Statistical Learning Theory, Optimization, and Neural Networks

Spencer Frei
(with thanks to Peter Bartlett and Alexander Rakhlin for slides)

UC Berkeley

August 1, 2022

Outline

Outline

Statistical Learning Theory

Aim: Predict an outcome y from some set $\mathcal{Y}$ of possible outcomes, on the basis of some observation x from a feature space $\mathcal{X}$.

Use *data set* of n pairs

$$\mathcal{S} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$$

to choose a function $f : \mathcal{X} \to \mathcal{Y}$ so that, for subsequent (x, y) pairs, $f(x)$ is a good prediction of y.

"Good" defined in terms of **loss function** $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$, e.g.,

- $\mathcal{Y} = \{-1, +1\}$: $\ell(f(x), y) = \mathsf{I}\{f(x) \neq y\}$ (0 if equal, 1 otherwise).
- $\mathcal{Y} = \mathbb{R}$, might have $\ell(f(x), y) = (f(x) - y)^2$.

Probabilistic Assumptions

Assume access to samples from unknown distribution:

- There is an (unknown) probability distribution P on $\mathcal{X} \times \mathcal{Y}$,
- The pairs $(X_1, Y_1), \ldots, (X_n, Y_n), (X, Y)$ are chosen independently according to P

The aim is to choose f with small *risk*:

$$L(f) = \mathbb{E}\ell(f(X), Y).$$

For instance, in pattern classification, this is the misclassification probability.

$$L(f) = L_{01}(f) = \mathbb{E}\mathsf{I}\{f(X) \neq Y\} = \mathsf{Pr}(f(X) \neq Y).$$

Key difficulty: our goals are in terms of unknown quantities related to unknown P. Have to use empirical data instead. Purview of statistics.

For instance, we can calculate the *empirical loss* of $f : \mathcal{X} \to \mathcal{Y}$

$$\widehat{L}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f(X_i))$$

The hope is that if $\widehat{L}(f)$ ('train error') is small, so is $L(f) = \mathbb{E}\ell(Y, f(X))$.

If we use data to construct an estimator $\widehat{f}_n(x) = \widehat{f}_n(x; X_1, Y_2, \ldots, X_n, Y_n)$, then the function $x \mapsto \widehat{f}_n(x)$ is random, since it depends on the random data $\mathcal{S} = (X_1, Y_1, \ldots, X_n, Y_n)$. Thus, the risk

$$\begin{aligned} L(\widehat{f}_n) &= \mathbb{E}\left[\ell(\widehat{f}_n(X), Y)|\mathcal{S}\right] \\ &= \mathbb{E}\left[\ell(\widehat{f}_n(X; X_1, Y_1, \ldots, X_n, Y_n), Y)|\mathcal{S}\right] \end{aligned}$$

is a random variable. We might aim for $\mathbb{E}L(\widehat{f}_n)$ small, or $L(\widehat{f}_n)$ small with high probability (over the training data).

Competing with the best predictor

The big question: is there a way to construct a learning algorithm with a guarantee that

$$L(\widehat{f}_n) - L(f^*) = L(\widehat{f}_n) - \inf_{f:\mathcal{X}\to\mathcal{Y} \text{ measurable}} L(f)$$

is small for large enough sample size n?

Or, more generally, suppose you work with some class of functions $\mathcal{F}$ that we hope captures well the relationship between X and Y (e.g. linear classifiers, neural networks). Can we guarantee that

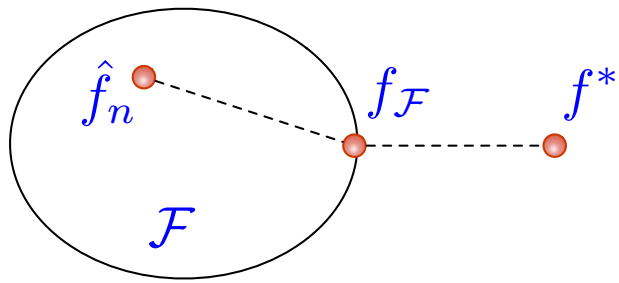
$$L(\widehat{f}_n) - \inf_{f\in\mathcal{F}} L(f)$$

is small for large n?

Outline

Approximation-Estimation Tradeoff

$$L(\widehat{f}_n) - L(f^*) = \underbrace{L(\widehat{f}_n) - \inf_{f \in \mathcal{F}} L(f)}_{\text{Estimation Error}} + \underbrace{\inf_{f \in \mathcal{F}} L(f) - L(f^*)}_{\text{Approximation Error}}$$



Clearly, the two terms are at odds with each other:

- Making $\mathcal{F}$ larger means smaller approximation error but (as we will see) larger estimation error
- Taking a larger sample n means smaller estimation error and has no effect on the approximation error.

We next describe how to characterize the estimation error.

Outline

Uniform Laws of Large Numbers: Motivation

Consider the performance of empirical risk minimization:

Choose $\widehat{f}_n \in \mathcal{F}$ to minimize $\widehat{L}(f)$, where $\widehat{L}$ is the *empirical risk*,

$$\widehat{L}(f) = P_n \ell(f(X), Y) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(X_i), Y_i).$$

For pattern classification, this is the proportion of training examples misclassified.

How does the excess risk, $L(\widehat{f}_n) - L(f_{\mathcal{F}})$ behave? We can write

$$L(\widehat{f}_n) - L(f_{\mathcal{F}}) = \left[L(\widehat{f}_n) - \widehat{L}(\widehat{f}_n)\right] + \boxed{\left[\widehat{L}(\widehat{f}_n) - \widehat{L}(f_{\mathcal{F}})\right]} + \boxed{\left[\widehat{L}(f_{\mathcal{F}}) - L(f_{\mathcal{F}})\right]}$$

Therefore,

$$\mathbb{E} L(\widehat{f}_n) - L(f_{\mathcal{F}}) \leq \mathbb{E}\left[L(\widehat{f}_n) - \widehat{L}(\widehat{f}_n)\right]$$

because second term is nonpositive and third is zero in expectation.

Uniform Laws of Large Numbers: Motivation

The term $L(\widehat{f}_n) - \widehat{L}(\widehat{f}_n)$ is not necessarily zero in expectation, since output on samples $\widehat{f}_n(x_i)$ doesn't have same distr. as $\widehat{f}_n(x)$ for test example (x, y). An easy upper bound is

$$L(\widehat{f}_n) - \widehat{L}(\widehat{f}_n) \leq \sup_{f \in \mathcal{F}} \left| L(f) - \widehat{L}(f) \right|,$$

and this motivates the study of uniform laws of large numbers.

