Lecture Notes of Advanced Machine Learning (CS 7140): Statistical Learning and Its Applications in Modern ML/AI

Hongyang R. Zhang

March 12, 2025

Contents

1 Overview

1.1 What is this course about? (Lecture 1)

Machine learning has been increased used in technology platforms and products, affecting our daily lives.[1] Machine learning involves a collection of models, algorithms, and engineering frameworks:

- Regression and classification: least squares estimation, logistic regression, LDA, bias-variance tradeoff, cross-validation.
- Neural networks and deep learning: CNNs, backpropagation, foundation models, language modeling.
- Unsupervised learning: dimension reduction (e.g., PCA), clustering, contrastive learning.
- Causal machine learning: study the cause-and-effect with a powerful machine learning model.
- Generative AI: diffusion models, multi-modal learning.
- NumPy, Sklearn, PyTorch, TensorFlow, Hugging Face.

This course aims to uncover the common statistical principles underlying the diverse array of methods. This class is mostly about the theoretical analysis of learning algorithms and models. Many of the techniques introduced in this course—which involve a beautiful blend of probability, linear algebra, and optimization—are separate fields in their respective discipline with independent interests outside of machine learning. For example, we will study the supreme of a complex random variable corresponding to the outcome of a learning algorithm applied to train a neural network model. We will show how to design estimation algorithms when we are working under distribution shifts between training and test datasets.

From a practical point of view, studying the underlying working mechanisms of a learning algorithm can deepen our understanding of how things work. For example, suppose we want to build a classifier to predict the topic of a document (e.g., sports, politics, technology, etc). We train a logistic regression model with word frequencies as features and obtain a training accuracy of 90% on 1000 training documents and a test accuracy of 85% on 1000 test documents.

- How reliable are these numbers? If we resample the training data, can we expect the same results?
- How much will the training and test accuracies increase if we double the number of training documents?
- What if we increase the number of features (e.g., use tri-occurrence of words)? Does regularization help?

[1] ChatGPT reportedly has 300 million weekly active users: CNCB news, 2024; Claude reportedly has 4.5 million monthly active users, anthropic.

There is obviously a clear gap between theoretical analysis and the practical performance of an algorithm. For instance, theoretical analysis is usually conducted under strong assumptions, which limit the implications one could draw from the theoretical results. The goal, instead, is to build a deeper understanding through theoretical analysis.

The course materials are divided into three parts: *fundamental concepts of statistical learning* (January), *generalization of neural networks and deep learning* (February), *statistical modeling of representation learning, reinforcement learning, and beyond* (March).[2]

1.2 Basic setup of supervised learning (Lecture 1)

Central questions: *Does minimizing training error lead to low test error? How does the generalization ability depend on the model architecture and the training algorithm?* It turns out that answering these questions is highly non-trivial as it also depends on the underlying data distribution.[3]

To formally study these questions, let us first describe the mathematical setup:

- Let $\mathcal{X}$ denote the feature space. Let $\mathcal{Y}$ denote the space of all possible outcomes. Binary classification example: $\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \{+1, -1\}$
- Consider the problem of predicting an output $y \in \mathcal{Y}$ given an input $x \in \mathcal{X}$.
- Let $\mathcal{H}$ be a set of hypotheses. Linear model example:

$$\mathcal{H} = \left\{ x \to \beta^\top x + \eta : \forall \beta \in \mathbb{R}^d, \eta \in \mathbb{R} \right\}$$

- Let $\ell : (\mathcal{X}, \mathcal{Y}) \times \mathcal{H} \to \mathbb{R}$ be a loss function. For example, the mean squared error (MSE) applied to linear models is

$$\ell((x, y), \beta) = \left(\beta^\top x + \eta - y \right)^2, \forall \beta \in \mathbb{R}^d, \forall \eta \in \mathbb{R}$$

- Given n training data samples, denoted by $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$, the training loss (or empirical risk) of a hypothesis $h \in \mathcal{H}$ is defined as

$$\hat{L}(h) = \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), y_i), \forall h \in \mathcal{H} \tag{1}$$

We make a critical assumption about the data-generating process. We assume that every x_i, y_i pair is drawn independently and identically from an unknown distribution $\mathbb{P}^\star$, supported on $\mathcal{X} \times \mathcal{Y}$.

The test loss (or expected risk) of a hypothesis $h \in \mathcal{H}$ is then given by

$$L(h) = \mathop{\mathbb{E}}_{(x,y)\sim\mathbb{P}^\star} \left[\ell(h(x), y) \right]. \tag{2}$$

[2] April will be dedicated to course project presentations.

[3] A recent empirical study highlights empirical scaling laws as key metrics for training large language models: paper (see also openai page).

Remarks:

- We have assumed the training and test distributions are the same. While this assumption does not hold exactly in practice, morally speaking, the training and test distributions have to be related.
- Formulating what it means to be related and not related, and dealing with the discrepancy between training and test data is studied under the area of domain adaptation or transfer learning.
- The independence assumption, which also does not hold exactly in practice, ensures that more training data gives us more information.

Empirical risk minimization: Consider minimizing the training loss

$$\hat{h}_{\text{ERM}} \leftarrow \operatorname*{argmin}_{h \in \mathcal{H}} \hat{L}(h). \tag{3}$$

What can say that the relationship between $\hat{L}(\hat{h}_{\text{ERM}})$ and $L(\hat{h}_{\text{ERM}})$? A key challenge is that the randomness of $\hat{h}_{\text{ERM}}$ now depends on $\hat{L}$. Thus, $\hat{L}(\hat{h}_{\text{ERM}})$ involves a correlation between the training data samples and the minimizing hypothesis. A central aspect we will tackle in the first part of the course is to develop the machinery to tackle this challenge.

Uniform convergence: We show provide statements of the following flavor

> With probability at least $1-\delta$, the gap between test loss and training loss of any hypothesis is upper bounded by some small ϵ, that is, $L(h) - \hat{L}(h) \leq \epsilon$, where the ϵ is generally a function that depends on δ and other aspects of the learning algorithm/model

More rigorously, we would like to show statements of the following:

$$\Pr\left[L(h) - \hat{L}(h) > \epsilon\right] \leq 1 - \delta, \tag{4}$$

where the randomness is on the training data samples drawn from $\mathbb{P}^\star$.

Equipped with such a statement, we will then apply the statement to the empirical risk minimizer $\hat{h}_{\text{ERM}}$, since the result essentially holds for any $h \in \mathcal{H}$, which also subsumes $\hat{h}_{\text{ERM}}$ as a special case.

1.3 Basic setup of neural networks (Lecture 1)

Consider the case of a basic one-layer network:

$$x \to \sum_{i=1}^{m} a_i \sigma(w_i^\top x + b_i), \text{ where} \tag{5}$$

- σ is the nonlinear activation function. Typical choices of σ: ReLU $x \to \max(0, x)$, sigmoid $x \to \frac{1}{1+\exp(-x)}$
- $Z = \{\alpha = (a_i, w_i, b_i)\}_{i=1}^m$ are trainable parameters of the network. By varying them we could define the function class $\mathcal{H}$ as

$$\mathcal{H} = \{f_\alpha : \forall \alpha \in Z\}$$

- $\mathcal{H}$ essentially represents a set of one-hidden-layer neural networks with m neurons
- We can define the weight matrix $W = [w_1, w_2, \ldots, w_m]$, and the bias vector $b = [b_1, b_2, \ldots, b_m]$. Thus, we may write map (5) as $x \to a^\top \sigma(Wx + b)$.

By extending the above setup, we may write a deep network as

$$f_\alpha(x) = \sigma_L \left(W_L \sigma_{L-1} \left(\cdots \sigma_2 \left(W_2 \sigma_1 \left(W_1 x + b_1 \right) + b_2 \right) \cdots \right) \right) \tag{6}$$

The depth of the network is given by L. The width is given by $\max(m_1, m_2, \ldots, m_l)$, i.e., the layer with the most neurons in the layer.

Motivating questions: *How could we analyze the training and test losses of a deep network? How well does a deep network generalize, and how does it depend on its depth and width? How does this ability to learn and to generalize rely on the data distributions, and what is the role of optimization algorithms used to train the network?*

Next lecture: In the next lecture, we will wrap up this overview by giving a setup about how to rigorously model transfer learning, and reason about estimation procedures whose test data is different from the training data. Then, we will dive deeper into the uniform convergence framework.

1.4 Statistical transfer learning (Lecture 2)

An important learning paradigm that has emerged in the past few years is transfer learning—transferring the knowledge from one task to help solve another task. How could we develop a more rigorous statistical modeling of transfer learning? A better understanding of this question has applications in NLP and language modeling, CV, robotics, to name a few.

Linear regression: Perhaps the simplest modeling framework is to examine transfer learning in linear regression tasks. For example, we may consider the case of two linear regression tasks, one called the source task and the other called the target task.

Suppose we have n_1 samples from the source task. We have n_2 samples from the target task. How could we use the samples from the source task to help estimate the target task? Concretely, let the samples of the source task be denoted by $(x_1^{(1)}, y_1^{(1)}), (x_2^{(1)}, y_2^{(1)}), \ldots, (x_{n1}^{(1)}, y_{n1}^{(1)})$, where every $x_i^{(1)}$ is a p-dimensional vector and $y_i^{(1)}$ is a real-valued outcome. We shall assume that they follow a linear relation specified by an unknown parameter $\beta^{(1)} \in \mathbb{R}^p$:

$$y_i^{(1)} = {x_i^{(1)}}^\top \beta^{(1)} + \epsilon_i^{(1)}, \text{ for all } i = 1, 2, \ldots, n_1 \tag{7}$$

Similarly, we denote the samples of the target task as $(x_1^{(2)}, y_1^{(2)}), (x_2^{(2)}, y_2^{(2)}), \ldots, (x_{n2}^{(2)}, y_{n2}^{(2)})$. In addition, they follow a linear relation specified by another unknown parameter $\beta^{(2)} \in \mathbb{R}^p$, which can be different from that of the source task:

$$y_i^{(2)} = {x_i^{(2)}}^\top \beta^{(2)} + \epsilon_i^{(2)}, \text{ for all } i = 1, 2, \ldots, n_2 \tag{8}$$

Now we can ask a few more concrete questions:

- How does the difference between $\beta^{(1)}$ and $\beta^{(2)}$ affect transfer learning performance?
- How does the difference between the feature vectors of source task and target task affect transfer learning?

More generally, we may say that the source task and the target task involve a distribution shift between their them. In the area of domain adaptation (Kouw and Loog, 2018):

- Covariate shift refers to scenarios where both tasks follow the same model conditioned on the features (i.e., $\beta^{(1)} = \beta^{(2)}$), but they have different feature distributions.
- Model shift refers to scenarios where the two tasks follow different models conditioned on the same features, that is $\beta^{(1)} \neq \beta^{(2)}$.

We may now ask, how does covariate shift and model shift affect transfer learning performance?

Transfer learning estimator: Typically, there are two strategies for transfer learning, one called hard transfer, where we hard-code the shared component across tasks, the other called soft transfer, where we use separate components for task, and encourage the separate components to be close to each other (Ruder, 2017).

In the context of linear regression, we can define an hard parameter sharing estimator as follows:

$$\hat{L}^{HPS}(\beta) = \frac{1}{n_1 + n_2} \left(\sum_{i=1}^{n_1} \left({x_i^{(1)}}^\top \beta - y_i^{(1)} \right)^2 + \sum_{j=1}^{n_2} \left({x_j^{(2)}}^\top \beta - y_j^{(2)} \right)^2 \right) \tag{9}$$

We may also elect to use a soft parameter sharing estimator instead:

$$\hat{L}^{SPS}(\beta, z) = \frac{1}{n_1 + n_2} \left(\sum_{i=1}^{n_1} \left({x_i^{(1)}}^\top (\beta + z) - y_i^{(1)} \right)^2 + \sum_{j=1}^{n_2} \left({x_j^{(2)}}^\top \beta - y_j^{(2)} \right)^2 \right) + \lambda \left\| z \right\|^2 \tag{10}$$

Essentially, by adjusting λ, we can adjust the magnitude of z, which then determines how far (and how close) the source and target task models are.

Optimality of the estimator: The above estimation algorithms are based on the best practices of practitioner (see the surveys above). Suppose we analyze their performances. However, how can we know that there are no better estimators out there? How could we understand the fundamental limits of estimation and optimization procedures? These are often called *lower bounds* on the performance of estimators, and it usually falls into the area of information theory (Duchi, 2019). In particular, we will touch on the framework of minimax lower bounds for transfer learning (though the scope of this is much broader than we'll cover in our lectures).

2 Uniform convergence and generalization

Recall that we have introduced the empirical risk and the expected risk of a hypothesis (denoted by $L(h)$ and $\hat{L}(h)$) for some h in a hypothesis class $\mathcal{H}$. Suppose we minimize the empirical risk to get $\hat{h}_{\text{ERM}}$. Two questions:

- Generalization gap: how does the expected and empirical risks compare for ERM, i.e., $L(\hat{h}_{\text{ERM}}) - \hat{L}(\hat{h}_{\text{ERM}})$? This is called the generalization gap.
- Excess risk: how well does ERM do with respect to the best possible hypothesis in the hypothesis class, i.e., $L(\hat{h}_{\text{ERM}}) - \min_{h \in \mathcal{H}} L(h)$? This is also called the excess risk.

A particularly fruitful framework for analyzing learning algorithms is the probably approximately correct (PAC) framework (Valiant, 1984):

> A learning algorithm A PAC learns a hypothesis class $\mathcal{H}$ if
>
> a) For any distribution $\mathbb{P}^\star$ supported over $\mathcal{X} \times \mathcal{Y}$, and any $\epsilon > 0$, $\delta > 0$
>
> b) Upon taking n I.I.D. samples from $\mathbb{P}^\star$, A produces an output $\hat{h} \in \mathcal{H}$ such that with probability at least $1 - \delta$ (over the randomness of the samples)
>
> $$L(\hat{h}) - \hat{L}(\hat{h}) \leq \epsilon$$
>
> c) Further, A runs in time polynomial in $n, d, \epsilon^{-1}, \delta^{-1}$ (where d is the dimension of the input)

Remark: Notice that the running time complexity places a bound on the sample complexity as well. We will assume that the empirical risk minimizer can be computed efficiently. For instance, think of a large neural network whose training loss can be efficiently reduced to reach zero using stochastic gradient descent

2.1 Learning a realizable, finite hypothesis class (Lecture 2)

The ERM framework is very general – we now give a concrete example to illustrate some basic results.

Assumptions (realizable, finite hypothesis): i) The size of the hypothesis space, $\mathcal{H}$, is finite; ii) There exists a hypothesis $h^\star \in \mathcal{H}$ such that $h^\star$ achieves perfect performance, i.e.,

$$L(h^\star) = \mathop{\mathbb{E}}_{(x,y) \in \mathbb{P}^\star} [\ell(h^\star(x), y)] = 0.$$

Under these assumptions, we shall prove the following property of ERM:

Under the above assumptions, with probability $1-\delta$,

$$L(\hat{h}_{\text{ERM}}) \leq \frac{\log(|\mathcal{H}|) + \log(\delta^{-1})}{n} \tag{11}$$

In particular, to reduce the expected risk below ϵ, we want $n \geq \frac{\log(|\mathcal{H}|)+\log(\delta^{-1})}{\epsilon}$. Remarks:

- The excess risk only grows logarithmically with the size of the hypothesis class, so we affort to use an exponentially large hypothesis space.
- The result is independent of $\mathbb{P}^\star$. This is nice because typically we don't know the true distribution.

Proof. We'd like to upper bound the probability of the bad event that $L(\hat{h}_{\text{ERM}}) > \epsilon$:

a) Let $B \subseteq \mathcal{H}$ be the set of bad hypotheses: $\{B \in \mathcal{H} : L(h) > \epsilon\}$

b) We can rewrite our goal as upper bounding the probability of selecting a bad hypothesis $\Pr[L(\hat{h}_{\text{ERM}}) > \epsilon] = \Pr[\hat{h}_{\text{ERM}} \in B]$

c) Recall the empirical risk of ERM is always zero because at least $\hat{L}(h^\star) = 0$

d) Hence for any "bad" hypothesis in B, they must have zero empirical risk

$$\Pr[\hat{h}_{\text{ERM}} \in B] \leq \Pr[\exists h \in B : \hat{L}(h) = 0]$$

e) Now we shall deal with the above in two steps. First, bound $\Pr[\hat{L}(h) = 0]$ for a fixed $h \in B$.

Notice that on a random example from $\mathcal{P}^\star$, the accuracy of h should be $1 - L(h)$. Since the training data is drawn IID from $\mathcal{P}^\star$, and $L(h) \geq \epsilon$ for any $h \in B$, we get that

$$\Pr[\hat{L}(h) = 0] \leq (1 - L(h))^n \leq (1 - \epsilon)^n \leq \exp^{-\epsilon n},$$

where we use the fact that $1 - x \leq \exp(-x)$.

f) Second, we want the above to hold simultaneously for all $h \in B$. We can apply the union bound to bound the probability of all bad events:

$$\begin{aligned}\Pr[\exists h \in B : \hat{L}(h) = 0] &\leq \sum_{h \in B} \Pr[\hat{L}(h) = 0] \\ &\leq |B| \exp(-\epsilon n) \\ &\leq |\mathcal{H}| \exp(-\epsilon n)\end{aligned}$$

By setting the above at most δ, we conclude that ϵ must be at least $\frac{\log(|\mathcal{H}|)+\log(\delta^{-1})}{n}$.

This concludes the proof for learning finite, realizable hypothesis spaces.

Takeaway: The proof of this result is elementary but illustrates an important pattern that will recur in more complex scenarios. We are interested in the expected risk, but only have access to empirical risk to choose the ERM:

- Step 1 (convergence): for a fixed h, show that $\hat{L}(h)$ is close to $L(h)$ with high probability
- Step 2 (uniform convergence): show that the above holds simultaneously for all hypotheses $h \in \mathcal{H}$

However, the assumptions are restrictive. There exists a perfect hypothesis (realizability). What happens when the problem is not realizable? To answer this, we introduce the tools of concentration estimates.

Second, the hypothesis class is finite. What happens when the number of hypotheses is infinite? To answer this, we need better ways of measuring the "size" of a set – leading to Rademacher complexity, VC, PAC-Bayes, etc.

2.2 Using uniform convergence to reason about generalization (Lecture 2)

We now give a high-level picture of the logic behind how we can use uniform convergence to reason about generalization (in the context of ERM). We'd like to show that ERM's excess risk is small:

$$\Pr\left[L(\hat{h}_{\text{ERM}}) - L(h^\star) \geq \epsilon\right] \leq \delta \tag{12}$$

We can expand the excess risk as

$$L(\hat{h}) - L(h^\star) = \underbrace{\left(L(\hat{h}) - \hat{L}(\hat{h})\right)}_{\text{Uniform convergence}} + \underbrace{\left(\hat{L}(\hat{h}) - \hat{L}(h^\star)\right)}_{\leq 0} + \underbrace{\left(\hat{L}(h^\star) - L(h^\star)\right)}_{\text{Concentration}} \tag{13}$$

We'll see how concentration estimates can be used to control this difference in the third part. However, the same reasoning does not apply to the first part because the ERM $\hat{h}_{\text{ERM}}$ depends on the training examples $\hat{L}$. In particular,

$$\hat{L}(\hat{h}_{\text{ERM}}) = \frac{1}{n}\sum_{i=1}^{n} \ell(\hat{h}_{\text{ERM}}(x_i), y_i). \tag{14}$$

Due to the correlation, the above is not a sum of independent random variables. The central thesis of uniform convergence is that if we could ensure that $L(h)$ and $\hat{L}(h)$ are close for all $h \in \mathcal{H}$, then $L(\hat{h}_{\text{ERM}})$ must be close to $\hat{L}(\hat{h}_{\text{ERM}})$ as well.

In summary, our goal of deriving a uniform convergence result can be stated as

$$\Pr[L(\hat{h}_{\text{ERM}}) - L(h^\star) \geq \epsilon] \leq \Pr\left[\left(\sup_{h\in\mathcal{H}} \left|L(h) - \hat{L}(h)\right|\right) \geq \frac{\epsilon}{2}\right] \leq \delta \tag{15}$$

In particular, the $1/2$ above comes from combining the error terms from the first and third parts together. Put it in words, we'd like to upper bound the probability that the largest difference between the empirical risk and the expected risk is larger than $\epsilon/2$.

2.3 Concentration estimates (Lecture 3)

Concentration inequalities are powerful tools from probability theory that show the average of independent random variables will be concentrated around its expectation. Concentration estimates are the basis of a large branch of learning theory (Bach, 2024) and high-dimensional statistics (Wainwright, 2019). They are one of the most basic tools for studying supervised learning algorithms and models (Zhang, 2023), primarily because much of supervised learning deals with in-distribution samples.

Example (mean estimation): Let $X_1, X_2, \ldots, X_n$ be independent and identically distributed random variables with mean $\mu = \mathbb{E}[X_i]$, for all $i = 1, 2, \ldots, n$. Define the empirical mean as

$$\hat{\mu}_n = \frac{1}{n} \sum_{i=1}^{n} X_i$$

How does $\hat{\mu}_n - \mu$ behave?

Three types of statements from probability:

- Consistency: by law of large numbers,

$$\hat{\mu}_n - \mu \to 0$$

- Asymptotic normality: let the variance of X_i (for all i) be equal to σ^2, by the central limit theorem, we have

$$\sqrt{n}(\hat{\mu}_n - \mu) \sim \mathcal{N}(0, \sigma^2)$$

- Tail estimates: for our purpose, we would like to draw a statement of the following type

$$\Pr[|\hat{\mu}_n - \mu| \geq \epsilon] \leq \delta$$

  For getting such tail estimates, we typically need to study the tail of a distribution, for instance, the tail of a Gaussian distribution, etc.

Markov's inequality: Let $Z \geq 0$ be a non-negative random variable. Then

$$\Pr[Z \geq t] \leq \frac{\mathbb{E}[Z]}{t}$$

Proof: Since Z is a non-negative quantity, we always have the condition that

$$t \mathbb{1}_{Z \geq t} \leq Z$$

To see this, notice that if $Z \geq t$, then the above is true. On the other hand, if $Z < t$, then the left-hand side above is zero, whereas $Z \geq 0$. Next, take the expectation over Z on both sides, we get

$$\mathbb{E}[t \mathbb{1}_{Z \geq t}] \leq \mathbb{E}[Z] \Rightarrow \mathbb{E}[\mathbb{1}_{Z \geq t}] \leq \frac{\mathbb{E}[Z]}{t}$$

Notice that $\mathbb{E}[\mathbb{1}_{Z \geq t}] = \Pr[Z \geq t]$. Thus, we have shown that Markov's inequality is true.

