IFT 6085 - Lecture 7
Elements of statistical learning theory

This version of the notes has not yet been thoroughly checked. Please report any bugs to the scribes or instructor.

Scribe(s): Brady Neal and Matthew Scicluna **Instructor:** Ioannis Mitliagkas

1 Summary

Up until now, we've had a crash course in optimization. Now, we move to the statistical learning theory crash course.

Lecture Narrative:

- Goal of statistical learning theory: determine how well a model performs on unseen data
- We really care about performance on unseen data, but all we have is the training data, so Empirical Risk Minimization (ERM) is used as a surrogate for minimizing the actual expected risk.
- Define the generalization gap and why it's the focus of study.
- Introduce Hoeffding's Inequality and the Union Bound
- Prove a bound on the generalization gap for more simple hypothesis classes (ones that are not only countable, but finite), using Hoeffding's Inequality and the Union Bound.
- If we discretize the weights in our models, we get that the sample complexity (amount of data needed) is linearly dependent on the number of weights.
- Briefly introduce VC dimension and how it extends the above bound to hypothesis classes that aren't necessarily finite or countable.

More General Generalization Narrative:

- This lecture is an example of the more classic generalization bounds.
- The bounds are relevant (tight enough) for simpler models (hypothesis classes), but they quickly lose relevance in the realm of deep learning. More specifically, because we have that dependence on the number of weights and because there are so many weights in a deep neural network, the bounds become vacuous (greater than 1).
- Current work is being done on finding non-vacuous bounds. One such approach is based on the the classic PAC-Bayes approach, which we see in the next lecture.

2 Introduction and Notation

The goal in machine learning is not to perform well on training data, but to perform well on unseen data. This is usually measured using a test set. The topic of "generalization" is about the difference of performance on training data and unseen data. For example, a model "generalizes well" if it performs roughly the same on test data as it does on training data. Statistical learning theory is largely concerned with theoretical bounds on this difference in performance, also known as the *generalization gap*.

In this lecture, we focus specifically on binary classification, but these results can be easily extended to multiclass classification, and there exist results for supervised learning in general.

Notation:

- $\mathcal{X}$ - domain set (input space)
- $\mathcal{Y}$ - label set (output space)
- n - number of training examples (the book uses m)
- $S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ - training set where $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$
  - Anything that is subscripted by S means that it is dependent on the training data S. For example, a learned hypothesis h_S and the empirical risk $\hat{R}_S$ with respect to S are both dependent on S.
- $(x_i, y_i) \sim \mathcal{D}$ - distribution over the data. Note that in our setup we have a joint distribution rather than just x_i being random and y_i being a deterministic function of x_i
- $\mathcal{H}$ - hypothesis class (class of possible models we can learn; examples below)
  - $\mathcal{H}_{\text{SVM}}$: class of possible SVMs on a dataset
  - $\mathcal{H}_{\text{LR}}$: class of possible logistic regression models on a dataset
  - $\mathcal{H}_{\text{NN}}$: class of possible neural networks of a fixed architecture on a dataset
  - $\mathcal{H} \subset \{h : \mathcal{X} \to \mathcal{Y}\}$: $\mathcal{H}$ is a subset of all possible functions that map from input space to output space. Choosing this subset (hypothesis class), $\mathcal{H}$, introduces inductive bias.
  - In the example of binary classification on a d dimensional real-valued dataset, we have $h(x_i) = \hat{y}_i$ where $h \in \mathcal{H}, x_i \in \mathbb{R}^d \equiv \mathcal{X}, \hat{y}_i \in \{0, 1\} \equiv \mathcal{Y}$
- $\ell(\hat{y}, y)$: loss, or error, function that measures the difference between the prediction, $\hat{y}$, and the true label, y (e.g. 0-1 loss, squared loss, etc.)
  - $\ell_{0-1}(\hat{y}, y) = \mathbb{1}(\hat{y} \neq y)$
  - $\ell_{\text{squared}}(\hat{y}, y) = (\hat{y} - y)^2$

3 Empirical Risk Minimization and Generalization Gap

The goal is to identify the hypothesis $h \in \mathcal{H}$ that gives the best performance on $\mathcal{D}$. If we knew $\mathcal{D}$ then we could evaluate h via the Risk:

Definition 1 (Risk)**.** *What we truly care about*

$$R[h] = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\ell(h(x), y)\right]$$

Unfortunately we cannot compute the Risk since we don't have access to $\mathcal{D}$. We can evaluate the performance of h on S, as a surrogate. This is called the empirical risk.

Definition 2 (Empirical Risk)**.** *A surrogate for risk*

$$\hat{R}_S[h] = \frac{1}{n}\sum_{i=1}^{n} \ell(h(x_i), y_i)$$

We usually use S to determine h, and we make this dependence explicit by writing h_S. In particular, one often chooses the h which minimizes $\hat{R}_S$. Notice that h_S is a random quantity since it depends on S randomly sampled from D.

Given some h_S, we can define the generalization gap:

Definition 3 (Generalization Gap)**.** *How we measure generalization*

$$\epsilon_{gen}(h_S) = |R[h_S] - \hat{R}_S[h_S]|$$

The generalization gap has complex dependence. It depends on the the variance in $\mathcal{D}$ because it is dependent on h_S. Both quantities in the generalization gap (risk and empirical risk) are random because they depend on the training data, which is random. Because $\epsilon_{\text{gen}}(h_S)$ is random, our bounds will be probabilistic bounds. We will have bounds that depend on δ and hold with probability $1 - \delta$. Other examples of things that affect the generalization gap are the length of training (related to early stopping) and the geometry of ℓ.

