Towards a Mathematical Theory of Machine Learning

Weinan E

Center for Machine Learning Research and
School of Mathematical Sciences

Peking University

Joint work with:

- **Chao Ma**, Stanford University
- **Stephan Wojtowytsch**, Texas A and M University
- **Lei Wu**, Peking University

Supported by **iFlytek.**

Machine learning is about solving some standard mathematical problems,

but typically in very high dimension!

Supervised learning:

Approximating a target function f^* using a finite training set

$$S = \{(\boldsymbol{x}_j, y_j = f^*(\boldsymbol{x}_j)), j \in [n] = \{1, 2, \cdots, n\}\}$$

Image classification:

We are interested in the function

$$f^* : \text{image} \rightarrow \text{ its content (category)}$$

We know the values of f^* on a finite sample (the labeled data). The goal is to find accurate approximation of f^*.

Unsupervised learning

Generating non-existing data: Pictures of FAKE human faces

https://arxiv.org/pdf/1710.10196v3.pdf

Unsupervised learning:

Approximating the underlying probability distribution using finite samples.

Generating pictures of fake human faces:

Approximating and sampling an unknown probability distribution.

- Random variable: pictures of human faces
- We don't know its probability distribution
- We do have a finite sample: pictures of real human faces
- We can approximate the unknown probability distribution and produce new samples
- These new samples are pictures of fake human faces.

Reinforcement learning:

Solving the Bellman equation for a *Markov decision process*.

Time series learning:

Approximating dynamical systems.

We have been solving these kinds of problems in (computational) mathematics for a long time!

After all,

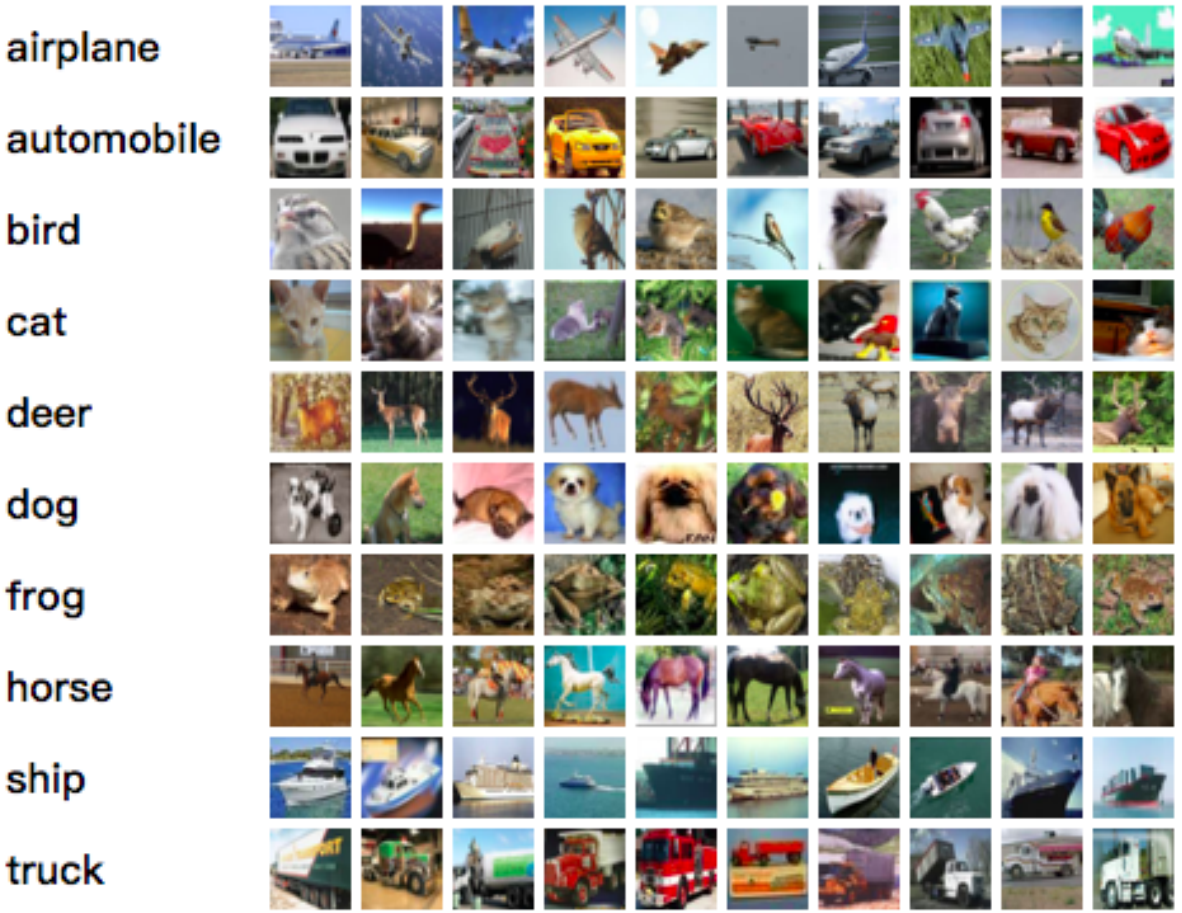
- approximating functions
- approximating and sampling probability distributions
- solving differential and difference equations

are all classical problems in numerical analysis.

So what is really the difference?

Dimensionality!

Dimensionality of the CIFAR-10 problem



Input dimension:

$$d = 32 \times 32 \times 3 = 3072$$

Classical approximation theory

Approximate a function using piecewise linear functions over a mesh of size h:

- $d =$ dimensionality of the problem
- $m =$ the total number of free parameters in the model

$$h \sim m^{-1/d}$$

$$|f^* - f_m| \sim h^2 |\nabla^2 f^*| \sim m^{-2/d} |\nabla^2 f^*|$$

To reduce the error by a factor of 10, we need to increase m by a factor of $10^{d/2}$.

Curse of dimensionality (CoD): As d grows, computational cost grows exponentially fast.

True for all classical algorithms, e.g. approximating functions using polynomials, or wavelets.

Apparently, deep neural networks can do much better in high dimension.

What is the magic?

Will focus on: supervised learning

Knowing the values of f^* on a finite **training dataset**

$$S = \{(\boldsymbol{x}_j, y_j = f^*(\boldsymbol{x}_j)), j \in [n] = \{1, 2, \cdots, n\}\}$$

find accurate approximations of the **target function** f^*.

Our main objective is to:

Minimize the **testing error** ("population risk" or "generalization error"):

$$\mathcal{R}(f) = \mathbb{E}_{\boldsymbol{x}\sim\mu}(f(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2 = \int_X (f(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2 d\mu$$

where μ is the distribution of $\boldsymbol{x}$ (say on a domain $X \subset \mathbb{R}^d$).

To be specific, we will take $X = [0, 1]^d$.

The three main components of the error

The total error: $f^* - \hat{f}$, where $\hat{f}$ = the output of the ML model.

Define:

- $f_m = \text{argmin}_{f \in \mathcal{H}_m} \mathcal{R}(f)$ = best approximation to f^* in $\mathcal{H}_m$
- $\tilde{f}_{n,m}$ = "best approximation to f^* in $\mathcal{H}_m$, using only the dataset S"

Decomposition of the error:

$$f^* - \hat{f} = \underbrace{f^* - f_m}_{\text{appr.}} + \underbrace{f_m - \tilde{f}_{n,m}}_{\text{estim.}} + \underbrace{\tilde{f}_{n,m} - \hat{f}}_{\text{optim.}}$$

- $f^* - f_m$ = *approximation error*, due entirely to the choice of the hypothesis space
- $f_m - \tilde{f}_{n,m}$ = *estimation error* — additional error due to the fact that we only have a finite dataset
- $\tilde{f}_{n,m} - \hat{f}$ = *optimization error* — additional error caused by training

Approximation Error

Benchmark: High dimensional integration

$$I(g) = \int_X g(\boldsymbol{x}) d\mu = \mathbb{E}_{\boldsymbol{x}\sim\mu} g, \quad I_m(g) = \frac{1}{m}\sum_{j=1}^{m} g(\boldsymbol{x}_j)$$

Grid-based quadrature rules (α is some fixed number):

$$I(g) - I_m(g) \sim \frac{C(g)}{m^{\alpha/d}}$$

Curse of dimensionality (CoD)!

