Lecture Notes of Advanced Machine Learning (CS 7140)

Hongyang R. Zhang

January 8, 2025

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.

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

- 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?

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

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

[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).

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

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

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

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

What can say that the relationship between $\hat{L}(\hat{h}_{\mathrm{ERM}})$ and $L(\hat{h}_{\mathrm{ERM}})$? A key challenge is that the randomness of $\hat{h}_{\mathrm{ERM}}$ now depends on $\hat{L}$. Thus, $\hat{L}(\hat{h}_{\mathrm{ERM}})$ 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, $L(h) - \hat{L}(h) \le \epsilon$, 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:

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

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 $\hat{h}_{\mathrm{ERM}}$, since the result essentially holds for any $h \in \mathcal{H}$, which also subsumes $\hat{h}_{\mathrm{ERM}}$ 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 $(x_1^{(1)}, y_1^{(1)}), (x_2^{(1)}, y_2^{(1)}), \ldots, (x_{n_1}^{(1)}, y_{n_1}^{(1)})$, 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 $\beta^{(1)} \in \mathbb{R}^p$:

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

Similarly, we denote the samples of the target task as $(x_1^{(2)}, y_1^{(2)}), (x_2^{(2)}, y_2^{(2)}), \dots, (x_{n_2}^{(2)}, y_{n_2}^{(2)})$. 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:

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

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 $1-\delta$,

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

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:

$$\Pr\left[L(\hat{h}_{\text{ERM}}) - \hat{L}(\hat{h}_{\text{ERM}}) \geq \epsilon\right] \leq \delta \tag{11}$$

We can expand the excess risk as

$$L(\hat{h}) - L(h^\star) = \underbrace{\left(L(\hat{h}) - \hat{L}(\hat{h})\right)}_{\text{Uniform convergence}} + \underbrace{\left(\hat{L}(\hat{h}) - \hat{L}(h^\star)\right)}_{\leq 0} + \underbrace{\left(\hat{L}(h^\star) - L(h^\star)\right)}_{\text{Concentration}} \tag{12}$$

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,

$$\hat{L}(\hat{h}_{\text{ERM}}) = \frac{1}{n}\sum_{i=1}^{n} \ell(\hat{h}_{\text{ERM}}(x_i), y_i). \tag{13}$$

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

$$\Pr[L(\hat{h}_{\text{ERM}}) - L(h^\star) \geq \epsilon] \leq \Pr\left[\left(\sup_{h\in\mathcal{H}} \left|L(h) - \hat{L}(h)\right|\right) \geq \frac{\epsilon}{2}\right] \leq \delta \tag{14}$$

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 $\epsilon/2$.

Next lecture: We'll talk about concentration estimates, which form the most fundamental techniques for dealing with IID random variables. Then we'll talk about a generalization bound for finite (not necessarily realizable) hypothesis classes. **Suggested reading:** Chapter 3.1-3.3 of Statistical learning theory lecture notes by Percy Liang.

References

Duchi, J. (2019). “Lecture notes for statistics 311/electrical engineering 377”. In: *URL: http://web.stanford.edu/class/stats311/lecture-notes.pdf* (page 6).

Kouw, W. M. and Loog, M. (2018). “An introduction to domain adaptation and transfer learning”. In: *arXiv preprint arXiv:1812.11806* (page 5).

Ruder, S. (2017). “An Overview of Multi-Task Learning in Deep Neural Networks”. In: *arXiv preprint arXiv:1706.05098* (page 5).

Valiant, L. G. (1984). “A theory of the learnable”. In: *Communications of the ACM* 27.11, pp. 1134–1142 (page 6).