Recitation 6: Deep Learning & Learning Theory

10-301/10-601 Introduction to Machine Learning (Summer 2023)
http://www.cs.cmu.edu/~hchai2/courses/10601
Released: July 6th, 2023
Quiz Date: July 11th, 2023
TAs: Alex, Andrew, Sofia, Tara, and Neural the Narwhal

1 Learning Theory

1.1 PAC Learning

Some Important Definitions
Basic notation:

1. - Probability distribution (unknown): $X \sim p^*$
   - **True function** (unknown): $c^* : X \to Y$
   - **Hypothesis space** $\mathcal{H}$ and **hypothesis** $h \in \mathcal{H} : X \to Y$
   - Training dataset $\mathcal{D} = \{x^{(1)}, \dots, x^{(N)}\}$
2. **True Error (expected risk)**
$$R(h) = P_{x \sim p^*(x)}(c^*(x) \neq h(x))$$
3. **Train Error (empirical risk)**
$$\begin{aligned}\hat{R}(h) &= P_{x \sim \mathcal{D}}(c^*(x) \neq h(x)) \\ &= \frac{1}{N} \sum_{i=1}^{N} 1(c^*(x^{(i)}) \neq h(x^{(i)})) \\ &= \frac{1}{N} \sum_{i=1}^{N} 1(y^{(i)} \neq h(x^{(i)}))\end{aligned}$$

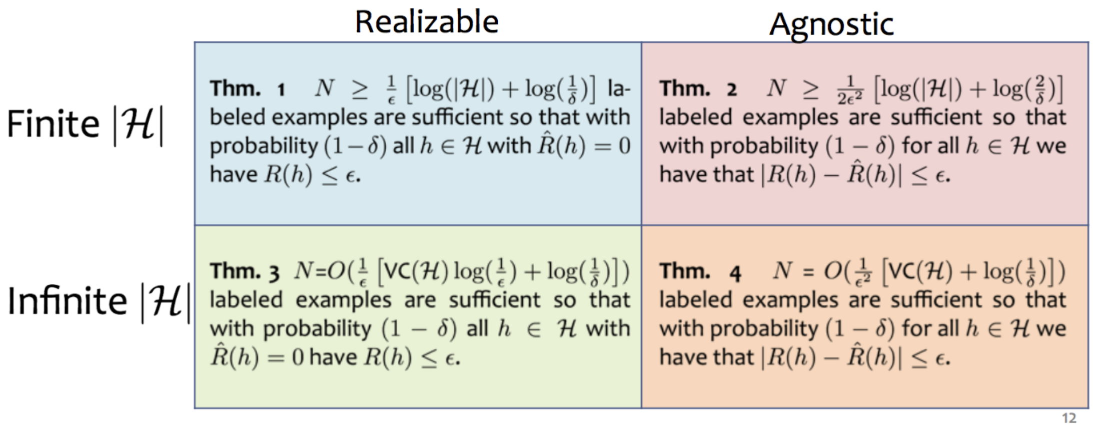
The **PAC criterion** is that we produce a high accuracy hypothesis with high probability. More formally,

$$P(\forall h \in \mathcal{H}, \underline{\hspace{6cm}} \leq \underline{\hspace{1.5cm}}) \geq \underline{\hspace{4cm}}$$

Sample Complexity is the minimum number of training examples N such that the PAC criterion is satisfied for a given ϵ and δ

Sample Complexity for 4 Cases: See Figure 1. Note that

- **Realizable** means $c^* \in \mathcal{H}$
- **Agnostic** means c^* may or may not be in $\mathcal{H}$