Monte Carlo: $\{\boldsymbol{x}_j, j \in [m]\}$ are i.i.d samples of μ

$$\mathbb{E}(I(g) - I_m(g))^2 = \frac{\mathsf{var}(g)}{m}, \quad \mathsf{var}(g) = \mathbb{E}g^2 - (\mathbb{E}g)^2$$

Implications to function approximation

Representation of functions using transforms:

Representing functions using Fourier transform:

$$f^*(\boldsymbol{x}) = \int_{\mathbb{R}^d} \hat{f}(\boldsymbol{\omega}) e^{i(\boldsymbol{\omega},\boldsymbol{x})} d\boldsymbol{\omega}.$$

Approximate using discrete Fourier transform on uniform grids:

$$f_m(\boldsymbol{x}) = \frac{1}{m} \sum_{j=1}^{m} \hat{f}(\boldsymbol{\omega}_j) e^{i(\boldsymbol{\omega}_j,\boldsymbol{x})}$$

The error suffers from CoD:

$$f^* - f_m \sim h^\alpha \sim m^{-\alpha/d}$$

"New" approach: Let π be a probability distribution

$$f^*(\boldsymbol{x}) = \int_{\mathbb{R}^d} a(\boldsymbol{\omega}) e^{i(\boldsymbol{\omega},\boldsymbol{x})} \pi(d\boldsymbol{\omega}) = \mathbb{E}_{\boldsymbol{\omega}\sim\pi} a(\boldsymbol{\omega}) e^{i(\boldsymbol{\omega},\boldsymbol{x})}$$

Let $\{\boldsymbol{\omega}_j\}$ be an **i.i.d. sample of** π, $f_m(\boldsymbol{x}) = \frac{1}{m}\sum_{j=1}^m a(\boldsymbol{\omega}_j) e^{i(\boldsymbol{\omega}_j,\boldsymbol{x})}$,

$$\mathbb{E}|f^*(\boldsymbol{x}) - f_m(\boldsymbol{x})|^2 = \frac{\mathsf{var}(f)}{m}$$

Note: $f_m(\boldsymbol{x}) = \frac{1}{m}\sum_{j=1}^m a_j \sigma(\boldsymbol{\omega}_j^T \boldsymbol{x}) =$ two-layer neural network with $\sigma(z) = e^{iz}$.

Conclusion:
Functions of the this type (i.e. can be expressed as this kind of expectation) can be approximated by two-layer neural networks with a dimension-independent error rate.

Approximation theory for the *random feature model*

- Let $\phi(\cdot; \boldsymbol{w})$ be a feature function parametrized by $\boldsymbol{w} \in \Omega$, e.g. $\phi(\boldsymbol{x}; \boldsymbol{w}) = \sigma(\boldsymbol{w}^T \boldsymbol{x})$.
  We will assume that ϕ is continuous and Ω is compact.
- Let π_0 be a fixed distribution for the random variable $\boldsymbol{w}$.
- Let $\{\boldsymbol{w}_j^0\}_{j=1}^m$ be a set of i.i.d samples drawn from π_0.

The random feature model (RFM) associated with the features $\{\phi(\cdot; \boldsymbol{w}_j^0)\}$ is given by

$$f_m(\boldsymbol{x}; \boldsymbol{a}) = \frac{1}{m} \sum_{j=1}^{m} a_j \phi(\boldsymbol{x}; \boldsymbol{w}_j^0).$$

What spaces of functions are "well approximated" (say with the same convergence rate as in Monte Carlo) **by the random feature model?**

- In classical approximation theory, these are a the Sobolev or Besov spaces: They are characterized by the convergence behavior for some specific approximation schemes.
- Direct and inverse approximation theorems.

Define the kernel function associated with the random feature model:

$$k(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}_{\boldsymbol{w}\sim\pi_0}[\phi(\boldsymbol{x}; \boldsymbol{w})\phi(\boldsymbol{x}'; \boldsymbol{w})]$$

Let $\mathcal{H}_k$ be the reproducing kernel Hilbert space (RKHS) induced by the kernel k.

Probabilistic characterization:
$f \in \mathcal{H}_k$ if and only if there exists $a(\cdot) \in L^2(\pi_0)$ such that

$$f(\boldsymbol{x}) = \int a(\boldsymbol{w})\phi(\boldsymbol{x}; \boldsymbol{w})d\pi_0(\boldsymbol{w}) = \mathbb{E}_{\boldsymbol{w}\sim\pi_0} a(\boldsymbol{w})\phi(\boldsymbol{x}; \boldsymbol{w})$$

and

$$\|f\|_{\mathcal{H}_k}^2 = \int a^2(\boldsymbol{w})d\pi_0(\boldsymbol{w}) = \mathbb{E}_{\boldsymbol{w}\sim\pi_0} a^2(\boldsymbol{w})$$

Direct and inverse approximation theorem

Theorem

For any $\delta \in (0,1)$, with probability $1-\delta$ over the samples $\{\boldsymbol{w}_j^0\}_{j=1}^m$, we have for any $f^* \in \mathcal{H}_k$

$$\inf_{a_1,\ldots,a_m} \|f^* - \frac{1}{m}\sum_{j=1}^m a_j \phi(\cdot;\boldsymbol{w}_j^0)\|_{L^2(\mu)} \lesssim \frac{\|f^*\|_{\mathcal{H}_k}}{\sqrt{m}}(1+\sqrt{\log(1/\delta)}).$$

Theorem

Let $(\boldsymbol{w}_j^0)_{j=0}^\infty$ be a sequence of i.i.d. random samples drawn from π_0. Let f^* be a continuous function on X. Assume that there exist a constant C and a sequence $\{(a_{j,m}), m \in \mathbb{N}^+, j \in [m]\}$ such that $\sup_{j,m}|a_{j,m}| \le C$ and

$$\lim_{m\to\infty} \frac{1}{m}\sum_{j=1}^m a_{j,m}\phi(\boldsymbol{x};\boldsymbol{w}_j^0) = f^*(\boldsymbol{x}),$$

for all $\boldsymbol{x} \in X$. Then with probability 1, $f^* \in \mathcal{H}_k$, and there exists a function $a^* \in L^\infty(\pi)$ such that

$$f^*(\boldsymbol{x}) = \int_\Omega a^*(\boldsymbol{w})\phi(\boldsymbol{x};\boldsymbol{w})d\pi_0(\boldsymbol{w}) = \mathbb{E}_{\boldsymbol{w}\sim\pi_0} a^*(\boldsymbol{w})\phi(\boldsymbol{x};\boldsymbol{w})$$

Moreover, $\|a^*\|_\infty \le C$.

Conclusion:

Roughly speaking, functions that are well approximated by the random feature models are functions which admit the integral representation:

$$f^*(\boldsymbol{x}) = \int_\Omega a^*(\boldsymbol{w})\phi(\boldsymbol{x};\boldsymbol{w})d\pi_0(\boldsymbol{w}) = \mathbb{E}_{\boldsymbol{w}\sim\pi_0} a^*(\boldsymbol{w})\phi(\boldsymbol{x};\boldsymbol{w})$$

$\mathcal{H}_k$ is about the right function space associated with the RFM.

Approximation theory for *two-layer neural networks*

Consider "scaled" two-layer neural networks:

$$f_m(\boldsymbol{x};\theta) = \frac{1}{m}\sum_{j=1}^{m} a_j \sigma(\boldsymbol{w}_j^T \boldsymbol{x}), \quad \sigma(t) = \max(0,t)$$

What class of functions are well-approximated by two-layer neural networks?

Integral representation: Consider functions $f : X = [0,1]^d \mapsto \mathbb{R}$ of the form

$$f(\boldsymbol{x}) = \int_\Omega a\sigma(\boldsymbol{w}^T\boldsymbol{x})\rho(da, d\boldsymbol{w}) = \mathbb{E}_\rho[a\sigma(\boldsymbol{w}^T\boldsymbol{x})], \quad \boldsymbol{x} \in X$$

- $\Omega = \mathbb{R}^1 \times \mathbb{R}^{d+1}$ is the parameter space
- ρ is a probability distribution on Ω

The actual values of the weights are not important. What's important is the probability distribution of the weights.