Note that while we use $\epsilon_{\text{gen}}(h_S)$ to say something about generalization, what we really care about minimizing is the risk $R[h_S]$. For example, we could have a hypothesis that always outputs the same thing. This would have $\epsilon_{\text{gen}}(h_S) = 0$, but it would have high risk. This means that we jointly care about minimizing $\hat{R}_S[h_S]$ and getting good bounds on $\epsilon_{\text{gen}}(h_S)$ to say something about $R[h_S]$. That's why generalization bounds will sometimes take the following form:

$$R[h_S] \leq \hat{R}_S[h_S] + \epsilon$$

We care about deriving bounds for $\epsilon_{\text{gen}}(h_S)$ because bounds give us the above ϵ. Also, $\epsilon_{\text{gen}}(h_S)$ directly tells us the difference between performance on training data and unseen data (i.e. how well the model generalizes).

4 Generalization Bound for Finite Hypothesis Classes

We want to show that given some arbitrary $\epsilon, \delta > 0$ we could choose an n such that $\epsilon_{\text{gen}}(h_S) \leq \epsilon$ with probability $\geq 1 - \delta$. If this condition holds for $\mathcal{H}$ we say it is *PAC Learnable*. The "probably" (*P*) part of PAC corresponds to $1 - \delta$ while the "approximately correct" (*AC*) part corresponds to ϵ. We show that any finite $\mathcal{H}$ has this property. We first derive a probabilistic bound on the following distance that holds for any $h \in \mathcal{H}$:

$$\left| R[h] - \hat{R}_S[h] \right|$$

We notice that this is just the absolute distance between the empirical average $\hat{R}_S[h]$ and its mean since:

$$\begin{aligned}
\mathbb{E}[\hat{R}_S[h]] &= \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} \ell(h(x_i), y_i)\right] \\
&= \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}[\ell(h(x_i), y_i)] \\
&= R[h]
\end{aligned}$$

Probabilistic bounds on such distances are called *concentration bounds*. We will use the following such bound:

Theorem 4 (Hoeffding's Inequality)**.** *Let $X_1, \cdots, X_n$ be independent random variables such that $X_i \in [0, 1]$. Let $W = \frac{1}{n}\sum_{i=1}^{n} X_i$. Then, for any $\epsilon > 0$:*

$$P\left(\mid W - \mathbb{E}[W] \mid \geq \epsilon\right) \leq 2\exp\left(-2n\epsilon^2\right)$$

Proof. Understanding Machine Learning [1] Appendix B.4. Note that the book considers the general case where $X_i \in [a, b]$ for $a < b \in \mathbb{R}$ □

Semantically, the expression above says that for any positive ϵ, our sample mean will be at least ϵ away from its expected value with a probability that decays exponentially with the number of training examples we have.

We can use the Hoeffding bound to decide how many samples we would need to take to guarantee that

$$P\left(\mid W - \mathbb{E}[W] \mid < \epsilon\right) > 1 - \delta$$

If we set $\delta = \exp\left(-2n\epsilon^2\right)$ we can solve to get $n = O\left(\frac{-\log(\delta)}{\epsilon^2}\right)$. Note: this is a lower bound on the sample size which guarantees the statement above. This quantity is often referred to as the *Sample Complexity*.

Now, suppose we consider an arbitrary $h \in \mathcal{H}$. For random variable $\hat{R}_S[h]$ we can use Hoeffding's inequality to get that:

$$P\left(\left| \hat{R}_S[h] - R[h] \right| \geq \epsilon\right) \leq 2\exp\left(-2n\epsilon^2\right)$$

We now extend this bound for $\epsilon_{\text{gen}}(h_S)$:

$$\begin{aligned}
P\left(\left| \hat{R}_S[h_S] - R[h_S] \right| \geq \epsilon\right) &\leq P\left(\max_{h\in\mathcal{H}} \left| \hat{R}_S[h_S] - R[h_S] \right| > \epsilon\right) \\
&= P\left(\bigcup_{h\in\mathcal{H}} \left\{\left| \hat{R}_S[h] - R[h] \right| > \epsilon\right\}\right) \\
&\overset{(a)}{\leq} \sum_{h\in\mathcal{H}} P\left(\left| \hat{R}_S[h] - R[h] \right| > \epsilon\right) \\
&= 2|\mathcal{H}|\exp\left(-2n\epsilon^2\right)
\end{aligned}$$

Where (a) follows using a union bound argument. We can prove this in the case of 2 events and then use induction. In this case $P(A \cup B) = P(A) + P(B) - P(A \cap B) \leq P(A) + P(B)$.

If we take

$$n = O\left(\frac{\log(\frac{|\mathcal{H}|}{\delta})}{\epsilon^2}\right)$$

We get the desired result:

$$P\left(\left| \hat{R}_S[h_S] - R[h_S] \right| \geq \epsilon\right) \leq 1 - \delta$$

For example, if we are dealing with 32 bit floating point numbers in d dimensions, then $|\mathcal{H}| = 2^{32\cdot d}$. We have that $\log|\mathcal{H}| = O(d)$ and for $\delta, \epsilon > 0$ we would choose $n = O\left(\frac{d - \log(\delta)}{\epsilon^2}\right)$.

References

[1] S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, New York, NY, USA, 2014. ISBN 1107057132, 9781107057135.