Roadmap: study the sup to

- understand when learning is possible,
- understand implications for sample complexity,
- design new algorithms (regularizer arises from upper bound on sup)

Loss Class

In what follows, we will work with a function class $\mathcal{G}$ on $\mathcal{Z}$, and for the learning application, $\mathcal{G} = \ell \circ \mathcal{F}$:

$$\mathcal{G} = \{(x, y) \mapsto \ell(f(x), y) : f \in \mathcal{F}\}$$

and $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$.

Uniform Laws of Large Numbers

For a class $\mathcal{G}$ of functions $g : \mathcal{Z} \to [0, 1]$, suppose that $Z_1, \ldots, Z_n, Z$ are i.i.d. on $\mathcal{Z}$, and consider

$$U = \sup_{g \in \mathcal{G}} \left| \mathbb{E}g(Z) - \frac{1}{n} \sum_{i=1}^{n} g(Z_i) \right| = \sup_{g \in \mathcal{G}} |Pg - P_n g| =: \|P - P_n\|_{\mathcal{G}}.$$

If U converges to 0, this is called a *uniform law of large numbers*. If this holds we can show estimation error goes to zero,

$$\begin{aligned} L(\widehat{f}_n) - \widehat{L}(\widehat{f}_n) &= \mathbb{E}\ell(\widehat{f}_n(X); Y) - \frac{1}{n} \sum_{i=1}^{n} \ell(\widehat{f}_n(X_i), Y_i) \\ &\leq \sup_{f \in \mathcal{F}} \left| \mathbb{E}[\ell(f(X); Y)] - \frac{1}{n} \sum_{i=1}^{n} \ell(f(X_i); Y_i) \right| \\ &= \sup_{g \in \mathcal{G}} \left| \mathbb{E}[g(Z)] - \frac{1}{n} \sum_{i=1}^{n} g(Z_i) \right| = \|P - P_n\|_{\mathcal{G}} \to 0. \end{aligned}$$

Glivenko-Cantelli Classes

Definition.

$\mathcal{G}$ is a **Glivenko-Cantelli class** for P if $\|P_n - P\|_{\mathcal{G}} \xrightarrow{P} 0$.

- P is a distribution on $\mathcal{Z}$,
- $Z_1, \ldots, Z_n$ are drawn i.i.d. from P,
- P_n is the empirical distribution (mass $1/n$ at each of $Z_1, \ldots, Z_n$),
- $\mathcal{G}$ is a set of measurable real-valued functions on $\mathcal{Z}$ with finite expectation under P,
- $P_n - P$ is an **empirical process**, that is, a stochastic process indexed by a class of functions $\mathcal{G}$, and
- $\|P_n - P\|_{\mathcal{G}} := \sup_{g \in \mathcal{G}} |P_n g - P g|$.

Glivenko-Cantelli Classes

Not all $\mathcal{G}$ are Glivenko-Cantelli classes. For instance,

$$\mathcal{G} = \{\mathrm{I}\{z \in S\} : S \subset \mathbb{R}, |S| < \infty\}.$$

Then for a continuous distribution P, $Pg = 0$ for any $g \in \mathcal{G}$, but $\sup_{g \in \mathcal{G}} P_n g = 1$ for all n.

So although $P_n g \xrightarrow{as} Pg$ for all $g \in \mathcal{G}$, this convergence is not uniform over $\mathcal{G}$. $\mathcal{G}$ is **too large**.

In general, how do we decide whether $\mathcal{G}$ is "too large"?

Outline

Growth Function

Definition.

For a class $\mathcal{F} \subseteq \{0,1\}^{\mathcal{X}}$ and $\{x_1, \ldots, x_n\} \subset \mathcal{X}$, let

$$\mathcal{F}(x_1^n) := \{(f(x_1), \ldots, f(x_n)) : f \in \mathcal{F}\} \subset \{0,1\}^n.$$

The **growth function** is

$$\Pi_{\mathcal{F}}(n) = \max\{|\mathcal{F}(x_1^n)| : x_1, \ldots, x_n \in \mathcal{X}\}$$

- $\mathbb{E}\|P - P_n\|_{\mathcal{F}} = O\left(\sqrt{\frac{\log(\Pi_{\mathcal{F}}(n))}{n}}\right)$.
- $\Pi_{\mathcal{F}}(n) \leq |\mathcal{F}|$.
- $\Pi_{\mathcal{F}}(n) \leq 2^n$.
  - If equals 2^n, $\mathcal{F}$ can fit any possible set of (random!) labels in $\{0,1\}^n$; no useful bound on $\mathbb{E}\|P - P_n\|_{\mathcal{F}}$.

Vapnik-Chervonenkis Dimension

Definition.

A class $\mathcal{F} \subseteq \{0,1\}^{\mathcal{X}}$ **shatters** $\{x_1, \ldots, x_d\} \subseteq \mathcal{X}$ means that $|\mathcal{F}(x_1^d)| = 2^d$. The Vapnik-Chervonenkis dimension of $\mathcal{F}$ is

$$\begin{aligned} d_{VC}(\mathcal{F}) &= \max\{d : \text{some } x_1, \ldots, x_d \in \mathcal{X} \text{ is shattered by } \mathcal{F}\} \\ &= \max\left\{d : \Pi_{\mathcal{F}}(d) = 2^d\right\}. \end{aligned}$$

Theorem (Vapnik-Chervonenkis)**.**

For $n \geq d_{VC}(\mathcal{F})$,

$$\mathbb{E}\|P - P_n\|_{\mathcal{F}} = O\left(\sqrt{\frac{d_{VC}\log(n/d_{VC})}{n}}\right).$$

Uniform convergence holds if VC dimension is finite.

VC-Dimension of ReLU Networks

Theorem (Bartlett-Harvey-Liaw-Mehrabian'19).

Consider the class $\mathcal{F}$ of $\{-1, 1\}$-valued functions computed by a network with L layers, p parameters, and k hidden units with ReLU activations. Then,

$$\mathrm{VCdim}(\mathcal{F}) = \tilde{\Theta}\left(pL\right).$$

In particular, for $n \gg pL$, empirical will be close to population:

$$\mathbb{E}\|P - P_n\|_{\mathcal{F}} \leq \tilde{O}\left(\sqrt{\frac{pL\log(n/pL)}{n}}\right).$$

Of course, in modern neural nets, not always the case that $n \gg pL$. Can we do better?

Outline

Rademacher Averages

Let $\epsilon_i \overset{\text{i.i.d.}}{\sim} \text{Unif}(\{-1, +1\})$ be 'random labels'. Conditionally on samples Z_1^n, measure a predictor $g \in \mathcal{G}$'s fit with these random labels:

$$\frac{1}{n} \sum_{i=1}^{n} \epsilon_i g(Z_i).$$

If g fits random labels well: close to 1; if not: $o_n(1)$.

