Chapter 1

The bias complexity trade-off

Let A be the learning algorithm, S the training set, $\mathcal{H}$ the hypothesis class and $\hat{h} \in \mathcal{H}$ such that $L_{\mathcal{D}}(\hat{h})$ is small. What we want to know is if there is a universal learner, i.e. an algorithm A that predicts the best $\hat{h}$ for any distribution $\mathcal{D}$.

1.1 No-Free-Lunch theorem

Theorem 1.1.1 (No-Free-Lunch)**.** *Let A be any learning algorithm for the task of binary classification with respect to the 0-1 loss over a domain $\mathcal{X}$. Let m be any number smaller than $\frac{|\mathcal{X}|}{2}$, representing a training set size. Then, there exists a distribution $\mathcal{D}$ over $\mathcal{X} \times \{0,1\}$ such that:*

- *There exists a function $f : \mathcal{X} \to \{0,1\}$ with $L_{\mathcal{D}}(f) = 0$.*
- *With probability of at least $\frac{1}{7}$ over the choice of $S \sim \mathcal{D}^m$ we have that $L_{\mathcal{D}}(A(S)) \geq \frac{1}{8}$.*

The meaning of this theorem is clear: a universal learner can't exist. In fact, no learner can succeed on all learning tasks. FIn other words, for every learner, there exists a task on which it fails, even though that task can be successfully learned by another learner.

Corollary 1.1.1.1. *Let $\mathcal{X}$ be an infinite domain set and let $\mathcal{H}$ be the set of all functions from $\mathcal{X}$ to $\{0,1\}$. Then, $\mathcal{H}$ is not PAC learnable.*

Firstly, we give the idea of the proof. Our training set is smaller than half of the domain, so we have no information on what happens on the other half. There exists some target function f that works on the other half in a way that contradicts our estimated labels.

Remark 1. A class $\mathcal{H}$ of all possible functions from $\mathcal{X}$ to $\{0,1\}$ (i.e. assuming no prior knowledge) is not a good idea since ii isn't PAC learnable.

Proof. Let's demonstate the previous corollary. We proceed by contradiction:

1. Assume PAC learnability:
   - We choose $\varepsilon < \frac{1}{8}$, $\delta < \frac{1}{7}$ (Note that $\mathcal{H}$ includes all functions).
   - By definition of PAC: there exists an algorithm A such that for very distribution $\mathcal{D}$, if realizable, when running the algorithm on m i.i.d. examples generated by $\mathcal{D}$, the algorithm returns a hypothesis h such that, with probability $\geq 1 - \delta$, $L_{\mathcal{D}}(A(S)) \leq \varepsilon$.
2. Apply No-Free-Lunch theorem to A:

- $|\mathcal{X}| > 2m$: for any ML algorithm (including A), there exists a distribution $\mathcal{D}$ for which with probability $\geq \frac{1}{7} > \delta$, $L_{\mathcal{D}}(A_S) \geq \frac{1}{8} > \varepsilon$.

3. Contradiction!

□

Remark 2. We want to choose a good hypotesis set:

- We need to use our prior knowledge about $\mathcal{D}$ to pick a good hypothesis set.
- We would like $\mathcal{H}$ to be large, so that it may contain a function h with small $L_S(h)$ and hopefully a small $L_{\mathcal{D}}(h)$.
- No-Free-Lunch theorem tells us that $\mathcal{H}$ can't be too large! In fact:
  - ▷ $\mathcal{H}$ too small $\implies$ large L_S and $L_{\mathcal{D}} \sim L_S$.
  - ▷ $\mathcal{H}$ too large $\implies$ small L_S and $L_{\mathcal{D}} \gg L_S$ (risk of overfitting).

1.2 Error decomposition

To have a more clear vision, we decompose the error of an $\text{ERM}_{\mathcal{H}}$ predictor into two components as follows. Let h_S be an $\text{ERM}_{\mathcal{H}}$ hypotesis. Then, we can write:

$$L_{\mathcal{D}}(h_S) = \varepsilon_{\text{app}} + \varepsilon_{\text{est}} \tag{1.1}$$

where:

$$\varepsilon_{\text{app}} = \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) \tag{1.2a}$$

$$\varepsilon_{\text{est}} = L_{\mathcal{D}}(h_S) - \varepsilon_{\text{app}} \tag{1.2b}$$

- **Approximation Error**:
  It is the minimum risk achievable by a predictor in the hypotesis class. This term measures how much risk we have because we restrict ourselves to a specific class, namely how much **inductive bias** we have. Other informations on ε_{app}:
  - ▷ It depends on the choice of $\mathcal{H}$.
  - ▷ It doesn't depend on the sample size m.
  - ▷ Under the realizability assumption: $\varepsilon_{\text{app}} = 0$
  - ▷ The larger $\mathcal{H}$ is, the smaller $\varepsilon_{\text{app}} = 0$ is.
- **Estimation Error**:
  It is the difference between the approximation error and the error achieved by the ERM predictor. It results because the empirical risk (i.e. the training error) is only an estimate of the true risk and so the predictor minimizing the empirical risk is only an estimate of the predictor minimizing the true risk. Other informations on ε_{est}:
  - ▷ It depends on the training set size m and on the size (complexity) of the hypothesis class.
  - ▷ ε_{est} increases with $|\mathcal{H}|$ and decreases with m.

Since our goal is to minimize the total risk, we face a tradeoff, called the **bias-complexity tradeoff**:

- Choosing $\mathcal{H}$ large $\implies$ decreases ε_{app}, increases ε_{est}. This leads to overfitting.
- Choosing $\mathcal{H}$ small $\implies$ decreases ε_{est}, increases ε_{app}. This leads to underfitting.

1.3 Train, test and validation sets

In most practical applications, the available set of examples is split into more sets with different purposes. In order:

- We estimate $L_{\mathcal{D}}(h)$ (with h selected with ERM).
- We extract a **test set** from the original one, namely a new set of samples not used for picking h. It is different from the training set and it leads to a more reliable estimation of $L_{\mathcal{D}}(h)$. We must not look to the test set until we have picked our final hypothesis. In practice, one set of sample is splitted into training set and test set.
- Sometimes the training set is divided into training set and **validation set**. The ladder is used for selecting the hyperparameters of the algorithm.

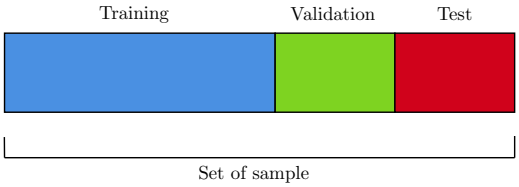

Figure 1.1: Scheme of the division of the set of examples S.

Hence, the training routine is:

- Train model on training set.
- Evaluate model on validation set.
- Tweak model according to the results on validation set. Restart from the first point.
- When the learning procedure has finished due to a particular ending condition, pick the model that performs better on validation set.
- Confirm the result on test set.