Chebyshev's inequality: Let X be a random variable with mean equal to μ. Then

$$\Pr[|X - \mu| \geq \epsilon] \leq \frac{\mathrm{Var}[X]}{\epsilon^2}$$

Proof: We will use Markov's inequality to get this result. Let $Z = (X - \mu)^2$ and let $t = \epsilon^2$. Notice that $Z \geq 0$. Thus, based on Markov's inequality

$$\Pr[Z \geq t] \leq \frac{\mathbb{E}[Z]}{t} = \frac{\mathbb{E}\left[(Z-\mu)^2\right]}{t} = \frac{\mathrm{Var}[Z]}{t},$$

which completes the proof.

Hoeffding's inequality: Let $Z_1, Z_2, \ldots, Z_n$ be n independent and identically distributed random variables drawn from a distribution with expectation μ and whose values are bounded from above by one.

Let $\hat{\mu}_n = \frac{1}{n}\sum_{i=1}^{n} Z_i$ denote the mean of the n random variables. Then, for any $\epsilon \in (0, 1)$, we have

$$\Pr[|\hat{\mu}_n - \mu| > \epsilon] \leq 2\exp(-2\epsilon^2 n) \tag{16}$$

The Hoeffding's inequality is a very powerful result when we work with the average of n random variables. Variants of this inequality (which is restricted to bounded random variables) are also called Chernoff bound.[4] Next, we shall see a proof through the use of moment generating functions (MGF).

Definition (Moment generating function): For a random variable X, its MGF is given by

$$M_X(t) := \mathbb{E}[\exp(tX)]$$

One can also think of the MGF in terms of Taylor's expansion of $\exp(tX)$ as

$$M_X(t) = 1 + t\,\mathbb{E}[X] + \frac{t^2}{2}\mathbb{E}\left[X^2\right] + \frac{t^3}{6}\mathbb{E}\left[X^3\right] + \cdots$$

Thus, the first moment is the mean of X. The second moment is the variance of X (assuming the mean of X is zero). And so on.

Property: The MGF of the sum of two independent random variables X_1 and X_2 is the product of the MGF of X_1 and X_2, respectively.

- To see this, notice that

$$M_{X+Y}(t) = \mathbb{E}\left[e^{t(X+Y)}\right] = \mathbb{E}\left[e^{tX}e^{tY}\right] = \mathbb{E}\left[e^{tX}\right]\mathbb{E}\left[e^{tY}\right] = M_X(t)M_Y(t)$$

- Here we have used that X and Y are independent, and hence e^{tX} and e^{tY} are independent, to conclude that $\mathbb{E}\left[e^{tX}e^{tY}\right] = \mathbb{E}\left[e^{tX}\right]\mathbb{E}\left[e^{tY}\right]$

The high-level idea for showing the Hoeffding's inequality is obtained by applying Markov's inequality to e^{tX} for some well-chosen value t. From Markov's inequality, we can derive the following useful inequality: for any $t > 0$,

$$\Pr[X \geq a] = \Pr\left[e^{tX} \geq e^{ta}\right] \leq \frac{\mathbb{E}\left[e^{tX}\right]}{e^{ta}}$$

[4] https://en.wikipedia.org/wiki/Chernoff_bound

In particular,

$$\Pr[X \geq a] \leq \min_{t>0} \frac{\mathbb{E}\left[e^{tX}\right]}{e^{ta}} \tag{17}$$

Similarly, for any $t < 0$,

$$\Pr[X \leq a] = \Pr[e^{tX} \geq e^{ta}] \leq \min_{t<0} \frac{\mathbb{E}\left[e^{tX}\right]}{e^{ta}}$$

Bounds for specific distributions are obtained by choosing appropriate values for t.

2.3.1 Chernoff bounds for the sum of Poisson trials

We now develop the most commonly used version of the Chernoff bound for the tail distribution of a sum of independent 0-1 random variables, which are also known as Poisson trials.[5]

Let $X_1, \ldots, X_n$ be a sequence of independent Poisson trials with $\Pr[X_i = 1] = p_i$. Let $X = \sum_{i=1}^{n} X_i$, and let

$$\mu = \mathbb{E}[X] = \mathbb{E}\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} \mathbb{E}[X_i] = \sum_{i=1}^{n} p_i$$

For a given $\delta > 0$, we are interested in bounds on $\Pr[X \geq (1+\delta)\mu]$ and $\Pr[X \leq (1-\delta)\mu]$, that is, the probability that X deviates from its expectation μ by $\delta\mu$ or more. To develop a Chernoff bound, we need to compute the moment generating function of X. We start with the MGF of each X_i:

$$\begin{aligned} M_{X_i}(t) = \mathbb{E}\left[e^{tX_i}\right] = p_i e^t + (1 - p_i) &= 1 + p_i(e^t - 1) \\ &\leq e^{p_i(e^t - 1)}, \end{aligned}$$

where in the last step, we have used the fact that, for any y, $1 + y \leq e^y$. Since the X_i's are independent from each other, we take the product of the n MGF to obtain

$$\begin{aligned} M_X(t) = \prod_{i=1}^{n} M_{X_i}(t) &\leq \prod_{i=1}^{n} e^{p_i(e^t - 1)} \\ &= \exp\left(\sum_{i=1}^{n} p_i(e^t - 1)\right) = e^{(e^t - 1)\mu} \end{aligned}$$

Based on this result, we now apply Markov's inequality: for any $t > 0$, we have

$$\begin{aligned} \Pr[X \geq (1+\delta)\mu] &= \Pr[e^{tX} \geq e^{t(1+\delta)\mu}] \\ &\leq \frac{\mathbb{E}\left[e^{tX}\right]}{e^{t(1+\delta)\mu}} \\ &\leq \frac{e^{(e^t - 1)\mu}}{e^{t(1+\delta)\mu}} \end{aligned}$$

For any $\delta > 0$, we can set $t = \ln(1+\delta) > 0$ to get

$$\Pr[X \geq (1+\delta)\mu] \leq \left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right)^{\mu}$$

[5] Poisson trials differ from Poisson random variables.

For $0 < \delta \leq 1$, we need to show that

$$\frac{e^{\delta}}{(1+\delta)^{1+\delta}} \leq e^{-\delta^2/3}$$

Taking the logarithm of both sides, we obtain

$$f(\delta) = \delta - (1+\delta)\ln(1+\delta) + \frac{\delta^2}{3} \leq 0$$

Computing the derivatives of $f(\delta)$, we have:

$$f'(\delta) = 1 - \frac{1+\delta}{1+\delta} - \ln(1+\delta) + \frac{2\delta}{3}$$
$$f''(\delta) = -\frac{1}{1+\delta} + \frac{2}{3}$$

We see that the second derivative is less than zero if $\delta < 1/2$, and it is positive otherwise. Hence, $f'(\delta)$ first decreases and then increases in the interval $[0, 1]$. Since $f'(0) = 0$ and $f'(1) < 0$, we conclude that $f'(\delta) \leq 0$ in the interval $[0, 1]$. Since $f(0) = 0$, it follows that $f(\delta) \leq 0$, which shows that for any δ between 0 and 1, we have

$$\Pr[X \geq (1+\delta)\mu] \leq e^{-\mu\delta^2/3}$$

2.3.2 Example: Coin flips

Let X be the number of heads in a sequence of n independent fair coin flipts. Applying the Chernoff bound, we have

$$\Pr\left[\left|X - \frac{n}{2}\right| \geq \frac{1}{2}\sqrt{6n\ln n}\right] \leq 2\exp\left(-\frac{1}{3}\frac{n}{2}\frac{6\ln n}{n}\right) = \frac{2}{n}$$

This demonstrates that the concentration of the number of heads around the mean $n/2$ is very tight. Most of the time, the deviations from the mean are on the order of $O(\sqrt{n\ln n})$.

Other use cases of Chernoff bound:

- Suppose we are interested in evaluating the probability that a particular gene mutation occurs in the population. Given a DNA sample, a lab test can determine if it carries the mutation. However, the lab test is expensive and we would like to obtain a relatively reliable estimate from a small number of tests.

2.3.3 Example: Gaussian random variables

In the next example, we look at the MGF of Gaussian random variables. Let $X \sim \mathcal{N}(0, \sigma^2)$. Then, $M_X(t) = e^{\sigma^2 t^2/2}$. To derive this, we use the definition of Gaussian probability density:

$$\begin{aligned} M_X(t) = \mathbb{E}\left[e^{tX}\right] &= \int \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{x^2 - 2\sigma^2 tx}{2\sigma^2}\right) dx \\ &= \int \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\sigma^2 t)^2 - \sigma^4 t^2}{2\sigma^2}\right) dx \\ &= \exp\left(\frac{\sigma^2 t^2}{2}\right) \end{aligned}$$

Based on the above MGF, we can derive a tail bound by plugging the form of MGF to equation (17) to get:

$$\Pr[X \geq \epsilon] \leq \inf_t \exp\left(\frac{\sigma^2 t^2}{2} - t\epsilon\right)$$

The infimum of the RHS is attained by setting $e = \frac{\epsilon}{\sigma^2}$, yielding:

$$\Pr[X \geq \epsilon] \leq \exp\left(-\frac{\epsilon^2}{2\sigma^2}\right)$$

What about non-Gaussian random variables? Notice that the bound still holds if we replace $M_X(t)$ with an upper bound. This motivates the following definition.

Sub-Gaussian: A mean zero random variable X is sub-Gaussian with parameter σ^2 if its MGF is bounded as follows:

$$M_X(t) \leq \exp\left(\frac{\sigma^2 t^2}{2}\right) \tag{18}$$

It follows that for sub-Gaussian X, we again have that $\Pr[X \geq \epsilon] \leq \exp(-\epsilon^2/(2\sigma^2))$.

2.4 Learning finite hypothesis space (Lecture 4)

Recall from last lecture that we developed the Hoeffding's inequality. Next we use this result to bound the excess risk of learning a finite hypothesis class (without the realizable condition).

Learning finite hypothesis classes: Let $\mathcal{H}$ be a finite hypothesis class. Let ℓ be the zero-one loss function: $\ell(h(x), y) = \mathbb{1}_{h(x)\neq y}$. Suppose we minimize the empirical risk to get the minimizer $\hat{h} \in \mathcal{H}$. Then, with probability at least $1 - \delta$ over the randomness of training samples, the excess risk must be bounded by

$$L(\hat{h}) - L(h^\star) \leq \sqrt{\frac{2(\log(|\mathcal{H}|) + \log(2\delta^{-1}))}{n}} \tag{19}$$

We can contrast this result with (11) (of learning finite, realizable hypothesis space). The difference is that we now get a slower convergence rate (n^{-1} to $n^{-1/2}$).

Proof: Recall that the excess risk can be decomposed to

$$L(\hat{h}_{\text{ERM}}) - L(h^\star) = L(\hat{h}_{\text{ERM}}) - \hat{L}(\hat{h}_{\text{ERM}}) + \underbrace{\hat{L}(\hat{h}_{\text{ERM}}) - \hat{L}(h^\star)}_{\leq 0} + \hat{L}(h^\star) - L(h^\star)$$

Notice that the second term is at most zero. Thus, we focus on the first and the third terms.

- Use Hoeffding's inequality to deal with the third term.
- Apply uniform convergence to upper bound the first term by $\sup_{h\in\mathcal{H}} \left|L(h) - \hat{L}(h)\right|$ (Use the union bound).

Step 1: bound $\Pr[L(h) - \hat{L}(h) \geq \epsilon]$ for a fixed $h \in \mathcal{H}$. For a fixed $h \in \mathcal{H}$, notice that $\hat{L}(h)$ is the averaged loss among n IID samples. Each of the loss terms is bounded between zero and one (with expectation equal to $L(h)$). Therefore, by Hoeffding's inequality,

$$\Pr\left[\left|L(h) - \hat{L}(h)\right| \geq \epsilon\right] \leq 2\exp(-2n\epsilon^2)$$

Step 2: apply Step 1 uniformly over all possible $h \in \mathcal{H}$. In particular, we can apply union bound over all the possible bad events to get

$$\Pr\left[\sup_{h\in\mathcal{H}}\left|L(h) - \hat{L}(h)\right| \geq \epsilon\right] \leq |\mathcal{H}| \cdot 2\exp(-2n\epsilon^2) = \delta$$

By setting $\epsilon = \sqrt{\frac{2(\log|\mathcal{H}|+\log(2\delta^{-1}))}{n}}$, we can get the probability set to δ.

Remarks: We have removed the realizable assumption by suffering a $\sqrt{n}$ factor in the generalization bound. The $\sqrt{n}$ factor arises from sampling noise. It makes sense that learning is faster when there is no noise.

2.5 Rademacher complexity (Lecture 4)

Both of our generalization bounds require finite hypothesis classes. What about infinite hypothesis classes?

We can't directly apply the union bound to infinite hypothesis classes. Need a more sophisticated way to measure complexity of a hypothesis class.

With Rademacher complexity, the key idea is symmetrization. Along the way, we need an extension of the Hoeffding's inequality to some function of bounded random variables, which is known as McDiarmid's inequality. Let us first start by motivating why we need these tools.

2.5.1 Motivation

Recall that, within the uniform convergence framework, we want to get a statement of the following

$$\Pr\left[L(\hat{h}) - L(h^\star) \geq \epsilon\right] \leq \Pr\left[\sup_{h\in\mathcal{H}}\left|L(h) - \hat{L}(h)\right| \geq \frac{\epsilon}{2}\right] \leq \delta \tag{20}$$

Since there are two cases here, let us denote

$$G_n := \sup_{h\in\mathcal{H}} L(h) - \hat{L}(h), \quad G_n' := \sup_{h\in\mathcal{H}} \hat{L}(h) - L(h)$$

Then we have that

$$\Pr\left[\sup_{h\in\mathcal{H}}\left|L(h) - \hat{L}(h)\right| \geq \frac{\epsilon}{2}\right] \leq \Pr\left[G_n \geq \frac{\epsilon}{2}\right] + \Pr\left[G_n' \geq \frac{\epsilon}{2}\right]$$

Let us focus on the first case, since the second case will be similar.

Now, our main object becomes G_n: This is a rather non-trivial function, because of taking the supremum over a sum of random variables.

Hence, let's look at its expectation first! Usually, when we encounter a complicated random variable, we start by examining its first moment, then second moment, and so on.

2.5.2 Defintion of Rademacher complexity

Our main object of interest is now $\mathbb{E}[G_n] = \mathbb{E}\left[\sup_{h\in\mathcal{H}}(L(h) - \hat{L}(h))\right]$. Recall that $\hat{L}(h)$ is the empirical risk, and $L(h)$ is the expected risk.

This quantity is still quite difficult because it depends on the expected risk, an expectation over the unknown data distribution. The key idea of symmetrization is to remove this expected risk term with a "simpler" term.

Definition of Rademacher complexity: Let us imagine n data points $Z'_1 = (x'_1, y'_1), Z'_2 = (x'_2, y'_2), \ldots, Z'_n = (x'_n, y'_n)$, sampled from the true data distribution. Then, clearly $L(h) = \mathbb{E}\left[\hat{L}'(h)\right]$, where $\hat{L}'(h)$ is the empirical risk on this "copy" dataset.

Hence,

$$\begin{aligned}
\mathbb{E}[G_n] &= \mathop{\mathbb{E}}_{Z_{1:n}}\left[\sup_{h\in\mathcal{H}}(L(h) - \hat{L}(h))\right] \\
&= \mathop{\mathbb{E}}_{Z_{1:n}}\left[\sup_{h\in\mathcal{H}} \mathop{\mathbb{E}}_{Z'_{1:n}}\left[\hat{L}'(h) - \hat{L}(h)\right]\right] \\
&= \mathop{\mathbb{E}}_{Z_{1:n}}\left[\mathop{\mathbb{E}}_{Z'_{1:n}}\left[\sup_{h\in\mathcal{H}}(\hat{L}'(h) - \hat{L}(h))\right]\right]
\end{aligned}$$

Let us remove the dependence on the copy dataset. To that end, we introduce IID Rademacher random variables $\sigma_1, \sigma_2, \ldots, \sigma_n$ sampled uniformly from $\{+1, -1\}$.

Notice that

$$\hat{L}'(h) - \hat{L}(h) = \sum_{i=1}^{n}(\ell(h(x'_i), y'_i) - \ell(h(x_i), y_i)),$$

is symmetric around zero. Hence, multiplying each individual term by σ_i does not change its distribution. Thus,

$$\begin{aligned}
\mathbb{E}[G_n] &\le \mathop{\mathbb{E}}_{Z_{1:n}, Z'_{1:n}}\left[\sup_{h\in\mathcal{H}}(\hat{L}'(h) - \hat{L}(h))\right] \\
&= \mathop{\mathbb{E}}_{Z_{1:n}, Z'_{1:n}, \sigma_{1:n}}\left[\sup_{h\in\mathcal{H}}\frac{1}{n}\sum_{i=1}^{n}\sigma_i(\ell(h(x'_i), y'_i) - \ell(h(x_i), y_i))\right] \\
&\le 2\mathop{\mathbb{E}}_{Z_{1:n}, \sigma_{1:n}}\left[\sup_{h\in\mathcal{H}}\frac{1}{n}\sum_{i=1}^{n}\sigma_i\ell(h(x_i), y_i)\right] \qquad (21)
\end{aligned}$$

The last line (without the factor of 2) is the Rademacher complexity of $\mathcal{H}$.

In summary, let $\mathcal{H}$ be a hypothesis class consisting of a class of real-valued functions. Example: two-layer neural nets, linear models. Define the Rademacher complexity (or Rademacher average) of $\mathcal{H}$ to be

$$R_n(\mathcal{H}) := \mathop{\mathbb{E}}_{Z_{1:n}, \sigma_{1:n}}\left[\sup_{h\in\mathcal{H}}\frac{1}{n}\sum_{i=1}^{n}\sigma_i\ell(h(x_i), y_i)\right],$$

where $Z_i = (x_i, y_i)$ is a random sample from the underlying data distribution, and σ_i is sampled uniformly from $\{+1, -1\}$.

2.5.3 Generalization bounds based on Rademacher complexity

To see the power of the concept we just introduced, here is a very general statement where we can always rely on when we work with supervised learning algorithms and models.

Generalization bounds based on Rademacher complexity: Define

$$\mathcal{A} = \{(x, y) \to \ell(h(x), y) : h \in \mathcal{H}\}$$

to be the loss function composed with the hypothesis space. Let $\mathcal{R}_n(\mathcal{A})$ denote the Rademacher complexity of the function class $\mathcal{A}$.

With probability at least $1 - \delta$,

$$L(\hat{h}_{\text{ERM}}) - L(h^\star) \leq 4R_n(\mathcal{A}) + \sqrt{\frac{2\log(2\delta^{-1})}{n}} \tag{22}$$

To be clear, recall that $\hat{h}_{\text{ERM}}$ is the empirical risk minimizer (ERM), $h^\star$ is the expected risk minimizer, and n is the size of the training set.

Proof sketch:

- Show that empirical Rademacher complexity is close to the expectation
- Use McDiarmid's inequality, which is essentially a concentration result for functions of IID random variables. We'll introduce this tool shortly
- Show that the Rademacher complexity upper bounds the excess risk. We've seen this logic from the motivation

In more detail, recall that $G_n = \sup_{h\in\mathcal{H}} L(h) - \hat{L}(h)$. Our first claim is to show that G_n is close to $\mathbb{E}[G_n]$:

$$\Pr[G_n \geq \mathbb{E}[G_n] + \epsilon] \leq \exp(-2n\epsilon^2) \tag{23}$$

This shows that G_n is indeed close to its expectation plus a small error. Hence it suffices to upper bound its expectation.

Our second claim is to show that $\mathbb{E}[G_n]$ is upper bounded by the Rademacher complexity.

$$\mathbb{E}[G_n] \leq 2R_n(\mathcal{A}).$$

We have already seen the proof of this claim. Combined together, we can derive (22).

2.5.4 Properties of Rademacher complexity

Boundedness:

$$R_n(\mathcal{H}) \leq \max_{h\in\mathcal{H}} \max_{x,y} \ell(h(x), y)$$

This only shows that the Rademacher complexity is bounded by some constant. Usually, we'd like to show it goes to zero as n goes to infinity.

Monotonicity: If $\mathcal{H}_1 \subseteq \mathcal{H}_2$, then $R_n(\mathcal{H}_1) \leq R_n(\mathcal{H}_2)$.

This is because we now take the supreme over a larger set, hence R_n increases.

Scaling: $R_n(c \cdot \mathcal{H}) = c \cdot R_n(\mathcal{H})$.

Lipschitz composition: Suppose ϕ is a Lipschitz-continuous function, bounded by some constant c_ϕ. Recall a function is Lipschitz-continuous if changing x only changes the function value by c_ϕ times. In addition, $\phi(0) = 0$.

Let $\phi \circ \mathcal{H} = \{(x, y) \to \phi(h(x), y) : h \in \mathcal{H}\}$, i.e., compose ϕ with h. Then, we have that

$$R_n(\phi \circ \mathcal{H}) \leq c_\phi \cdot R_n(\mathcal{H}).$$

The proof essentially uses the contraction property of the sum of Rademacher random variables. For details, see Corollary 3.17 of Ledoux and Talagrand, 2013. This property is useful because we can start by studying a simpler hypothesis class, and then compose more functions with the class without going through the calculation again.

Convex hull: The convex hull of a set $\mathcal{H}$ is when we take all possible linear combinations of the hypotheses in $\mathcal{H}$. One useful property is that taking the convex of $\mathcal{H}$ preserves the Rademacher complexity: $R_n(\mathcal{H}) = R_n(\text{convex-hull}(\mathcal{H}))$.

Proof: Notice that the supreme over the convex hull must be attained at a vertex of the convex hull. This property is quite useful if we want to calculate the Rademacher complexity of a polytope. Though it is an infinite set, it suffices to examine the vertices of the polytope. We'll use this property when we examine the ℓ_1-regularization.

2.5.5 Wrapping up the proof of (22) (Lecture 6)

Recall that (23) that we'd like to show G_n and its expectation $\mathbb{E}[G_n]$ are close to each other. In order to complete this result, we'll use a tool called McDiarmid's inequality.