Given dataset $S = (Z_i)_1^n \overset{\text{i.i.d.}}{\sim} P$, the **empirical Rademacher complexity** of a function class $\mathcal{G}$ is

$$\hat{\mathfrak{R}}_S(\mathcal{G}) := \mathbb{E}_\epsilon \left[\sup_{g \in \mathcal{G}} \frac{1}{n} \sum_{i=1}^{n} \epsilon_i g(Z_i) \right].$$

If for any ϵ_i exists $g \in \mathcal{G}$ that can fit random labels, then $\mathfrak{R}_S(\mathcal{G}) = 1$.

The **Rademacher complexity** of $\mathcal{G}$ is

$$\mathfrak{R}(\mathcal{G}) := \mathbb{E}_S \hat{\mathfrak{R}}_S(\mathcal{G}) = \mathbb{E}_S \mathbb{E}_\epsilon \left[\sup_{g \in \mathcal{G}} \frac{1}{n} \sum_{i=1}^{n} \epsilon_i g(Z_i) \right].$$

Uniform Laws and Rademacher Complexity

Theorem.

For any $\mathcal{G}$,

$$\mathbb{E}\|P - P_n\|_{\mathcal{G}} \leq 2\mathfrak{R}_n(\mathcal{G})$$

If $\mathcal{G} \subset [0,1]^{\mathcal{Z}}$, then

$$\frac{1}{2}\mathfrak{R}_n(\mathcal{G}) - \sqrt{\frac{\log 2}{2n}} \leq \mathbb{E}\|P - P_n\|_{\mathcal{G}} \leq 2\mathfrak{R}_n(\mathcal{G}),$$

Thus, $\mathfrak{R}_n(\mathcal{G}) \to 0$ iff $\mathbb{E}\|P - P_n\|_{\mathcal{G}} \to 0$.

Thus, uniform convergence over $g \in \mathcal{G}$ equivalent to Rademacher complexity is $o_n(1)$.

Outline

Rademacher Complexity: Structural Results

There is a 'Rademacher calculus' which makes calculating Rademacher complexity of a function class easier easier.

Theorem.

Subsets $\mathcal{F} \subseteq \mathcal{G}$ implies $\hat{\mathfrak{R}}_n(\mathcal{F}) \leq \hat{\mathfrak{R}}_n(\mathcal{G})$.

Scaling $\hat{\mathfrak{R}}_n(c\mathcal{F}) = |c|\hat{\mathfrak{R}}_n(\mathcal{F})$.

Plus Constant For $|g(X)| \leq 1$,
$|\mathbb{E}\|R_n\|_{\mathcal{F}+g} - \mathbb{E}\|R_n\|_{\mathcal{F}}| \leq \sqrt{2\log 2/n}$.

Convex Hull $\hat{\mathfrak{R}}_n(\mathrm{co}\mathcal{F}) = \|R_n\|_{\mathcal{F}}$, where $\mathrm{co}\mathcal{F}$ is the convex hull of $\mathcal{F}$.

Contraction If $\ell : \mathbb{R} \times \mathbb{R} \to [-1, 1]$ has $\hat{y} \mapsto \ell(\hat{y}, y)$ 1-Lipschitz for all y, then for $\ell \circ \mathcal{F} = \{(x, y) \mapsto \ell(f(x), y)\}$, $\mathfrak{R}_n(\ell \circ \mathcal{F}) \leq 2\mathfrak{R}_n(\mathcal{F}) + c/\sqrt{n}$.

Rademacher Complexity of ReLU Networks

Consider L-layer ReLU networks $\sigma(t) = \max(0, t)$ with bounded norm:

$$\mathcal{F}_B := \big\{ x \mapsto W_L \sigma(W_{L-1} \cdots \sigma(W_1 x) \cdots),\ W_k \in \mathbb{R}^{m_k \times m_{k-1}},$$
$$m_0 = \dim(x),\ m_L = 1,\ m_k \in \mathbb{N}\ (k \neq 1, L),\ \|W_i\|_F \leq B\ \forall i \big\}.$$

Theorem.

For a training set $(X_1, Y_1), \ldots, (X_n, Y_n) \in \mathcal{X} \times \{\pm 1\}$ with $\|X_i\| \leq 1$ a.s.,

$$\mathfrak{R}_n(\mathcal{F}_B) \leq \frac{(2B)^L}{\sqrt{n}}.$$

Note that the *width* of the network $(\max_k m_k)$ does not play a role here—only the norms.

This means that we do not need number of samples to be larger than the width of the network provided the norms of the weights are bounded.

Rademacher Complexity of ReLU Networks: Proof Sketch

Consider two-layer $f(x) = u^\top \sigma(Wx)$ first. For $W^\top = (w_1 \cdots w_k)$, we use positive homogeneity, i.e. $\sigma(|c|t) = |c|\sigma(t)$:

$$\hat{\mathfrak{R}}(\mathcal{F}_R) = \frac{1}{n}\mathbb{E}_{\epsilon_i}\left[\sup_{\|u\|,\|W\|_F \le B} \sum_{i=1}^n \epsilon_i \sum_{j=1}^n u_j \sigma(\langle w_j, x_i\rangle)\right]$$

$$= \frac{1}{n}\mathbb{E}_{\epsilon_i}\left[\sup_{\|u\|,\|W\|_F \le B} \sum_{j=1}^m u_j \|w_j\|_2 \sum_{i=1}^n \epsilon_i \sigma(\langle w_j/\|w_j\|_2, x_i\rangle)\right]$$

$$\le \frac{1}{n}\mathbb{E}_{\epsilon_i}\left[\left(\sup_{\|u\|,\|W\|_F \le B} \sum_{j=1}^m |u_j| \|w_j\|_2\right) \max_{j\in[m]} \left|\sum_{i=1}^n \epsilon_i \sigma(\langle w_j/\|w_j\|_2, x_i\rangle)\right|\right]$$

$$\le \frac{B^2}{n}\mathbb{E}_{\epsilon_i}\left[\max_{j\in[m]} \left|\sum_{i=1}^n \epsilon_i \sigma(\langle w_j/\|w_j\|_2, x_i\rangle)\right|\right] \qquad \text{(Cauchy-Schwarz)}$$

$$\le \frac{B^2}{n}\mathbb{E}_{\epsilon_i}\left[\sup_{\|\bar{w}\|\le 1} \left|\sum_{i=1}^n \epsilon_i \sigma(\langle \bar{w}, x_i\rangle)\right|\right] \qquad \text{(then use contraction, C-S)}$$

Recap

We showed that the excess risk of ERM $\widehat{f}_n$ can be bounded by

$$\begin{aligned}\mathbb{E}[L(\widehat{f}_n) - \widehat{L}(\widehat{f}_n)] &= \mathbb{E}\ell(\widehat{f}_n(x); y)] - \frac{1}{n}\sum_{i=1}^{n} \ell(\widehat{f}_n(X_i), Y_i) \\ &\leq \mathbb{E}\|P - P_n\|_{\mathcal{G}} \leq 2\mathfrak{R}_n(\mathcal{G}).\end{aligned}$$

Rademacher complexity is easier to deal with due to Rademacher calculus.

