky, 2017; . . .):
  - Useful for heavily-regularized models, or if observe large margin *a posteriori*.

Problems with uniform / *a posteriori* generalization bounds

- **Uniform convergence bounds** (Vapnik & Chervonenkis, 1971):
  Never use (except maybe when $\log N = O(1)$).
- **Margin/norm-based generalization bounds** (e.g., Schapire, Freund, Bartlett, & Lee, 1998; Bartlett, Foster, & Telgarsky, 2017; . . .):
  - Useful for heavily-regularized models, or if observe large margin *a posteriori*.

  Unclear if appropriate for (say) large neural nets, at least as used in practice:

  1. Find "overfitted" (interpolating) model with gradient descent.
  2. Then tune/regularize a bit.

Problems with uniform / *a posteriori* generalization bounds

- **Uniform convergence bounds** (Vapnik & Chervonenkis, 1971):
  Never use (except maybe when $\log N = O(1)$).
- **Margin/norm-based generalization bounds** (e.g., Schapire, Freund, Bartlett, & Lee, 1998; Bartlett, Foster, & Telgarsky, 2017; ...):
  - Useful for heavily-regularized models, or if observe large margin *a posteriori*.

  Unclear if appropriate for (say) large neural nets, at least as used in practice:
  1. Find "overfitted" (interpolating) model with gradient descent.
  2. Then tune/regularize a bit.

- Inductive bias (e.g., "gradient descent $\rightarrow$ least norm solution") is critical, but only part of the explanation for generalization.

Problems with uniform / *a posteriori* generalization bounds

- **Uniform convergence bounds** (Vapnik & Chervonenkis, 1971):
  Never use (except maybe when $\log N = O(1)$).
- **Margin/norm-based generalization bounds** (e.g., Schapire, Freund, Bartlett, & Lee, 1998; Bartlett, Foster, & Telgarsky, 2017; . . .):
  - Useful for heavily-regularized models, or if observe large margin *a posteriori*.

  Unclear if appropriate for (say) large neural nets, at least as used in practice:

  1. Find "overfitted" (interpolating) model with gradient descent.
  2. ~~Then tune/regularize a bit.~~
- Inductive bias (e.g., "gradient descent $\rightarrow$ least norm solution") is critical, but only part of the explanation for generalization.

  E.g., under what circumstances will the norm small?

Some surprising behavior

(Belkin, H., Ma, & Mandal, 2018; Belkin, H., & Xu, 2019)

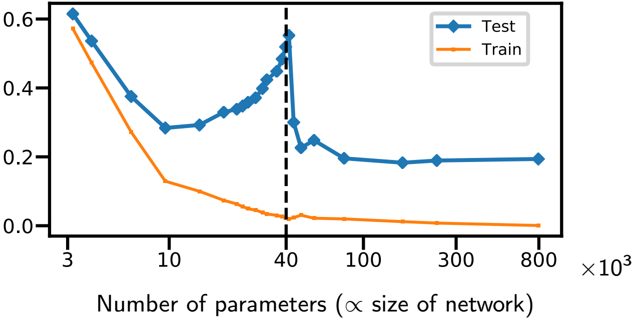
Fit two-layer neural network to training data with gradient descent.

Some surprising behavior

(Belkin, H., Ma, & Mandal, 2018; Belkin, H., & Xu, 2019)

Fit two-layer neural network to training data with gradient descent.

Mean squared error train/test on MNIST vs. # parameters

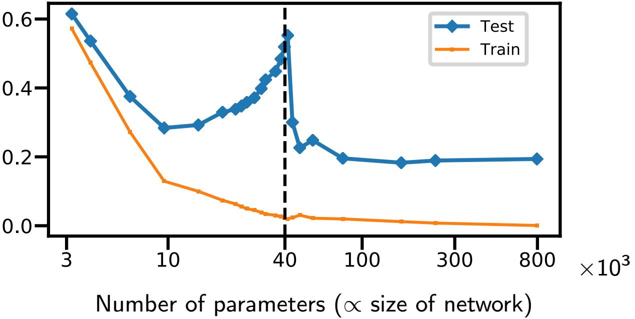


Some surprising behavior

(Belkin, H., Ma, & Mandal, 2018; Belkin, H., & Xu, 2019)

Fit two-layer neural network to training data with gradient descent.

Mean squared error train/test on MNIST vs. # parameters



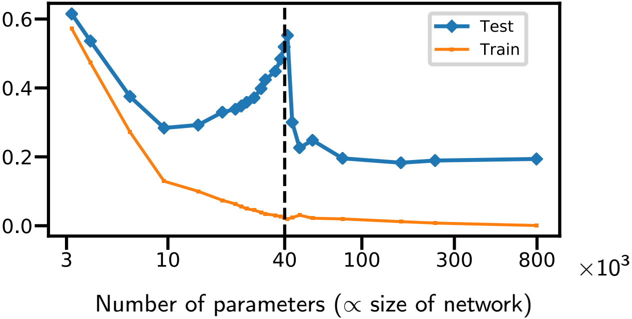
We prove this happens for certain "linearized" two-layer neural nets in some stylized settings. (Norm of predictor shows similar cusp.)