McDiarmid's inequality: the bounded differences inequality

- Let f be a function satisfying the following bounded differences condition for all $i = 1, 2, \ldots, n$ and all $x_1, x_2, \ldots, x_n$, and any x_i':

$$\left|f(x_1, x_2, \ldots, x_n) - f(x_1, \ldots, x_{i-1}, x_i', x_{i+1}, \ldots, x_n)\right| \leq c_i$$

  In other words, modifying one coordinate does not change the function value too much.

- Let $X_1, X_2, \ldots, X_n$ be independent random variables. Then

$$\Pr\left[|f(X_1, X_2, \ldots, X_n) - \mathbb{E}\left[f(X_1, X_2, \ldots, X_n)\right]| \geq \epsilon\right] \leq \exp\left(-\frac{2\epsilon^2}{\sum_{i=1}^n c_i^2}\right) \tag{24}$$

Remark: McDiarmid's inequality generalizes Hoeffding's inequality. Let the function $f(x_1, x_2, \ldots, x_n) = \frac{1}{n}\sum_{i=1}^n x_i$ be the empirical mean. Suppose every x_i and x_i' are bounded from above by one. Then, changing x_i to x_i' changes the function value by at most $\frac{2}{n}$, leading to $c_i = \frac{2}{n}$ for all $i = 1, 2, \ldots, n$. Hence, the sum of c_i^2 is equal to $\frac{4}{n}$. Plugging this to (24) recovers the Hoeffding's inequality.

This result is quite powerful, since it holds for any independent random variables, and f can be complex (say a neural net). As long as this function is not too sensitive to perturbations in one coordinate, we get good concentration.

Proof of McDiarmid's inequality: Recall the definition of a martingale sequence

- A sequence of $Z_0, Z_1, \ldots, Z_n$ is a martingale sequence with respect to another sequence of random variables $X_1, X_2, \ldots, X_n$ if and only if Z_i only depends on $X_{1:i}$ and $\mathbb{E}\left[Z_i | X_{1:i-1}\right] = Z_{i-1}$
- For a martingale sequence, we have the concentration because of Azuma-Hoeffding's inequality: Suppose $Z_{1:t}$ is a martingale sequence such that $\mathbb{E}\left[Z_i | Z_{<i}\right] = 0$ and $|Z_i| \leq c_i$. Then with probability at least $1 - \delta$,

$$\sum_{i=1}^{n} Z_i \leq \sqrt{2\Big(\sum_{i=1}^{n} c_i^2\Big) \log(\delta^{-1})} \tag{25}$$

  For reference, see Chapter 12.4 of Mitzenmacher and Upfal (2017).[6]

We're going to construct a sequence

$$Z_i = \mathbb{E}\left[f(X_1, X_2, \ldots, X_n) \mid X_{1:i}\right] \tag{26}$$

Let's check the following properties

- Clearly, Z_i only depends on the values of $X_{1:i}$, but not of the remaining sequence
- Besides,

$$\mathbb{E}\left[Z_i \mid X_{1:i-1}\right] = \mathbb{E}\left[\mathbb{E}\left[f(X_1, X_2, \ldots, X_n) \mid X_{1:i}\right] \mid X_{1:i-1}\right] = Z_{i-1}$$

- Note that for $i = 0$,

$$Z_0 = \mathbb{E}\left[f(X_1, X_2, \ldots, X_n)\right]$$

- For $i = n$,

$$Z_n = \mathbb{E}\left[f(X_1, \ldots, X_n) \mid X_{1:n}\right] = f(X_1, \ldots, X_n)$$

Using these notations, we'd like to bound the following

$$\Pr[Z_n - Z_0 \geq \epsilon]$$

Let $D_i = Z_i - Z_{i-1}$. Clearly, $Z_n - Z_0 = \sum_{i=1}^{n} D_i$. We're going to show that $|D_i|$ is bounded by c_i. Therefore, using Azuma-Hoeffding's inequality, we can get an exponentially decreasing tail bound for $Z_n - Z_0$. We write down both Z_i and Z_{i-1} as follows:

$$\begin{aligned} Z_{i-1} &= \mathbb{E}\left[f(X_1, \ldots, X_n) \mid X_{1:i-1}\right] \\ Z_i &= \mathbb{E}\left[f(X_1, \ldots, X_n) \mid X_{1:i}\right] \end{aligned}$$

The key difference is whether we condition on X_i or not above. Imagine the set of possible realizations of X_i. We'll consider the maximum and minimum achieving realizations

$$\begin{aligned} L_i &= \inf_x \mathbb{E}\left[f(X_1, \ldots, X_{i-1}, X_i = x, \ldots, X_n) \mid X_{1:i-1}, X_i = x\right] - Z_{i-1} \\ U_i &= \sup_x \mathbb{E}\left[f(X_1, \ldots, X_{i-1}, X_i = x, \ldots, X_n) \mid X_{1:i-1}, X_i = x\right] - Z_{i-1} \end{aligned}$$

[6] See also https://en.wikipedia.org/wiki/Azuma%27s_inequality

Clearly, D_i is somewhere between L_i and U_i. Suppose L_i is achieved by $X_i = x_L$ and U_i is achieved by $X_i = x_U$.

By the bounded difference assumption, for any fixed values $X_{1:n}$ except X_i (being x_L or x_U), they are at most c_i apart. Taking expectation over $X_{1:i-1}$ and note that they are independent of X_i (key!), we have that

$$\begin{aligned}|U_i| &= |\mathbb{E}\left[f(X_1, \ldots, X_{i-1}, X_i = x_U, \ldots, X_n) - f(X_1, \ldots, X_{i-1}, X_i, \ldots, X_n) \mid X_{1:i-1}\right]| \\ &\leq c_i,\end{aligned}$$

by the bounded difference condition on X_i. And similarly, $|L_i| \leq c_i$. Thus, we have verified that the sequence $D_1, D_2, \ldots, D_n$ is bounded. In addition, we can check that they are a Martingale sequence based on the properties listed above.

Having checked through McDiarmid's inequality, it remains to show that G_n satisfies the bounded difference condition. Recall that $G_n = \sup_{h \in \mathcal{H}} L(h) - \hat{L}(h)$. This quantity naturally arises as we derive a uniform convergence claim. Recall that $\hat{L}$ is the empirical risk. Consider changing Z_i to Z_i' in G_n, we'd like to bound the difference

$$\left| \left(\sup_{h \in \mathcal{H}} L(h) - \hat{L}(h) \right) - \left(\sup_{h \in \mathcal{H}} \left(L(h) - \hat{L}(h) + \frac{1}{n}(\ell(Z_i, h) - \ell(Z_i', h)) \right) \right) \right| \leq \frac{1}{n}$$

This is because the loss values are within 0 and 1. Taking supreme cannot increase the difference.

In summary, we now put all the pieces together to complete the proof of (22).

- We'd like to show that, for $\epsilon = 4R_n(\mathcal{A}) + \sqrt{\frac{2\log(2\delta^{-1})}{n}}$,

$$\Pr[L(\hat{h}_{\text{ERM}}) - L(h^\star) \geq \epsilon] \leq \Pr\left[\sup_{h \in \mathcal{H}} \left| L(h) - \hat{L}(h) \right| \geq \frac{\epsilon}{2}\right] \leq \delta$$

- The latter reduces to showing that

$$\Pr\left[\sup_{h \in \mathcal{H}} \left| L(h) - \hat{L}(h) \right| \geq \frac{\epsilon}{2}\right] \leq \Pr[G_n \geq \frac{\epsilon}{2}] + \Pr[G_n' \geq \frac{\epsilon}{2}]$$

- By the steps above, we have

$$\begin{aligned}\Pr[G_n \geq \frac{\epsilon}{2}] &\leq \Pr\left[G_n - \mathbb{E}\left[G_n\right] \geq \sqrt{\frac{\log(2\delta^{-1})}{2n}}\right] \\ &\leq \exp\left(-2n\left(\frac{\log(2\delta^{-1})}{2n}\right)\right) = \frac{\delta}{2}\end{aligned}$$

  The first line is because by (21), $\mathbb{E}\left[G_n\right] \leq 2R_n(\mathcal{A})$.

- A similar proof applies to show that $\Pr[G_n' \geq \frac{\epsilon}{2}] \leq \frac{\delta}{2}$

2.6 Examples of Rademacher complexity (Lecture 5)

So far, we have set up Rademacher complexity for bounding the complexity of an infinite hypothesis class. Now, let's apply Rademacher complexity for the case when the function class is finite. Recall that we have already derived a nice generalization bound for finite hypothesis classes, using concentration inequalities (see result (19)). Still, this is a nice sanity check through a different route. Besides, we'll derive a useful result that will be helpful later on.

2.6.1 Learning finite hypothesis classes

Massart's finite lemma: Let $\mathcal{H}$ be a set of functions/hypotheses. Let M^2 be a bound on the second moment of functions of $\mathcal{H}$:

$$\sup_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} (f(Z_i))^2 \leq M^2,$$

where $Z_i = (x_i, y_i)$ denotes an input data sample. Then, the empirical Rademacher complexity is bounded by:

$$\hat{R}_n(F) \leq \sqrt{\frac{2M^2 \log(|\mathcal{H}|)}{n}} \tag{27}$$

Notice that if we combine the above result (27) with the Rademacher complexity-based generalization bound of equation (22), we recover the result in equation (19).

Proof: For simplicity, let us denote $W_f = \frac{1}{n} \sum_{i=1}^{n} \sigma_i f(Z_i)$. Recall that the empirical Rademacher complexity is defined as

$$\hat{R}_n(F) = \mathop{\mathbb{E}}_{\sigma_{1:n}} \left[\sup_{f \in \mathcal{H}} W_f \mid Z_{1:n} \right]$$

Let us write down the moment generating function of $\sup_{f \in \mathcal{H}} W_f$, and we'll use the convexity of the exponential function

$$\begin{aligned} \exp\left(t \cdot \mathbb{E}\left[\sup_{f \in \mathcal{H}} W_f \mid Z_{1:n} \right] \right) &\leq \mathbb{E}\left[\exp\left(t \cdot \sup_{f \in \mathcal{H}} W_f \right) \mid Z_{1:n} \right] \\ &= \mathbb{E}\left[\sup_{f \in \mathcal{H}} \exp\left(t \cdot W_f \right) \mid Z_{1:n} \right] \end{aligned}$$

The above can be further bounded by

$$|\mathcal{H}| \cdot \mathbb{E}\left[\exp\left(t \cdot W_f \right) \mid Z_{1:n} \right]$$

Finally, recall that W_f is the sum of n independent random variables. Hence, we invoke the product property of the individual MGFs, to get that $M_{W_f}(t)$ is the product of the individual random variables. Recall that we've conditioned on $Z_{1:n}$, which can be treated as constants. Thus, the randomness is now just on $\sigma_{1:n}$—each σ_i is either $+1$ or -1, scaled by $f(Z_i)$ (which is a constant conditioned on Z_i). We can show that σ_i is sub-Gaussian with parameter 1, since we have the fact that for any random variable X bounded between a and b (for some $a < b$), then X must be $(b-a)^2/4$ sub-Gaussian.[7]

As a result, W_f is sub-Gaussian with parameter at most $\frac{1}{n^2} \sum_{i=1}^{n} (f(Z_i))^2 \leq M^2/n$. Therefore, we now get that

$$\mathbb{E}\left[\exp(t \cdot W_f) \mid Z_{1:n} \right] \leq \exp\left(\frac{t \cdot M^2}{2n} \right)$$

[7] For a proof see this blog post: https://statisticaloddsandends.wordpress.com/2018/10/05/bounded-random-variables-are-sub-gaussian/

We can now conclude that

$$\exp(t\hat{R}_n(f)) \leq |\mathcal{H}| \exp\left(\frac{tM^2}{2n}\right)$$

Taking log on both sides and dividing by t, we get:

$$\hat{R}_n(\mathcal{H}) \leq \frac{\log(|\mathcal{H}|)}{t} + \frac{tM^2}{2n}$$

By setting t to minimize the right-hand side above, we conclude that $\hat{R}_n(\mathcal{H}) \leq \sqrt{\frac{2\log(|\mathcal{H}|)M^2}{n}}$.

We can apply the above result along with (22) to derive the result for learning from a bounded set of 0-1 valued functions.

2.6.2 Generalization bounds for ℓ_2-norm/ℓ_1-norm constrained linear models

The next two examples involve learning a linear model from a norm-bounded hypothesis space. The first example applies to Ridge regression, for instance, training models with weight decay. The second example applies to LASSO regression, i.e., training with ℓ_1-regularization instead. Next, we'll derive the Rademacher complexity of these two examples.

Rademacher complexity of ℓ_2 balls: Let $\mathcal{H} = \{z \to \langle w, z\rangle \mid w \in \mathcal{W}\}$ be a set of linear functions such that $\|w\| \leq B$ for any $w \in \mathcal{W}$. Furthermore, assume the data distribution has a bounded second moment:

$$\mathop{\mathbb{E}}_{z\sim\mathbb{P}}\left[\|z\|^2\right] \leq C^2$$

Then we have

$$R_n(\mathcal{H}) = \mathbb{E}\left[\sup_{h\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^n h(z_i)\right] \leq \frac{B\cdot C}{\sqrt{n}}$$

Proof: The key idea is to exploit the linear algebraic structure of the Rademacher complexity

$$\begin{aligned} R_n(\mathcal{H}) = \mathbb{E}\left[\sup_{h\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^n \sigma_i h(z_i)\right] &= \mathbb{E}\left[\sup_{w\in\mathcal{W}} \frac{1}{n}\sum_{i=1}^n \sigma_i \langle w, z_i\rangle\right] \\ &\leq \mathbb{E}\left[\sup_{w\in\mathcal{W}} \left\|\frac{w}{n}\right\| \cdot \left\|\sum_{i=1}^n \sigma_i z_i\right\|\right] \\ &\leq \frac{B}{n}\cdot \mathbb{E}\left[\left\|\sum_{i=1}^n \sigma_i z_i\right\|\right] \end{aligned}$$

by the condition placed on $\mathcal{W}$. Finally, by Jensen's inequality, the above is at most

$$\frac{B}{n}\cdot\sqrt{\mathbb{E}\left[\left\|\sum_{i=1}^n \sigma_i z_i\right\|^2\right]} = \frac{B}{n}\cdot\sqrt{\sum_{i=1}^n \mathbb{E}\left[\|z_i\|^2\right]} \leq \frac{B\cdot C}{\sqrt{n}}$$

Rademacher complexity of ℓ_1 balls: In some applications, we have a finite but large set of features, and we believe that only a small subset of them are relevant to our task. For such applications, ℓ_1-regularization has proven to be a working strategy (e.g., pruning, compression). Assume that the entries of the input vector are bounded from above by some constant $C > 0$: $\max_{j=1}^{d} z_i[j] \leq C$, for $i = 1, 2, \ldots, n$. Let $\mathcal{H} = \{z \to \langle w, z \rangle \mid \|w\|_1 \leq B\}$ be a set of linear functions. Then, the Rademacher complexity of $\mathcal{H}$ is bounded as follows

$$R_n(\mathcal{H}) \leq \frac{B \cdot C \cdot \sqrt{2 \log(2d)}}{\sqrt{n}}$$

Proof: The key idea is that the ℓ_1 ball is the convex hull of $2d$ weight vectors aligned with the basis

$$W = \{Be_1, -Be_1, Be_2, -Be_2, \ldots, Be_d, -Be_d\}$$

Since the Rademacher complexity of this class is equal to the Rademacher complexity of its convex hull, we just look at the finite class W since

$$R_n(\mathcal{H}) = \mathbb{E}\left[\sup_{w \in W} \frac{1}{n} \sum_{i=1}^{n} \sigma_i \langle w, z_i \rangle\right]$$

Since W is a finite set, we can apply Massart's finite lemma (see (27)). In order to do so, we shall first check the moment condition:

$$\frac{1}{n} \sum_{i=1}^{n} (\sigma_i \langle w, z_i \rangle)^2 \leq \|w\|_1^2 \cdot \|z\|_\infty^2 = B^2 \cdot C^2$$

Thus, by result (27), we now get that

$$\hat{R}_n(\mathcal{H}) \leq \sqrt{\frac{2B^2C^2 \log(2d)}{n}}$$

2.6.3 Binary classification

In the next example, we study the case of binary classification. For an input with feature x and label y, we focus on the set of linear predictors with bounded ℓ_2-norm: $\mathcal{W} = \{w : \|w\| \leq B\}$. Next, we select a loss function for our linear predictor. The output of the linear predictor is given by $y \cdot \langle w, x \rangle$.

- Zero-one loss: $\ell(m) = \mathbb{1}_{m \leq 0}$
- Hinge loss: $\ell(m) = \max(0, 1 - m)$
- Logistic loss: $\ell(m) = \log(1 + \exp(-m))$

The function class is $\mathcal{H} = \{z \to \langle z, w \rangle \mid w \in \mathcal{W}\}$. The loss function composed with $\mathcal{H}$ is thus

$$\mathcal{A} = \{(x, y) \to \ell(y \langle w, x \rangle) \mid w \in \mathcal{W}\}$$

- The linear predictors are constrained inside an ℓ_2-norm bounded ball
- Both the Hinge loss and the logistic loss are Lipschitz-continuous

- Using the composition property of Rademacher complexity (see Section 2.5.4), we get a generalization bound for solving binary classification using Hinge/logistic loss

The zero-one loss, however, is not Lipschitz, so we cannot directly apply our result to the zero-one loss. Notice that the zero-one loss is not sensitive to the norm of w. Think of two points close to zero, one positive and the other negative. To handle this situation, we can use the margin loss.

Margin loss: Penalizes whenever we don't predict correctly by at least a margin of γ

$$\phi_\gamma(m) = \mathbb{1}_{m \leq \gamma}$$

Then, we smooth the margin loss between 0 and γ into a straight line, leading to the following somewhat complicated form

$$\tilde{\phi}_\gamma(m) = \begin{cases} 1 & \text{if } m \leq 0 \\ 1 - \frac{m}{\gamma} & \text{if } 0 < m < \gamma \\ 0 & \text{if } m \geq \gamma \end{cases}$$

Key observation: $\tilde{\phi}_\gamma$ is γ^{-1}-Lipschitz continuous (meaning that $\left|\tilde{\phi}_\gamma(m_1) - \tilde{\phi}_\gamma(m_2)\right| \leq \frac{|m_1 - m_2|}{\gamma}$).

Margin-sensitive zero-one loss for linear classifiers: Let $\mathcal{H}$ be a set of linear models. With probability at least $1 - \delta$,

$$L_0(\hat{h}_{\text{ERM}}) \leq \min_{h \in \mathcal{H}} L_\gamma(h) + \frac{4R_n(\mathcal{H})}{\gamma} + \sqrt{\frac{2\log(2\delta^{-1})}{n}}$$

Proof sketch:

- Using composition of R_n, we have: $R_n(\tilde{\phi}_\gamma \circ \mathcal{H}) \leq \gamma^{-1} R_n(\mathcal{H})$
- Using Rademacher complexity applied to $\tilde{L}_\gamma$, we get the relation between $\tilde{L}_\gamma(\hat{h}_{\text{ERM}})$ and $\min_{h \in \mathcal{H}} \tilde{L}_\gamma(h)$
- Notice that $\phi_0 \leq \tilde{\phi}_\gamma \leq \phi_\gamma$. Thus,

$$\min_{h \in \mathcal{H}} \tilde{L}_\gamma \leq \min_{h \in \mathcal{H}} L_\gamma, \text{ and } \quad L_0(\hat{h}_{\text{ERM}}) \leq \tilde{L}_\gamma(\hat{h}_{\text{ERM}}).$$

2.7 Matrix completion (Lecture 6)

Suppose you are asked to build a recommendation system. This system is used to predict the review rating of a movie. Related examples: Book reviews on Amazon and music recommendations on YouTube.

Rank-constrained minimization: Given an input array M, find X as close to M as possible on the observed entries, while subject to a (low) rank constraint:

$$\begin{aligned} \min_X \quad & \sum_{(i,j) \in \Omega} (X_{i,j} - M_{i,j})^2 \\ \text{s.t.} \quad & \text{rank}(X) = r \end{aligned}$$

Nuclear norm relaxation: The rank constraint makes the problem non-convex, search space increases at a combinatorial rate (although good algorithms exist). Instead, we could relax the rank constraint into a nuclear norm constraint: given a matrix X, let $\|X\|_\star = \sum_j \lambda_j(X)$—the sum of all the singular values of X. One could show that $\|X\|_\star$ is convex in X; this is called the "nuclear" norm of X.[8] Instead, solve the following, which is convex in X:

$$\begin{aligned} \min_X \quad & \sum_{(i,j)\in\Omega} (X_{i,j} - M_{i,j})^2 \\ \text{s.t.} \quad & \|X\|_\star \le dr, \end{aligned}$$

where we may replace dr with another parameter as needed.

Low-rank factorization: An alternative way to formulate the above is by writing it as a convex optimization problem:

$$\min \sum_{(i,j)\in\Omega} (X_{i,j} - M_{i,j})^2 + \lambda \|X\|_\star$$

This is a semi-definite program (SDP), which is convex in X. Instead, we may consider the re-parametrized optimization problem:

$$\min_{U,V} \sum_{(i,j)\in\Omega} (U_i^\top V_j - M_{i,j})^2,$$

where U_i, V_j are both r-dimensional vectors, for any i, j. Put together, $U = [U_1, U_2, \ldots, U_{d_1}]$, $V = [V_1, V_2, \ldots, V_{d_2}]$ are both rank-r matrices.

How well does nuclear norm minimization work? Imagine we want to recover an unknown low-rank matrix $M \in \mathbb{R}^{d_1 \times d_2}$. We observe m entries from M uniformly at random. Then, given another matrix X, we may define the mean squared recovery error as

$$\frac{1}{d_1}\frac{1}{d_2}\sum_{i=1}^{d_1}\sum_{j=1}^{d_2}(X_{i,j} - M_{i,j})^2$$

Without further assumptions, we would need to see the entire matrix:

- Imagine the unknown matrix only has an extremely large entry but remains nonzero elsewhere
- Without sampling this entry, we are only observing zeros

Therefore, we shall assume that M is bounded in all entries—this is a reasonable assumption for modeling many applications (e.g., movie ratings). In general, one may assume that M is incoherent (meaning that the row norms of the low-rank factors are bounded at the order of $O(n^{-1})$).