For ReLU networks with each layer's weights bounded by B, we can guarantee that excess risk is small if $n \gg (2B)^{2L}$.

A natural question is whether or not we can truly find the ERM $\widehat{f}_n$, since neural network optimization is non-convex in general. More details on this in next tutorial.

Outline

Uniform convergence and benign overfitting

The last slides have focused on trying to establish conditions under which

$$\mathbb{E}[L(\widehat{f}_n) - \widehat{L}(\widehat{f}_n)] \leq 2\mathfrak{R}_n(\ell \circ \mathcal{F}) \overset{n\to\infty}{\longrightarrow} 0.$$

We saw that Rademacher complexity of a function class grows as its ability to fit random labels increases:

$$\hat{\mathfrak{R}}_n(\mathcal{F}) = \mathbb{E}_{\epsilon_i \overset{\text{i.i.d.}}{\sim} \text{Unif}(\{\pm 1\})} \left[\sup_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \epsilon_i f(x_i) \right].$$

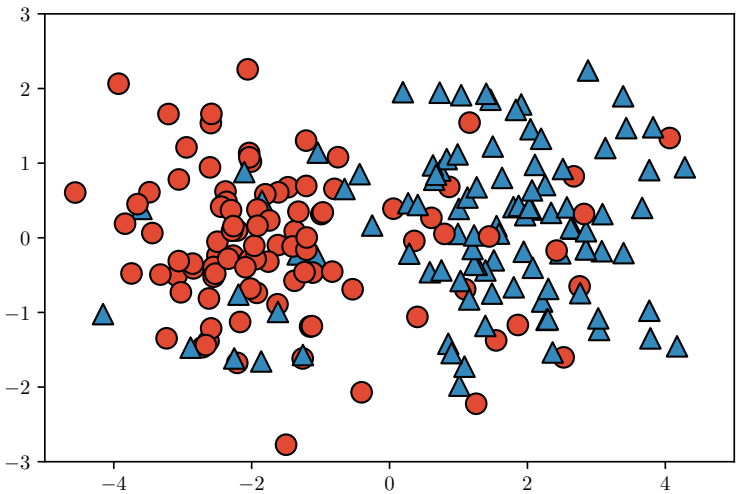
What if we consider a distribution P and hypothesis class $\mathcal{F}$ where $L(f) \geq \sigma^2 > 0$ for all $f \in \mathcal{F}$, and $\widehat{f}_n$ is an *interpolator*: it achieves $\widehat{L}(\widehat{f}_n) = 0$. Then $\widehat{f}_n$ *must* fit the noise, and $\hat{\mathfrak{R}}_n(\mathcal{F}) = 1$.

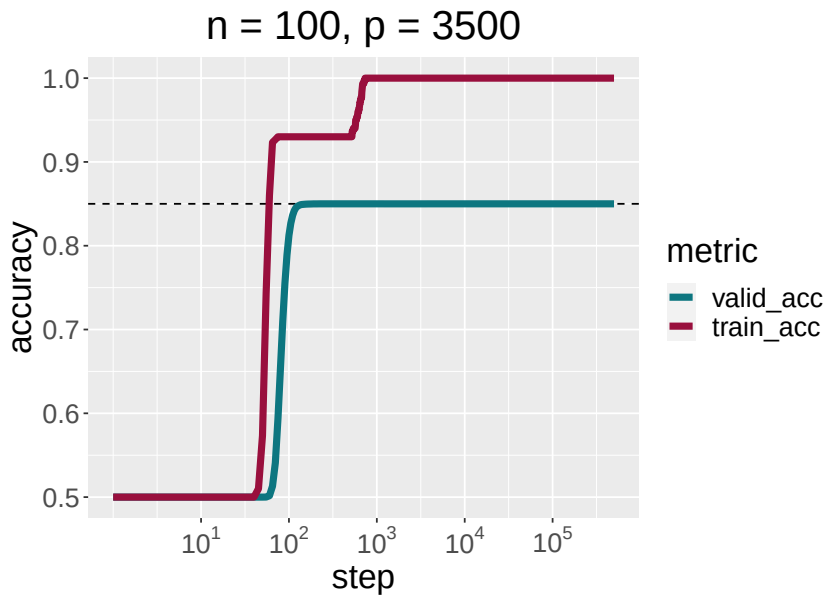
Cannot hope for $\mathbb{E}[L(\widehat{f}_n) - \widehat{L}(\widehat{f}_n)] \to 0$ in this setting! At best, can get

$$\mathbb{E}[L(\widehat{f}_n) - \widehat{L}(\widehat{f}_n)] = \mathbb{E}[L(\widehat{f}_n)] \to \sigma^2.$$

Uniform convergence and benign overfitting

Consider the following experiment: you have Gaussian mixture model data, but you have a 'noisy' distribution where every sample has 15% chance of having a random label. Then train a two-layer network by gradient descent on this (noisy) data.





The neural network achieves 100% training accuracy and *simultaneously* optimal test accuracy (85%)—provably so [F.–Chatterji–Bartlett'22].

This seems in conflict with many intuitions developed by uniform convergence. See: tutorial & talks tomorrow.

Outline

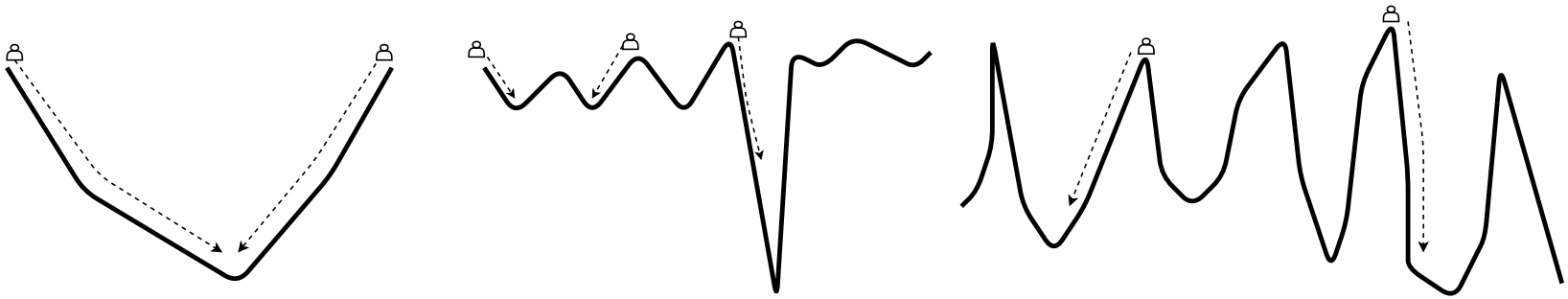
Neural Network Optimization

In the previous slides, we focused on the generalization aspect of neural networks and identified conditions under which $\widehat{L}(\widehat{f}_n) \approx L(\widehat{f}_n)$.

