Theory of Deep Learning

Manik Bhandari
Department of Computational Data Science
Indian Institute of Science
Bangalore, India
mbbhandarimanik2@gmail.com

Abstract

Notes of Theory of Deep Learning mainly from lectures at IISC.

1 Introduction

Define error of a classifier (aka *true error*) as probability of it making a mistake given a random data point.

$$L_{D,f}(h) = \underset{x \sim D}{P}[h(x) \neq f(x)] = D(\{x : h(x) \neq f(x)\})$$

where D is the distribution from where a data point x is drawn, f is a known *correct* function which always gives the correct labels to a data point. By this definition $D(A)$ is the probability of observing a random point x from A.

Define *training error* or *emperical risk* as

$$L_S(h) = \frac{|i \in [m] : h(x_i) \neq y_i|}{m}$$

where S is the training *set* (it is actually a sequence since points can repeat and classifiers often take into account their order) of the form $\{(x_i, y_i)\}$. If you naively minimize this emperical risk then you are likely to overfit. To avoid it, you use some prior knowledge about the *kind of classifier* that can possibly fit to the data and restrict your hypothesis search space to those types of classifiers.

This kind of restriction induces a *bias* in the model (aka *inductive bias*). In this setting, define

$$h_S = ERM_h(S) \in \underset{h \in \mathcal{H}}{argmin}\, L_S(h).$$

This is a tradeoff – choosing a restricted $\mathcal{H}$ can add too much bias but choosing a large $\mathcal{H}$ may lead to overfitting.

Finite hypothesis class If we restrict $\mathcal{H}$ to have an upper bound on its size then ERM_h will not overfit if we have *large* training data (how large will depend on size of $\mathcal{H}$).

Realizability Assumption There exists $h^* \in \mathcal{H}$ such that $L_{D,f}(h^*) = 0$ i.e. it never makes a mistake which means that $L_S(h^*) = 0$. Since this is the least possible error, this means that for every ERM hypothesis $L_S(h_S) = 0$. We are however interested in true error of h_S i.e. $L_{D,f}(h_S)$.

iid assumption Assume that elements of S are identically and independently distributed according to D denoted by $S \sim D^m$.

Now, we would like to have an h_S such that $L_{D,f}(h_S)$ is not too large. Let's say h_S *fails* if $L_{D,f}(h_S) > \epsilon$.

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.

We want to upper bound the probability of sampling a training set that leads to a failure i.e $D^m(S : L_{D,f}(h_S) > \epsilon)$. Define bad hypothesis as $\mathcal{H_B} = \{h \in \mathcal{H}\} : L_{D,f}(h_S) > \epsilon$ and misleading training sets as $M = \{S : \exists h \in \mathcal{H_B}, L_S(h) = 0\}$. So, all the training sets for which h_S fails must be misleading (there can be other misleading sets also). So

$$\{S : L_{D,f}(h_S) > \epsilon\} \subseteq M = \bigcup_{h \in \mathcal{H_B}} \{S : L_S(h) = 0\}.$$

This means that

$$D^m(\{S : L_{D,f}(h_S) > \epsilon\}) \leq D^m(M) = D^m(\bigcup_{h \in \mathcal{H_B}} \{S : L_S(h) = 0\}).$$

Take union bound of RHS to get

$$D^m(\{S : L_{D,f}(h_S) > \epsilon\}) \leq \sum_{h \in \mathcal{H_B}} D^m(\{S : L_S(h) = 0\}) = \sum_{h \in \mathcal{H_B}} \left(\prod_{i=1}^{m} D(\{x_i : h(x_i) = f(x_i)\}) \right).$$

and since $h \in \mathcal{H_B}$, $D(\{x_i : h(x_i) = f(x_i)\}) \leq 1 - \epsilon$. But each x_i is iid over D so $D^m(\{S : L_S(h) = 0 \leq (1-\epsilon)^m \leq e^{-\epsilon m}$. As m goes large, the probability of finding a misleading set reduces. Therefore

$$D^m(\{S : L_{D,f}(h_S) > \epsilon\}) \leq |\mathcal{H_B}| e^{-\epsilon m} \leq |\mathcal{H}| e^{-\epsilon m}.$$