E, Ma and Wu (2018, 2019), (related work in Barron (1993), Klusowski and Barron (2016), Bach (2017), E and Wojtowytsch (2020))

The Barron space

Definition (Barron space)

Consider function $f : X \mapsto \mathbb{R}$. Define the *"Barron norm"*

$$\|f\|_{\mathcal{B}} := \inf_{\rho \in \Psi_f} \mathbb{E}_\rho |a| \|\boldsymbol{w}\|_1.$$

where $\Psi_f = \{\rho : f(\boldsymbol{x}) = \mathbb{E}_\rho a\sigma(\boldsymbol{w}^T\boldsymbol{x})\}$, the set of possible representations for f.

Define the set of *Barron functions*

$$\mathcal{B} = \{f \in C(X) : \|f\|_{\mathcal{B}} < \infty\}$$

E, Chao Ma, Lei Wu (2019)

Structural theorem

Theorem

Let f be a Barron function. Then $f = \sum_{i=1}^{\infty} f_i$ where $f_i \in C^1(\mathbb{R}^d \setminus V_i)$ where V_i is a k-dimensional affine subspace of $\mathbb{R}^d$ for some $0 \leq k \leq d-1$.

As a consequence, distance functions to curved surfaces are not Barron functions.

- $f_1(\boldsymbol{x}) = \text{dist}(\boldsymbol{x}, \mathbb{S}^{d-1})$, then f_1 is not a Barron function.
- $f_2(\boldsymbol{x}) = \|\boldsymbol{x}\|$, f_2 is a Barron function.

E and Wojtowytsch (2020)

Direct approximation theorem

Theorem

For any $f^* \in \mathcal{B}$, there exists a two-layer network $f_m(\cdot;\theta)$ such that

$$\|f^* - f_m(\cdot;\theta)\|_{L^2(\mu)} \lesssim \frac{\|f^*\|_{\mathcal{B}}}{\sqrt{m}}.$$

Moreover,

$$\|\theta\|_{\mathcal{P}} \lesssim \|f^*\|_{\mathcal{B}}$$

Path norm:

$$\|\theta\|_{\mathcal{P}} = \frac{1}{m}\sum_{k=1}^{m} |a_k| \|\boldsymbol{w}_k\|_1,$$

if $f_m(\boldsymbol{x};\theta) = \frac{1}{m}\sum_{j=1}^{m} a_j \sigma(\boldsymbol{w}_j^T \boldsymbol{x})$

– discrete analog of the Barron norm, but for the parameters.

Inverse approximation theorem

Theorem

Let f^* be a continuous function. Assume there exist a constant C and a sequence of two-layer neural networks $\{f_m\}$ such that

$$\frac{1}{m}\sum_{k=1}^{m}|a_k|\|\boldsymbol{w}_k\|_1 \leq C, m \in \mathbb{N}^+,$$

$$f_m(\boldsymbol{x}) \to f^*(\boldsymbol{x})$$

for all $\boldsymbol{x} \in X$, then $f^* \in \mathcal{B}$, i.e. there exists a probability distribution ρ^* on Ω, such that

$$f^*(\boldsymbol{x}) = \int_\Omega a\sigma(\boldsymbol{w}^T\boldsymbol{x})\rho^*(da, d\boldsymbol{w}) = \mathbb{E}_{\rho^*}a\sigma(\boldsymbol{w}^T\boldsymbol{x})$$

for all $\boldsymbol{x} \in X$ and $\|f^*\|_{\mathcal{B}} \leq C$.

Conclusion: Roughly speaking, functions that are well approximated by two-layer neural networks are functions that admit the above integral representation.
Barron space is the right function space associated with two-layer neural networks.

Extensions:

- Extension to residual neural networks (E, Ma and Wu (2019, 2020)):
  - where in place of the Barron space, we have the "*flow-induced function space*".
- Extension to multi-layer neural networks, but results unsatisfactory.
  - Need a natural way of representing "continuous" multi-layer neural networks as expectations over probability distributions on the parameter space, i.e. the analog of:

$$f(\boldsymbol{x}) = \mathbb{E}_{\rho}[a\sigma(\boldsymbol{w}^T\boldsymbol{x})], \quad \boldsymbol{x} \in X$$

Estimation Error

We are minimizing the **training error**:

$$\hat{\mathcal{R}}_n(f) = \frac{1}{n}\sum_j (f(\boldsymbol{x}_j) - f^*(\boldsymbol{x}_j))^2 = \frac{1}{n}\sum_j \ell(\theta, \boldsymbol{x}_j)$$

But what we are really interested in is to minimize the **testing error**:

$$\mathcal{R}(f) = \int_X (f(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2 d\mu$$

The Runge phenomenon

What happens outside the training dataset?

Example: Polynomial interpolation on equally spaced grid points

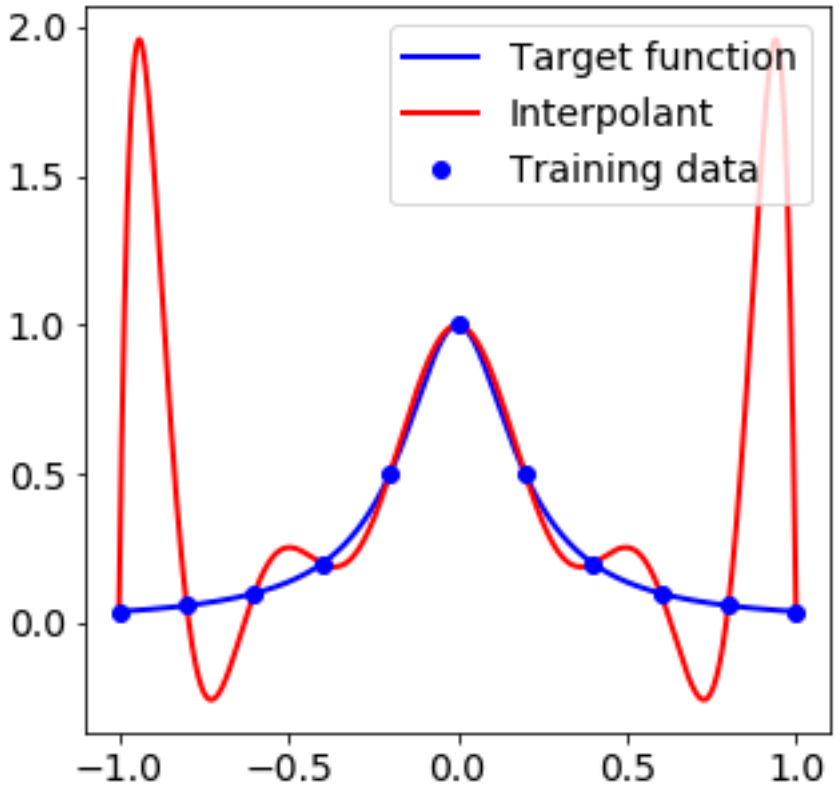


Figure: The Runge phenomenon: $f^*(x) = \frac{1}{1+25x^2}$

Generalization gap = difference between training and testing errors

generalization gap:

$$|\mathcal{R}(\hat{f}) - \hat{\mathcal{R}}_n(\hat{f})| = |\mathbb{E}_{\boldsymbol{x}\sim\mu}\hat{g}(\boldsymbol{x}) - \frac{1}{n}\sum_{j=1}^{n}\hat{g}(\boldsymbol{x}_j)|$$

where $\hat{g}(\boldsymbol{x}) = (\hat{f}(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2$.

Naively, one might expect:

$$\mathbb{E}(\text{generalization gap})^2 = O(1/n)$$

This is not necessarily true since $\hat{f}$ is highly correlated with $\{\boldsymbol{x}_j\}$.