In the remainder, we will discuss approaches for showing $\widehat{L}(\widehat{f}_n)$ is small when $\widehat{f}_n$ is a neural network trained by gradient descent.

Outline

Convexity and gradient descent



A differentiable function $g : \mathbb{R}^d \to \mathbb{R}$ is *convex* if for all $w, v \in \mathbb{R}^d$,

$$g(w) + \langle \nabla g(w), v - w \rangle \leq g(v).$$

If twice-differentiable, convex if $\nabla^2 g(w) \succeq 0$.

Gradient descent with learning rate $\eta > 0$ is,

$$w_{t+1} = w_t - \eta \nabla g(w_t).$$

Convexity and gradient descent

Gradient descent can efficiently find minimizers of smooth convex functions. For example, consider $\|\nabla g(w)\| \leq L_1$. Short proof: let w^* be a minimizer of g. Then for $\eta \leq \epsilon / L_1^2$,

$$\begin{aligned}
&\|w_t - w^*\|^2 - \|w_{t+1} - w^*\|^2 \\
&\quad = 2\eta \langle \nabla g(w_t), w_t - w^* \rangle - \eta^2 \|\nabla g(w_t)\|^2 \\
&\quad \geq 2\eta \left[(g(w_t) - g(w^*)) - \eta L_1^2/2 \right] \qquad \text{(convexity, smoothness)} \\
&\quad \geq 2\eta \left[g(w_t) - g(w^*) - \epsilon/2 \right] \qquad \text{(} \eta \leq \epsilon / L_1^2 \text{)}.
\end{aligned}$$

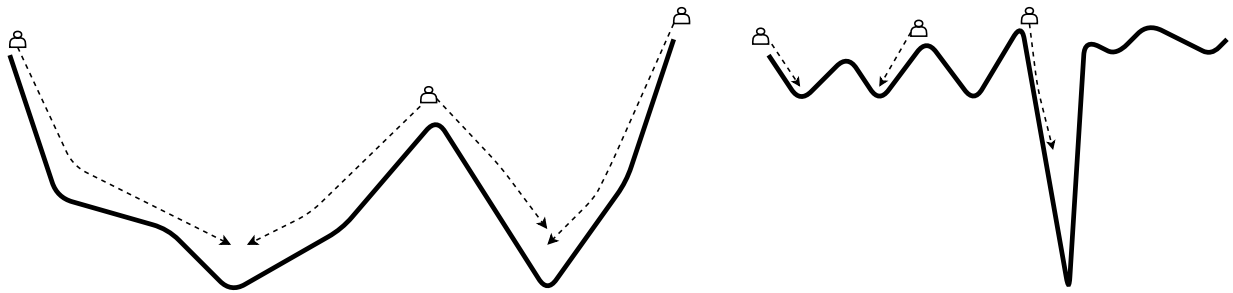
If $g(w_t) \geq g(w^*) + \epsilon$, then we get:

$$\|w_{t+1} - w^*\|^2 \leq \|w_t - w^*\|^2 - \eta\epsilon.$$

So, distance from optimal point will decrease until we get optimal value, thus will eventually get $g(w_t) \leq g(w^*) + \epsilon$.

Outline

Nonconvexity and gradient descent



In general, if your objective is non-convex but smooth, the best you can hope for is to find stationary points: $\|\nabla g(w)\| \approx 0$.

In some structured non-convex problems, we can guarantee that $\|\nabla g(w)\| \approx 0$ implies $g(w) \approx \text{OPT}$.

Polyak–Lojasiewicz (PL) inequality

Definition.

A differentiable function $g : \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ achieving minimum value OPT satisfies the **PL inequality** if there exist $\alpha, \mu > 0$ such that,

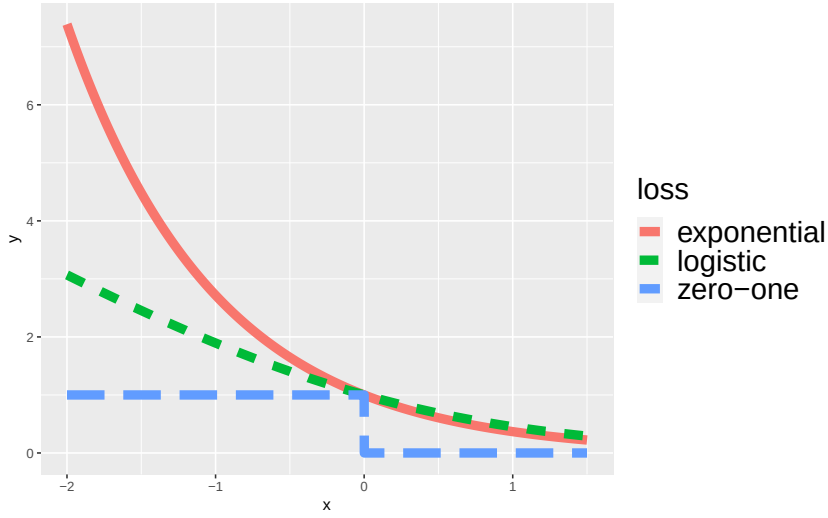
$$\text{for all } w, \quad \|\nabla g(w)\|^\alpha \geq \mu(g(w) - \text{OPT}).$$

PL inequality can be satisfied by non-convex functions.

If PL inequality is satisfied and g is smooth, gradient descent drives $\|\nabla g(w_t)\| \to 0$ and thus $g(w_t) \to \text{OPT}$.

PL inequality appears to unify much of neural network optimization works [F.-Gu'21, Liu-Zhu-Belkin'21]

Establishing PL inequality for neural networks



Consider a classification task ($y_i \in \{\pm 1\}$), training with the exponential loss (similar analysis for logistic loss):

$$\ell(yf(x;W)) := \exp\big(-yf(x;W)\big), \qquad \hat{L}(W) := \frac{1}{n}\sum_{i=1}^{n} \ell(y_i f(x_i;W)).$$

The exponential loss is a convex surrogate for the 0-1 loss (classification error). If we can minimize this, we get an upper bound for 0-1 loss. Same idea holds for logistic loss.