Some surprising behavior

(Belkin, H., Ma, & Mandal, 2018; Belkin, H., & Xu, 2019)

Fit two-layer neural network to training data with gradient descent.

Mean squared error train/test on MNIST vs. # parameters



We prove this happens for certain "linearized" two-layer neural nets in some stylized settings. (Norm of predictor shows similar cusp.)

Why do we observe good performance even when "overfitted"?

Risk bounds for prediction rules that interpolate

- Most of existing theory doesn't provide *a priori* guarantees for models that *interpolate* (noisy) training data.

Risk bounds for prediction rules that interpolate

- Most of existing theory doesn't provide *a priori* guarantees for models that *interpolate* (noisy) training data.
- Notable exception: **nearest neighbor** (Cover & Hart, 1967)

$$\mathrm{Err}(\mathrm{NN}) \quad \xrightarrow{n\to\infty} \quad 2 \times \mathrm{OPT} \qquad \text{(sort of)}$$

Risk bounds for prediction rules that interpolate

- Most of existing theory doesn't provide *a priori* guarantees for models that *interpolate* (noisy) training data.
- Notable exception: **nearest neighbor** (Cover & Hart, 1967)

$$\mathrm{Err}(\mathrm{NN}) \quad \stackrel{n\to\infty}{\longrightarrow} \quad 2 \times \mathrm{OPT} \qquad \text{(sort of)}$$

- Other interpolating models (Belkin, H., & Mitra, 2018)
  1. Simplicial interpolation (plausibly similar to ReLU networks)

$$\mathrm{Err}(\mathrm{SI}) \quad \stackrel{n\to\infty}{\longrightarrow} \quad (1 + 2^{-\Omega(d)}) \times \mathrm{OPT}$$

(under Massart noise condition, in $\mathbb{R}^d$).

Risk bounds for prediction rules that interpolate

- Most of existing theory doesn't provide *a priori* guarantees for models that *interpolate* (noisy) training data.
- Notable exception: **nearest neighbor** (Cover & Hart, 1967)

$$\mathrm{Err}(\mathrm{NN}) \quad \overset{n\to\infty}{\longrightarrow} \quad 2 \times \mathrm{OPT} \qquad \text{(sort of)}$$

- Other interpolating models (Belkin, H., & Mitra, 2018)
  1. Simplicial interpolation (plausibly similar to ReLU networks)

     $$\mathrm{Err}(\mathrm{SI}) \quad \overset{n\to\infty}{\longrightarrow} \quad (1 + 2^{-\Omega(d)}) \times \mathrm{OPT}$$

     (under Massart noise condition, in $\mathbb{R}^d$).
  2. Weighted & interpolated nearest neighbor

     $$\mathrm{MSE}(\mathrm{WINN}) \quad \leq \quad \mathrm{OPT} + O(n^{-2\alpha/(2\alpha+d)})$$

     when true regression function is α-Hölder smooth, in $\mathbb{R}^d$.

Risk bounds for prediction rules that interpolate

- Most of existing theory doesn't provide *a priori* guarantees for models that *interpolate* (noisy) training data.
- Notable exception: **nearest neighbor** (Cover & Hart, 1967)

$$\mathrm{Err(NN)} \quad \xrightarrow{n\to\infty} \quad 2 \times \mathrm{OPT} \qquad \text{(sort of)}$$

- Other interpolating models (Belkin, H., & Mitra, 2018)
  1. Simplicial interpolation (plausibly similar to ReLU networks)

$$\mathrm{Err(SI)} \quad \xrightarrow{n\to\infty} \quad (1 + 2^{-\Omega(d)}) \times \mathrm{OPT}$$

     (under Massart noise condition, in $\mathbb{R}^d$).
  2. Weighted & interpolated nearest neighbor

$$\mathrm{MSE(WINN)} \quad \leq \quad \mathrm{OPT} + O(n^{-2\alpha/(2\alpha+d)})$$

     when true regression function is α-Hölder smooth, in $\mathbb{R}^d$.

Would be great to have such results for interpolating neural nets, or even kernel machines.

Concluding remarks

How to manage exploration for interactive learning?

Concluding remarks

How to manage exploration for interactive learning?

- Reductions provide way to use advances in supervised learning to do better interactive learning.

Concluding remarks

How to manage exploration for interactive learning?

- Reductions provide way to use advances in supervised learning to do better interactive learning.
- **However**:
  Existing reductions crucially rely on generalization bounds.

Concluding remarks

How to manage exploration for interactive learning?

- Reductions provide way to use advances in supervised learning to do better interactive learning.
- **However**:
  Existing reductions crucially rely on generalization bounds.
  - Perhaps consequence of statistical learning framework . . .
  - Need better understanding of function classes we want to use (e.g., “practical” neural nets)

Concluding remarks

How to manage exploration for interactive learning?