Bounding the generalization gap

Use the naive bound:

$$|\mathcal{R}(\hat{f}) - \hat{\mathcal{R}}_n(\hat{f})| \leq \sup_{f \in \mathcal{H}_m} |\mathcal{R}(f) - \hat{\mathcal{R}}_n(f)|$$

Theorem

Given a function class $\mathcal{H}$, for any $\delta \in (0,1)$, with probability at least $1-\delta$ over the random samples $S = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n)$,

$$\sup_{h \in \mathcal{H}} \left| \mathbb{E}_{\boldsymbol{x}} [h(\boldsymbol{x})] - \frac{1}{n} \sum_{i=1}^{n} h(\boldsymbol{x}_i) \right| \leq 2\widehat{\mathrm{Rad}}_n(\mathcal{H}) + \sup_{h \in \mathcal{H}} \|h\|_\infty \sqrt{\frac{\log(2/\delta)}{2n}}.$$

$$\sup_{h \in \mathcal{H}} \left| \mathbb{E}_{\boldsymbol{x}} [h(\boldsymbol{x})] - \frac{1}{n} \sum_{i=1}^{n} h(\boldsymbol{x}_i) \right| \geq \frac{1}{2}\widehat{\mathrm{Rad}}_n(\mathcal{H}) - \sup_{h \in \mathcal{H}} \|h\|_\infty \sqrt{\frac{\log(2/\delta)}{2n}}.$$

Rademacher complexity of a function space $\mathcal{H}$

The Rademacher complexity of a function space measures its ability to fit random noise on a set of data points.

Definition: Let $\mathcal{H}$ be a set of functions, and $S = (\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n)$ be a set of data points. The Rademacher complexity of $\mathcal{H}$ with respect to S is defined as

$$\widehat{\mathrm{Rad}}_n(\mathcal{H}) = \frac{1}{n}\mathbb{E}_\xi \left[\sup_{h \in \mathcal{H}} \sum_{i=1}^{n} \xi_i h(\boldsymbol{x}_i) \right],$$

where $\{\xi_i\}_{i=1}^n$ are i.i.d. random variables taking values ± 1 with equal probability.

- If $\mathcal{H}$ = unit ball in C^0:

$$\widehat{\mathrm{Rad}}_n(\mathcal{H}) \sim O(1)$$

- If $\mathcal{H}$ = unit ball in Lipschitz space:

$$\widehat{\mathrm{Rad}}_n(\mathcal{H}) \sim O(1/n^{1/d})$$

Another form of CoD! (note that n is the size of the training dataset).

As d grows, the size of the training dataset needed grows exponentially fast.

Rademacher complexity of RKHS

Theorem

Assume that $\sup_{\boldsymbol{x}} k(\boldsymbol{x}, \boldsymbol{x}) \leq 1$. *Let* $\mathcal{H}_k^Q = \{f : \|f\|_{\mathcal{H}_k} \leq Q\}$. *Then,*

$$\widehat{\mathrm{Rad}}_n(\mathcal{H}_k^Q) \leq \frac{Q}{\sqrt{n}}.$$

Rademacher complexity of Barron space

Theorem

Let $\mathcal{F}_Q = \{f \in \mathcal{B}, \|f\|_{\mathcal{B}} \leq Q\}$. Then we have

$$\widehat{\mathrm{Rad}}_n(\mathcal{F}_Q) \leq 2Q\sqrt{\frac{2\ln(2d)}{n}}$$

Neyshbur et al. (2015), Bach (2017)

Generalization error analysis for two-layer neural networks

$$\mathcal{L}_n(\theta) = \hat{\mathcal{R}}_n(\theta) + \lambda\sqrt{\frac{\log(2d)}{n}}\|\theta\|_{\mathcal{P}}, \qquad \hat{\theta}_n = \text{argmin } \mathcal{L}_n(\theta).$$

Theorem

Assume that the target function $f^* : X \mapsto [0,1] \in \mathcal{B}$. There exist constants C_0, such that if $\lambda \geq C_0$, for any $\delta > 0$, then with probability at least $1-\delta$ over the choice of training set, we have

$$\mathcal{R}(\hat{\theta}_n) \lesssim \left(\frac{\|f^*\|_{\mathcal{B}}^2}{m} + \|f^*\|_{\mathcal{B}}\sqrt{\frac{\log(2d)}{n}}\right) + \sqrt{\frac{\log(4C_2/\delta) + \log(n)}{n}}.$$

For Barron functions, not only do good two-layer neural network approximations exist, they can be found using only a finite training dataset (achieves "Monte Carlo error rate").

E, Chao Ma and Lei Wu (2018)

A priori vs. a posteriori estimates

- A priori estimates: RHS depends on the target function, not the size of the parameter

$$\mathcal{R}(\hat{\theta}_n) \lesssim \frac{\|f^*\|_{\mathcal{B}}^2}{m} + \|f^*\|_{\mathcal{B}} \sqrt{\frac{\log(2d)}{n}} + O(n^{-1/2})$$

- A posteriori estimates: The RHS does not depend on the target function but the size of the parameter

$$|\mathcal{R}(\theta) - \hat{\mathcal{R}}_n(\theta)| \lesssim \|\theta\|_* \sqrt{\frac{\log(2d)}{n}} + O(n^{-1/2})$$

  - $\|\theta\|_*$ can be arbitrarily large.
  - The connection to f^* is still missing.

The Training Process

Can we find good solutions efficiently using gradient descent?

$$\min_{\theta} \hat{\mathcal{R}}_n(\theta) = \frac{1}{n} \sum_{j=1}^{n} (f(\boldsymbol{x}_j, \theta) - f^*(\boldsymbol{x}_j))^2$$

is a non-convex function in a high dimensional space.

1. Can gradient descent converge fast?

3rd source of **CoD**: the convergence rate.

2. Does the solution we obtain **generalize well (i.e. have small testing error)**?

Hardness of gradient-based training algorithms

- Let $h(\cdot;\theta)$ be any parametric model such that $Q(\theta) = \mathbb{E}_{\boldsymbol{x}\sim\mu}\|\nabla_\theta h(\boldsymbol{x};\theta)\|_2^2 < \infty$
- Let $\mathcal{R}^f(\theta) = \mathbb{E}_{\boldsymbol{x}\sim\mu}[(h(\boldsymbol{x};\theta) - f(\boldsymbol{x}))^2]$, the loss function associated with f.

Lemma

Let $\mathcal{F} = \{f_1, \ldots, f_M\}$ be an orthonormal class, i.e., $\langle f_i, f_j\rangle_{L^2(\mu)} = \delta_{i,j}$. We have

$$\frac{1}{M}\sum_{i=1}^{M}[\|\nabla\mathcal{R}^{f_i}(\theta) - \overline{\nabla\mathcal{R}^f(\theta)}\|_2^2] \leq \frac{Q(\theta)}{M}.$$

where $\overline{\nabla\mathcal{R}^f(\theta)} = \frac{1}{M}\sum_{j=1}^{M}\nabla\mathcal{R}^{f_j}(\theta)$.

We only have limited ability to distinguish target functions using gradients if there are many orthonormal functions in the function class.

- If $M = \exp(d)$, then the variance of the gradients is exponentially small.
- **The convergence rate for gradient-based training algorithms must suffer from CoD!**

Shamir (2018)

Barron space is such a function class

Lemma

Let $s > 0$ be a fixed number, $\mathcal{B}^s = \{f \in \mathcal{B} : \|f\|_{\mathcal{B}} \lesssim (1+s)^2 d^2\}$. Then $\mathcal{B}^s$ contains at least $(1+s)^d$ orthonormal functions.

Proof:

- Consider the set of orthogonal functions:

$$\mathcal{G}_m = \left\{ \cos(2\pi \mathbf{b}^T \boldsymbol{x}) \ : \ \sum_{i=1}^{d} b_i \leq m, b_i \in \mathbb{N}_+ \right\}.$$

Barron (1993)

Barron space is the right object for approximation theory, but is too big for training.

The right function space for two-layer NNs (error does not suffer from CoD):

- should be in-between the RKHS and Barron space;
- could very well be initialization dependent.