Establishing PL inequality for neural networks

PL holds if $\exists \alpha, c > 0$ s.t. $\|\nabla \hat{L}(W)\|^\alpha \geq c\hat{L}(W) = c \cdot \frac{1}{n} \sum_{i=1}^{n} \ell(y_i f(x_i; W))$.

To show PL, can use variational def'n of norm: for any V, $\|V\| \leq 1$,

$$\begin{aligned}
\left\|\nabla \hat{L}(W^{(t)})\right\| &\geq \langle \nabla \hat{L}(W^{(t)}), -V \rangle \\
&= \frac{1}{n} \sum_{i=1}^{n} -\ell'(y_i f(x_i; W^{(t)}) \cdot y_i \langle \nabla f(x_i; W^{(t)}), V \rangle \\
&= \frac{1}{n} \sum_{i=1}^{n} \ell(y_i f(x_i; W^{(t)})) \cdot \underbrace{y_i \langle \nabla f(x_i; W^{(t)}), V \rangle}_{(*)\text{: 'gradient margin'}}.
\end{aligned}$$

If we can show $(*) \geq c$, then we have established PL inequality.

Establishing PL inequalities for neural networks

For any V, $\|V\|_F \le 1$, we have the lower bound

$$\left\|\nabla \hat{L}(W^{(t)})\right\| \ge \frac{1}{n}\sum_{i=1}^{n} \ell(y_i f(x_i; W^{(t)})) \cdot \underbrace{y_i \langle \nabla f(x_i; W^{(t)}), V\rangle}_{(*)\text{: 'gradient margin'}}.$$

The hope is $(*) \ge c$.

In words: can we construct a linear combination of the features (given by $x_i \mapsto \nabla f(x_i; W^{(t)})$) to classify the labels y_i well? If so, then PL inequality holds at time t.

The quantity $\nabla f(x_i; W^{(t)})$ changes in hard-to-predict ways over time. A more tractable quantity to understand: $\nabla f(x_i; W^{(0)})$, where $W^{(0)}$ is randomly-initialized. Then, at least, if $\|W^{(t)} - W^{(0)}\|$ is small, we could hope that $\nabla f(x_i; W^{(t)}) \approx \nabla f(x_i; W^{(0)})$, and then a path forward is to bound from below

$$y_i \langle \nabla f(x_i; W^{(0)}), V\rangle \overset{(?)}{\ge} c.$$

Outline

Neural Tangent Kernel

For simplicity consider two-layer networks with activation ϕ:

$$f(x; W) = \sum_{j=1}^{m} a_j \phi(\langle w_j, x \rangle), \qquad a_j \sim \text{Unif}(\{\pm 1/\sqrt{m}\}), \qquad a = (a_j) \in \mathbb{R}^m.$$

The gradient of f with respect to W is,

$$\nabla f(x; W) = D_x^W a x^\top, \qquad D_x^W = \text{diag}\big(\phi'(\langle w_j, x \rangle)\big) \in \mathbb{R}^{m \times m}.$$

Suppose $W = W^{(0)} \in \mathbb{R}^{m \times d}$ has i.i.d. $N(0, 1)$ entries. Clearly, $\nabla f(x; W^{(0)})$ is random. We want to understand when we can construct $V^\top = (v_1 \ v_2 \ \dots \ v_m)$ s.t.

$$y_i \langle \nabla f(x_i; W^{(0)}), V \rangle = y_i \sum_{j=1}^{m} a_j \langle v_j, \boxed{x_i \phi'(\langle w_j^{(0)}, x_i \rangle)} \rangle \overset{(?)}{\geq} c.$$

Neural Tangent Kernel

Can define a (random) function $\hat{K}$ in terms of 'features' of gradient of network at initialization:

$$\begin{aligned}\hat{K}(x,z) &= \langle \nabla f(x; W^{(0)}), \nabla f(z; W^{(0)}) \rangle \\ &= \langle D_x^W a x^\top, D_z^W a z^\top \rangle \\ &= \mathrm{trace}(x a^\top D_x^W D_z^W a z^\top) \\ &= \langle x, z \rangle \cdot a^\top D_x^W D_z^W a^\top \\ &= \langle x, z \rangle \cdot \frac{1}{m} \sum_{j=1}^m \phi'(\langle w_j^{(0)}, x \rangle) \phi'(\langle w_j^{(0)}, z \rangle) \\ &= \frac{1}{m} \sum_{j=1}^m \langle x \phi'(\langle w_j^{(0)}, x \rangle), z \phi'(\langle w_j^{(0)}, z \rangle) \rangle\end{aligned}$$

As the width of the network $m \to \infty$, this becomes *deterministic*.

$$\hat{K}(x,z) \overset{m\to\infty}{\longrightarrow} K(x,z) := \mathbb{E}_{w \sim \mathrm{N}(0,I)}[\langle x \phi'(\langle w, x \rangle), z \phi'(\langle w, z \rangle) \rangle].$$

This limiting kernel is the NTK. Extends to different architectures, too.

[Jacot–Gabriel–Hongler'18], [AllenZhu–Li–Song'19], [Arora–Du–Hu–Li–Wang'19], [Du–Zhai–Poczos–Singh'19], [Zou–Cao–Zhou–Gu'19], and many others

Neural Tangent Kernel

Gradients of network at random init serve as (random) feature embeddings. These embeddings determine the random kernel $\hat{K}(x,z) = \langle \nabla f(x; W^{(0)}), \nabla f(z; W^{(0)}) \rangle$, whose ∞-width limit is NTK:

$$K(x,z) = \mathbb{E}_{w \sim \mathrm{N}(0,I)}[\langle x\phi'(\langle w,x\rangle), z\phi'(\langle w,z\rangle)\rangle]$$
$$= \int \langle \boxed{x\phi'(\langle w,x\rangle)}, z\phi'(\langle w,z\rangle)\rangle p_{\mathrm{N}(0,I)}(w)dw.$$

This motivates assumptions like the following [Ji–Telgarsky'20]:

Assumption.

There exists $u = u(w) : \mathbb{R}^d \to \{w \in \mathbb{R}^d : \|w\| \le 1\}$, s.t.

$$\forall i,\ y_i \int \boxed{\langle x_i\phi'(\langle w,x_i\rangle)}, u(w)\rangle p_{\mathrm{N}(0,I)}(w)dw \ge \gamma > 0.$$

In words: the data can be classified well in infinite-dimensional space using the 'tangent features' $\boxed{\nabla\phi(\langle w,x\rangle)}$ using weights $v(w)p(w)dw$.

Optimization with NTK

Assumption.