- Reductions provide way to use advances in supervised learning to do better interactive learning.
- **However**:
  Existing reductions crucially rely on generalization bounds.
  - Perhaps consequence of statistical learning framework ...
  - Need better understanding of function classes we want to use (e.g., "practical" neural nets)

Thanks!

Simons Institute for the Theory of Computing

National Science Foundation (CCF-1740833, DMR-153491)

Sloan Foundation

3. Extra

(Sub-optimal) alternative

Explore-then-exploit:

1. Pick uniformly random actions in first τ rounds.
2. Obtain $\hat{\pi} := \mathrm{AMO}(\{(x_i, \hat{\boldsymbol{r}}_i\}))_{i=1}^{\tau}$.
3. Use $\hat{\pi}$ in remaining $T - \tau$ rounds.

(Sub-optimal) alternative

Explore-then-exploit:

1. Pick uniformly random actions in first τ rounds.
2. Obtain $\hat{\pi} := \mathrm{AMO}(\{(x_i, \hat{\boldsymbol{r}}_i\}))_{i=1}^{\tau}$.
3. Use $\hat{\pi}$ in remaining $T - \tau$ rounds.

Optimal τ still depends on uniform generalization bound for Π.

- But seems more benign, and easy to "adapt" to favorable conditions (Langford & Zhang, 2007).

(Sub-optimal) alternative

Explore-then-exploit:

1. Pick uniformly random actions in first τ rounds.
2. Obtain $\hat{\pi} := \mathrm{AMO}(\{(x_i, \hat{\boldsymbol{r}}_i\}))_{i=1}^{\tau}$.
3. Use $\hat{\pi}$ in remaining $T - \tau$ rounds.

Optimal τ still depends on uniform generalization bound for Π.

- But seems more benign, and easy to "adapt" to favorable conditions (Langford & Zhang, 2007).

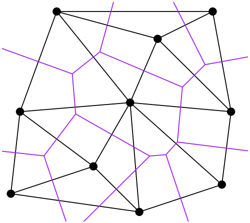
Other alternatives: replace bounds with resampling methods (e.g., permutation tests, bootstrap). Can these be made optimal?

Simplicial interpolation

- IID training examples $(x_1, y_1), \ldots, (x_n, y_n)$ from $\mathbb{R}^d \times [0, 1]$

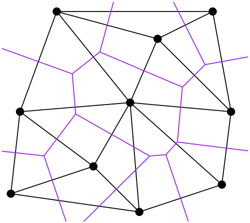
Simplicial interpolation

- IID training examples $(x_1, y_1), \ldots, (x_n, y_n)$ from $\mathbb{R}^d \times [0, 1]$
- Partition convex hull C of $(x_i)_{i=1}^n$ into simplices with x_i as vertices (via Delaunay triangulation)



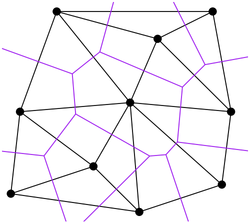
Simplicial interpolation

- IID training examples $(x_1, y_1), \ldots, (x_n, y_n)$ from $\mathbb{R}^d \times [0,1]$
- Partition convex hull C of $(x_i)_{i=1}^n$ into simplices with x_i as vertices (via Delaunay triangulation)
- Define $\hat{\eta}(x)$ on each simplex by affine interp. of vertices' labels



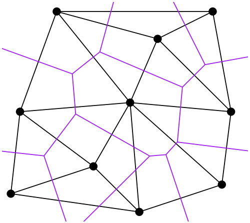
Simplicial interpolation

- IID training examples $(x_1, y_1), \ldots, (x_n, y_n)$ from $\mathbb{R}^d \times [0, 1]$
- Partition convex hull C of $(x_i)_{i=1}^n$ into simplices with x_i as vertices (via Delaunay triangulation)
- Define $\hat{\eta}(x)$ on each simplex by affine interp. of vertices' labels
- Result is piecewise linear on C. (Punt on what to do outside of C.)



Simplicial interpolation

- IID training examples $(x_1, y_1), \ldots, (x_n, y_n)$ from $\mathbb{R}^d \times [0,1]$
- Partition convex hull C of $(x_i)_{i=1}^n$ into simplices with x_i as vertices (via Delaunay triangulation)
- Define $\hat{\eta}(x)$ on each simplex by affine interp. of vertices' labels
- Result is piecewise linear on C. (Punt on what to do outside of C.)
- For classification, let $\hat{f}$ be plug-in classifier via $\hat{\eta}$.



Comparison to nearest neighbor

Restrict attention to a single simplex, with vertices $x_1, \ldots, x_{d+1}$.

- Suppose $\Pr(y = 1 \mid x) < 1/2$ for all points in the simplex
- Suppose training data has

$$y_1 = \cdots = y_d = 0$$

but $y_{d+1} = 1$ (due to noise, say).