[8] See https://en.wikipedia.org/wiki/Matrix_norm

Recovery result: Let dr be the upper bound we place on the matrix in the nuclear norm minimization program. Let $\hat{M}$ denote the output of the program. Then, the recovery error of $\hat{M}$ is at most:

$$2r\sqrt{\frac{\log(d)}{md}},$$

where $d = \max(d_1, d_2)$. For example, if we want the test error less than ϵ, then the number of samples needs to be at least

$$m \geq \frac{dr^2}{\epsilon^2 \log(d)}$$

We're going to use Rademacher complexity to tackle this. Here we can think that nuclear norm minimization constrains how large the matrix can be. Thus, we define the hypothesis class to be the set of nuclear norm bounded and entrywise bounded matrices as follows:

$$\mathcal{H} = \left\{X \in \mathbb{R}^{d_1 \times d_2} \mid \|X\|_\star \leq dr, \|X\|_\infty \leq 1\right\}$$

Let $\Omega \subseteq [d_1] \times [d_2]$ be the indices of the set of observed entries. Let X_Ω denote the observed matrix that pads zero for the unobserved entries. Then, we define the Rademacher complexity of $\mathcal{H}$ as

$$R_n(\mathcal{H}) = \mathop{\mathbb{E}}_{\Omega, \sigma_{1:m}} \left[\sup_{X \in \mathcal{H}} \frac{1}{m} \sum_{(i,j) \in \Omega} \sigma_{i,j} \cdot |X_{i,j} - M_{i,j}|\right]$$

Here we're going to use the following result. Suppose $\mathcal{H}$ is a set of matrices whose nuclear norm is bounded by C:

$$\mathcal{H} = \left\{X \in \mathbb{R}^{d_1 \times d_2} \mid \|X\|_\star \leq C\right\}$$

Then, the Rademacher complexity of nuclear norm bounded matrices is at most

$$R_n(\mathcal{H}) = \mathop{\mathbb{E}}_{\Omega} \left[\sup_{X \in \mathcal{H}} \frac{1}{m} \sum_{(i,j) \in \Omega} \sigma_{i,j} |X_{i,j} - M_{i,j}|\right] \leq 2C\sqrt{\frac{\log d}{md}} \tag{28}$$

Recall our result from (22), if $m \geq dr^2 \log(d\delta^{-1})/\epsilon^2$, then the above Rademacher complexity is at most ϵ. Therefore, based on (22), we could say that with probability at least $1 - \delta$, the recovery error of the minimizer must be at most $O(\epsilon)$.

It remains to show that (28) is true. By symmetry, the absolute function inside $R_n(\mathcal{H})$ can be skipped:

$$R_n(\mathcal{H}) = \mathbb{E}\left[\sup_{X \in \mathcal{H}} \frac{1}{m} \sum_{(i,j) \in \Omega} \sigma_{i,j}(X_{i,j} - M_{i,j})\right]$$

Let us introduce a zero-one matrix Σ such that $\Sigma_{i,j} = \sigma_{i,j}$ if $(i,j) \in \Omega$, and $\Sigma_{i,j} = 0$ otherwise. Hence, the above is equal to

$$\begin{aligned}\mathop{\mathbb{E}}_{\Omega, \sigma_{1:n}} \left[\sup_{X \in \mathcal{H}} \frac{1}{m} \langle \Sigma, X - M\rangle\right] &\leq \mathbb{E}\left[\sup_{X \in \mathcal{H}} \frac{1}{m} \|\Sigma\|_2 \cdot \|X - M\|_\star\right] \\ &\leq \frac{2C}{m} \mathbb{E}\left[\|\Sigma\|_2\right]\end{aligned}$$

Above we use the fact that both X, M have nuclear norm bounded above by C. Hence, to finish the proof, we are left to deal with the expectation of $\|\Sigma\|_2$. Here we'll use the claim that

$$\mathbb{E}\left[\|\Sigma\|_2\right] \leq \sqrt{\frac{m \log d}{d}}$$

Showing this result requires using some facts related to the spectrum of random matrices, we'll cover this result at a later point in class. In summary, we have thus shown that

$$R_n(\mathcal{H}) \leq 2C\sqrt{\frac{\log(d)}{md}}$$

2.8 Two-layer neural networks (Lecture 7)

Setup: Consider a two-layer ReLU neural net with m hidden units. Let $\Phi = (w \in \mathbb{R}^m, U \in \mathbb{R}^{m \times d})$ denote the parameters for the first and second layers, respectively.

Given an input vector $x \in \mathbb{R}^d$, let

$$f_\Phi(x) = w^\top \phi(Ux) \in \mathbb{R},$$

denote the output of the network, where $\phi(\cdot)$ is the activation function that is 1-Lipschitz continuous. Let us assume the training set $\{x_i\}_{i=1}^n \subseteq \mathbb{R}^d$ lies in some bounded ℓ_2-norm ball.

First result: Let $\mathcal{H} = \{f_\Phi \mid \|w\| \leq B', \|U\|_2 \leq B\}$. Then the Rademacher complexity of $\mathcal{H}$ is bounded by

$$R_n(\mathcal{H}) \leq 2BB'\sqrt{\frac{m}{n}}$$

Imagine we use a two-layer neural network to classify MNIST digits. If we use a network with m neurons. This result says that the Rademacher complexity scales with the operator norm of the first layer weights and the second layer weights.

Proof: Use composition property of Rademacher complexity and result for ℓ_2-norm bounded linear predictors; Let's first write the Rademacher complexity

$$R_n(\mathcal{H}) = \mathop{\mathbb{E}}_{\sigma_{1:n}}\left[\sup_{\Phi \in \mathcal{H}} \frac{1}{n}\sum_{i=1}^n \sigma_i w^\top \phi(Ux)\right]$$

Let's first deal with the supremum over the second layer weight w by setting its direction to be the same as the other vector

$$R_n(\mathcal{H}) \leq B' \cdot \mathop{\mathbb{E}}_{\sigma_{1:n}}\left[\sup_{\Phi \in \mathcal{H}} \left\|\frac{1}{n}\sum_{i=1}^n \sigma_i \phi(Ux_i)\right\|_2\right]$$

Next, because there are m neurons. Hence (by Cauchy-Schwartz inequality), we reduce the above to less than

$$B' \cdot \sqrt{m} \cdot \mathbb{E}\left[\sup_{\|u\| \leq B} \left|\frac{1}{n}\sum_{i=1}^n \sigma_i \phi(u^\top x_i)\right|\right]$$

The trick here is that since the operator norm of the first layer weight U is bounded by B, the norm of every row of U (or neuron) is also bounded by B. It remains to deal with the

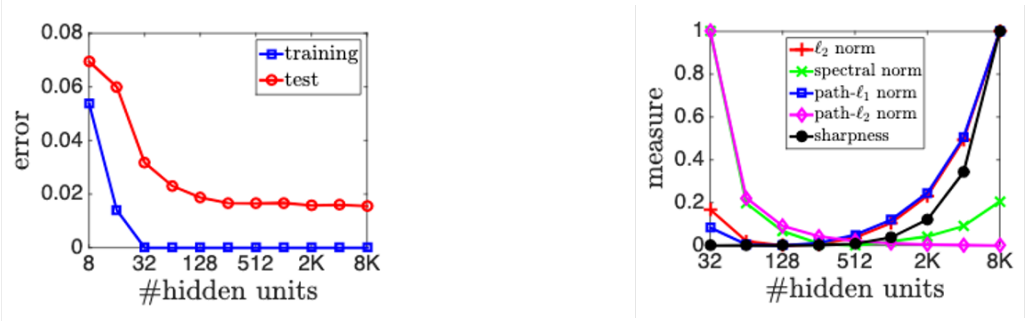


Figure 1: Vary the number of neurons m in a two-layer neural net. Path norm scales nicely with the test error.

above. Here, we can use our composition property: ϕ is 1-Lipschitz continuous. Furthermore, the absolute value can be divided into two parts. Hence, the above is at most

$$2B' \cdot \sqrt{m} \cdot \mathbb{E}\left[\sup_{\|u\| \le B} \frac{1}{n} \sum_{i=1}^{n} \sigma_i u^\top x_i\right]$$

To finish the proof, we invoke our Rademacher complexity bound for ℓ_2-norm bounded linear predictors, to get that the above is at most $2B'B\sqrt{m/n}$.

Provided with the Rademacher complexity of two-layer neural nets, we can get a generalization bound for neural networks, adding an extra factor of $\sqrt{\log(\delta^{-1})/n}$.

Second result: The first result scales with $\sqrt{m}$ – number of hidden units. Hence, the result gets worse when we over-parametrize with more hidden units. Empirically, researchers have observed that generalization error can improve as number of hidden units increases. Here's the result from training a two-layer neural net on MNIST (Neyshabur et al., 2017).

We'll a generalization bound that does not depend on the number of hidden units m using the path norm $P(w, U) = \sum_{i=1}^{m} |w_i| \cdot \|u_i\|$, where u_i is the i-th row of U. Let $\mathcal{H} = \{f_\Phi \mid P(w, U) \le B\}$. Then, the Rademacher complexity of $\mathcal{H}$ is bounded as follows

$$R_n(\mathcal{H}) \le B\sqrt{\frac{1}{n}}$$

Remarks: The second result implies the first result. To see this, we use the Cauchy-Schwartz inequality:

$$P(w, U) = \sum_{i=1}^{m} |w_i| \cdot \|u_i\| \le \sqrt{\sum_{i=1}^{m} w_i^2} \cdot \sqrt{\sum_{i=1}^{m} \|u_i\|^2} \le B' \cdot B \cdot \sqrt{m}$$

Technically speaking, the path norm still scales with the number of hidden units. However, as we saw from the previous slide, some neurons turn out to be quite small in the experiments, resulting in a small path norm even as m increases!

Proof: Let's first write the Rademacher complexity. Recall that u_i is the i-th row of U. We expand out the Rademacher complexity using

$$w^\top \phi(Ux_i) = \sum_{j=1}^{m} w_j \phi(u_j^\top x_i)$$

Let $\bar{u}_j = u_j / \|u_j\|$. Because ReLU activation is 1-Homogeneous, meaning that $\phi(cx) = c\phi(x)$, hence

$$w^\top \phi(Ux_i) = \sum_{j=1}^{m} w_j \|u_j\| \phi\big(\bar{u}_j^\top x_i\big)$$

We apply this expansion to the Rademacher complexity

$$R_n(\mathcal{H}) = \mathbb{E}\left[\sup_{\Phi} \frac{1}{n} \sum_{i=1}^{n} \sigma_i \left(\sum_{j=1}^{m} w_j \|u_j\| \phi(\bar{u}_j^\top x_i)\right)\right]$$

Taking the supremum over Φ means that we'll maximize the path norm above! Hence, after switching the order of the sum, we can get

$$R_n(\mathcal{H}) \leq B \cdot \mathbb{E}\left[\max_{1 \leq j \leq m} \left|\frac{1}{n} \sum_{i=1}^{n} \sigma_i \phi(\bar{u}_j^\top x_i)\right|\right] \leq 2B\sqrt{\frac{1}{n}}$$

3 Generalization of neural networks and deep learning

In the previous section, we saw the concept of uniform convergence, and showed that with this concept, we can get pretty strong results on a variety of problems, including ℓ_2/ℓ_1-regularized linear functions, matrix completion, and two-layer neural networks. However, it turns out that moving beyond these settings with the techniques we've developed is quite challenging. A couple of key challenges are:

- In terms of the sample/learning complexity, how could we go beyond two layers? How does this complexity depend on the data distribution?
- How could we incorporate the inductive bias induced by the choice of specific optimization algorithms into the learning complexity?

The goal of this section is to tackle the above questions. In particular, we shall begin by establishing the folklore intuition that the learning complexity of an ML model scales with its number of parameters. This kind of folklore is especially intriguing in the context of deep networks, as these models have millions (and now billions) of parameters. However, this high complexity clearly doesn't explain/corroborate with the empirical results we typically see with model fine-tuning (now prompt tuning).

3.1 The concept of shattering and VC dimension (Lecture 8)

Motivation about shattering: Imagine we want to bound the complexity of a set of half-spaces passing through the origin:

$$F = \left\{ z \to \mathbb{1}_{w^\top z \geq 0} \mid w \in \mathbb{R}^d \right\}$$

For example, think of w as a linear predictor in the case of binary classification problems. Question: What should the complexity of F be?

Notice that this function class depends on d parameters. For example, there are 2^d possible output vectors for a given input. However, while F is an infinite set, it only has finitely many possible behaviors on a given set of n samples $x_1, x_2, \ldots, x_n$.

Example: Consider two points in two dimensions: $z_1 = [3, 0], z_2 = [-3, 0]$. Recall the class of half-spaces passing through the origin can be written down as

$$\mathcal{F} = \left\{ z \to \mathbb{1}_{w^\top z \geq 0} \mid w \in \mathbb{R}^2 \right\}$$

For this example, the second coordinate of w does not matter. Hence, the number of possible outputs here is only 2, i.e., $[1, 0], [-1, 0]$ (instead of 4). In particular, this implies that the empirical Rademacher complexity of $\{z_1, z_2\}$ is the same as $\{[1, 0], [-1, 0]\}$ (since we essentially just take the sup over this output set).

More generally, for two function classes $\mathcal{F}$ and $\mathcal{F}'$, if

$$\{[f(z_1), f(z_2), \ldots, f(z_n)] \mid f \in \mathcal{F}\} = \left\{[f'(z_1), f'(z_2), \ldots, f'(z_n)] \mid f' \in \mathcal{F}'\right\}$$

Then, $\hat{R}_n(\mathcal{F}) = \hat{R}_n(\mathcal{F}')$. In other words, what matters is the output behavior of the function on the n samples. This observation is useful when we analyze Boolean functions (i.e., those that return either 0 or 1), an important example being the loss function class $\mathcal{A} = \{z \to \ell(z, h) \mid h \in \mathcal{H}\}$ where ℓ is the zero-one loss. This observation regarding the output behavior of $\mathcal{F}$ is captured by the shattering coefficient.

Definition of shattering: Let $\mathcal{F}$ be a family of functions that map a dataset Z to a finite set (such as $\{0,1\}$). The shattering coefficient of $\mathcal{F}$ is the maximum number of behaviors observed on n samples:

$$s(\mathcal{F},n) = \max_{\{z_1,z_2,\ldots,z_n\}\subseteq Z} |\{[f(z_1),f(z_2),\ldots,f(z_n)] \mid f\in\mathcal{F}\}|$$

We can use Massart's finite lemma (see (27)) to bound the empirical Rademacher complexity using the shattering coefficient. In particular, suppose $\mathcal{F}$ involves a set of Boolean functions, then clearly the second moment of any such function on n inputs is at most 1:

$$\sup_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n}(f(z_i))^2 \le 1$$

Hence, by setting $M=1$ in (27), we get that the empirical Rademacher complexity of $\mathcal{F}$ is at most

$$\hat{R}_n(\mathcal{F}) \le \sqrt{\frac{2\log(s(\mathcal{F},n))}{n}}$$

In essence, we have reduced the complexity of learning a possibly infinite class of functions down to a finite number (i.e., the shattering).

As a remark, in the case of zero-one loss, the shattering coefficient $s(\mathcal{F},n)$ is at most 2^n. Hence, the empirical Rademacher complexity is at most $\sqrt{\frac{2\log(2^n)}{n}} = \sqrt{2}$. Therefore, to get meaningful results, we would like the above quantity to go to zero instead, as n goes to infinity.

Example (threshold functions): Consider n real values placed on a line. We claim that the function class $\mathcal{F} = \{z \to \mathbb{1}_{z\ge t} \mid t\in\mathbb{R}\}$ has shattering coefficient $s(\mathcal{F},n) = n+1$.

To see this, we notice that on n real-valued inputs, all of the possible outputs are: $[0,0,\ldots,0],[1,0,\ldots,0],[1,1,\ldots,0],\ldots,[1,1,\ldots,1]$, which has $n+1$ different outcomes in total.

Now we are ready to define the concept of VC dimension.

Definition of VC dimension: The VC dimension of a family of functions $\mathcal{H}$ with Boolean outputs is the maximum number of samples that can be shattered by $\mathcal{H}$. In other words,

$$VC(\mathcal{H}) = \sup_n \mathbb{1}[s(\mathcal{H},n) = 2^n]$$

Thus, the VC dimension of $\mathcal{H}$ captures the maximum number of samples whose output behaviors can always be realized using some function $h\in\mathcal{H}$.

Example (intervals): Let $\mathcal{H} = \{z \to \mathbb{1}[z\in[a,b]] \mid a,b\in\mathbb{R}\}$. We can verify that $s(\mathcal{H},1) = 2$ (fix this point and vary a,b), and $s(\mathcal{H},2) = 2$ (fix two different points and vary a,b to achieve all four outcomes). However, $s(\mathcal{H},3) = 7 < 2^3$—because after fixing the three points, no matter how we set a,b, we cannot leave out the middle point. In other words, we cannot realize the outcome $[1,0,1]$. As a result, the VC dimension of $\mathcal{H}$ is equal to 2.

Example (VC dimension of finite-dimensional function classes): Let $\mathcal{F}$ be a real-valued function class. Let $\mathcal{H} = \{x \to \mathbb{1}[f(x) \geq 0] \mid f \in \mathcal{F}\}$ be the set of binary classifiers defined by $\mathcal{F}$. Then we have that

$$VC(\mathcal{H}) \leq dim(\mathcal{F})$$

where the dimension of $\mathcal{F}$ refers to the number of basis elements in $\mathcal{F}$. For example, if $\mathcal{F} = \left\{x \to w^\top x \mid w \in \mathbb{R}^d\right\}$, then $dim(\mathcal{F}) = d$. This result thus connects the algebraic dimension of $\mathcal{F}$ with the combinatorial dimension of $\mathcal{H}$.

Proof: Take any n samples $x_1, x_2, \ldots, x_n$ with $n > dim(\mathcal{F})$. We show that these n points cannot be completely shattered by $\mathcal{H}$. In other words, we'd like to find some direction that $\mathcal{F}$ does not cover. To achieve this, consider the linear map $M(f) = [f(x_1), f(x_2), \ldots, f(x_n)] \in \mathbb{R}^n$ that takes a function $f \in \mathcal{F}$ and outputs a vector on the n samples.

By definition, the dimension of $\{M(f) \mid f \in \mathcal{F}\}$ is at most $dim(\mathcal{F})$. Since $n > dim(\mathcal{F})$, there must exist a nonzero vector $c \in \mathbb{R}^n$ such that $\langle c, M(f)\rangle = 0$ for all $f \in \mathcal{F}$. In particular, let us assume without loss of generality that there exists an entry of c that is negative. Otherwise, we could just consider $-c$ with the same argument (since c is nonzero).

Thus, for all functions $f \in \mathcal{F}$, we have that

$$\sum_{i=1}^{n} c_i f(x_i) = 0$$

In particular, we now separate the sum into two buckets depending on whether c_i is positive or negative, and re-write the above as

$$\sum_{i:c_i \geq 0} c_i f(x_i) + \sum_{i:c_i < 0} c_i f(x_i) = 0$$

Now suppose that the VC dimension of $\mathcal{H}$ is greater than or equal to the dimension of $\mathcal{F}$. Then, for any zero one labeling of $x_1, x_2, \ldots, x_n$, that is $y_1, y_2, \ldots, y_n$, there must exist some function $f \in \mathcal{F}$ such that

$$h = [x_i \to \mathbb{1}[f(x_i) \geq 0] = y_i] \in \mathcal{H}$$

Based on these premises, we now derive a contradiction, which will imply that the n samples cannot be shattered by $\mathcal{F}$. This will imply that the VC dimension of $\mathcal{H}$ should be at most $dim(\mathcal{F})$.

To derive this contradiction, let us consider a function h such that: $h(x_i) = 1$ (hence $f(x_i) \geq 0$) whenever $c_i \geq 0$; $h(x_i) = 0$ (hence $f(x_i) < 0$) whenever $c_i < 0$. For such an h, the weighted sum of $c_i f(x_i)$ must be strictly positive. This is a contradiction!

Remark: by invoking the set of linear half spaces through the origin, we can achieve the above dimension d. For this case, we shall construct some set of d points that can be shattered by $\mathcal{F}$, and the set of d basis vectors will do. In particular, let us create d samples corresponding to the basis vectors as

$$\begin{aligned} z_1 &= [1, 0, \ldots, 0] \\ z_2 &= [0, 1, \ldots, 0] \\ &\ldots \\ z_d &= [0, 0, \ldots, 1] \end{aligned}$$

We'd like to show that for any binary vector $y \in \{0,1\}^d$, we can find a linear predictor $w \in \mathbb{R}^d$ such that

$$\begin{aligned}\mathbb{1}[w^\top z_1 \geq 0] &= y_1 \\ \mathbb{1}[w^\top z_2 \geq 0] &= y_2 \\ &\cdots \\ \mathbb{1} w^\top z_n \geq 0 &= y_n\end{aligned}$$

We construct the linear predictor w as follows. Let $I \subseteq \{1, 2, \ldots, d\}$ be the set of indices on which $y \in \{0,1\}^d$ has value 1. We then set $w_i = 1$ for any $i \in I$ and set $w_i = -1$ for any $i \notin I$. Now, we can verify that the above conditions all hold. This implies that the VC dimension of these linear half spaces in dimension d must be at least d. In conclusion, the VC dimension is equal to d.

Connecting shattering to Rademacher complexity: For a function class $\mathcal{H}$ whose VC dimension is at most d, its shattering coefficient is at most the following

$$s(\mathcal{H}, n) \leq \sum_{i=0}^{d} \binom{n}{i} \leq \begin{cases} 2^n & \text{if } n \leq d \\ \left(\frac{en}{d}\right)^d & \text{if } n > d \end{cases} \tag{29}$$

Now, imagine the case that $n > d$. Using this result we shall have that

$$\log(s(\mathcal{A}, n)) = \log(s(\mathcal{H}, n)) \leq d(\log(n) + 1 - \log(d)),$$

where $\mathcal{A}$ is the loss function class of $\mathcal{H}$. Thus, we now have a result that bounds the empirical Rademacher complexity of $\mathcal{A}$ using the VC dimension of $\mathcal{H}$ as follows:

$$\hat{R}_n(\mathcal{A}) \leq \sqrt{\frac{2\log(s(\mathcal{F}, n))}{n}} \leq \sqrt{\frac{2d(\log(n) + 1 - \log(d))}{n}} \leq \sqrt{\frac{2VC(\mathcal{H})\ln(en)}{n}} \tag{30}$$

By plugging this result into (22), we thus got a learning bound that only depends on the VC dimension of the hypothesis class $\mathcal{H}$ and the number of samples n.