There exists $u = u(w) : \mathbb{R}^d \to \{z \in \mathbb{R}^d : \|z\| \le 1\}$, s.t.

$$\forall i,\ y_i \int \langle x_i \phi'(\langle w, x_i \rangle), u(w) \rangle p_{\mathrm{N}(0,I)}(w) dw \boxed{\ge \gamma > 0}.$$

For m large and $w_j^{(0)} \overset{\text{i.i.d.}}{\sim} \mathrm{N}(0, I)$ we have by law of large numbers,

$$y_i \cdot \frac{1}{m} \sum_{j=1}^{m} \langle x_i \phi'(\langle w_j^{(0)}, x_i \rangle), u(w_j^{(0)}) \rangle$$

$$\approx y_i \cdot \int \langle x_i \phi'(\langle w, x_i \rangle), u(w) \rangle p_{\mathrm{N}(0,I)}(w) dw \boxed{\ge \gamma}.$$

If we let $v_j := a_j u(w_j^{(0)})$, using $\nabla f(x; W) = \mathrm{Diag}\big(\phi'(\langle w_j, x \rangle)\big) a x^\top$,

$$y_i \langle \nabla f(x_i; W^{(0)}), V \rangle = \frac{1}{m} \sum_{j=1}^{m} y_i \langle x_i \phi'(\langle w_j^{(0)}, x_i \rangle), u(w_j^{(0)}) \rangle \boxed{\gtrsim \gamma}.$$

Optimization with NTK

Lemma.

If there exists $u = u(w) : \mathbb{R}^d \to \{z \in \mathbb{R}^d : \|z\| \le 1\}$, s.t.

$$\forall i,\ y_i \int \langle x_i \phi'(\langle w, x_i \rangle), u(w) \rangle p_{\mathrm{N}(0,I)}(w) dw \ge \gamma > 0,$$

then for m large, there exists $\|V\|_F \le 1$ s.t. for all i,

$$\forall i,\ \boxed{y_i \langle \nabla f(x_i; W^{(0)}), V \rangle \ge \gamma/2}.$$

This gives PL inequality $\boxed{\text{when } \|W^{(t)} - W^{(0)}\| \text{ small}}$ ('NTK regime'):

$$\left\|\nabla \hat{L}(W^{(t)})\right\| \ge \frac{1}{n} \sum_{i=1}^{n} \ell(y_i f(x_i; W^{(t)})) \cdot \boxed{y_i \langle \nabla f(x_i; W^{(t)}), V \rangle}$$

$$\approx \frac{1}{n} \sum_{i=1}^{n} \ell(y_i f(x_i; W^{(t)})) \cdot \boxed{y_i \langle \nabla f(x_i; W^{(0)}), V \rangle}$$

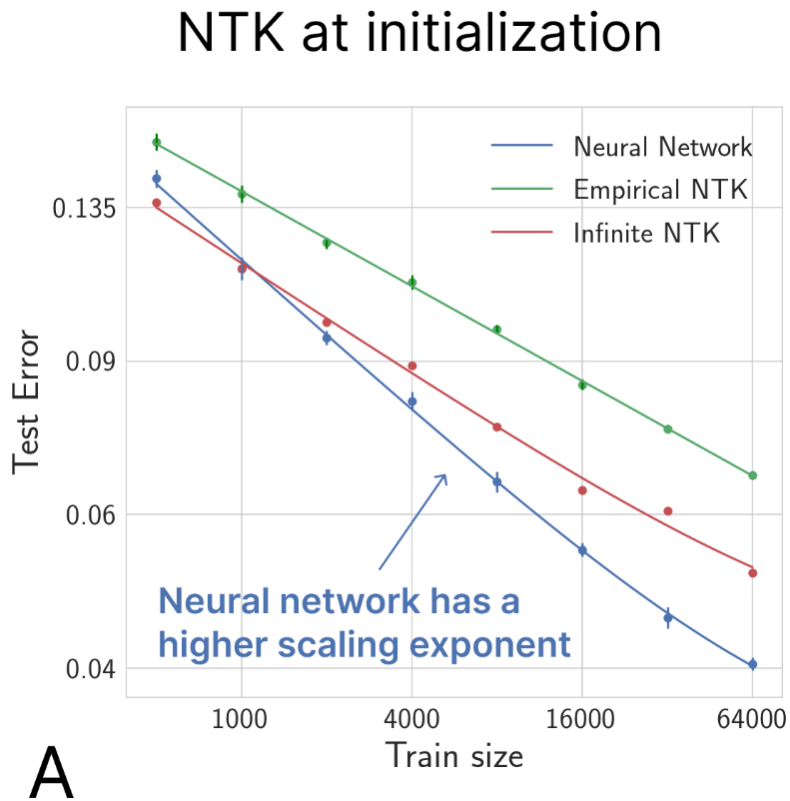
$$\ge \boxed{\frac{\gamma}{2}} \cdot \frac{1}{n} \sum_{i=1}^{n} \ell(y_i f(x_i; W^{(t)})) = \frac{\gamma}{2} \hat{L}(W^{(t)}).$$

Limitations of Neural Tangent Kernel

The NTK approximation relies upon the fact that

$$\nabla f(x_i; W^{(t)}) \approx \nabla f(x_i; W^{(0)}).$$

This ignores the ability for neural networks to learn data-dependent features: the weights $W^{(0)}$ do not incorporate any data-dependent information, which should enable better sample complexity.



NTK seems to be missing a very key part of neural net training in practice.

Sample complexity scaling laws on variant of CIFAR10, taken from [Vyas–Bansal–Nakkiran'22]

Limitations of Neural Tangent Kernel

A number of works provide explicit separations between what is learnable with any kernel method (e.g., NTK) and neural networks in the 'feature learning regime' [Yehudai–Shamir'19; AllenZhu–Li'19; Damian–Lee–Soltanolkotabi'22;...]

For instance, consider learning a single ReLU neuron from i.i.d. samples,

$$x_i \sim \mathsf{Unif}(\{x \in \mathbb{R}^d : \|x\| \le 1\}), \quad y_i = \max(0, \langle w^*, x_i \rangle + b).$$

Then the NTK cannot efficiently ($\mathrm{poly}(d)$-size weights/random features) approximate the labels y [Yehudai–Shamir'19], but neural networks in the 'feature learning' regime can [Mei–Bai–Montanari'18; Soltanolkotabi'17; Yehudai–Shamir'20; F.–Cao–Gu'20; ...].