	Realizable	Agnostic
Finite $\lvert\mathcal{H}\rvert$	**Thm. 1** $N \geq \frac{1}{\epsilon}\left[\log(\lvert\mathcal{H}\rvert) + \log(\frac{1}{\delta})\right]$ labeled examples are sufficient so that with probability $(1-\delta)$ all $h \in \mathcal{H}$ with $\hat{R}(h) = 0$ have $R(h) \leq \epsilon$.	**Thm. 2** $N \geq \frac{1}{2\epsilon^2}\left[\log(\lvert\mathcal{H}\rvert) + \log(\frac{2}{\delta})\right]$ labeled examples are sufficient so that with probability $(1-\delta)$ for all $h \in \mathcal{H}$ we have that $\lvert R(h) - \hat{R}(h)\rvert \leq \epsilon$.
Infinite $\lvert\mathcal{H}\rvert$	**Thm. 3** $N{=}O(\frac{1}{\epsilon}\left[\mathsf{VC}(\mathcal{H})\log(\frac{1}{\epsilon}) + \log(\frac{1}{\delta})\right])$ labeled examples are sufficient so that with probability $(1-\delta)$ all $h \in \mathcal{H}$ with $\hat{R}(h) = 0$ have $R(h) \leq \epsilon$.	**Thm. 4** $N = O(\frac{1}{\epsilon^2}\left[\mathsf{VC}(\mathcal{H}) + \log(\frac{1}{\delta})\right])$ labeled examples are sufficient so that with probability $(1-\delta)$ for all $h \in \mathcal{H}$ we have that $\lvert R(h) - \hat{R}(h)\rvert \leq \epsilon$.

Figure 1: Sample Complexity for 4 Cases

The **VC dimension** of a hypothesis space $\mathcal{H}$, denoted VC($\mathcal{H}$) or $d_{VC}(\mathcal{H})$, is the maximum number of points such that there exists at least one arrangement of these points and a hypothesis $h \in \mathcal{H}$ that is consistent with any labelling of this arrangement of points.

To show that VC$(\mathcal{H}) = n$:

-
-

Questions

1. For the following examples, write whether or not there exists a dataset with the given properties that can be shattered by a linear classifier.

   - 2 points in 1D
   - 3 points in 1D
   - 3 points in 2D
   - 4 points in 2D

   How many points can a linear boundary (with bias) classify exactly for d-Dimensions?

2. Consider a rectangle classifier (i.e. the classifier is uniquely defined 3 points $x_1, x_2, x_3 \in \mathbb{R}^2$ that specify 3 out of the four corners), where all points within the rectangle must equal 1 and all points outside must equal -1

   (a) Which of the configurations of 4 points in figure 2 can a rectangle shatter?

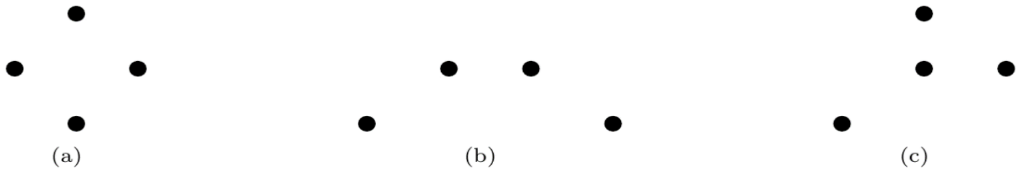


Figure 2

   (b) What about the configurations of 5 points in figure 3?

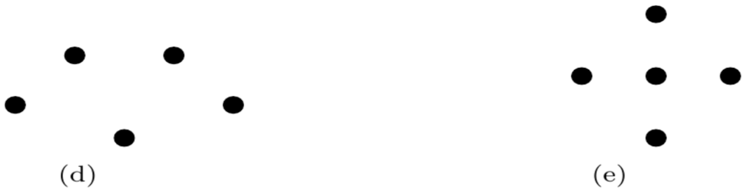


Figure 3

3. Let $x_1, x_2, ..., x_n$ be n random variables that represent binary literals ($x \in \{0,1\}^n$). Let the hypothesis class $\mathcal{H}_n$ denote the conjunctions of no more than n literals in which each variable occurs at most once. Assume that $c^* \in \mathcal{H}_n$.

   Example: For $n = 4$, $(x_1 \wedge x_2 \wedge x_4), (x_1 \wedge \neg x_3) \in \mathcal{H}_4$

   Find the minimum number of examples required to learn $h \in \mathcal{H}_{10}$ which guarantees at least 99% accuracy with at least 98% confidence.

2 Convolutional Neural Networks

2.1 Dance Dance Convolution

Consider the following 4 x 4 image and 2x2 filter below.

1	3	-2	4
0	8	6	5
2	1	-9	0
4	-1	3	7

1	2
-2	-1

1. Assume that there is no padding and stride $= 1$. What are the dimensions of the output, and what is the value in the bottom right corner of the output image?
2. Now assume that we having padding $= 1$. Given that, what are the new dimensions of the output, and the new value in the bottom right corner?

