

Lecture Notes of Advanced Machine Learning (CS 7140)

Hongyang R. Zhang

January 6, 2025

1 Overview (Lecture 1)

1.1 What is this course about?

Machine learning has been increasingly 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.

¹ChatGPT 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

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:

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

- Let $\ell : (\mathcal{X}, \mathcal{Y}) \times \mathcal{H} \rightarrow \mathbb{R}$ 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 $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$, the training loss (or empirical risk) of a hypothesis $h \in \mathcal{H}$ is defined as

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

²April will be dedicated to course project presentations.

³A 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}^* , supported on $\mathcal{X} \times \mathcal{Y}$.

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

$$L(h) = \mathbb{E}_{(x,y) \sim \mathbb{P}^*} [\ell(h(x), y)]. \quad (2)$$

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

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

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}_{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:

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

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

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

1.3 Basic setup of neural networks

Consider the case of a basic one-layer network:

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

- σ is the nonlinear activation function. Typical choices of σ : ReLU $x \rightarrow \max(0, x)$, sigmoid $x \rightarrow \frac{1}{1+\exp(-x)}$
- $Z = \{\alpha = (a_i, w_i, b_i)\}_{i=1}^m$ 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 $W = [w_1, w_2, \dots, w_m]$, and the bias vector $b = [b_1, b_2, \dots, b_m]$. Thus, we may write map (5) as $x \rightarrow a^\top \sigma(Wx + b)$.

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

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

The depth of the network is given by L . The width is given by $\max(m_1, m_2, \dots, m_L)$, 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?*

1.4 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.

P.S.

- Have feedback or want to ask a question for the instructor? Leave a note here: <https://forms.gle/hPZ8u3d7niDJf1AJ7>.
- Want to see the latex source code? You can find the tex files here: <https://github.com/hongyangsg/cs7140>.