Optimization without NTK for linearly separable data

Recall the PL inequality idea from before: for $f(x; W)$ neural net, $\ell(z) = \exp(-z)$, and for any V, $\|V\| \le 1$,

$$\left\|\nabla \hat{L}(W^{(t)})\right\| \ge \langle \nabla \hat{L}(W^{(t)}), -V \rangle$$

$$= \frac{1}{n} \sum_{i=1}^{n} \ell(y_i f(x_i; W^{(t)})) \cdot \underbrace{y_i \langle \nabla f(x_i; W^{(t)}), V \rangle}_{(*)\text{: 'gradient margin'}}.$$

Restrict to two-layer nets $f(x; W) = a^\top \phi(Wx)$ with 'leaky' activations, $\phi'(z) \ge \alpha > 0$. Then

$$\nabla f(x_i; W^{(t)}) = \mathrm{Diag}\left(\phi'(\langle w_j^{(t)}, x_i \rangle)\right) a x^\top, \qquad \text{and so}$$

$$y_i \langle \nabla f(x_i; W^{(t)}), V \rangle = \sum_{j=1}^{m} a_j \phi'(\langle w_j^{(t)}, x_i \rangle) \langle y_i x_i, v_j \rangle \overset{(?)}{\ge} c.$$

Optimization without NTK for linearly separable data

Suppose the data is linearly separable, so $\exists u$, $\|u\| \le 1$, s.t. $y_i \langle x_i, u \rangle \ge \gamma$ for all i. Then let V be the matrix with rows

$$v_j = a_j u, \quad \|V\|_F^2 = \sum_{j=1}^{m} a_j^2 \|u\|^2 \le 1,$$

and thus

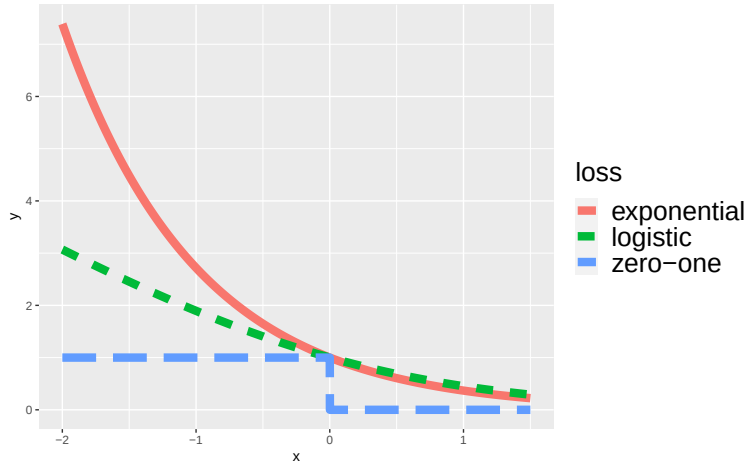
$$\begin{aligned} y_i \langle \nabla f(x_i; W^{(t)}), V \rangle &= \sum_{j=1}^{m} a_j \phi'(\langle w_j^{(t)}, x_i \rangle) \langle y_i x_i, v_j \rangle \\ &= \frac{1}{m} \sum_{j=1}^{m} \phi'(\langle w_j^{(t)}, x_i \rangle) \cdot y_i \langle x_i, u \rangle \qquad (a_j^2 = 1/m) \\ &\ge \alpha \cdot \gamma. \qquad (\phi'(z) \ge \alpha) \end{aligned}$$

This gives,

$$\left\| \nabla \hat{L}(W^{(t)}) \right\| \ge \frac{1}{n} \sum_{i=1}^{n} \ell(y_i f(x_i; W^{(t)})) \cdot y_i \langle \nabla f(x_i; W^{(t)}), V \rangle \ge \alpha\gamma \hat{L}(W^{(t)}).$$

Outline

Generalizations of the PL inequality



For classification problems, we never minimize the 0-1 loss directly, but instead minimize convex, decreasing, nonnegative surrogates of the 0-1 loss: exponential, logistic loss, hinge loss, sigmoid loss, etc. Key reason:

$$\mathbb{P}(yf(x;W) < 0) = \mathbb{P}(\ell(yf(x;W)) > \ell(0)) \leq \frac{\mathbb{E}[\ell(yf(x;W))]}{\ell(0)}.$$

Calculations above relied upon exponential loss satisfying $-\ell'(z) = \ell(z)$. But if ℓ is convex and decreasing, then $-\ell'(z)$ is non-negative and decreasing, so $-\ell'(yf(x;W))$ can *also* serve as a proxy for the 0-1 loss.

Generalizations of the PL inequality

Suppose we have a convex, non-negative, decreasing ℓ, so $-\ell'$ is non-negative and decreasing. Further suppose we have for all t, there is some $V = V^{(t)}$ with $\|V\| \le 1$ and

$$\text{for all } i, \quad \boxed{y_i \langle \nabla f(x_i; W^{(t)}), V \rangle \ge c}.$$

If we use variational definition of norm again,

$$\begin{aligned}
\left\| \nabla \hat{L}(W^{(t)}) \right\| &\ge \langle \nabla \hat{L}(W^{(t)}), -V \rangle \\
&= \frac{1}{n} \sum_{i=1}^{n} -\ell'(y_i f(x_i; W^{(t)})) \cdot \boxed{y_i \langle \nabla f(x_i; W^{(t)}), V \rangle} \\
&\ge \boxed{c} \cdot \frac{1}{n} \sum_{i=1}^{n} -\ell'(y_i f(x_i; W^{(t)})) =: c \hat{G}(W^{(t)}),
\end{aligned}$$

where we have a *proxy* loss $\hat{G}(W^{(t)}) := \frac{1}{n} \sum_{i=1}^{n} -\ell'(y_i f(x_i; W^{(t)}))$, and have thus established a *proxy PL inequality* [F.–Gu'21]. Thus stationary points have small $\hat{G}(W^{(t)})$, and thus small 0-1 loss.

Optimization without NTK?

Understanding optimization without NTK in finite-width neural networks is a (big) open problem, even for two layer nets.

Via PL-inequality, to show optimization it suffices to show that *you can classify points well using the net's gradient as features*:

$$\text{Construct } V = V^{(t)} \text{ s.t.} \quad \forall i,\ y_i \langle \nabla f(x_i; W^{(t)}), V \rangle \geq c.$$

Current approaches generally rely upon considering structured data distributions $\mathcal{D}$ with $\{(x_i, y_i)\} \overset{\text{i.i.d.}}{\sim} \mathcal{D}$, then understanding the types of features that different neurons in the neural net learn. This gives a handle on $\nabla f(x_i; W^{(t)})$.

See, e.g., [AllenZhu–Li'20], [Zou–Cao–Li–Gu'21], [F.–Chatterji–Bartlett'22], [Cao–Belkin–Gu'22], ...

Some questions we haven't discussed yet

- "Implicit bias": there are many neural networks that can achieve zero training error, thanks to overparameterization.
  - Can we differentiate between those minima found by *gradient descent* from others?
  - What does this mean for generalization?
  - How does this depend on architecture?
  - What role do regularizers/optimization parameters play here?
- How do we 'optimally' choose optimization hyperparameters? Step size, initialization variance, and how these depend on width, depth, architecture, ...