The proof of (29) is skipped and can be found in Section 3.11, Liang, 2016.

VC dimension of deep neural networks: It has been shown that for a network with W weight parameters and L number of layers, the VC dimension of such classes of neural networks with ReLU activation (or piecewise linear activation) function is at most $O(WL\log(W))$ (Bartlett et al., 2019). Therefore, we could now combine this result with (30)—together, we now have the folklore statement that the learning complexity of a deep neural network scales with the number of parameters of the network!

3.2 Analyzing over-parameterized neural networks using neural tangent kernels (Lecture 9)

3.2.1 Basics of kernel methods

The performance of a machine learning model breaks down into two parts:

$$L(\hat{f}) - \inf_f L(f) = \underbrace{L(\hat{f}) - \inf_{f \in F} L(f)}_{\text{Estimation error}} + \underbrace{\inf_{f \in F} L(f) - \inf_f L(f)}_{\text{Approximation error}}$$

Both optimization and generalization results aim to reduce estimation error. Approximation error results, on the other hand, relate to the expressivity of a function class.

Example (linear models and kernel features): For linear methods (i.e., $f_w(x) = w^\top x$ for some parameter $w \in \mathbb{R}^d$ and input feature vector $x \in \mathbb{R}^d$), the estimation error is small, but the approximation error is large.

Kernel methods represent one way to move beyond linear methods. In kernel methods, we replace $\langle x, w\rangle$ with $\langle \phi(x), w\rangle$, where $\phi(x)$ is an arbitrary feature map of x. Then, we may use the kernel features in a regression model. Suppose we are minimizing the mean squared error using a kernel method:

$$\ell(w) = \frac{1}{2n}\sum_{i=1}^{n}(\langle \phi(x_i), w\rangle - y_i)^2$$

Definition of kernels: A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a positive semi-definite kernel if and only if for every subset of inputs $\{x_1, x_2, \ldots, x_n\} \subseteq \mathcal{X}$, the matrix $K \in \mathbb{R}^{n\times n}$ defined by using the kernel map

$$K_{i,j} = k(x_i, x_j), \text{ for every } 1 \le i, j \le n$$

is a positive semi-definite matrix.[9]

Examples (kernels):

- Linear kernels: $k(x, x') = \langle x, x'\rangle$. This follows from the fact that $K = XX^\top$ is positive semi-definite.
- Gaussian kernels: $k(x, x') = \exp\left(-\frac{\|x-x'\|^2}{2\sigma^2}\right)$. This is followed by the fact that either the sum or the product of two kernels (the element-wise product (Hadamard product) of two PSD matrices is still PSD) is still a kernel.
- Polynomial kernels: $k(x, x') = (1 + \langle x, x'\rangle)^p$.

Exercise: verify that both the polynomial kernel and the Gaussian kernel are indeed kernels

Alternative definitions of kernels through feature maps: Recall that a feature map ϕ maps an input x to a feature embedding $\phi(x)$. Consider the kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ defined by

$$k(x, x') = \langle \phi(x), \phi(x')\rangle$$

This must be a kernel. To show this, we will follow the definition of a kernel by taking a vector α and let

$$\alpha^\top K\alpha = \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\langle \phi(x_i), \phi(x_j)\rangle = A^\top A,$$

where $A = \sum_{i=1}^{n}\alpha_i\phi(x_i)$.

[9] Recall that a symmetric matrix K is positive semi-definite if for every x, $x^\top Kx \ge 0$.

3.2.2 Motivation for introducing the neural tangent kernels

It has been widely observed that gradient-based optimization algorithms often converge to small training errors on a complex neural network model. A widely believed explanation for this surprising phenomenon is that the neural net is over-parametrized. The neural tangent kernel represents one of the earliest attempts to mathematically formulate the theory of over-parameterized neural networks. The neural tangent kernel arises from the dynamics of the predictions applied to the training data.

Problem setup: Given a training dataset $\{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$, let $w \in \mathbb{R}^N$ represent the parameters of a neural network. Let $f_w(x)$ denote the output of the network. We shall restrict our attention to the mean squared loss (for solving regression problems)

$$\ell(w) = \frac{1}{2}\sum_{i=1}^{n}(f_w(x_i) - y_i)^2$$

For the purpose of illustration, let us consider the gradient flow update. Let $w(t)$ denote the iterate configuration at time $t \geq 0$. Then, $w(t)$ evolves according to

$$\frac{dw(t)}{dt} = -\nabla\ell(w(t))$$

Learning dynamics: We now show that the dynamics are characterized by a positive semi-definite matrix $H(t) \in \mathbb{R}^{n\times n}$, defined as

$$H_{i,j}(t) = \left\langle \frac{\partial f_{w(t)}(x_i)}{\partial w_i}, \frac{\partial f_{w(t)}(x_j)}{\partial w_j} \right\rangle \tag{31}$$

Then, let the network outputs at time t be defined as

$$u(t) = [f_{w(t)}(x_1), f_{w(t)}(x_2), \ldots, f_{w(t)}(x_n)],$$

for any $t \geq 0$. Finally, let $y = [y_1, y_2, \ldots, y_n]$. Then, we will show that the network outputs will follow the dynamics as

$$\frac{du(t)}{dt} = -H(t)(u(t) - y)$$

Proof: Recall that the evolution of the network parameters follows an update as

$$\frac{dw(t)}{dt} = -\nabla\ell(w(t)) = -\sum_{i=1}^{n}(f_{w(t)}(x_i) - y_i)\frac{df_{w(t)}(x_i)}{dw}$$

Using chain rule and multi-variate calculus, the evolution of the network output satisfies

$$\frac{df_{w(t)}(x_i)}{dt} = \left\langle \frac{df_{w(t)}(x_i)}{dw}, \frac{dw(t)}{dt} \right\rangle$$

By applying the first step to the second step, we get

$$\begin{aligned}\frac{df_{w(t)}(x_i)}{dt} &= \left\langle \frac{df_{w(t)}(x_i)}{dw}, -\sum_{j=1}^{n}(f_{w(t)}(x_j)-y_j)\frac{df_{w(t)}(x_j)}{dw} \right\rangle \\ &= -\sum_{j=1}^{n}(f_{w(t)}(x_j)-y_j)\underbrace{\left\langle \frac{df_{w(t)}(x_i)}{dw}, \frac{df_{w(t)}(x_j)}{dw} \right\rangle}_{i,j\text{-th entry of the } H(t) \text{ matrix}} \\ &= -H(t)(u(t)-y)\end{aligned}$$

In summary, the learning dynamics are captured by a symmetric matrix $H(t)$. Now, how does the NTK arise? We will define an ultra-wide neural net whose width goes to infinity. In the limit, it can be shown that the matrix $H(0)$ remains constant during training, i.e., equal to $H(0)$. Moreover, under a random initialization of parameters, $H(0)$ converges to a deterministic kernel matrix $H^\star$ — the Neural Tangent Kernel (NTK).

3.2.3 Defining the neural tangent kernel

Two-layer neural networks: Consider the mapping

$$x \to \sum_{i=1}^{m} a_i \sigma(w_i^\top x), \tag{32}$$

where σ represents a nonlinear activation function (such as ReLU or sigmoid). Now consider the mapping rescaled by $\frac{1}{\sqrt{m}}$:

$$x \to \frac{1}{\sqrt{m}} \sum_{i=1}^{m} a_i \sigma(w_i^\top x)$$

We shall linearize the right-hand side around the initialization $w_i(0)$, for all $i = 1, 2, \ldots, m$; Essentially, performing Taylor's expansion to derive the following

$$\begin{aligned}x &\to \frac{1}{\sqrt{m}} \sum_{i=1}^{m} a_i \left(\sigma((w_i(0))^\top x) + (w_i - w_i(0))^\top x \sigma'(w_i(0)^\top x)\right) \\ &= \frac{1}{\sqrt{m}} \sum_{i=1}^{m} a_i \left(\sigma((w_i(0))^\top x) - (w_i(0))^\top x \sigma'((w_i(0))^\top x)\right) + \frac{1}{\sqrt{m}} x^\top \left(\sum_{i=1}^{m} a_i w_i \sigma'((w_i(0))^\top x)\right)\end{aligned}$$

Defining the ultra-wide kernel: Recall that the dynamics of predictions is governed by $H(t) \in \mathbb{R}^{n\times n}$ (see (31)). Recall that f is given in (32). Therefore, the i,j-th entry of the kernel matrix $H(t)$ is

$$H_{i,j}(0) = x_i^\top x_j \sum_{r=1}^{m} \left\langle \frac{a_r \sigma'((w_r(0)^\top x_i))}{\sqrt{m}}, \frac{a_r \sigma'((w_r(0))^\top x_j)}{\sqrt{m}} \right\rangle$$

Suppose we sample every $w_r(0)$ from a standard Gaussian distribution. In addition, suppose we sample a_r uniformly between $\{+1, -1\}$. Then, one can view the above as the average of m independent random variables. When m becomes very large, then by the law of large numbers, the average is close to its expectation. This gives rise to the Neural Tangent Kernel evaluated at x,x_j as follows:

$$H^\star_{i,j} = x_i^\top x_j \mathop{\mathbb{E}}_{w\sim\mathcal{N}(0,\mathrm{Id})} \left[\sigma'(w^\top x_i)\sigma'(w^\top x_j)\right]$$

3.2.4 Convergence analysis (Lecture 10)

Based on the NTK defined above, we now show that the neural network stays close to initialization long enough to get a small loss value. There are essentially two steps here:

- Step 1: For a sufficiently wide network, the randomly initialized neural net is close to the expectation – the NTK
- Step 2: For a sufficiently wide network, the kernel matrix at time t remains close to the initialization

$H(0)$ is close to $H^\star$ for sufficiently large m: Suppose the activation function (e.g., ReLU) is 1-Lipschitz continuous. Suppose for every input x_i, the Euclidean norm of x_i is less than 1, for any $i = 1, 2, \ldots, n$. For any $\epsilon > 0$, if $m \geq O(n^2 \log(\delta^{-1})/\epsilon^2)$ (recall n is the number of samples at the input), then with probability at least $1 - \delta$, we have that

$$\|H(0) - H^\star\|_F \leq \epsilon$$

To show that this is true, we will first show that every entry of $H(0)$ is close to $H^\star$ with probability $1 - O(\delta/n^2)$. First, we can see that every individual entry of $H(0)$ is bounded from above by 1:

$$\left| x_i^\top x_j \cdot \sigma'((w_r(0))^\top x_i) \sigma'((w_r(0))^\top x_j) \right| \leq 1$$

Now we shall apply Hoeffding's inequality (see (16)), and set the deviation as ϵ/n. Then, the failure probability becomes

$$2 \exp\left(-\frac{2\epsilon^2 m}{n^2} \right)$$

When $m \geq O(\frac{n^2 \log(n\delta^{-1})}{\epsilon^2})$, the above probability can be reduced below $O(\delta/n^2)$. By taking a union bound over all the entries of $H(0)$, we have that the above will hold with probability $1 - \delta$.

Finally, we convert the entry-wise error bound to the operator norm bound:

$$\|H(0) - H^\star\|_F \leq \sqrt{n^2 \cdot \frac{\epsilon^2}{n^2}} = \epsilon$$

$H(t)$ remains close to $H^\star$: Suppose that $y_i = O(1)$ for all $i = 1, 2, \ldots, n$. Let $t > 0$, suppose that for all $0 < s < t$, $u_i(s) = O(1)$ for all $i = 1, 2, \ldots, n$. Then if $m \geq O(\frac{n^6 t^2 \log(n\delta^{-1})}{\epsilon^2})$, with probability $1 - \delta$,

$$\|H(t) - H^\star\|_F \leq \epsilon$$

There are two steps in the proof of this result:

- We first show that every weight vector remains close to the initialization if the width is large
- We then show that this implies the kernel matrix remains close to the NTK

Consider the movement of a single weight vector w_r:

$$\begin{aligned}
\|w_r(t) - w_r(0)\|_2 &= \left\| \int_0^t \frac{dw_r(\tau)}{d\tau} d\tau \right\|_2 \\
&= \left\| \int_0^t \frac{1}{\sqrt{m}} \sum_{i=0}^{n} (u_i(\tau) - y_i) a_r x_i \sigma'(w_r(\tau)^\top x_i) d\tau \right\|_2 \\
&\leq \int_0^t \left\| \frac{1}{\sqrt{m}} \sum_{i=0}^{n} (u_i(\tau) - y_i) a_r x_i \sigma'(w_r(\tau)^\top x_i) d\tau \right\|_2 \\
&\leq \frac{1}{\sqrt{m}} \sum_{i=0}^{n} \int_0^t \left\| (u_i(\tau) - y_i) a_r x_i \sigma'(w_r(\tau)^\top x_i) \right\|_2 d\tau \leq O\left(\frac{tn}{\sqrt{m}}\right),
\end{aligned}$$

where we use the fact that the inputs x_i, y_i and $a_r, u_i(\tau)$ are all bounded by some constants, and the derivative of σ is also bounded by a constant (e.g., sigmoid).

Consider the movement of a single entry of the kernel matrix

$$\begin{aligned}
H_{i,j}(t) - H_{i,j}(0) =& \frac{1}{m} \left| \sum_{r=1}^{m} \left(\sigma'(w_r(t)^\top x_i) \sigma'(w_r(t)^\top x_j) - \sigma'(w_r(0)^\top x_i) \sigma'(w_r(0)^\top x_j) \right) \right| \\
\leq& \frac{1}{m} \left| \sum_{r=1}^{m} \sigma'(w_r(t)^\top x_i) (\sigma'(w_r(t)^\top x_j) - \sigma'(w_r(0)^\top x_j)) \right| \\
&+ \frac{1}{m} \left| \sum_{r=1}^{m} \sigma'(w_r(0)^\top x_j) (\sigma'(w_r(t)^\top x_i - \sigma'(w_r(0)^\top x_i)) \right| \\
\lesssim& \frac{1}{m} \sum_{r=1}^{m} \max_{i=1}^{n} \left\| \sigma'(w_r(t)^\top x_i) - \sigma'(w_r(0)^\top x_i) \right\|_2 \\
\leq& O\left(\frac{tn}{\sqrt{m}}\right)
\end{aligned}$$

Taken together, we have shown that if m is large enough, then $H(t)$ will stay close to $H^\star$ throughout the entire gradient descent dynamic.

3.2.5 Implications of the neural tangent kernel analysis (Lecture 10)

The dynamics of network outputs are governed by the following approximation using the NTK matrix

$$\frac{du(t)}{dt} \approx -H^\star(u(t) - y)$$

We'll use this to describe two implications: i) Convergence to global minima; ii) Explaining why NTK converges faster using original labels than using random labels.

Notice that this is a linear dynamical system. Denote the eigen-decomposition of $H^\star$ as $\sum_{i=1}^n \lambda_i v_i v_i^\top$. Now we consider the dynamics of $u(t)$ on each eigenvector separately. Consider one eigenvector v_i for instance. We have

$$\frac{dv_i^\top u(t)}{dt} \approx -v_i^\top H^\star(u(t) - y) = -\lambda_i v_i^\top u(t) + \lambda_i v_i^\top y,$$

which is equivalent to

$$\frac{dv_i^\top (u(t) - y)}{dt} = -\lambda_i v_i^\top (u(t) - y)$$

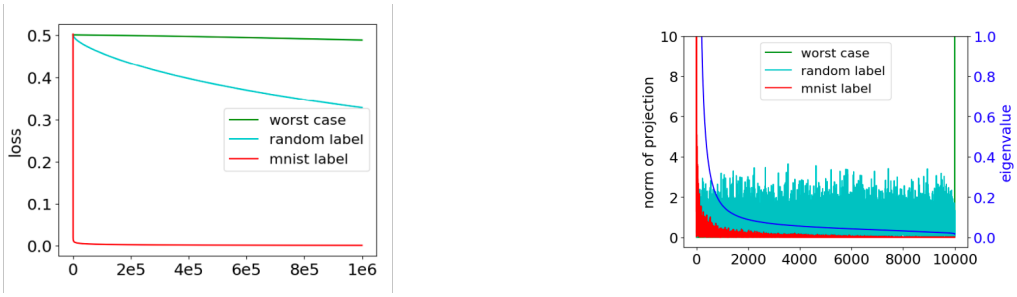


Figure 2: Illustration of the loss and the projected values onto the eigenspace.

One function that satisfies this differential equation is when

$$v_i^\top (u(t) - y) = \exp(-\lambda_i t)$$

Notice that $u(t) - y$ is the difference between predictions and training labels at time t.

We would like to show that the difference converges to zero when t is large. The above implies that each component of $u(t) - y$ projected to eigenvector v_i converges to zero exponentially fast at a rate of $\exp(-\lambda_i t)$.

Also notice that the eigenvectors $\{v_i\}_i$ form an orthonormal basis. Since we know that each component converges to zero, together they imply that $u(t) - y$ must also converge to zero. This shows that the outputs will eventually converge to the training labels, meaning the training loss will go to zero.

Next, we can see that each component goes to zero at different rates. For larger eigenvalues, the convergence is faster. Hence in order to have fast convergence, we would like the projections of the labels onto the top eigenvectors to be large.

For a set of labels, if they align with top eigenvectors, then gradient descent converges quickly. On the other hand, if the projections are uniform, or aligns with eigenvectors with small eigenvectors, then gradient descent converges with a slow rate. Below, we compare convergence rates of gradient descent between using original labels, random labels, and worst-case labels. Data: two classes of MNIST. Model: $f(a, w) = \frac{1}{\sqrt{m}} \sum_{i=1}^m a_i \sigma'(w_i^\top x)$.

3.3 Implicit regularization in over-parameterized matrix factorization (Lecture 11)

Recall that we discussed the matrix completion problem in lecture 6. Now, we shall study the gradient descent dynamic of this problem in an over-parameterized setting.

Warm-up example: Suppose we are doing linear regression on n input samples:

$$\ell(w) = \frac{1}{n} \sum_{i=1}^n (x_i^\top \beta - y_i)$$

Consider gradient descent:

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

Suppose n is smaller than the dimension of $w \in \mathbb{R}^d$. Suppose $w_0 = 0$. Then, given that $\nabla \ell(w_t) = \frac{2}{n} \sum_{i=1}^n (x_i^\top \beta - y_i) x_i$, w_{t+1} will always lie inside the span of $\{x_1, x_2, \ldots, x_n\}$, whose dimension is smaller than d. In other words, gradient descent searches inside a space whose dimension is much smaller than the actual dimension of the entire space.

Problem setup: Suppose there is an underlying rank-r matrix $X^\star \in \mathbb{R}^{d\times d}$. In particular, for the illustration of the analysis, we shall focus on the case of $X^\star$ being positive semi-definite. As for the input to the problem, suppose we have m random Gaussian matrices whose entries are all drawn from $N(0,1)$, denoted as $A_1, A_2, \ldots, A_m$, all with dimension d by d. In addition, suppose they are symmetric, for example, we could transform each A_i as $(A_i + A_i^\top)/2$. Let $y_i = \langle A_i, UU^\top\rangle$, for $i = 1, 2, \ldots, n$.

A solution would be to find another positive semi-definite matrix $UU^\top$ so that the distance between $UU^\top$ and $X^\star$ is minimized. In particular, we shall consider the case that $U \in \mathbb{R}^{d\times d}$. Then, we set up a mean squared loss as follows:

$$\ell(U) = \frac{1}{n}\sum_{i=1}^{n}\Big(\langle UU^\top, A_i\rangle - y_i\Big)^2$$

In practice, we see that this loss function has many solutions that render its loss to near zero, even though $UU^\top$ can be quite far from $X^\star$. A key observation is that the generalization behavior of U should depend on the initialization. Next we shall make this precise by analyzing the gradient descent dynamic.

Let $U_0 = \alpha\,\mathrm{Id}$, for some small values of α. Then, let the gradient descent update be given by

$$U_{t+1} = U_t - \eta\nabla\ell(U_t) = (\mathrm{Id} - \eta M_t)U_t,$$

where

$$M_t = \frac{1}{m}\sum_{i=1}^{m}\langle A_i, U_tU_t^\top - X^\star\rangle A_i$$

Complete analysis in the rank-1 case: We shall restrict the case to $r = 1$, where we can write $X^\star = u^\star u^{\star\top}$, for some $u^\star \in \mathbb{R}^d$. Let $\mathrm{Id}_{u^\star} = \mathrm{Id} - \frac{u^\star u^{\star\top}}{\|u^\star\|^2}$. For simplicity, let's take the norm of $u^\star$ as 1. Then, we decompose the iterates U_t into two parts into the subspace of $u^\star$ and its complement:

$$\begin{aligned} U_t &= u^\star u^{\star\top}U_t + (\mathrm{Id} - \mathrm{Id}_{u^\star}\,u^{\star\top})U_t \\ &:= u^\star r_t^\top + E_t, \end{aligned}$$

where we denote by $r_t = U_t^\top u^\star$, and $E_t = (\mathrm{Id} - \mathrm{Id}_{u^\star})U_t$.

We will show that the spectral norm and the Frobenius norm of the "error term" E_t remains small throughout the iterations, whereas the "signal" term r_t grows exponentially fast (in the sense that the norm of r_t grows to the norm of $u^\star$.) Note that any solution with $\|r_t\| = 1$ and $E_t = 0$ will give exact recovery, and for the purpose of this section we will show that $\|r_t\|$ will converge approximately to 1 and E_t stays small.