Training two-layer neural networks under conventional scaling

$$f_m(\boldsymbol{x}; \boldsymbol{a}, \boldsymbol{W}) = \sum_{j=1}^{m} a_j \sigma(\boldsymbol{w}_j^T \boldsymbol{x}) = \boldsymbol{a}^T \sigma(\boldsymbol{W}\boldsymbol{x}),$$

Initialization:

$$a_j(0) = 0, \quad \boldsymbol{w}_j(0) \sim \mathcal{N}(0, I/d), \quad j \in [m]$$

Define the associated Gram matrix $K = (K_{ij})$:

$$K_{i,j} = \frac{1}{n} \mathbb{E}_{\boldsymbol{w} \sim \mathcal{N}(0, I/d)} [\sigma(\boldsymbol{w}^T \boldsymbol{x}_i) \sigma(\boldsymbol{w}^T \boldsymbol{x}_j)].$$

The associated random feature model: $\{\boldsymbol{w}_j\} = \{\boldsymbol{w}_j^0\}$ are frozen, only allow $\{a_j\}$ to vary.

Gradient descent dynamics

$$\frac{da_j}{dt}(t) = -\nabla_{a_j}\hat{\mathcal{R}}_n \sim O(\|\boldsymbol{w}_j\|) = O(1)$$
$$\frac{d\boldsymbol{w}_j}{dt}(t) = -\nabla_{\boldsymbol{w}_j}\hat{\mathcal{R}}_n \sim O(|a_j|) = O\left(\frac{1}{\lambda_n m}\right)$$

where $\lambda_n = \lambda_{\min}(K)$

In the "highly over-parametrized regime" (i.e. $m \gg n$), we have time scale separation: the dynamics of $\boldsymbol{w}$ is effectively frozen.

Highly over-parametrized regime

Jacot, Gabriel and Hongler (2018): "**neural tangent kernel**" regime

- **Good news**: exponential convergence (Du et al (2018))
- **Bad news**: converged solution is no better than that of the random feature model (E, Ma, Wu (2019), Arora et al (2019), ...)

Theorem

Denote by $\{f_m(\boldsymbol{x}; \tilde{\boldsymbol{a}}(t), \boldsymbol{W}_0))\}$ *the solution of the gradient descent dynamics for the random feature model. For any* $\delta \in (0,1)$*, assume that* $m \gtrsim n^2 \lambda_n^{-4} \delta^{-1} \ln(n^2 \delta^{-1})$*. Then with probability at least* $1 - 6\delta$*, we have*

$$\hat{\mathcal{R}}_n(\boldsymbol{a}(t), \boldsymbol{W}(t)) \leq e^{-m\lambda_n t} \hat{\mathcal{R}}_n(\boldsymbol{a}(0), \boldsymbol{W}(0))$$

$$\sup_{\boldsymbol{x} \in \mathcal{S}^{d-1}} |f_m(\boldsymbol{x}; \boldsymbol{a}(t), \boldsymbol{W}(t)) - f_m(\boldsymbol{x}; \tilde{\boldsymbol{a}}(t), \boldsymbol{W}_0)| \lesssim \frac{(1 + \sqrt{\ln(\delta^{-1})})^2}{\lambda_n \sqrt{m}}.$$

This is an effectively linear regime.

Mean-field formulation

$$\mathcal{H}_m = \{f_m(\boldsymbol{x}) = \frac{1}{m}\sum_{j=1}^{m} a_j \sigma(\boldsymbol{w}_j^T \boldsymbol{x})\}$$

Let

$$I(\boldsymbol{u}_1, \cdots, \boldsymbol{u}_m) = \hat{\mathcal{R}}_n(f_m), \quad \boldsymbol{u}_j = (a_j, \boldsymbol{w}_j)$$

Lemma:. $\{\boldsymbol{u}_j(\cdot)\}$ is a solution of the gradient descent dynamics

$$\frac{d\boldsymbol{u}_j}{dt} = -\nabla_{\boldsymbol{u}_j} I(\boldsymbol{u}_1, \cdots, \boldsymbol{u}_m), \quad \boldsymbol{u}_j(0) = \boldsymbol{u}_j^0, \quad j \in [m]$$

if and only if

$$\rho_m(d\boldsymbol{u}, \cdot) = \frac{1}{m}\sum_{j=1}^{m} \delta_{\boldsymbol{u}_j(\cdot)}$$

is a solution of

$$\partial_t \rho = \nabla(\rho \nabla V), \quad V = \frac{\delta \hat{\mathcal{R}}_n}{\delta \rho}$$

Chizat and Bach (2018), Mei, Montanari and Nguyen (2018), Rotskoff and Vanden-Eijnden (2018), Sirignano and Spiliopoulos (2018)

This is the gradient flow under the Wasserstein metric

$$\partial_t \rho = \nabla(\rho \nabla V), \quad V = \frac{\delta \hat{\mathcal{R}}_n}{\delta \rho}$$

Long time decay theorem under the condition of **displacement convexity.**

Unfortunately, in general displacement convexity does not hold in the current setting.

Convergence of gradient flow

If the initial condition ρ_0 has full support and if the gradient flow dynamics converges, then it must converge to a global minimizer.

Theorem

Let ρ_t be a solution of the Wasserstein gradient flow such that

- *ρ_0 is a density on the cone $\Theta := \{|a|^2 \leq |\boldsymbol{w}|^2\}$.*
- *Every open cone in Θ has positive measure with respect to ρ_0*

Then the following are equivalent.

1. *The velocity potentials $\frac{\delta \mathcal{R}}{\delta \rho}(\rho_t, \cdot)$ converge to a unique limit as $t \to \infty$.*
2. *$\mathcal{R}(\rho_t)$ decays to minimum Bayes risk as $t \to \infty$.*

If either condition is met, the unique limit is zero. If also ρ_t converges in Wasserstein metric, then the limit ρ_∞ is a minimizer.

Chizat and Bach (2018, 2020), Wojtowytsch (2020)

In the over-parametrized regime, which global minimum gets selected?

Jaim Cooper (2018):

Generically, the set of global minimizers of the empirical risk forms a submanifold of dimension $m - n$.

Idea of proof: Use Sard's theorem.

There are n conditions:

$$f(\boldsymbol{x}_j, \theta) = f^*(\boldsymbol{x}_j), \quad j \in [n]$$

and $\theta \in \mathbb{R}^m$.

