Machine Learning

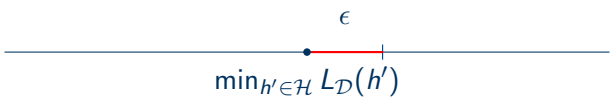
Lecture 15: Generalization 2

Hao Tang

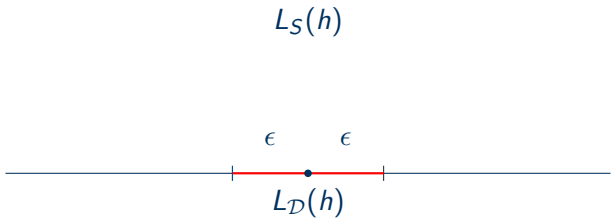
October 26, 2022

PAC learning

$L_{\mathcal{D}}(A(S))$



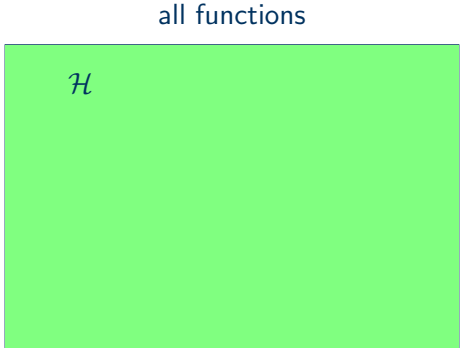
Uniform convergence



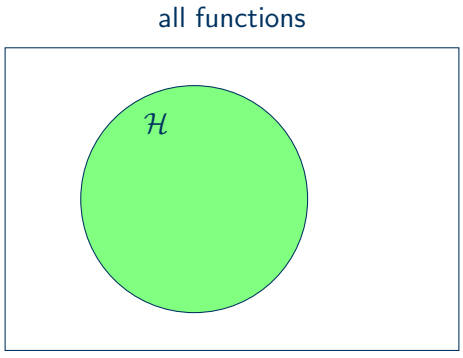
No free lunch theorem

all functions

No free lunch theorem



No free lunch theorem



Error decomposition

- The generalization error can be decomposed into

$$L_{\mathcal{D}}(h) = \left[L_{\mathcal{D}}(h) - \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') \right] + \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h'). \tag{1}$$

- Estimation error can be controlled if we do ERM and have uniform convergence.
- Approximation error can be controlled by changing the size of $\mathcal{H}$.

Generalization bounds

- Many (but not all) generalization bounds have the following form.
- With probability $1 - \delta$, for all $h \in \mathcal{H}$

$$L_{\mathcal{D}}(h) \leq L_S(h) + \sqrt{\frac{C(\mathcal{H})}{n}} + \sqrt{\frac{\log(1/\delta)}{2n}}. \tag{2}$$

- n is the number of samples.
- $C(\mathcal{H})$ is a capacity measure of $\mathcal{H}$.
- There is a family of uniform convergence results.

Sample complexity

- How many samples do we need to achieve a certain error?
- How large should n to get to ϵ?

$$\sqrt{\frac{C(\mathcal{H})}{n}} + \sqrt{\frac{\log(1/\delta)}{2n}} \leq \epsilon \tag{3}$$

- In other words,

$$n = O\left(\frac{C(\mathcal{H}) + \log(1/\delta)}{\epsilon^2}\right) \tag{4}$$

VC generalization bounds

- Vapnik-Chervonenkis generalization bounds

$$L_{\mathcal{D}}(h) \leq L_S(h) + 2\sqrt{\frac{8d\log(en/d) + 2\log(4/\delta)}{n}} \tag{5}$$

- d is called the VC dimension.
- For linear classifiers $\mathcal{H} = \{x \mapsto w^\top \phi(x) : w \in \mathbb{R}^p\}$, VC-dim$(\mathcal{H}) = p + 1$.
- For multilayer perceptrons with p edges, VC-dim$(\mathcal{H}) = O(p \log p)$.
- These results are independent of learning algorithms.
- In particular, it is independent of how ERM is done.

Capacity measure of $\mathcal{H}$

- Shattering
- Norm
- Margin

Shattering

- Given n data points, there are 2^n ways of label them $\{+1, -1\}$.
- A set of n points is shattered by $\mathcal{H}$ if there is an arrangement of n points such that classifiers in $\mathcal{H}$ can produce all 2^n ways of labeling.
- VC dimension is the largest number of points that $\mathcal{H}$ can shatter.

Shattering points in 2D

- We could shatter 3 points with a line in 2D.
- However, we cannot shatter 4 points with a line in 2D.
- The VC dimension of lines in 2D is 3.
- In general, linear classifiers with p parameters have VC dimension $p + 1$.
- We can again shatter 4 points with a 2-layer MLP in 2D.
- Neural networks have larger VC dimension than linear classifiers.

Interpreting generalization bounds

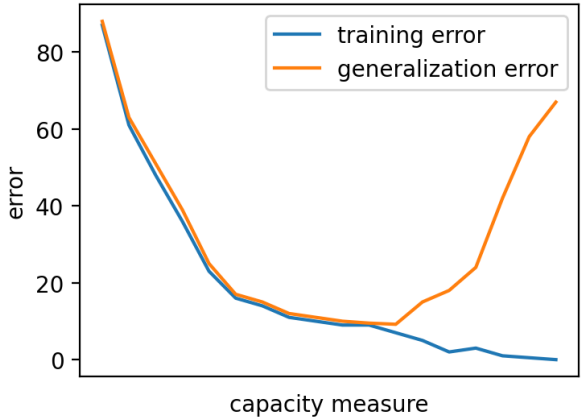
- VC generalization bounds

$$L_{\mathcal{D}}(h) \leq L_S(h) + 2\sqrt{\frac{8d\log(en/d) + 2\log(4/\delta)}{n}} \tag{6}$$

- When $\mathcal{H}$ is large, $\min_{h\in\mathcal{H}} L_S(h)$ can be low.