Based on the gradient update above, we derive the update for E_t:

$$\begin{aligned} E_{t+1} &= (\mathrm{Id} - \mathrm{Id}_{u^\star})(\mathrm{Id} - \eta M_t)U_t \\ &= E_t - \eta(\mathrm{Id} - \mathrm{Id}_{u^\star})M_tU_t \end{aligned} \tag{33}$$

We shall also assume that the set of measurement matrices $(A_1, A_2, \ldots, A_m)$ satisfies the restricted isometry property. A set of linear measurement matrices $A_1, \ldots, A_m$ satisfies

(r, δ)-restricted isometry property (RIP) if for any d by d matrix X with rank at most r, we have that

$$(1-\delta)\left\|X\right\|_F^2 \leq \frac{1}{m}\sum_{i=1}^{m}\langle A_i, X\rangle^2 \leq (1+\delta)\left\|X\right\|_F^2 \tag{34}$$

In the following, we shall assume that the set of measurement matrices satisfies $(4, \delta)$-RIP with $\delta \leq c$ (for some small enough constant c). One corollary of the above definition is that for any matrices $X, Y \in \mathbb{R}^{d\times d}$ with rank at most r, we also have:

$$\left|\frac{1}{m}\sum_{i=1}^{m}\langle A_i, X\rangle\langle A_i, Y\rangle - \langle X, Y\rangle\right| \leq \delta\left\|X\right\|_F\left\|Y\right\|_F \tag{35}$$

If Y is at most rank r but X may not be, then we have the following instead:

$$\left|\frac{1}{m}\sum_{i=1}^{m}\langle A_i, X\rangle\langle A_i, Y\rangle - \langle X, Y\rangle\right| \leq \delta\left\|X\right\|_\star\left\|Y\right\|_F \tag{36}$$

Convergence of gradient descent from small initialization in over-parameterized matrix sensing: Suppose $\alpha \leq \delta\sqrt{\frac{1}{d}\log(\frac{1}{\delta})}$ and $\eta \lesssim c\delta^2\log^{-1}(\frac{1}{\delta\alpha})$. Then after $T = \Theta(\frac{(\alpha\delta)^{-1}}{\eta})$ iterations, we have that

$$\left\|U_T U_T^\top - X^\star\right\|_F \lesssim \delta\log(\delta^{-1}) \tag{37}$$

Error dynamics: Based on the update rule (33) for E_t, we have that

$$\left\|E_{t+1}\right\|_F^2 = \left\|E_t\right\|_F^2 - 2\eta\langle E_t, (\mathrm{Id}-\mathrm{Id}_{u^\star})M_tU_t\rangle + \eta^2\left\|(\mathrm{Id}-\mathrm{Id}_{u^\star})M_tU_t\right\|_F^2$$

When η is sufficiently small, and $\left\|M_t\right\|_F, \left\|U_t\right\|_2$ are both bounded from above, the third term on the right-hand side is negligible compared to the second term. Therefore, we focus on the second term.

$$\begin{aligned}\langle E_t, (\mathrm{Id}-\mathrm{Id}_{u^\star})M_tU_t\rangle &= \frac{1}{m}\sum_{i=1}^{m}\langle A_i, U_tU_t^\top - X^\star\rangle\langle A_i, (\mathrm{Id}-\mathrm{Id}_{u^\star})E_tU_t^\top\rangle \\ &= \frac{1}{m}\sum_{i=1}^{m}\langle A_i, U_tU_t^\top - X^\star\rangle\langle A_i, E_tU_t^\top\rangle\end{aligned}$$

We use (35) to analyze the above, but one gap here is that $U_tU_t^\top$ is not necessarily low-rank. Thus, we decompose it into a low-rank part and an error part with small trace norm.

$$\frac{1}{m}\sum_{i=1}^{m}\langle A_i, U_tU_t^\top - X^\star - E_tE_t^\top\rangle\langle A_i, E_tr_tu^{\star\top}\rangle \tag{38}$$

$$\geq\langle U_tU_t^\top - X^\star - E_tE_t^\top, E_tr_tu^{\star\top}\rangle - \delta\left\|U_tU_t^\top - X^\star - E_tE_t^\top\right\|_F\left\|E_tr_tu^{\star\top}\right\|_F \tag{39}$$

$$\geq\langle U_tU_t^\top - X^\star - E_tE_t^\top, E_tr_tu^{\star\top}\rangle - 2\delta\left\|E_tr_t\right\| \tag{40}$$

Above we use the fact that $\left\|U_tU_t^\top - X^\star E_tE_t^\top\right\|_F, \|r_t\|, \|u^\star\|$ are all bounded from above by 1. As for the $E_tE_t^\top$ part, we first note that $\langle A_i, E_tE_t^\top\rangle^2 \geq 0$. Thus, we can focus on the cross terms.

$$\begin{aligned}\frac{1}{m}\sum_{i=1}^m \langle A_i, E_tE_t^\top\rangle\langle A_i, E_tr_tu^{\star\top}\rangle &\geq \langle E_tE_t^\top, E_tr_tu^{\star\top}\rangle - \delta\left\|E_tE_t^\top\right\|_\star \left\|E_tr_tu^{\star\top}\right\| \\ &\geq \langle E_tE_t^\top, E_tr_tu^{\star\top}\rangle - \frac{\delta}{2}\|E_tr_t\|,\end{aligned}$$

where we are using the fact that $\left\|E_tE_t^\top\right\|_\star$ is bounded from above by some constant. The other cross term would be

$$\begin{aligned}&\frac{1}{m}\sum_{i=1}^m \langle A_i, E_tE_t^\top\rangle\langle A_i, U_tU_t^\top - X^\star - E_tE_t^\top\rangle \\ \geq&\langle E_tE_t^\top, U_tU_t^\top - X^\star - E_tE_t^\top\rangle - \delta\left\|E_tE_t^\top\right\|_\star \left\|U_tU_t^\top - X^\star - E_tE_t^\top\right\|_F \\ \geq&\langle E_tE_t^\top, U_tU_t^\top - X^\star\rangle - \delta\left\|E_tE_t^\top\right\|_\star\end{aligned}$$

Because $E_t^\top X^\star = 0$ and also $\left\|U_tU_t^\top - X^\star - E_tE_t^\top\right\|_F$ is bounded from above by 1. All put together, we can conclude that

$$\langle E_t, (\mathrm{Id} - \mathrm{Id}_{u^\star})M_tU_t\rangle \geq \langle U_tU_t^\top - X^\star, E_tU_t^\top\rangle - 2.5\delta\|E_tr_t\| - \delta\|E_t\|_F^2$$

Next, we have

$$\begin{aligned}\langle U_tU_t^\top - X^\star, E_tU_t^\top\rangle = \langle U_tU_t^\top, E_tU_t^\top\rangle &= \langle U_t^\top, U_t^\top E_tU_t^\top\rangle \\ &= \langle U_t^\top, E_t^\top E_tU_t^\top\rangle \\ &= \langle E_tU_t^\top, E_tU_t^\top\rangle \geq 0\end{aligned}$$

In summary, we have proved that

$$\|E_{t+1}\|_F^2 \leq (1+\eta\delta)\|E_t\|_F^2 + 2.5\delta\eta\|E_tr_t\| + O(\eta^2) \tag{41}$$

We could also control the growth of the largest singular value of E_t as

$$\begin{aligned}E_{t+1} &= E_t - \eta(\mathrm{Id} - \mathrm{Id}_{u^\star})M_tU_t &(42)\\ &= E_t - \eta(\mathrm{Id} - \mathrm{Id}_{u^\star})E_tU_t^\top U_t + \eta(\mathrm{Id} - \mathrm{Id}_{u^\star})(M_tU_t - E_tU_t^\top U_t) &(43)\end{aligned}$$

Next, notice that $(\mathrm{Id} - \mathrm{Id}_{u^\star})(U_tU_t^\top - X^\star)U_t = E_tU_t^\top U_t$, and

$$\begin{aligned}&\left\|(\mathrm{Id} - \mathrm{Id}_{u^\star})M_tU_t - (\mathrm{Id} - \mathrm{Id}_{u^\star})(U_tU_t^\top - X^\star)U_t\right\|_2 \\ \leq&4\delta\left(\left\|U_tU_t^\top - X^\star - E_tE_t^\top\right\|_F + \left\|E_tE_t^\top\right\|_\star\right)\|U_t\|_2 \\ \leq&8\delta\|U_t\|_2 \leq 8\delta(\|r_t\| + \|E_t\|_2)\end{aligned}$$

Here, we are using two more properties about RIP measurements: if X is at most rank r, then

$$\left\|\frac{1}{m}\sum_{i=1}^m \langle A_i, X\rangle A_iR - XR\right\|_2 \leq \delta\|X\|_F\|R\|_2 \tag{44}$$

In addition, for any X, we have

$$\left\| \frac{1}{m} \sum_{i=1}^{m} \langle A_i, X \rangle A_i R - XR \right\|_2 \leq \delta \|X\|_\star \|R\|_2 \tag{45}$$

The same is true if we have another matrix U where we insert before A_i:

$$\left\| \frac{1}{m} \sum_{i=1}^{m} \langle A_i, X \rangle U A_i R - UXR \right\|_2 \leq \delta \|U\|_2 \|X\|_\star \|R\|_2 \tag{46}$$

Therefore, we can conclude that

$$\begin{aligned} \|E_{t+1}\|_2 &\leq \left\| E_t(\mathrm{Id} - \eta U_t^\top U_t) \right\|_2 + 2\eta\delta(\|r_t\| + \|E_t\|_2) \\ &\leq (1 + 2\eta\delta) \|E_t\|_2 + 2\eta\delta \|r_t\| \end{aligned}$$

Signal dynamics: Next, we show that

$$\left\| r_{t+1} - (1 + \eta(1 - \|r_t\|^2)) r_t \right\| \leq \eta \|E_t\|_2^2 \|r_t\| + 2\eta\delta(\|E_t\|_2 + \|r_t\|) \tag{47}$$

By the update rule $U_{t+1} + (\mathrm{Id} - \eta M_t) U_t$. We have that

$$r_{t+1} = U_{t+1}^\top u^\star = U_t^\top (\mathrm{Id} - \eta M_t^\top) u^\star = r_t - \eta U_t^\top M_t^\top u^\star$$

Recall that $M_t = \frac{1}{m} \sum_{i=1}^{m} \langle A_i, U_t U_t^\top - X^\star \rangle A_i$. Using (44) and (46), we have that

$$\left\| r_{t+1} - (r_t - \eta U_t^\star (U_t U_t^\top - X^\star) u^\star) \right\| \leq \delta \left(\left\| U_t U_t^\top - X^\star - E_t E_t^\top \right\|_F + \left\| E_t E_t^\top \right\|_\star \right) \|U_t\|_2 \tag{48}$$

Here we observe that

$$U_t^\top (U_t U_t^\top - X^\star) u^\star = U_t^\top U_t r_t - r_t = (r_t r_t^\top + E_t^\top E_t) r_t - r_t = (\|r_t\|^2 - 1) r_t - E_t^\top E_t r_t \tag{49}$$

Therefore, we have that

$$\left\| r_{t+1} - (1 + \eta(1 - \|r_t\|^2)) r_t \right\| \leq \eta \left\| E_t^\top E_t r_t \right\| + 2\eta\delta \|U_t\|_2 \tag{50}$$

The above suggests that r_t will eventually drift towards a vector whose norm is equal to one.

Putting things together, we essentially showed that E_t will remain relatively smaller than r_t, since E_t only grows by a rate of $1 + \eta\delta$. Meanwhile, r_t grows by a rate of $1 + O(\eta)$, when it is still bounded away from 1.

3.4 Benign overfitting (Lecture 12)

Benign overfitting represents yet another approach to tackle the mystery behind the generalization of deep neural networks (Bartlett et al., 2020). The basic hypothesis behind benign overfitting stipulates that for a prediction rule $\hat{f}$, we can decompose it into

$$\hat{f} = \hat{f}_0 + \Delta,$$

where $\hat{f}_0$ is a simple component useful for prediction, Δ is a useful component useful for benign overfitting. While Δ is not useful for prediction, it is benign. In the setting of over-parameterized matrix sensing from the previous section, we see that $u^\star {r_t}^\top$ corresponds to $\hat{f}_0$, whereas E_t relates to Δ.

To better understand the phenomenon of benign overfitting, we shall now consider the case of linear regression.

- We denote the covariates as $x \in \mathbb{R}^d$ and the label as $y \in \mathbb{R}$.
- We assume that x is sub-Gaussian, meaning that there exists $c > 0$ such that $\Pr\left[\|x\|^2 > c\,\mathbb{E}\left[\|x\|^2\right]\right] \leq \delta$. Moreover, $\mathbb{E}[y \mid x] = x^\top \theta^\star$.
- Let us define

$$\Sigma = \mathbb{E}\left[xx^\top\right] = \sum_{i=1}^{d} \lambda_i v_i v_i^\top, \text{ for } i = 1, 2, \ldots, d$$
$$\theta^\star = \arg\min_\theta \mathbb{E}\left[(y - x^\top\theta)^2\right]$$
$$\sigma^2 = \mathbb{E}\left[(y - x^\top\theta^\star)^2\right]$$

A key concept here to understand benign overfitting is the *minimum norm interpolator*. Let $X \in \mathbb{R}^{n\times d}$ denote the features of n samples. Let $y = [y_1, y_2, \ldots, y_n]$ denote the labels. We have the estimator $\hat{\theta} = (X^\top X)^\dagger X^\top y$, which solves the problem

$$\min \|\theta\|^2$$
$$\text{s.t. } \|X\theta - y\|^2 = \min_\beta \|X\beta - y\|^2$$

Now we define the excess prediction risk of $\hat{\theta}$ as

$$\begin{aligned} R(\hat{\theta}) &:= \mathop{\mathbb{E}}_{x,y}\left[(y - x^\top\hat{\theta})^2\right] - \min_\theta \mathop{\mathbb{E}}_{x,y}\left[(y - x^\top\theta)^2\right] \\ &= \mathop{\mathbb{E}}_{x,y}\left[(y - x^\top\hat{\theta})^2 - (y - x^\top\theta^\star)^2\right] \\ &= (\hat{\theta} - \theta^\star)^\top \Sigma (\hat{\theta} - \theta^\star), \end{aligned}$$

where we are using the fact that $\mathbb{E}\left[xx^\top\right] = \Sigma$ and $\mathbb{E}[y|x] = x^\top\theta^\star$. Therefore, Σ determines the importance of parameter directions. Here is the question: when can the label noise be hidden in $\hat{\theta}$ without hurting the prediction accuracy?

Characterization of benign overfitting (Bartlett et al., 2020): There exists universal constants b, c such that the excess risk of the minimum norm interpolator satisfies:

$$R(\hat{\theta}) \leq c\left(\text{bias}(\theta^\star, \Sigma, n) + \sigma^2\left(\frac{k^\star}{n} + \frac{n}{R_{k^\star}(\Sigma)}\right)\right),$$

where $k^\star = \min\{k \geq 0 : r_k(\Sigma) \geq bn\}$, and

$$r_k(\Sigma) := \frac{\sum_{i>k}\lambda_i}{\lambda_{k+1}}$$
$$R_k(\Sigma) := \frac{\left(\sum_{i>k}\lambda_i\right)^2}{\sum_{i>k}\lambda_i^2}$$

And the bias is given by

$$\text{bias}(\theta^\star, \Sigma, n) = \left\|\theta^\star_{k+1:\infty}\right\|^2_{\Sigma_{k+1:\infty}} + \|\theta^\star_{1:k}\|^2_{\Sigma^{-1}_{1:k}}\left(\frac{\sum_{i>k}\lambda_i}{n}\right)^2$$

Recall that the benign overfitting prediction rule $\hat{f}$ decomposes as $\hat{f}_0 + \Delta$.

- $\hat{f}_0$ is the prediction component: $k^\star$-dimensional subspace corresponding to $\lambda_1, \ldots, \lambda_{k^\star}$
- Δ is the benign overfitting component: orthogonal subspace. Δ is benign only if $R_{k^\star} \gg n$

Intuition: To avoid harming prediction accuracy, the noise energy must be distributed across many unimportant directions.

3.5 PAC-Bayes generalization bounds (Lecture 13)

Next, we present a PAC-Bayes generalization bound that depends on the trace of the Hessian. Our bound can be related to the notion of trace norm, which can be viewed as a measure of model complexity. For instance, in the context of matrix recovery, the trace norm refers to the sum of the singular values of a matrix and links to the recovery quality of a matrix completion algorithm (Srebro and Shraibman, 2005).

To motivate this analysis, let's suppose that we have a pretrained model and we would like to fine-tune this model on a downstream task of interest. The weights of this pretrained model can be viewed as our prior belief of the target hypothesis in the PAC-Bayes analysis. Once we have learned a model through fine-tuning the pretrained model on the target task data, we can view this fine-tuned as our adjusted posterior in PAC-Bayes analysis.

More precisely, let $\mathcal{D} \subseteq \mathcal{X} \times \mathcal{Y}$ be an unknown data distribution, supported on the feature space $\mathcal{X}$ and the label space $\mathcal{Y}$. Given n random samples $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ drawn from $\mathcal{D}$, the empirical loss measured by loss function ℓ applied to a model f_W with $W \in \mathbb{R}^d$ is:

$$\hat{L}(W) = \frac{1}{n} \sum_{i=1}^{n} \ell(f_W(x_i), y_i).$$

The population loss is

$$L(W) = \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} \left[\ell(f_W(x), y)\right].$$

It is sufficient to think that the empirical loss is less than the population loss, and the goal is to bound the gap between $\hat{L}(W)$ and $L(W)$ from above (Shalev-Shwartz and Ben-David, 2014). With this setting in mind, we now describe several PAC-Bayesian bounds, which incorporate the prior into the analysis without assuming that the prior is "correct."

Occam bound: Here we assume a countable hypothesis class, and the algorithm outputs a single hypothesis. More specifically, let $\mathcal{H}$ be a countable hypothesis class. Let the loss function be bounded $\ell(z, h) \leq [0, 1]$. Let $\mathcal{P}$ be any prior distribution over $\mathcal{H}$. Then with probability at least $1 - \delta$, we have that

$$\forall h \in \mathcal{H}, L(h) \leq \hat{L}(h) + \sqrt{\frac{\log\left(\frac{1}{\mathcal{P}(h)}\right) + \log(\delta^{-1})}{2n}}$$

Proof: By Hoeffding's inequality, we have that for any fixed $h \in \mathcal{H}$:

$$\Pr[L(h) \geq \hat{L}(h) + \epsilon] \leq \exp(-2n\epsilon^2)$$

If we set the RHS to $\delta\mathcal{P}(h)$, then we get that with probability at least $\delta\mathcal{P}(h)$,

$$L(h) \geq \hat{L}(h) + \sqrt{\frac{\log\left(\frac{1}{\delta\mathcal{P}(h)}\right)}{2n}}$$

Applying the union bound across all of $h \in \mathcal{H}$, we have that with probability at most δ, there exists an $h \in \mathcal{H}$ such that the above is true. By taking the contra-positive, we have derived Occam's bound.

There are two things lacking about the Occam bound. It only applies to countable $\mathcal{H}$, which does not include the set of all weight vectors, for example. It only embraces half of the Bayesian story. While we have a prior $\mathcal{P}(h)$, only a single $h \in \mathcal{H}$ is returned rather than a full posterior $\mathcal{Q}(h)$. This is fixed by McAllester's bound.

PAC-Bayesian theorem (McAllester): Let the loss function be bounded between zero and one. Let $\mathcal{P}$ be any prior distribution over $\mathcal{H}$. Let $\mathcal{Q}$ be any posterior distribution over $\mathcal{H}$, which is a function of the training data. Then with probability at least $1 - \delta$,

$$\mathop{\mathbb{E}}_{h\sim\mathcal{Q}}[L(h)] \leq \mathop{\mathbb{E}}_{h\in\mathcal{Q}}\left[\hat{L}(h)\right] + \sqrt{\frac{KL(\mathcal{Q}||\mathcal{P}) + \log(2n\delta^{-1})}{2(n-1)}},$$

which holds for all posterior $\mathcal{Q}$.

Proof: For a prior $\mathcal{P}$, the change of measure inequality states that for any measure $\mathcal{Q}$,

$$\log \mathop{\mathbb{E}}_{f\in\mathcal{P}}[\exp(\phi(f))] \geq \mathop{\mathbb{E}}_{f\sim\mathcal{Q}}[\phi(f)] - D_{KL}(\mathcal{Q}||\mathcal{P}),$$

for any function $\phi : \mathcal{H} \to \mathbb{R}$. To see this, let us define a distribution $\mathcal{P}_G$ via its density with respect to $\mathcal{P}$ as:

$$\frac{d\mathcal{P}_G}{d\mathcal{P}}(f) = \frac{\exp(\phi(f))}{\mathbb{E}_{h\sim\mathcal{P}}[\exp(\phi(h))]}$$

This is called the Gibbs measure. Notice that

$$\begin{aligned}
D_{KL}(\mathcal{Q}||\mathcal{P}) - D_{KL}(\mathcal{Q}||\mathcal{P}_G) &= \int \left(\log\left(\frac{d\mathcal{Q}}{d\mathcal{P}}\right) - \log\left(\frac{d\mathcal{Q}}{d\mathcal{P}_G}\right)\right) d\mathcal{Q} \\
&= \int \log\left(\frac{d\mathcal{P}_G}{d\mathcal{P}}\right) d\mathcal{Q} \\
&= \mathop{\mathbb{E}}_{f\sim\mathcal{Q}}\left[\log\left(\frac{\exp(\phi(f))}{\mathbb{E}_{h\sim\mathcal{P}}[\exp(\phi(h))]}\right)\right] \\
&= \mathop{\mathbb{E}}_{f\sim\mathcal{Q}}[\phi(f)] - \log \mathop{\mathbb{E}}_{h\sim\mathcal{P}}[\exp(\phi(h))]
\end{aligned}$$

Thus,

$$\begin{aligned}
\log \mathop{\mathbb{E}}_{h\sim\mathcal{P}}[\exp(\phi(h))] &= \mathop{\mathbb{E}}_{f\sim\mathcal{Q}}[\phi(f)] - D_{KL}(\mathcal{Q}||\mathcal{P}) + D_{KL}(\mathcal{Q}||\mathcal{P}_G) \\
&\geq \mathop{\mathbb{E}}_{f\sim\mathcal{Q}}[\phi(f)] - D_{KL}(\mathcal{Q}||\mathcal{P}),
\end{aligned}$$

since the KL divergence is always non-negative. Notice that the inequality holds if and only if $\mathcal{Q} = \mathcal{P}_G$.