The escape phenomenon

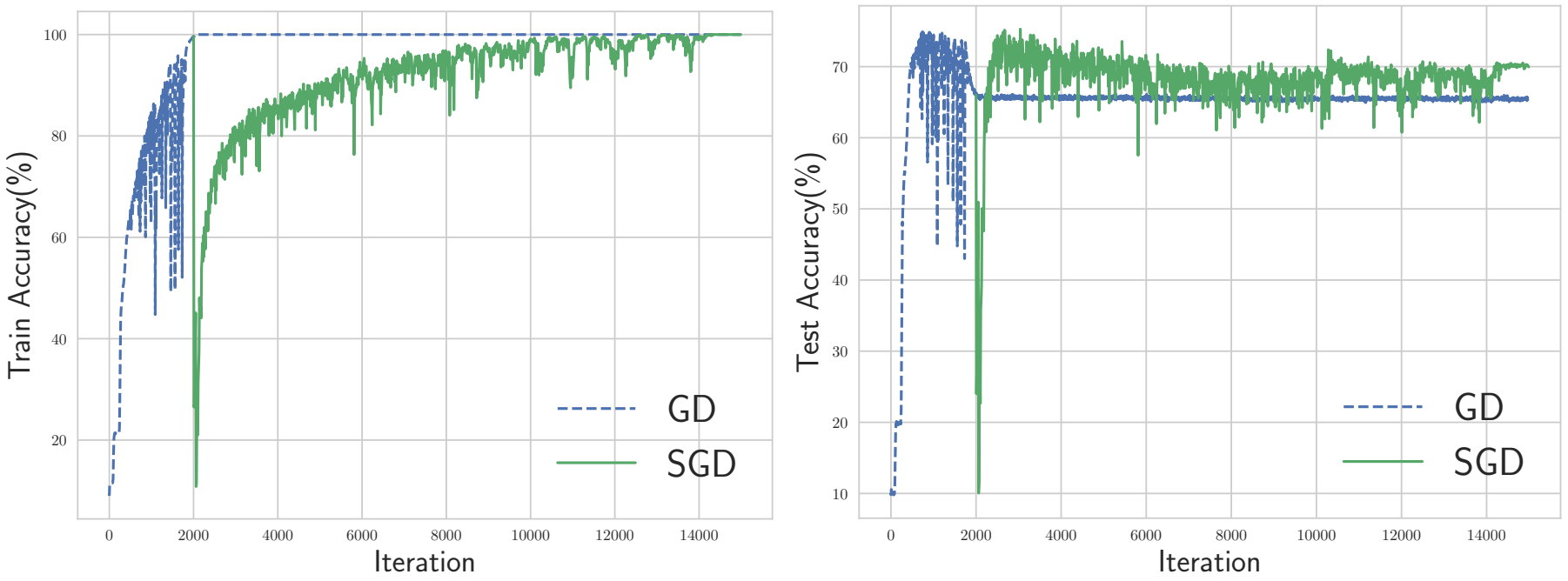


Figure: Fast escape phenomenon in fitting corrupted FashionMNIST.

- This suggests that GD solutions can be **dynamically unstable** for SGD.

Quantifying this phenomenon: Linear stability

- **Linearizing GD:** Let $H(\theta) = \nabla^2 \hat{\mathcal{R}}_n(\theta)$. Linearzing GD around θ^* gives

$$\begin{aligned}\theta_{t+1} - \theta^* &= \theta_t - \theta^* - \eta(\nabla \hat{\mathcal{R}}_n(\theta) - \nabla \hat{\mathcal{R}}_n(\theta^*)) \\ &\approx (I - \eta H(\theta^*))^t (\theta_0 - \theta^*).\end{aligned}$$

- **Stability condition:**

$$\underbrace{\lambda_1(H(\theta^*))}_{\text{flatness}} \leq \frac{2}{\eta}.$$

The edge of stability (EoS)

In practice, GD often settles at the **edge of stability** (EoS) (Wu, Ma and E (NeurIPS 2018)).

η	0.01	0.05	0.1	0.5	1
FashionMNIST	53.5 ± 4.3	39.3 ± 0.5	19.6 ± 0.15	3.9 ± 0.0	1.9 ± 0.0
CIFAR10	198.9 ± 0.6	39.8 ± 0.2	19.8 ± 0.1	3.6 ± 0.4	-
prediction $2/\eta$	200	40	20	4	2

- See Jastrzebski et al. (ICLR 2020); Cohen et al. (ICLR 2021) for some analysis of this striking phenomenon.

Linear stability analysis for SGD

- Over-parametrized: Assume $\hat{\mathcal{R}}_n(\theta^*) = 0$ for the global minimum θ^*.
- Linearizing the SGD dynamics around θ^*:

$$\tilde{\theta}_{t+1} = \tilde{\theta}_t - \frac{\eta}{B} \sum_{j \in I_t} \nabla^2 \ell_j(\theta^*)(\tilde{\theta}_t - \theta^*). \tag{1}$$

and let $H_j = \nabla^2 \ell_j(\theta^*)$.

The one-dimensional case

- The SGD iteration:

$$\theta_{t+1} = (1 - \eta \frac{1}{B} \sum_{j \in I_t} H_j)\theta_t,$$

$$\begin{aligned} \mathbb{E}\theta_{t+1} &= (1 - \eta a)\mathbb{E}\theta_t, \\ \mathbb{E}\theta_{t+1}^2 &= \left[(1 - \eta a)^2 + \frac{\eta^2}{B} s^2\right] \mathbb{E}\theta_t^2, \end{aligned}$$

where

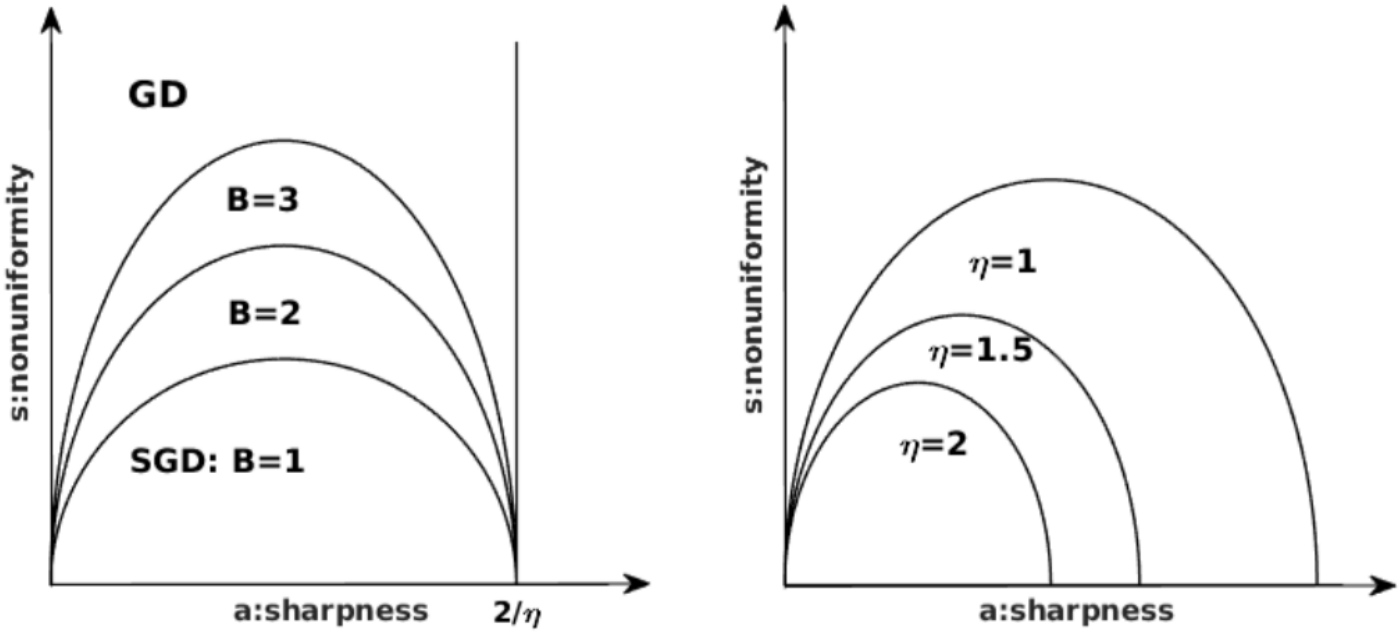
$$a = \frac{1}{n} \sum_{i=1}^{n} H_i, \quad \text{“sharpness”}$$

$$s = \sqrt{\frac{1}{n} \sum_{i=1}^{n} H_i^2 - H^2} \quad \text{“non-uniformity”}$$

- **Stability condition:**

$$(1 - \eta a)^2 + \frac{\eta^2}{B} s^2 \leq 1.$$

The stability digram



- The learning rate and batch size play different roles in the global minima selection.
- Compared with GD, SGD prefers more uniform solutions.