Take log both sides to get

$$\log D^m(\{S : L_{D,f}(h_S) > \epsilon\}) \leq \log |\mathcal{H}| - \epsilon m \implies m \leq \frac{\log(|\mathcal{H}|/\delta)}{\epsilon}$$

where $\delta = D^m(\{S : L_{D,f}(h_S) > \epsilon\})$. This also implies that if m is large enough i.e. $m \geq \frac{\log(|\mathcal{H}|/\delta)}{\epsilon}$ then $L_{D,f}(h_S) \leq \epsilon$ with probability $1 - \delta$ of choosing the iid samples S.

So with the ERM_h rule, your hypothesis will be *probably* $(1 - \delta)$ *approximately* (ϵ) *correct* (PAC). Note that the size m does not depend upon the underlying distribution or labeling function.

PAC Learnability A hypothesis class $\mathcal{H}$ is PAC learnable if $\exists\ m_{\mathcal{H}} : (0,1)^2 \to \mathbb{N}$ and a learning algorithm such that
For every $(\epsilon, \delta) \in (0,1)$, for every distribution $\mathcal{D}$ over $\mathcal{X}$ and for every labeling function $f : \mathcal{X} \to (0,1)$
If the realizability assumption holds over $\mathcal{H}, \mathcal{D}, f$
then running the algorithm on $m > m_{\mathcal{H}}(\epsilon, \delta)$ samples generated iid from $\mathcal{D}$ and labeled by f gives a hypothesis h such that
with probability at least $1 - \delta$ over the choice of examples, $L_{\mathcal{D},f}(h) \leq \epsilon$.

Sample complexity $m_{\mathcal{H}} : (0,1)^2 \to \mathbb{N}$ defines the *sample complexity* of learning $\mathcal{H}$ i.e. how many samples are required to get a PAC solution. Let it be the *minimum function* that satisfies the criteria of PAC learnability.

Sample complexity of finite hypothesis class Every finite hypothesis class is PAC learnable with sample complexity $m \leq \lceil \frac{\log(|\mathcal{H}|/\delta)}{\epsilon} \rceil$

Removing realizability assumption Assuming that such an h^* exists such that
$\underset{x\sim\mathcal{D}}{P}[h^*(x) = f(x)] = 1$ is too strong. Not only might such an h^* not exists, your features might not be discriminative enough. Instead assume that $\mathcal{D}$ is a joint distribution over domain points $\mathcal{X}$ and labels $\mathcal{Y}$. Now, true error

$$L_D(h) = \underset{(x,y\ \sim\ \mathcal{D})}{P}[h(x) \neq y] = \mathcal{D}(\{(x,y) : h(x) \neq y\})$$

Bayes optimal predictor is the best labeling function defined as

$$f_{\mathcal{D}}(x) = [insertbrackets]1 if \mathbb{P}[y = 1|x] \geq 0.50 o.w$$

Agnostic PAC Learnability A hypothesis class $\mathcal{H}$ is agnostic PAC learnable w.r.t. a set $\mathbb{Z}$ and a loss function $l : \mathcal{Z} \rightarrow \mathbb{R}_+$ if there exists a function $m_{\mathcal{H}} : (0, 1)^2 \rightarrow \mathbb{N}$ and a learning algorithm such that

for *every* $\epsilon, \delta \in (0, 1)$ and for *every* $\mathcal{D}$ over $\mathcal{Z}$ when the algorithm is run $m \geq m_{\mathcal{H}}(\epsilon, \delta)$ samples iid from $\mathcal{D}$, the algo returns a hypothesis $h \in \mathcal{H}$ such that with probability $1 - \delta$ over the training samples

$$L_{\mathcal{D}}(h) = \underset{h' \in \mathcal{H}}{min} L_{\mathcal{D}}(h') + \epsilon$$

where $L_{\mathcal{D}}(h) = \mathbb{E}_{z \sim \mathcal{D}}[l(h, z)]$.