Next, in order to prove McAllester's bound, we consider the function $2(n-1)(\Delta(f))^2$ as ϕ, where $\Delta(f) = L(f) - \hat{L}_n(f)$. Applying this to the above result gives

$$\mathop{\mathbb{E}}_{h\sim\mathcal{P}}\left[\exp\left(2(n-1)(\Delta(h))^2\right)\right] \geq \exp\left(2(n-1)\mathop{\mathbb{E}}_{f\sim\mathcal{Q}}\left[(\Delta(f))^2\right] - D_{KL}(\mathcal{Q}||\mathcal{P})\right)$$

Let us take the expectation over the training sample set S (recall that $\mathcal{Q}$ depends on the training samples) on both sides and notice that the prior $\mathcal{P}$ does not depend on the training samples. We get

$$\mathop{\mathbb{E}}_{S}\left[\mathop{\mathbb{E}}_{h\sim\mathcal{P}}\left[\exp(2(n-1)(\Delta(h))^2)\right]\right] \geq \mathop{\mathbb{E}}_{S}\left[\exp\left(2(n-1)\mathop{\mathbb{E}}_{f\sim\mathcal{Q}}\left[(\Delta(f))^2\right] - D_{KL}(\mathcal{Q}||\mathcal{P})\right)\right]$$

For a fixed f, we can bound $\Delta(f)$ using Hoeffding as:

$$\Pr[|\Delta(f)| \geq \epsilon] \leq 2\exp(-2n\epsilon^2),$$

which holds for any positive $\epsilon < 1$. Therefore, we have

$$\begin{aligned}
\mathop{\mathbb{E}}_{S}\left[\exp(2(n-1)(\Delta(f))^2)\right] &= \int_0^\infty \Pr\left[\exp\left(2(n-1)(\Delta(f))^2\right) \geq s\right] ds \\
&\leq 1 + \int_1^\infty \Pr\left[|\Delta(f)| \geq \sqrt{\frac{\log(s)}{2(n-1)}}\right] ds \\
&\leq 1 + \int_1^\infty 2\exp\left(-2n\cdot\frac{\log(s)}{2(n-1)}\right) ds \\
&= 1 + 2(n-1) \leq 2n
\end{aligned}$$

As for the last step, we notice that

$$-\int_1^\infty s^{-1-\frac{1}{n-1}} ds = -\int_1^\infty d\left((n-1)s^{-\frac{1}{n-1}}\right) = n-1$$

Then, by Markov's inequality,

$$\Pr\left[\exp\left(2(n-1)\mathop{\mathbb{E}}_{f\sim\mathcal{Q}}\left[(\Delta(f))^2\right] - D_{KL}(\mathcal{Q}||\mathcal{P})\right) \geq \frac{2n}{\delta}\right] \leq \delta$$

That is, with probability at least $1-\delta$,

$$2(n-1)\mathop{\mathbb{E}}_{f\sim\mathcal{Q}}\left[(\Delta(f))^2\right] \leq \log\left(\frac{2n}{\delta}\right) + D_{KL}(\mathcal{Q}||\mathcal{P})$$

Using Jensen's inequality, the left is greater than

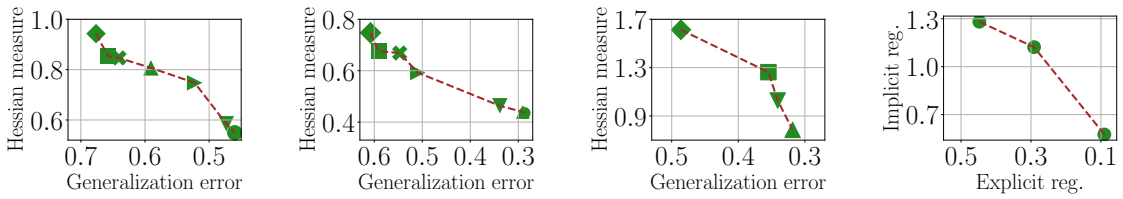
$$2(n-1)\left(\mathop{\mathbb{E}}_{f\sim\mathcal{Q}}\left[\Delta(f)\right]\right)^2$$

Taking square root on both sides gives us the McAllester's bound.

PAC-Bayesian bound (Catoni): Another version of the PAC-Bayesian theorem is shown by Catoni, 2007 (see also Theorem 3.5, Alquier, 2021). Suppose the loss function $\ell(f_W(x), y)$ lies in a bounded range $[0, C]$ given any $x \in \mathcal{X}$ with label y. For any $\beta \in (0,1)$ and $\delta \in (0,1)$, with probability at least $1-\delta$, the following holds:

$$L_{\mathcal{Q}}(W) \leq \frac{1}{\beta}\hat{L}_{\mathcal{Q}}(W) + \frac{C\left(KL(\mathcal{Q}||\mathcal{P}) + \log\frac{2\sqrt{n}}{\delta}\right)}{2\beta(1-\beta)n}, \tag{51}$$

where $\mathcal{P}$ refers to the *prior* distribution, C refers to the upper bound on the loss value ℓ. For analyzing fine-tuning, we view $\mathcal{P}$ as centered at the pretrained model, with covariance matrix $\sigma^2\,\mathrm{Id}_p$.



(a) ResNet-50 (b) ResNet-50 (c) BERT-Base (d) Average $\mathcal{D}_i$

Figure 3: 3a, 3b, 3c: The Hessian measures accurately correlate with empirical generalization errors for seven fine-tuning methods. 3d: $\mathcal{D}_i$ is smaller for fine-tuning with explicit distance-based regularization than implicit early-stopped regularization. Each dot represents the generalization error and the Hessian measures for one fine-tuned model using a particular regularization method.

3.6 Noise sensitivity generalization bounds (Lecture 14)

Let W be any learned hypothesis within the hypothesis space, denoted as $\mathcal{H}$. This generalization bound will apply uniformly to W within the hypothesis space. We state this result, including the required assumptions, as follows. Assume that the loss function ℓ is bounded between 0 and C for a fixed constant $C > 0$ on the data distribution $\mathcal{D}$. Suppose $\ell(f_W(\cdot), \cdot)$ is twice-differentiable in W and the Hessian matrix $\nabla^2[\ell(f_W(\cdot), \cdot)]$ is Lipschitz continuous within the hypothesis space. Suppose for any W in $\mathcal{H}$, the trace norm of the Hessian is less than α:

$$\alpha := \max_{W \in \mathcal{H}} \max_{(x,y) \sim \mathcal{D}} \operatorname{Tr}\left[\nabla^2 \ell(f_W(x), y)\right], \tag{52}$$

and the ℓ_2-norm of W is at most r for any $W \in \mathcal{H}$. Then, for any W in $\mathcal{H}$, with probability at least $1 - \delta$ for any $\delta > 0$, the following must hold, for any ϵ close to zero:

$$L(W) \leq (1+\epsilon)\hat{L}(W) + (1+\epsilon)\sqrt{\frac{C\alpha r^2}{n}} + O\Big(n^{-\frac{3}{4}} \log(\delta^{-1})\Big). \tag{53}$$

An important takeaway of this bound is that we could now measure the RHS in practice, and this leads to a non-vacuous bound on the generalization error. We experiment with seven methods, including fine-tuning with and without early stopping, distance-based regularization, label smoothing, mixup, etc. Figure 3 shows that the Hessian measure (i.e., the second part of equation (53)) correlates with the generalization errors of these methods. Second, we plot the Hessian measure (averaged over all layers) between fine-tuning with implicit (early stopping) regularization and explicit (distance-based) regularization. We find that the Hessian measure is smaller for explicit regularization than implicit regularization.

Proof Sketch: We provide a high-level illustration of the proof of equation (52). Let $\mathcal{Q}$ denote the *posterior* distribution. Specifically, we consider $\mathcal{Q}$ as being centered at the learned hypothesis W (which could be anywhere within the hypothesis space), given by a Gaussian distribution $\mathcal{N}(W, \sigma^2 \operatorname{Id}_p)$, where Id_p denotes the p by p identity matrix. Given a sample $U \sim \mathcal{N}(0, \sigma^2 \operatorname{Id}_p)$, let the perturbed loss be given by

$$\ell_{\mathcal{Q}}(f_W(x), y) = \mathop{\mathbb{E}}_{U}\left[\ell(f_{W+U}(x), y)\right]. \tag{54}$$

Then, let $\hat{L}_{\mathcal{Q}}(W)$ be the averaged value of $\ell_{\mathcal{Q}}(f_W(\cdot), \cdot)$, taken over n empirical samples from the training dataset. Likewise, let $L_{\mathcal{Q}}(W)$ be the population average of $\ell_{\mathcal{Q}}(f_W(\cdot), \cdot)$, in expectation over an unseen data sample from the underlying data distribution.

By Taylor's expansion of $\ell_{\mathcal{Q}}$, we show that:

$$L_{\mathcal{Q}}(W) = L(W) + \frac{\sigma^2}{2} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} \left[\mathrm{Tr}\left[\nabla^2 \ell(f_W(x), y)\right]\right] + O(\sigma^3) \tag{55}$$

$$\hat{L}_{\mathcal{Q}}(W) = \hat{L}(W) + \frac{\sigma^2}{2n} \sum_{i=1}^{n} \mathrm{Tr}\left[\nabla^2 \ell(f_W(x_i), y_i)\right] + O(\sigma^3). \tag{56}$$

Since the Hessian operator is Lipschitz continuous by assumption, we can bound the gap between the above two quantities with ϵ-covering arguments. By plugging in these results back to the PAC-Bayes bound of equation (51), after some calculation, we can get:

$$L(W) \leq \frac{1}{\beta}\hat{L}(W) + \frac{\sigma^2(1-\beta)\alpha}{2\beta} + \frac{Cr^2/2\sigma^2}{2\beta(1-\beta)n} + O\left(\sigma^3 + \frac{\sigma^2\sqrt{p}}{\sqrt{n}} + \frac{\log(\delta^{-1})}{n}\right). \tag{57}$$

In particular, the above uses the fact that the ℓ_2-norm of W is less than r for any $W \in \mathcal{H}$. The KL divergence is discussed below. Suppose $\mathcal{P} = N(X, \Sigma)$ and $\mathcal{Q} = N(Y, \Sigma)$ are both Gaussian distributions with mean vectors given by $X \in \mathbb{R}^p, Y \in \mathbb{R}^p$, and population covariance matrix $\Sigma \in \mathbb{R}^{p\times p}$. The KL divergence between $\mathcal{P}$ and $\mathcal{Q}$ is equal to

$$KL(\mathcal{Q}||\mathcal{P}) = \frac{1}{2}(X-Y)^\top \Sigma^{-1}(X-Y).$$

Specifically, if $\Sigma = \sigma^2 \,\mathrm{Id}_p$, then the above simplifies to

$$KL(\mathcal{Q}||\mathcal{P}) = \frac{\|X-Y\|_2^2}{2\sigma^2}.$$

By choosing σ^2 and β to minimize equation (57), we will obtain equation (53). This summarizes the high-level proof idea.

4 Statistical modeling of representation learning

In many modern ML settings, one would like to use a large, pretrained model, to solve a downstream task. Instead of training from scratch, these models provide a representation of the input that are then used to make predictions on the input data. While this paradigm has been quite useful in practice, practitioners have also observed problematic outcomes. For instance, negative transfer refers to a situation where transfer learning from a source task to predict a target task negatively hurts the prediction performance, as compared to learning using the target task data alone. However, exactly how and why negative transfer occurs has not been fully resolved — Next, we provide a rigorous study of this phenomenon in the setting of transfer learning linear regression.

4.1 Transfer learning linear regression

We first introduce the problem setup. Suppose there are n_1 samples drawn from a p-dimensional distribution D_1 with real-valued labels in a *source* task. Suppose there are n_2 samples drawn from a p-dimensional distribution D_2 with real-valued labels in the *target* task. Let $x_1^{(i)}, x_2^{(i)}, \ldots, x_{n_i}^{(i)}$ denote feature covariates. Let $y_1^{(i)}, y_2^{(i)}, \ldots, y_{n_i}^{(i)}$ be the prediction labels. We assume a linear model specified by an unknown parameter vector $\beta^{(i)} \in \mathbb{R}^p$:

$$y_j^{(i)} = \left(x_j^{(i)}\right)^\top \beta^{(i)} + \varepsilon_j^{(i)}, \quad \text{for any} \quad j = 1, \ldots, n_i,$$

where $\varepsilon_j^{(i)} \in \mathbb{R}$ denotes a random noise variable with mean zero and variance σ^2. We refer to the first task as the *source* task and the second task as the *target* task for ease of presentation. We assume $n_2 > p$ while n_1 can be either less or greater than p.

The design matrix. In linear regression, there is typically a distinction between fixed designs vs. random designs. Fixed designs refer to a situation where the covariates are deterministic, e.g., think of the population demographic data for one example. By contrast, random designs correspond to cases where the covariates may involve observational uncertainties. For this section, we shall study random designs where the covariates are given by

$$x_j^{(i)} = \left(\Sigma^{(i)}\right)^{\frac{1}{2}} z_j^{(i)}, \text{ for } i \in \{1, 2\} \text{ and for any } j = 1, \ldots, n_i,$$

where $z_j^{(i)}$ consists of independent and identically distributed entries of mean zero and variance one, and $\Sigma^{(1)} \in \mathbb{R}^{p \times p}$ and $\Sigma^{(2)} \in \mathbb{R}^{p \times p}$ denote the population covariance matrices. Let $X^{(i)} = (x_1^{(i)}, \ldots, x_{n_i}^{(i)})^\top \in \mathbb{R}^{n_i \times p}$ denote a matrix that corresponds to all the covariates of task i, for $i \in \{1, 2\}$.

Let $Y^{(1)} \in \mathbb{R}^{n_1}$, $Y^{(2)} \in \mathbb{R}^{n_2}$ be the vectors of labels. Then, consider the following objective, parameterized by a shared variable $B \in \mathbb{R}^p$:

$$\ell(B) = \left\| X^{(1)} B - Y^{(1)} \right\|^2 + \left\| X^{(2)} B - Y^{(2)} \right\|^2. \tag{58}$$

Notice that B provides a shared feature vector for both tasks. This way of estimation is consistent with the hard parameter sharing (HPS) architecture that has been used in the earlier literature (Caruana, 1997). Thus, we shall call the minimizer of $\ell(B)$ the HPS estimator, denoted as $\hat{\beta}_2^{\text{HPS}}$ for short.

Bias and variance: The risk of $\hat{\beta}_i^{\text{HPS}}$ over an unseen sample (x, y) of the target task i is given by (under the mean squared loss):

$$\mathop{\mathbb{E}}_{x,y}\left[\left\|y - x^\top \hat{\beta}_i^{\text{HPS}}\right\|^2\right] = \left\|\Sigma^{(i)\,\frac{1}{2}}\left(\hat{\beta}_i^{\text{HPS}} - \beta^{(i)}\right)\right\|^2 + \sigma^2.$$

The excess risk is the difference between the above risk and the expected risk of the population risk optimizer:

$$L(\hat{\beta}_i^{\text{HPS}}) := \left\|\Sigma^{(i)\,\frac{1}{2}}\left(\hat{\beta}_i^{\text{HPS}} - \beta^{(i)}\right)\right\|^2. \tag{59}$$

Next, we present a bias-variance decomposition of the excess risk of HPS. Denote the sum of both tasks' sample covariance matrix as:

$$\hat{\Sigma} = {X^{(1)}}^\top X^{(1)} + {X^{(2)}}^\top X^{(2)}. \tag{60}$$

The bias and variance formulas are defined as

$$L_{\text{bias}} = \left\|\Sigma^{(2)\,\frac{1}{2}}\hat{\Sigma}^{-1}\left({X^{(1)}}^\top X^{(1)}\right)\left(\beta^{(1)} - \beta^{(2)}\right)\right\|^2, \tag{61}$$

$$L_{\text{var}} = \sigma^2\,\text{Tr}\left[\Sigma^{(2)}\hat{\Sigma}^{-1}\right]. \tag{62}$$

Proof: The proof of the above is based on algebraic calculations. By minimizing $\ell(B)$ in (59), we obtain the minimizer as

$$\hat{\beta}_2^{\text{HPS}} = \left((X^{(1)})^\top X^{(1)} + (X^{(2)})^\top X^{(2)}\right)^{-1}\left((X^{(1)})^\top Y^{(1)} + (X^{(2)})^\top Y^{(2)}\right)$$

By simplifying the above a little bit, we get

$$\begin{aligned}\hat{\beta}_2^{\text{HPS}} = \beta^{(2)} &+ \left((X^{(1)})^\top X^{(1)} + (X^{(2)})^\top X^{(2)}\right)^{-1}\left((X^{(1)})^\top \varepsilon^{(1)} + (X^{(2)})^\top \varepsilon^{(2)}\right) \\ &+ \left((X^{(1)})^\top X^{(1)} + (X^{(2)})^\top X^{(2)}\right)^{-1}(X^{(1)})^\top X^{(1)}(\beta^{(1)} - \beta^{(2)})\end{aligned}$$

Plugging this into the excess risk, we get that in expectation over the randomness of $\varepsilon^{(1)}$ and $\varepsilon^{(2)}$,

$$\mathbb{E}\left[L(\hat{\beta}_2^{\text{HPS}})\right] = L_{\text{bias}} + L_{\text{var}}$$

Implications: We can now compare the above bias-variance to that of single-task learning on the target task. That would just be

$$\sigma^2\,\text{Tr}\left[\Sigma^{(2)}((X^{(2)})^\top X^{(2)})^{-1}\right],$$

which is the variance of the ordinary least squares (OLS) estimator for the target task. The bias and variance decomposition above allows us to reason about transfer effects by comparing the bias and variance of HPS with that of OLS. The bias of HPS is always larger than that of OLS. On the other hand, the variance of HPS is always lower than that of OLS. By Woodbury matrix identity, the variance of OLS is always higher than formula (62):

$$\text{Tr}\left[\Sigma^{(2)}\left({X^{(2)}}^\top X^{(2)}\right)^{-1}\right] \geq \text{Tr}\left[\Sigma^{(2)}\hat{\Sigma}^{-1}\right]. \tag{63}$$

Thus, the transfer effect of HPS is determined by the bias-variance decomposition: the bias always increases while the variance always decreases. Whether or not the transfer effect of HPS is positive depends on which effect dominates.

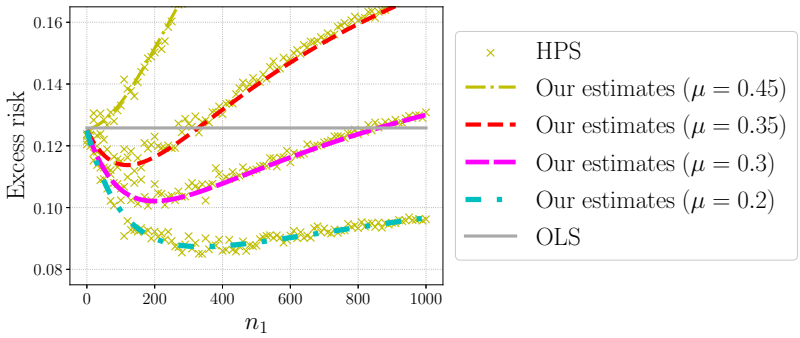


Figure 4: We illustrate the performance of HPS vs. OLS (ordinary least squares) for different levels of model shifts, indicated by $\mu \approx \left\|\beta^{(1)} - \beta^{(2)}\right\| / \sqrt{2}$. We consider two linear regression tasks and vary the sample size of the source task, denoted as n_1, and μ, while holding the target task as fixed. The gray line refers to the performance of OLS. Any point above the gray line represents negative transfer, while any point below represents positive transfer. In this simulation, we set $p = 100$, $n_2 = 300$, and $\sigma^2 = \frac{1}{4}$. We sample the covariates X from a p-dimensional isotropic Gaussian and plot the excess risk of HPS (see equation (59)).

Numerical example: We will consider a random-effects model. Each $\beta^{(i)}$ consists of two components, in this case, one shared by all tasks and one that is task-specific. Let β_0 be the shared component and γ_i be the i-th task-specific component. For any i, the i-th model vector is equal to $\beta^{(i)} = \beta_0 + \gamma_i$. The entries of the task-specific component γ_i are drawn independently from a Gaussian distribution with mean zero and variance $p^{-1}\mu^2$, for a parameter $\mu > 0$. In expectation, the Euclidean distance between the two model vectors is equal to $2\mu^2$.

Then, we have the following phase transition in the random-effects model. With high probability over the randomness of training samples and model vectors.

1. If $\mu^2 \frac{\mathrm{Tr}\left[\Sigma^{(1)}\right]}{p} \leq \frac{\sigma^2 p}{2(n_2 - p)}$, then the transfer effect is always positive:

$$L(\hat{\beta}_2^{\mathrm{HPS}}) \leq L(\hat{\beta}_2^{\mathrm{OLS}}). \tag{64}$$

2. If $\frac{\sigma^2 p}{2(n_2-p)} < \mu^2 \frac{\mathrm{Tr}\left[\Sigma^{(1)}\right]}{p} < \frac{\sigma^2 n_2}{2(n_2-p)}$, then there exists a deterministic value $n_0 > 0$ (which may not be an integer) such that if $n_1 \leq n_0$, then equation (64) holds; otherwise if $n_1 > n_0$, then

$$L(\hat{\beta}_2^{\mathrm{OLS}}) \leq L(\hat{\beta}_2^{\mathrm{HPS}}). \tag{65}$$

3. If $\mu^2 \frac{\mathrm{Tr}\left[\Sigma^{(1)}\right]}{p} \geq \frac{\sigma^2 n_2}{2(n_2-p)}$, then equation (65) holds for any $n_1 > p$.

Figure 4 demonstrates an illustration of this tradeoff.

4.1.1 Minimax lower bounds

The above estimator seems like a natural choice in practice. However, how would we know if there won't be another estimator that is much better the above? Concretely, suppose we are trying to estimate an unknown parameter denoted as $\beta^{(2)}$. We are given n_2 samples

drawn from a linear parametric model, following $\beta^{(2)}$, with isotropic Gaussian covariates $X^{(2)}$, contaminated by Gaussian noise with variance σ^2. Then, we are given n_1 samples from another linear parametric model, following $\beta^{(1)}$, again with isotropic Gaussian covariates $X^{(1)}$, contaminated by Gaussian noise with variance σ^2. The parameter vectors belong to the set

$$\Theta(\mu) = \left\{ \beta^{(1)} \in \mathbb{R}^p, \beta^{(2)} \in \mathbb{R}^p : \|\beta^{(1)} - \beta^{(2)}\| \leq \mu, \|\beta^{(1)}\| \leq 1, \|\beta^{(2)}\| \leq 1 \right\}.$$

Let $\hat{\beta}$ be any estimation procedure that, given the above $n_1 + n_2$ samples, produces an estimate of the unknown vector $\beta^{(2)}$. We prove the following minimax rate on the estimation error of $\hat{\beta}$. Assume that $n_1 \geq (1+\tau)p$ and $n_2 \geq (1+\tau)p$ for a constant $\tau > 0$. For any $(\beta^{(1)}, \beta^{(2)})$ within the set $\Theta(\mu)$, we have that

$$\inf_{\hat{\beta}} \sup_{\Theta(\mu)} \mathbb{E}\left[\frac{1}{n_2} \left\| \widetilde{X}^{(2)} \left(\hat{\beta} - \beta^{(2)} \right) \right\|^2 \right] \geq c\Big(\min\Big(\frac{n_1^2 \mu^2}{(n_1+n_2)^2}, \frac{\sigma^2 p}{n_2} \Big) + \frac{\sigma^2 p}{n_1 + n_2} \Big), \tag{66}$$

where the expectation is over the randomness of $X^{(1)}, X^{(2)}, Y^{(1)}, Y^{(2)}$ and an independently drawn $\widetilde{X}^{(2)}$ that follows the same distribution as $X^{(2)}$. c is a fixed constant that does not grow with p.