Extension to high dimensions

- Similar analyses can be extended for high-dimensional cases

$$\lambda_{\max}\left\{(I-\eta H)^2+\frac{\eta^2}{B}\Sigma\right\}\leq 1,$$

where

$$H=\frac{1}{n}\sum_i H_i,\quad \Sigma=\frac{1}{n}H_i^2-H^2$$

- Simplification: Let

$$a=\lambda_{\max}(H),\quad s^2=\lambda_{\max}(\Sigma)$$

then a necessary condition is

$$0\leq a\leq\frac{2}{\eta},\quad 0\leq s\leq\frac{\sqrt{B}}{\eta}.$$

Experiments

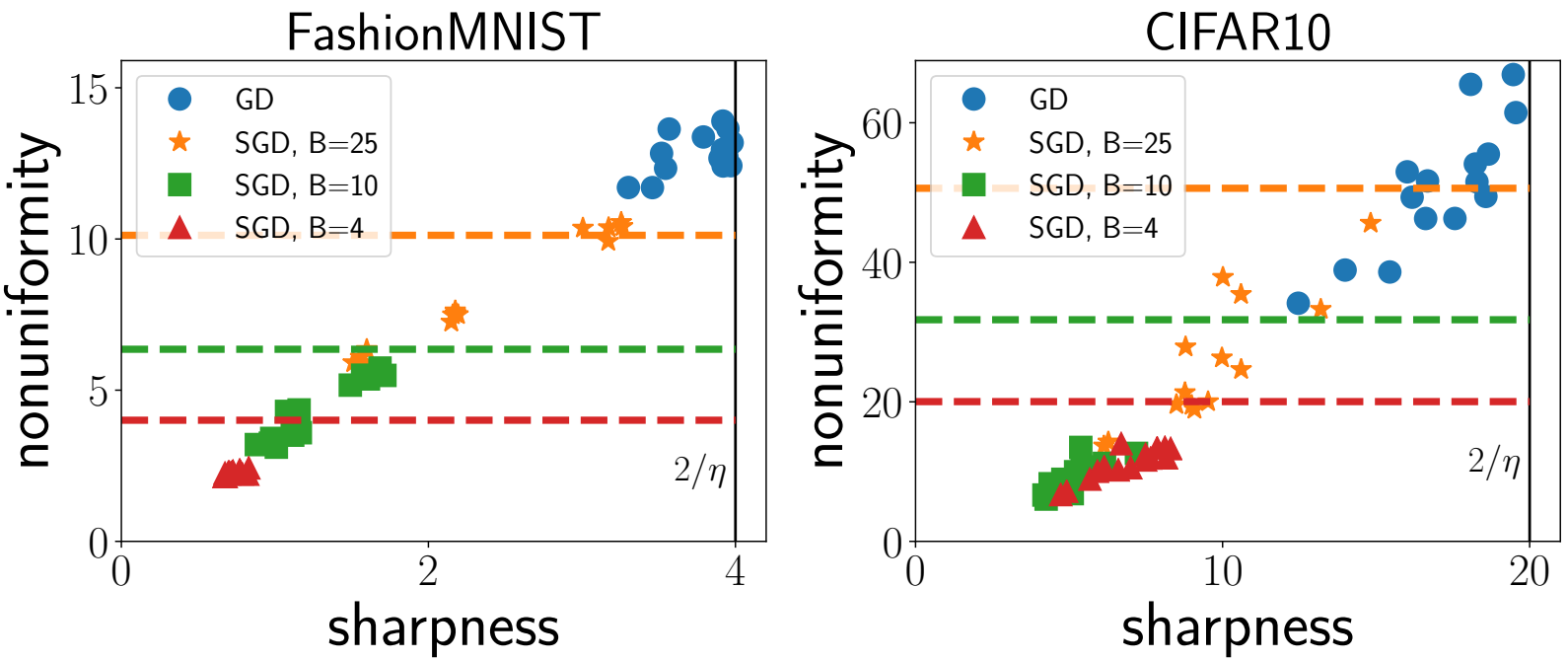


Figure: The sharpness-non-uniformity diagram for the minima selected by SGD. Different colors denote solutions found by SGD with different batch sizes.

- SGD favors more uniform solutions.
- Non-uniformity is nearly proportional to the sharpness.
- SGD favors flatter minima.

Flat minima hypothesis

SGD converges to flatter solutions and flatter solutions generalize better.

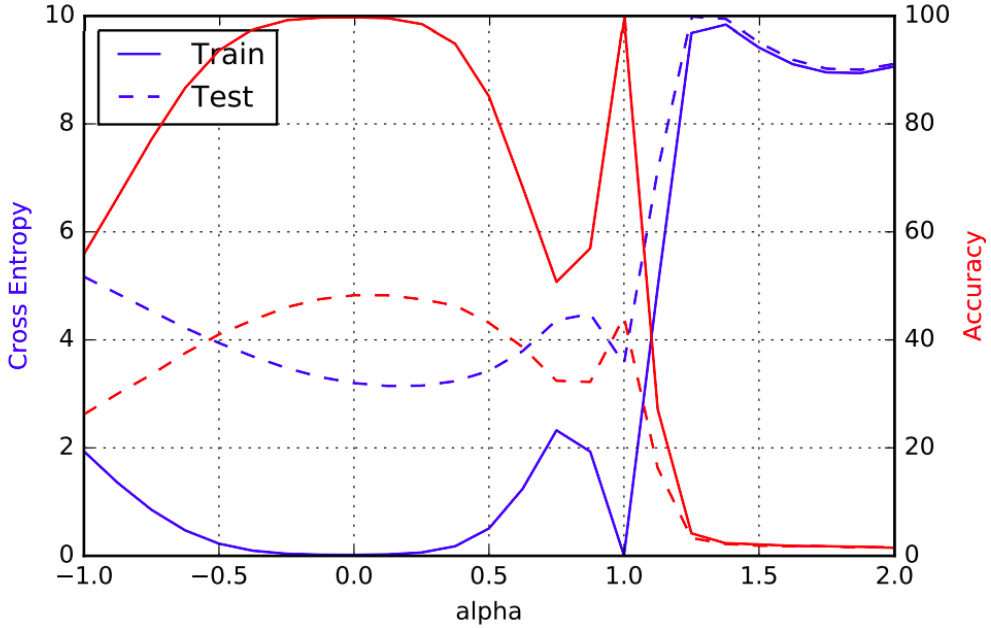
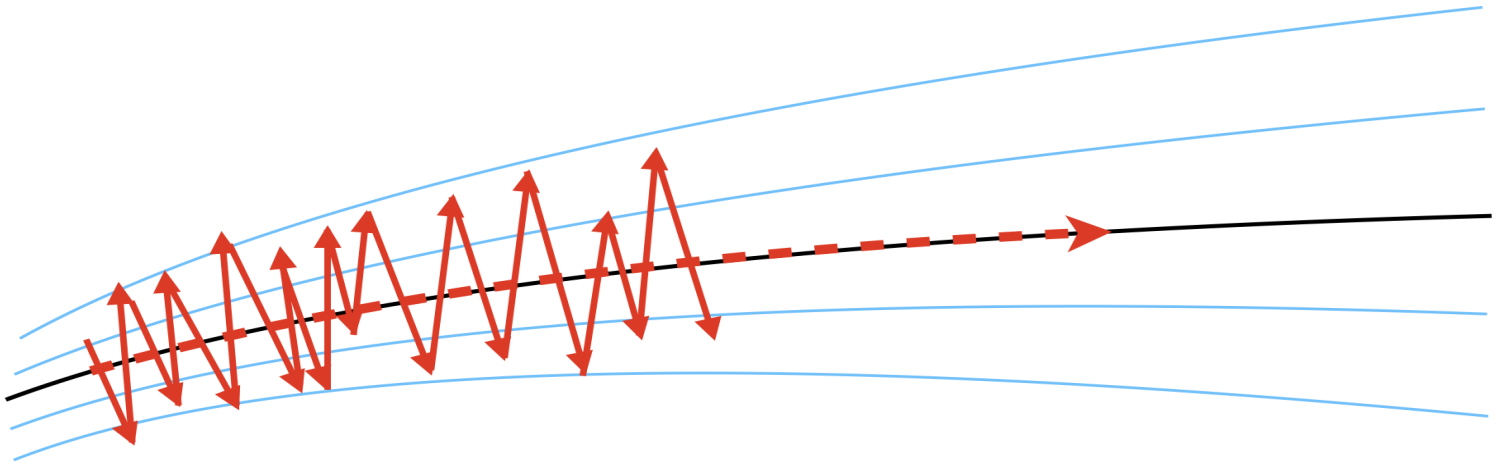


Figure: The landscape for for $\theta(\alpha) := (1-\alpha)\theta_{SGD} + \alpha\theta_{GD}$.

Hochreiter and Schmidhuber, 1995; Keskar et al., 2016

Exploring the global minima manifold

- For over-parameterized models, global minima form a submanifold.
- SGD oscillates around the manifold, bouncing back and forth inside the valley .
- Claim: it moves slowly towards flatter minimum on the manifold.