2.2 Concepts

1. What are filters?
   - Filters (also called kernels) are feature extractors in the form of a small matrix used in convolutional neural layers. They usually have a width, height, depth, stride, padding, channels (output) associated with them.
2. What are convolutions?
   - We sweep the filter around the input tensor and take matrix dot products based on factors such as filter size, stride, padding. The matrix dot products form a new tensor, which is the output of a convolutional layer.
3. What are some benefits of CNNs over fully connected (also called dense) layers?
   - Good for image-related machine learning (learns the kernels that do feature engineering)
   - Pseudo translational invariance
   - Parameter efficient

2.3 Parameters

Suppose that we want to classify images that belong to one of ten possible classes (i.e. `[cat, dog, bird, turtle, ..., horse]`). The images come in RGB format (one channel for each color), and are downsampled to dimension `128x128`.

Figure 4 illustrates one such image from the MS-COCO dataset[1].

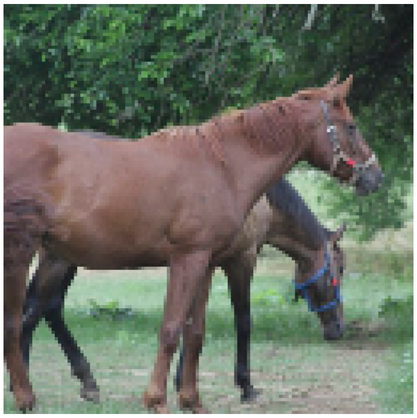


Figure 4: Image of a horse from the MS-COCO dataset, downsampled to 128x128

We construct a Convolutional Neural Network that has the following structure: the input is first max-pooled with a 2x2 filter with stride 2 and 3 output channels. The results are then sent to a convolutional layer that uses a 17x17 filter of stride 1 and 12 output channels. Those values are then passed through a max-pool with a 3x3 filter with stride 3 and also 12 output channels. The result is then flattened and passed through a fully connected layer (ReLU activation) with 128 hidden units followed by a fully connected layer (softmax activation) with 10 hidden units. We say that the final 10 hidden units thus represent the categorical probability for each of the ten classes. With enough labeled data, we can simply use some optimizer like SGD to train this model through backprogation.

Note: By default, please assume we have bias terms in all neural network layers unless explicitly stated otherwise.

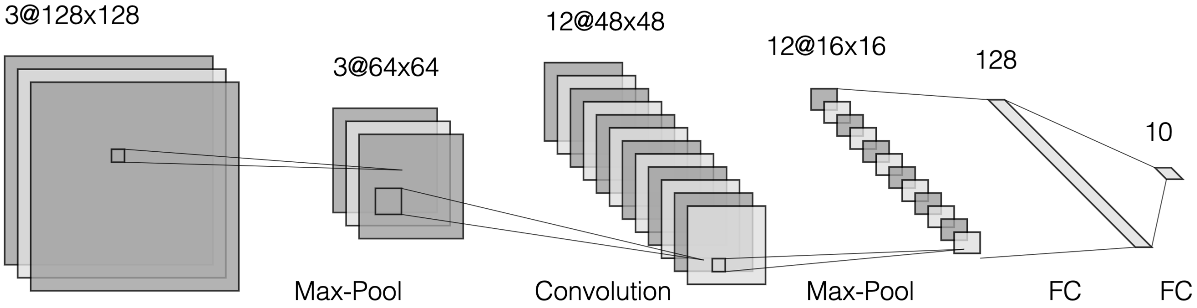


Figure 5: Full CNN structure, illustrated

[1] https://cocodataset.org/

1. How many parameters are in this network for the convolutional components?

2. How many parameters are in this network for the fully connected (also called dense) components?

3. From these parameter calculations, what can you say about convolutional layers and fully connected layers in terms of parameter efficiency[2]? Why do you think this is the case?

2.4 Links

Visualization of convolutional filter sweep steps https://github.com/vdumoulin/conv_arithmetic

Visualization of convolutional filter smooth sweep with outputs https://www.youtube.com/watch?v=f0t-OCG79-U

Visualization of neural network layer outputs http://cs231n.stanford.edu/

The architecture used there is (conv→ relu → conv → relu → pool) x3 → fc → softmax

[2] the ratio between the number of parameters from some layer type and the total number of parameters.