- When $\mathcal{H}$ is large, d becomes large.

Capacity-generalization tradeoff



Optimization

- We can only do ERM for a limited number of cases, for example, $w = (XX^\top)^{-1}Xy$ in linear regression.

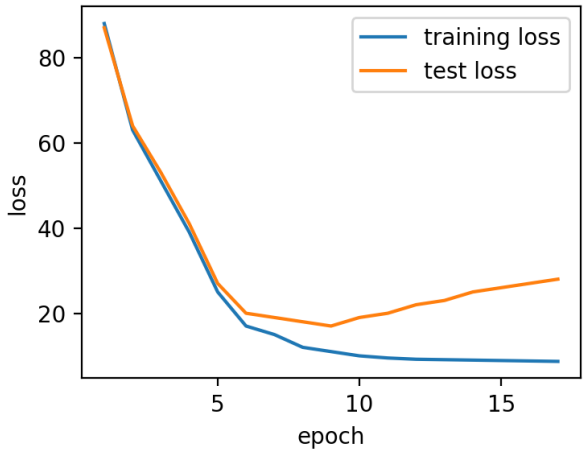
- Recall that the convergence of an optimization algorithm tells us how many iterations we need (how large t should be) to get to

$$L_S(h_t) - \min_{h \in \mathcal{H}} L_S(h) < \epsilon. \tag{7}$$

- The number of iterations (or gradient updates) is often devided by the number of training samples.

- A pass through a data set is called an epoch.

Optimization



Optimization

- We care about generalization of zero-one loss, not the cross entropy or the log likelihood.
- Cross entropy or the log likelihood are called surrogate losses.
- Surrogate losses are easier to optimize than the task loss, and usually have some connection to the task loss.
- For example, log loss is easier to optimize than zero-one loss, and is a smooth approximation of zero-one loss.

Error decomposition

- Optimization error
  - Mismatch between the surrogate loss and the task loss
  - Controlled by the optimization algorithm
- Estimation error
  - Controlled if we do ERM and have uniform convergence
  - Controlled by the capacity of $\mathcal{H}$ and the size of the training set
- Approximation error
  - Controlled by the capacity of $\mathcal{H}$

Underfitting

- A model is underfitting if there is another model that has a lower training.
- A model h is underfitting if there is f such that $L_S(f) < L_S(h)$.
- The better f is unknown unless we find it.
- All models are underfitting with respect to ERM.
- When people say a model is underfitting, they simply mean there is room to improve the training error.

Overfitting

- A model is overfitting if there is another model that has a higher training error but a lower test eror.
- A model h is overfitting if there is f such that $L_S(f) > L_S(h)$ and $L_{S'}(f) < L_{S'}(h)$.
- The better f is unknown unless we find it.
- Models can overfit, even though the gap $|L_S(h) - L_{S'}(h)|$ between training and test is not large.
- When people say a model is overfitting, they simply mean there is a large gap between the training and test error.

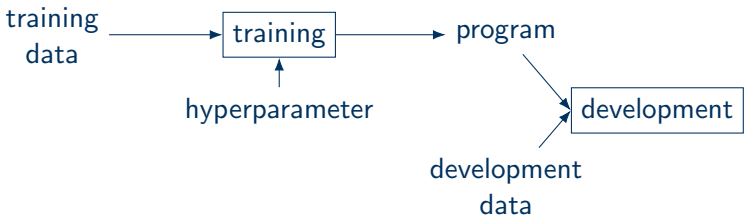
In practice

- We minimize a *surrogate* loss on the training set S, i.e., doing ERM.
- We can only do ERM approximately most of the time, because of optimization difficulty.
- Suppose training gives us $\hat{h}$.
- We use a test set S' and measure *task* loss $L_{S'}(\hat{h})$ to approximate generalization error.
- We hope $L_{\mathcal{D}}(\hat{h})$ is small when $L_{S'}(\hat{h})$ is small.

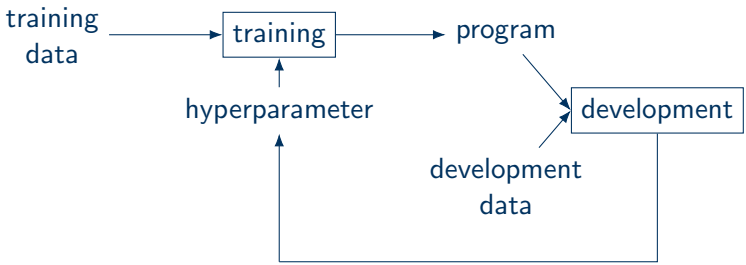
Test set

- Test error on a test set is used to approximate generalization error.
- Test set is supposed to be considered as an indepdent data drawn from the unknown distribution.
- Sometimes we have hyperparameters (not learned from data) we need to tune, for example, the step size in stochastic gradient descent.
- What's the problem of using the test set to tune hyperparameters?

How to measure generalization



How to measure generalization



How to measure generalization