Chao Ma et al. (2022)

Effective dynamics close to the minima manifold

- Consider the SDE approximating SGD:

$$d\boldsymbol{x}_t = -\nabla f(\boldsymbol{x}_t)dt + \sqrt{\eta}D(\boldsymbol{x}_t)d\mathbf{W}_t.$$

- For a simple example, let $f(x, y) = h(x)y^2$, $h(x) > 0$. The global minima manifold is given by: $\{y = 0\}$.
  Assume the noise covariance is proportional to the Hessian on the minima manifold:

$$D^2(\boldsymbol{x}) = \frac{\sigma^2}{2}\nabla^2 f(x, 0) = \begin{bmatrix} 0 & 0 \\ 0 & \sigma^2 h(x) \end{bmatrix}.$$

- The SDE can be written as

$$\begin{aligned} dx_t &= -h'(x_t)y_t^2 dt \\ dy_t &= 2h(x_t)y_t dt + \sqrt{\eta h(x_t)}\sigma dW_t. \end{aligned}$$

Effective dynamics close to the minima manifold

The original dynamics:

$$dx_t = -h'(x_t)y_t^2 dt$$
$$dy_t = 2h(x_t)y_t dt + \sqrt{\eta h(x_t)}\sigma dW_t.$$

Close to the minima manifold, y_t is small. Hence, the x-dynamics is much slower than the y-dynamics.

- Quasi-static analysis: Assumes y_t is close to the equilibrium given x_t:

$$dx_t = -\mathbb{E}_y h'(x_t)y_{t,\infty}^2 dt$$
$$dy_{t,\tau} = 2h(x_t)y_{t,\tau} d\tau + \sqrt{\eta h(x_t)}\sigma dW_\tau.$$

- The local equilibrium for y is given by $y_{t,\infty} \sim \mathcal{N}(0, \frac{\eta\sigma^2}{4})$. Hence we have

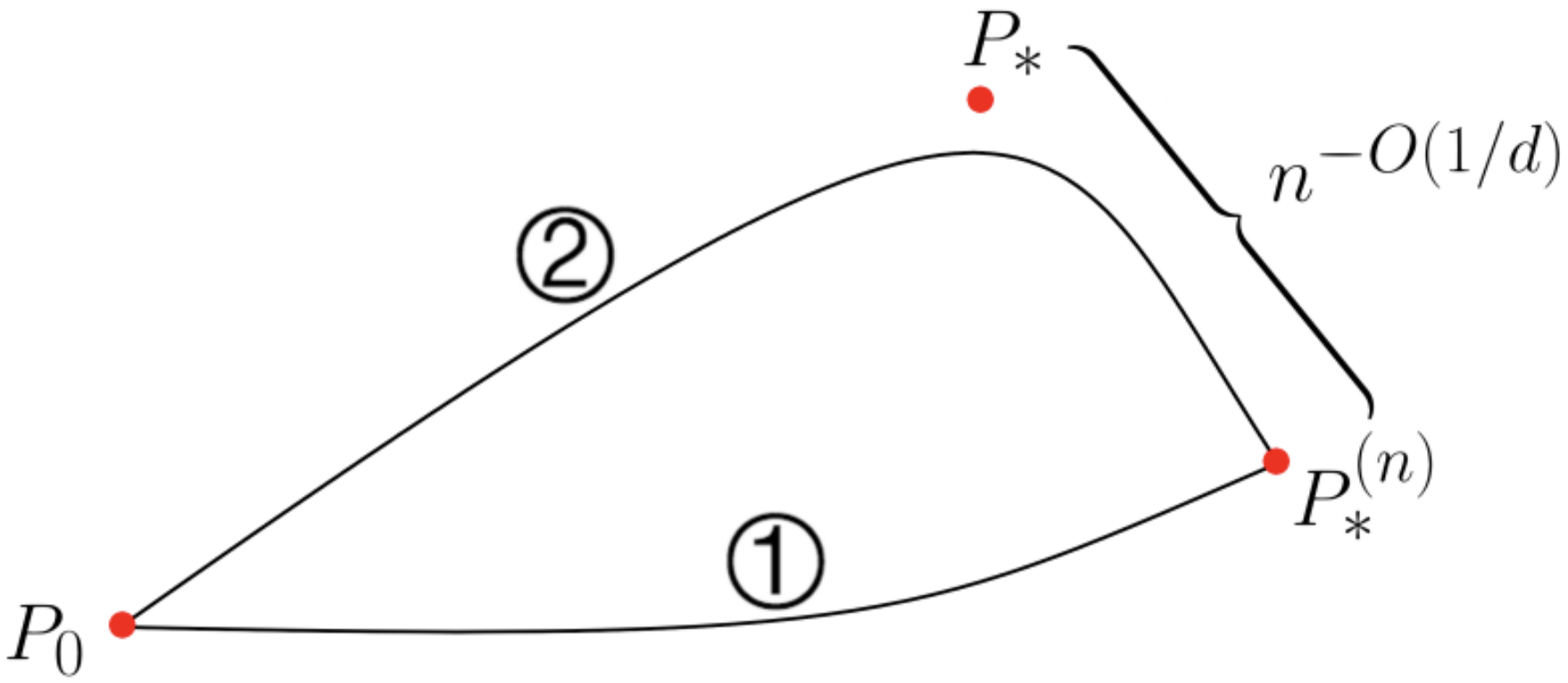
$$\frac{dx_t}{dt} = -\frac{\eta\sigma^2 h'(x_t)}{4}.$$

- This is a gradient flow that minimizes h (the flatness)!

Unsupervised learning: Approximating probability distributions

The memorization phenomenon: The training process ultimately converges to the empirical distribution $P_*^{(n)}$.

Can early stopping give us approximations whose error rate does not suffer from CoD?



Hongkang Yang and E (2020, 2021)

The Curse of Memory in Approximation by RNNs

Theorem

Let $\{H_t^*\}_{t\in\mathbb{R}}$ be a family of continuous, linear, causal, regular and time-homogeneous target functionals. Suppose there exist constants $\alpha \in \mathbb{N}_+, \beta, \gamma > 0$ such that $y_i(t) := H_t^*(e_i) \in C^{(\alpha+1)}(\mathbb{R})$, where $e_i(s) := e_i \mathbf{1}_{\{s\geq 0\}}$ with $\{e_i\}_{i=1}^d$ as standard basis vectors in $\mathbb{R}^d$, and $e^{\beta t} y_i^{(k)}(t) = o(1)$ as $t \to +\infty$, with $\sup_{t\geq 0} \frac{|e^{\beta t} y_i^{(k)}(t)|}{\beta^k} \leq \gamma$, $i = 1, \ldots, d$, $k = 1, \ldots, \alpha+1$. Then there exists a universal constant $C(\alpha) > 0$ only depending on α, such that for any $m \in \mathbb{N}_+$, there exists a sequence of width-m RNN functionals $\{\hat{H}_t\}_{t\in\mathbb{R}}$ such that

$$\sup_{t\in\mathbb{R}} \|H_t^* - \hat{H}_t\| \leq \frac{C(\alpha)\gamma d}{\beta m^\alpha}.$$

Curse of memory: $H_t^*(e_1) \sim t^{-\omega} \Rightarrow m \sim \mathcal{O}(\omega \epsilon^{-\frac{1}{\omega}})$.

Zhong Li, Jiequn Han, Weinan E, Qianxiao Li (2020)

Reinforcement learning

Existing work focuses on the "classical" situation when the state and action spaces are finite (and small).

Almost no work for the situation when the state/action spaces are big or high dimensional, in which case we must use function approximation.

Difficulty: Reinforcement learning involves all the aspects discussed so far:

- function approximation
- learning dynamical systems
- learning probability distributions
- generalization gap
- training is done online and can not be decoupled

Summary

- The central theme is about understanding high dimensional functions.
- A reasonable mathematical picture is taking shape.
  - approximation theory in high dimension
  - global minimum selection and late stage training
- Theorems vs. insight.
  - carefully designed numerical experiments
  - asymptotic analysis