The lower bound in the right-hand side of equation (66) involves two parts:

- For the first part, if μ is large, the source samples are not helpful, so the rate $\frac{\sigma^2 p}{n_2}$ is the OLS rate using only the target task samples. If μ is small and n_1 is large, then the rate μ^2 appears if we use the OLS estimator for the source task.
- For the second part, $\frac{\sigma^2 p}{n_1+n_2}$ is the rate in the case without model drift, i.e., $\beta^{(1)} = \beta^{(2)}$.

Takeaway: It turns out that the naive estimator described above is nearly optimal, with one caveat. When μ is very high, then we essentially throw away the source task data. Otherwise when μ is very small, then we combine source and target task data as HPS. It can be shown that this adjusted HPS estimator now matches the minimax lower bound.

Proof sketch: The proof of the minimax lower bound is via a covering argument. In particular, we design a cover of the parameter vectors, taking into account the effect of model shifts (Wainwright, 2019).

4.1.2 Extension to multiple tasks

Suppose there are t tasks whose feature covariates are all equal to $X \in \mathbb{R}^{n \times p}$. The label vector of the i-th task follows a linear model with an unknown p dimensional vector $\beta^{(i)}$, for $i = 1, 2, \ldots, t$:

$$Y^{(i)} = X\beta^{(i)} + \varepsilon^{(i)}. \tag{67}$$

We assume that $X = Z\Sigma^{\frac{1}{2}} \in \mathbb{R}^{n \times p}$ is a random matrix.

We combine the samples from all the tasks with a shared feature matrix B and a task-specific prediction vector for each task. Specifically, let $B \in \mathbb{R}^{p \times r}$ denote the shared feature matrix and let $A = [A_1, A_2, \ldots, A_t] \in \mathbb{R}^{r \times t}$ denote the combined prediction variables. We remark that unlike the two-task setting concerning covariate shifts, we have both A and B as

part of the model in the multi-task setting. The reason is two-fold. First, in this paper, we focus on the underparameterized setting. Thus, in the two-task setting, one can show that the optimal rank of B would be one, i.e., $r = 1$, in which case B becomes a scalar. Second, to tackle more than two tasks, we need to incorporate more of the structural information from the other tasks into the model (Ando and Zhang, 2005), which is encoded as the low-rank structures of A and B. By contrast, in the two-task setting, we can focus on the setting where we have a shared parameter vector for both tasks.

We can now write down the loss objective as follows:

$$\ell(A, B) = \sum_{j=1}^{t} \left\| XBA_j - Y^{(j)} \right\|^2, \tag{68}$$

In particular, we focus on the case where $r < t$. Otherwise, if $r \geq t$, the problem reduces to single-task learning (for reference, see Proposition 1 from (Wu et al., 2020)).

Let $(\hat{A}, \hat{B})$ denote the global minimizer of $\ell(A, B)$. In particular, we compute the global minimizer of $\ell(A, B)$ by first solving $\hat{B}$ as a function of $\hat{A}$—this turns out to be $(X^\top X)^{-1} X^\top Y$ multiplied by $A^\top (AA^\top)^+$, with $(AA^\top)^+$ denoting the pseudoinverse of $AA^\top$. Next, we could find the optimal $\hat{A}$ by taking the rank-r SVD of $Y^\top X (X^\top X)^{-1} X^\top Y$ to get the leading r left singular vectors as $U_r \in \mathbb{R}^{t \times r}$. Then, we can obtain $\hat{B}\hat{A}$ as $(X^\top X)^{-1} X^\top Y U_r U_r^\top$.

Deriving the HPS estimator: For the optimization objective $\ell(A, B)$ in equation (68), using the local optimality condition $\frac{\partial \ell}{\partial B} = 0$, we can obtain $\hat{B}$ as a function of A:

$$\hat{B}(A) = (X^\top X)^{-1} X^\top Y A^\top (AA^\top)^+, \tag{69}$$

where $Y := [Y^{(1)}, Y^{(2)}, \ldots, Y^{(t)}]$ and $(AA^\top)^+$ denotes the pseudoinverse of $AA^\top$. Plugging $\hat{B}(A)$ into equation (68), we obtain the following objective that depends only on A (in matrix notation):

$$g(A) = \left\| X (X^\top X)^{-1} X^\top Y A^\top (AA^\top)^+ A - Y \right\|_F^2. \tag{70}$$

Note that $A^\top (AA^\top)^+ A$ is a projection onto the subspace spanned by the rows of A. For simplicity, we write it into the form

$$A^\top (AA^\top)^+ A = U_A U_A^\top,$$

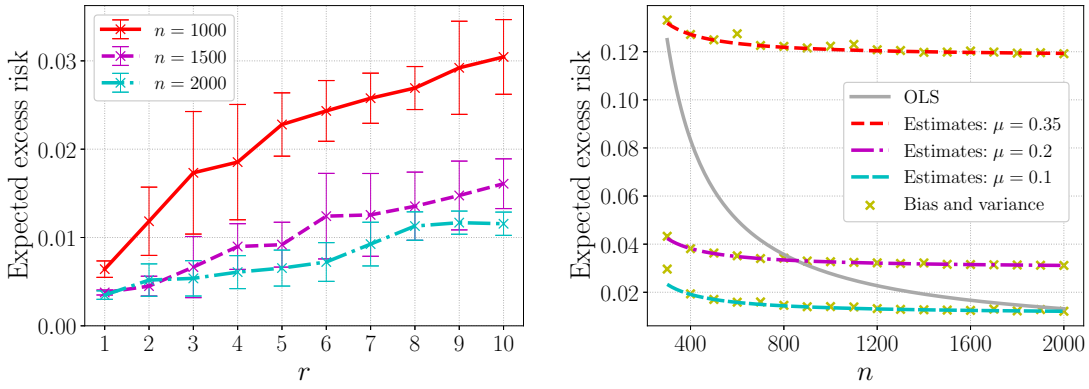
where $U_A \in \mathbb{R}^{t \times r}$ is a $t \times r$ partial orthonormal matrix (i.e. $U_A^\top U_A = \mathrm{Id}_{r \times r}$). Hence, we also denote the function $g(A)$ by $g(U_A)$.

Now, to find the optimal solution for U_A, we write $X(X^\top X)^{-1} X^\top = P_X$ (which is an n by n projection matrix of rank p), and expand $g(A)$ as:

$$\begin{aligned} g(A) = \left\| P_X Y U_A U_A^\top - Y \right\|_F^2 &= \left\| P_X Y U_A U_A^\top \right\|_F^2 + \|Y\|_F^2 - 2\langle P_X Y U_A U_A^\top, Y \rangle \\ &= \|Y\|_F^2 - \langle P_X Y U_A U_A^\top, Y \rangle = \|Y\|_F^2 - \langle U_A U_A^\top, Y^\top P_X Y \rangle, \end{aligned}$$

where we use the fact that $P_X^2 = P_X$ since it is a projection matrix and $U_A^\top U_A = \mathrm{Id}_{r \times r}$. Thus, to minimize $g(A)$, we just need to maximize the inner project between $Y^\top P_X Y$ and $U_A U_A^\top$—this can be achieved by finding the best rank-r SVD of $Y^\top P_X Y$ and taking the top-r singular vectors, which form the partial orthonormal matrix $U_r = U_{\hat{A}}$. Lastly, we can insert U_r as A back to $\hat{B}A$ to obtain $\hat{B}\hat{A}$ as the HPS estimators for all tasks from 1 to t.

We illustrate the behavior of the HPS estimator in the multitask setting as follows.



(a) Varying the rank of the shared matrix B

(b) Varying model shift μ

Figure 5: Illustration of transfer effects in the random-effects model with multiple source tasks. In particular, μ indicates the extent of model shift — higher values of μ means greater differences between model vectors across tasks. Figure 5a shows that for rank r from 1 to 10, the lowest excess risk of task t is achieved at $r = 1$ (averaged over three random seeds). Figure 5b fixes $r = 1$ and varies μ, n. Similar to Figure 4, the transfer effect of HPS can be either positive or negative depending on μ, n.

4.2 A brief foray into reinforcement learning

In reinforcement learning, the interactions between the agent and the environment are often described by an infinite horizon, discounted Markov decision process (MDP), specified by:

- A state space $\mathcal{S}$, which may be finite or infinite. For mathematical convenience, we will assume that $\mathcal{S}$ is finite or countably infinite.
- An action space $\mathcal{A}$, which may also be discrete or infinite.
- A transition function $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$, where $\Delta(\mathcal{S})$ is the space of probability distributions over $\mathcal{S}$ (i.e., the probability simplex). $P(s'|s, a)$ is the probability of transitioning into state s' upon taking action a in state s. We use $P_{s,a}$ to denote the vector $P(\cdot|s, a)$.
- A reward function $r : \mathcal{S} \times \mathcal{A} \to [0, 1]$, where $r(s, a)$ is the immediate reward associated with taking action a in state s. More generally, $r(s, a)$ could be a random variable where the distribution depends on s, a.
- A discount factor $\gamma \in [0, 1)$, which defines a horizon for the problem.
- An initial state distribution $\mu \in \Delta(\mathcal{S})$, which specifies how the initial state s_0 is generated.

The objective, policies, and values. In a given MDP $M = (\mathcal{S}, \mathcal{A}, P, r, \gamma, \mu)$, the agent interacts with the environment according to the following protocol. The agent starts at some state $s_0 \sim \mu$; at each time step $t = 0, 1, 2, \ldots$, the agent takes an action $a_t \in \mathcal{A}$, obtains the immediate reward $r_t = r(s_t, a_t)$, and observes the next state s_{t+1} sampled according to $s_{t+1} \sim P(\cdot|s_t, a_t)$. The interaction record at time t,

$$r_t = (s_0, a_0, r_0, s_1, \ldots, s_t, a_t, r_t)$$

is called a trajectory, which includes the observed state at time t.

In the most general setting, a policy specifies a decision-making strategy in which the agent chooses actions adaptively based on the history of observations. Precisely, a policy is a mapping from a trajectory to an action, i.e., $\pi : \mathcal{H} \to \Delta(\mathcal{A})$ where $\mathcal{H}$ is the set of all possible trajectories (of all lengths) and $\Delta(\mathcal{A})$ is the space of probability distributions over $\mathcal{A}$.

Values. We now define values for general policies. For a fixed policy and a starting state $s_0 = s$, we define the value function $V_M^\pi : \mathcal{S} \to \mathbb{R}$ as the discounted sum of future rewards

$$V_M^\pi(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | \pi, s_0 = s\right]$$

where expectation is with respect to the randomness of the trajectory, that is, randomness in state transitions and stochasticity of π.

Similarly, the action-value (or Q-value) function $Q_M^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is defined as

$$Q_M^\pi(s, a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | \pi, s_0 = s, a_0 = a\right]$$

Goal. Given a state s, the goal of the agent is to find a policy π that maximizes t he value, i.e., the optimization problem the agent seeks to solve is:

$$\max_\pi V_M^\pi(s),$$

where the max is over all possibly non-stationary and randomized policies.

Example (Navigation). Imagine the navigation of a robot or a delivery agent. The state of the agent is their current location. The four actions might be moving one step along each of east, west, north, or south. The transitions in the simplest setting are deterministic. Taking the north action moves the agent one step north of their location, assuming the step size is standardized. The agent might have a goal state g they are trying to reach, and the reward is zero until the agent reaches the goal, and one upon reaching the goal state. Since the discount factor $\gamma < 1$, there is incentive to reach the goal state earlier in the trajectory. As a result, the optimal behavior in this setting corresponds to finding the shortest path from the initial state to the goal state, and the value function of a state, given a policy is γ^d, where d is the number of steps required by the policy to reach the goal state.

Conversational agent. This is another fairly general RL problem. The state of an agent can be the current transcript of the conversation, along with additional information about the context of the conversation, characteristics of other agents or human in the conversation. Actions depend on the domain, for instance, we can think of the statement to make in the conversation. Conversational agents may be designed for task completion, such as travel assistant, tech support, or virtual receptionist. For these cases, where may be a pre-defined set of slots which the agent needs to fill before they can find a good solution. For instance, in the case of a travel agent, these may correspond to the dates, source, destination, and mode of travel. The actions may correspond to the text queries to fill the slots.

For task completion, reward is defined as a binary outcome on whether the task was completed or not, such as whether the travel was successfully booked or not. Depending on

the domain, we could further refine it based on quality or price of travel package. For more general conversations, the reward is whether the conversation was satisfactory to the other agents/humans or not.

Example (Strategic games). This is another popular use case where RL has been used to achieve human level performance in Go, Chess, Poker, etc. The usual setting consists of states being the current game board, actions being the potential next moves and reward being the eventual win/loss outcome on a more detailed score when it's defined in the game. These are also called multi-agent RL.

Stationary policies. Stationary policies satisfy the following consistency conditions. In particular, suppose that π is a stationary policy. Then V^π and Q^π satisfy the following Bellman consistency equations for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$,

$$\begin{aligned} V^\pi(s) &= Q^\pi(s, \pi(s)) \\ Q^\pi(s,a) &= r(s,a) + \gamma \mathop{\mathbb{E}}_{a\sim\pi, s'\sim P(\cdot|s,a)} \left[V^\pi(s')\right] \end{aligned}$$

Let us denote P^π as the transition matrix on state-action pairs induced by a stationary matrix π, specifically:

$$P^\pi_{(s,a),(s',a')} := P(s'|s,a)\pi(a'|s')$$

In particular, for deterministic policies we have

$$P^\pi_{(s,a),(s',a')} := \begin{cases} P(s'|s,a) & \text{if } a' = \pi(s') \\ 0 & \text{if } a' \neq \pi(s') \end{cases}$$

Let V^π be a vector of length $\mathcal{S}$ and Q^π, r as vectors of length $|\mathcal{S}| \cdot |\mathcal{A}|$. Let P be a matrix where $P_{(s,a),s'} = P(s'|s,a)$. With this notation, we can verify that

$$Q^\pi = r + \gamma P V^\pi, \text{ and } Q^\pi = r + \gamma P^\pi Q^\pi,$$

leading to the solution that $Q^\pi = (\mathrm{Id} - \gamma P^\pi)^{-1} r$.

A remarkable and convenient property of MDPs is that there exists a stationary and deterministic policy that simultaneously maximizes $V^\pi(s)$ for all $s \in \mathcal{S}$. In particular, the Bellman optimality equations (Bellman, 1956) give a precise characterization of the optimal value function.

4.2.1 Finite-horizon Markov decision processes

In some cases, it's natural to work with finite-horizon (and time-dependent) MDPs. Here, a finite horizon, time-dependent MDP $M = (\mathcal{S}, \mathcal{A}, \{P\}_h, \{r\}_h, H, \mu)$ is specified as follows

- A state space $\mathcal{S}$
- An action space $\mathcal{A}$
- A time-dependent transition function $P_h : \mathcal{S} \times \mathcal{A} \to \Delta(S)$
- A time-dependent reward function $r_h : \mathcal{S} \times \mathcal{A} \to [0, 1]$
- An integer H which defines the horizon of the problem

- An initial state distribution $\mu \in \Delta(\mathcal{S})$ which specifies how the initial state s_0 is generated

Here, for a policy π, a state s, and $h \in \{0, \ldots, H-1\}$, we define the value function $V_h^\pi : \mathcal{S} \to \mathbb{R}$ as

$$V_h^\pi(s) = \mathbb{E}\left[\sum_{t=h}^{H-1} r_h(s_t, a_t)|pi, s_h = s\right]$$

Similarly, the state-action value function Q_h^π is defined as

$$Q_h^\pi(s, a) = \mathbb{E}\left[\sum_{t=h}^{H-1} r_h(s_t, a_t)|\pi, s_h = s, a\right]$$

Given a state a, the goal of the agent is to find a policy π that maximizes the value $\max_\pi V^\pi(s)$.

When the MDP is known, solving the optimal policy can be achieved in polynomial time using LP optimization.

4.3 Multi-armed bandits

Above we require knowledge of the MDP. Here, we consider another decision making framework. For $t = 1, \ldots, T$,

- Select decision $\pi^t \in \Pi :== \{1, 2, \ldots, A\}$
- Observe reward $r^t \in \mathbb{R}$

The protocol proceeds in T rounds. At each round $t \in [T]$, the learning agent selects a discrete decision $\pi^t \in \Pi = \{1, \ldots, K\}$ using historical data. Based on the decision π^t, the learner receives a random reward $r_a \in [-1, 1]$ from $R(a) \in \Delta([-1, 1])$, which has mean reward

$$\mathop{\mathbb{E}}_{r_a \sim R(a)}[r_a] = \mu_a$$

Every iteration t, the learner picks an arm $I_t \in [1, 2, \ldots, K]$. The cumulative regret is defined as

$$R_T = T \cdot \max_i \mu_i - \sum_{t=0}^{T-1} \mu_{I_t}$$

Denote $a^\star = \arg\min_i \mu_i$ as the optimal arm. Define gap $\Delta_a = \mu_{a^\star} - \mu_a$ for any arm a.

The upper confidence bound algorithm is one approach to tackle the MAB problem. For simplicity, we allocate the first K rounds to pull each arm once. Then, for $t = 1 \to T - K$,

- Execute arm $I_t = \arg\max_{i \in [K]} \left(\hat{\mu}_i^t + \sqrt{\frac{\log(TK/\delta)}{N_i^t}}\right)$
- Observe $r_t := r_{I_t}$

where I_t is the index of the arm that is picked by the algorithm at iteration t, and the counts of each arm at each iteration t is

$$N_a^t = 1 + \sum_{i=1}^{t-1} \mathbf{1}\{I_i = a\}$$

The upper confidence bound states that with probability at least $1 - \delta$,

$$|\hat{\mu}_a^t - \mu_a| \leq 2\sqrt{\frac{\ln(TK/\delta)}{N_a^t}}$$

where $\hat{\mu}_a^t$ is the empirical mean for each arm. It can be shown that UCB achieves a regret of at most $O(\sqrt{KT\ln(KT/\delta)} + K)$.

Suggested readings: Chapter 1 in Agarwal et al., 2019.

References

Agarwal, A., Jiang, N., Kakade, S. M., and Sun, W. (2019). "Reinforcement learning: Theory and algorithms". In: *CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep* 32, p. 96 (page 60).

Alquier, P. (2021). "User-friendly introduction to PAC-Bayes bounds". In: *arXiv preprint arXiv:2110.11216* (page 48).

Ando, R. K. and Zhang, T. (2005). "A framework for learning predictive structures from multiple tasks and unlabeled data". In: *Journal of Machine Learning Research* 6.Nov, pp. 1817–1853 (page 55).

Bach, F. (2024). *Learning theory from first principles*. MIT press (page 11).

Bartlett, P. L., Harvey, N., Liaw, C., and Mehrabian, A. (2019). "Nearly-tight VC-dimension and pseudodimension bounds for piecewise linear neural networks". In: *Journal of Machine Learning Research* 20.63, pp. 1–17 (page 34).

Bartlett, P. L., Long, P. M., Lugosi, G., and Tsigler, A. (2020). "Benign overfitting in linear regression". In: *Proceedings of the National Academy of Sciences* 117.48, pp. 30063–30070 (pages 44, 45).

Bellman, R. (1956). "Dynamic programming and Lagrange multipliers". In: *Proceedings of the National Academy of Sciences* 42.10, pp. 767–769 (page 58).

Caruana, R. (1997). "Multitask learning". In: *Machine learning* 28, pp. 41–75 (page 51).

Catoni, O. (2007). "PAC-Bayesian supervised classification: the thermodynamics of statistical learning". In: *arXiv preprint arXiv:0712.0248* (page 48).

Duchi, J. (2019). "Lecture notes for statistics 311/electrical engineering 377". In: *URL: http://web.stanford.edu/class/stats311/lecture-notes.pdf* (page 7).

Kouw, W. M. and Loog, M. (2018). "An introduction to domain adaptation and transfer learning". In: *arXiv preprint arXiv:1812.11806* (page 7).

Ledoux, M. and Talagrand, M. (2013). *Probability in Banach Spaces: isoperimetry and processes*. Springer Science & Business Media (page 19).

Liang, P. (2016). *CS229T/STAT231: Statistical learning theory (Winter 2016)* (page 34).

Mitzenmacher, M. and Upfal, E. (2017). *Probability and computing: Randomization and probabilistic techniques in algorithms and data analysis*. Cambridge university press (page 20).

Neyshabur, B., Bhojanapalli, S., McAllester, D., and Srebro, N. (2017). "Exploring generalization in deep learning". In: *Advances in neural information processing systems* 30 (page 29).

Ruder, S. (2017). "An Overview of Multi-Task Learning in Deep Neural Networks". In: *arXiv preprint arXiv:1706.05098* (page 7).

Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding machine learning: From theory to algorithms*. Cambridge university press (page 46).

Srebro, N. and Shraibman, A. (2005). "Rank, trace-norm and max-norm". In: *International conference on computational learning theory*. Springer, pp. 545–560 (page 46).

Valiant, L. G. (1984). "A theory of the learnable". In: *Communications of the ACM* 27.11, pp. 1134–1142 (page 8).

Wainwright, M. J. (2019). *High-dimensional statistics: A non-asymptotic viewpoint*. Vol. 48. Cambridge university press (pages 11, 54).

Wu, S., Zhang, H. R., and Ré, C. (2020). "Understanding and improving information transfer in multi-task learning". In: *ICLR* (page 55).

Zhang, T. (2023). *Mathematical analysis of machine learning algorithms*. Cambridge University Press (page 11).