Machine Learning: Generalization 1

Hao Tang

February 27, 2024
Machine learning is about programming with data

- Minimizing a loss function on a data set produces a program.
- How do we know if the program is correct?
Correctness of classical programs

- A program is correct if it has the desired behavior on all input.

- Correctness is achieved through mathematical proofs and careful engineering.
Correctness of learned programs

- Imagine we have trained a binary classifier.
- We know the loss on the training set.
- Even on the training set, the loss might not be 0.
- Can we say anything about the loss outside of the training set?
- Is it even possible? What assumptions do we need?
What happens if the data is Gaussian?
What happens if the data is Gaussian?
What happens if the data is Gaussian?
Gaussian data

• We need to know
  – the two distributions are Gaussian
  – their means
  – their variances.

• The decision boundary
  – can be found without training
  – is optimal (in the sense that no other boundary achieves a lower error).

• Next questions
  – What if we don’t know the means?
  – What if we don’t know the variances?
  – What if the two distributions are not Gaussian?
  – What if we don’t know what the distributions are?
Generalization

training data → training → classifier
Generalization
Generalization

training data → training → classifier

test data
Generalization

- Training data
- Classifier
- Test data
- Training
- Testing
Generalization

• A data set is said to be **i.i.d. (independent and identically distributed)** if the data points come from the same distribution and are statistically independent from each other.

• A function (or program) is said to **generalize** to data within a distribution if the function achieves a low error on data drawn from that distribution *in expectation*.

• In particular, if a function generalizes then the function has to achieve a low error on both the training set and the test set.

• We do not know the distribution, and only have data drawn from the distribution.

• The only assumption is i.i.d. data.
Generalization

• There exists a distribution $\mathcal{D}$ where both the training data and the test data are drawn from.

• The training set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ includes i.i.d. samples drawn from $\mathcal{D}$.

• The training error for a loss $\ell$ and a program $h$ is defined as

$$L_S(h) = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, h(x_i)).$$  \hfill (1)

• If we have a test set $S'$, then $L_{S'}(h)$ is the error on the test set (or test error for short) for a program $h$. 
Generalization

• The **generalization error** for a program $h$ is defined as

$$L_D(h) = \mathbb{E}_{(x,y) \sim D}[\ell(y, h(x))].$$  \hspace{1cm} (2)

• The test error $L_{S'}(h)$ of a program $h$ is an estimate of the generalization error $L_D(h)$.

$$\mathbb{E}_{S \sim D^n}[L_S(h)] = \mathbb{E}_{S' \sim D'^n}[L_{S'}(h)] = L_D(h)$$  \hspace{1cm} (3)

• The goal of learning is to find a program $h$ with low generalization error $L_D(h)$. 
Learning algorithms and hypothesis classes

- A learning algorithm is a function that takes a data set of size $m$ and returns a function from the hypothesis class $\mathcal{H}$.

- A hypothesis class $\mathcal{H}$ is the set of possible programs of a particular form.

- For example, a linear classifier is $\mathcal{H} = \{x \mapsto w^\top x : w \in \mathbb{R}^d\}$. 

A hypothesis class $\mathcal{H}$ is PAC-learnable with a learning algorithm $A$ if for any distribution $\mathcal{D}$, and any $\epsilon > 0$ and $0 \leq \delta \leq 1$, there exists $N > 0$ such that

$$\mathbb{P}_{S \sim \mathcal{D}^n} \left[ L_{\mathcal{D}}(A(S)) - \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') > \epsilon \right] < \delta$$

(4)

for any $n \geq N$. 

**Probably approximately correct**
• The data set $S$ is a random variable.

• $A(S)$ is a program returned by $A$ after training on $S$.

• $L_D(A(S))$ is also a random variable.

• $\min_{h'\in\mathcal{H}} L_D(h')$ is the best error we can achieve among all programs in $\mathcal{H}$.

• $\epsilon$ is the error tolerance, the approximately correct part.

• $\delta$ is the confidence probability, the probably part.
Imagine we do the following experiment many many times.

1. Draw a training set $S$ and obtain a trained program $A(S)$.
2. Evaluate $L_D(A(S)) - \min_{h' \in \mathcal{H}} L_D(h')$.
3. Repeat

On average, the chance of seeing $L_D(A(S)) - \min_{h' \in \mathcal{H}} L_D(h') > \epsilon$ is $\delta$.

Think of $\epsilon$ and $\delta$ as something small, close to 0.

With high probability $1 - \delta$, the two terms $L_D(A(S))$ and $\min_{h' \in \mathcal{H}} L_D(h')$ only differ by a small amount $\epsilon$.

With high probability, the program learned by $A$ achieves a similar error to the best program in $\mathcal{H}$. 

Probably approximately correct
PAC learning

- PAC learnability is merely a definition.

- The minimum number of samples required, $N$, also known as sample complexity, is a function of $\mathcal{H}$, $\epsilon$, and $\delta$.

- We can now ask, “is the set of linear classifiers PAC learnable if we minimize the zero-one loss on a training set?”
Empirical risk minimization

• Minimizing the loss on a training set is also known as **empirical risk minimization (ERM)**.

\[
A_{\text{ERM}, \mathcal{H}}(S) = h_{\text{ERM}} = \arg\min_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, h(x_i))
\]  

• The set of linear classifiers is \( \mathcal{H}_{\text{lin}} = \{ w \mapsto w^\top x : w \in \mathbb{R}^d \} \).

• We can now formally ask, “is the set of linear classifiers \( \mathcal{H}_{\text{lin}} \) PAC learnable with ERM?”

• And if so, how does the sample complexity \( N \) depends on \( \mathcal{H}_{\text{lin}}, \epsilon, \) and \( \delta \)?
The universe of all programs

• Instead of choosing the set linear classifiers $\mathcal{H}_{\text{lin}} = \{x \mapsto w^\top x : w \in \mathbb{R}^d\}$, can we choose $\mathcal{H}_{\text{universe}} = \{\text{any function in the universe}\}$?

• We can now ask, “is $\mathcal{H}_{\text{universe}}$ PAC learnable with ERM or with any other learning algorithms?”
Probably approximately correct

A hypothesis class $\mathcal{H}$ is PAC-learnable with a learning algorithm $A$ if for any distribution $\mathcal{D}$, and any $\epsilon > 0$ and $0 \leq \delta \leq 1$, there exists $N > 0$ such that

$$\mathbb{P}_{S \sim \mathcal{D}^n} \left[ L_{\mathcal{D}}(A(S)) - \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') > \epsilon \right] < \delta$$

(6)

for any $n \geq N$. 
Suppose $|\mathcal{X}| = 2m$. For any learning algorithm $A$, there is a distribution $\mathcal{D}$ and $f : \mathcal{X} \rightarrow \{0, 1\}$ such that $L_{\mathcal{D}}(f) = 0$, but

$$\mathbb{P}_{S \sim \mathcal{D}^m} \left[ L_{\mathcal{D}}(A(S)) \geq \frac{1}{10} \right] \geq \frac{1}{10}.$$  

(7)
No free lunch theorem

- The 2 and 10 are constants that can be relaxed.

- In words, for any learning algorithm, there exists a distribution and a perfect function, but the learning algorithm has a sufficiently large error with sufficiently high probability.

- What should we do?

- Don’t compare to the best $f$ in the universe.

- Compare to the best in the hypothesis space.
No free lunch theorem

all functions
No free lunch theorem

all functions

$\mathcal{H}$
No free lunch theorem

all functions

\[ \mathcal{H} \]