Sample complexity

The **sample complexity** of a machine learning algorithm represents the number of training-samples that it needs in order to successfully learn a target function.

More precisely, the sample complexity is the number of training-samples that we need to supply to the algorithm, so that the function returned by the algorithm is within an arbitrarily small error of the best possible function, with probability arbitrarily close to 1.

There are two variants of sample complexity:

- The weak variant fixes a particular input-output distribution;
- The strong variant takes the worst-case sample complexity over all input-output distributions.

The No free lunch theorem, discussed below, proves that, in general, the strong sample complexity is infinite, i.e. that there is no algorithm that can learn the globally-optimal target function using a finite number of training samples.

However, if we are only interested in a particular class of target functions (e.g, only linear functions) then the sample complexity is finite, and it depends linearly on the VC dimension on the class of target functions.[1]

Contents

Definition

Let X be a space which we call the input space, and Y be a space which we call the output space, and let Z denote the product $X \times Y$. For example, in the setting of binary classification, X is typically a finite-dimensional vector space and Y is the set $\{-1, 1\}$.

Fix a hypothesis space $\mathcal{H}$ of functions $h\colon X \to Y$. A learning algorithm over $\mathcal{H}$ is a computable map from Z^* to $\mathcal{H}$. In other words, it is an algorithm that takes as input a finite sequence of training samples and outputs a function from X to Y. Typical learning algorithms include empirical risk minimization, without or with Tikhonov regularization.

Fix a loss function $L: Y \times Y \to \mathbb{R}_{\geq 0}$, for example, the square loss $L(y, y') = (y - y')^2$, where $h(x) = y'$. For a given distribution ρ on $X \times Y$, the **expected risk** of a hypothesis (a function) $h \in \mathcal{H}$ is

$$\mathcal{E}(h) := \mathbb{E}_\rho[L(h(x), y)] = \int_{X \times Y} L(h(x), y)\, d\rho(x, y)$$

In our setting, we have $h = \mathcal{A}(S_n)$, where $\mathcal{A}$ is a learning algorithm and $S_n = ((x_1, y_1), \ldots, (x_n, y_n)) \sim \rho^n$ is a sequence of vectors which are all drawn independently from ρ. Define the optimal risk

$$\mathcal{E}^*_{\mathcal{H}} = \inf_{h \in \mathcal{H}} \mathcal{E}(h).$$

Set $h_n = \mathcal{A}(S_n)$, for each n. Note that h_n is a random variable and depends on the random variable S_n, which is drawn from the distribution ρ^n. The algorithm $\mathcal{A}$ is called **consistent** if $\mathcal{E}(h_n)$ probabilistically converges to $\mathcal{E}^*_{\mathcal{H}}$. In other words, for all $\epsilon, \delta > 0$, there exists a positive integer N, such that, for all $n \geq N$, we have

$$\Pr_{\rho^n}[\mathcal{E}(h_n) - \mathcal{E}^*_{\mathcal{H}} \geq \varepsilon] < \delta.$$

The **sample complexity** of $\mathcal{A}$ is then the minimum N for which this holds, as a function of ρ, ϵ, and δ. We write the sample complexity as $N(\rho, \epsilon, \delta)$ to emphasize that this value of N depends on ρ, ϵ, and δ. If $\mathcal{A}$ is **not consistent**, then we set $N(\rho, \epsilon, \delta) = \infty$. If there exists an algorithm for which $N(\rho, \epsilon, \delta)$ is finite, then we say that the hypothesis space $\mathcal{H}$ is **learnable**.

In others words, the sample complexity $N(\rho, \epsilon, \delta)$ defines the rate of consistency of the algorithm: given a desired accuracy ϵ and confidence δ, one needs to sample $N(\rho, \epsilon, \delta)$ data points to guarantee that the risk of the output function is within ϵ of the best possible, with probability at least $1 - \delta$.[2]

In probably approximately correct (PAC) learning, one is concerned with whether the sample complexity is *polynomial*, that is, whether $N(\rho, \epsilon, \delta)$ is bounded by a polynomial in $1/\epsilon$ and $1/\delta$. If $N(\rho, \epsilon, \delta)$ is polynomial for some learning algorithm, then one says that the hypothesis space $\mathcal{H}$ is **PAC-learnable**. Note that this is a stronger notion than being learnable.

Unrestricted hypothesis space: infinite sample complexity

One can ask whether there exists a learning algorithm so that the sample complexity is finite in the strong sense, that is, there is a bound on the number of samples needed so that the algorithm can learn any distribution over the input-output space with a specified target error. More formally, one asks whether there exists a learning algorithm $\mathcal{A}$, such that, for all $\epsilon, \delta > 0$, there exists a positive integer N such that for all $n \geq N$, we have

$$\sup_\rho \left(\Pr_{\rho^n}[\mathcal{E}(h_n) - \mathcal{E}^*_{\mathcal{H}} \geq \varepsilon] \right) < \delta,$$

where $h_n = \mathcal{A}(S_n)$, with $S_n = ((x_1, y_1), \ldots, (x_n, y_n)) \sim \rho^n$ as above. The No Free Lunch Theorem says that without restrictions on the hypothesis space $\mathcal{H}$, this is not the case, i.e., there always exist "bad"

distributions for which the sample complexity is arbitrarily large.[1]

Thus, in order to make statements about the rate of convergence of the quantity

$$\sup_{\rho}\left(\Pr_{\rho^n}[\mathcal{E}(h_n)-\mathcal{E}^*_{\mathcal{H}}\geq\varepsilon]\right),$$

one must either

- constrain the space of probability distributions ρ, e.g. via a parametric approach, or
- constrain the space of hypotheses $\mathcal{H}$, as in distribution-free approaches.

Restricted hypothesis space: finite sample-complexity

The latter approach leads to concepts such as VC dimension and Rademacher complexity which control the complexity of the space $\mathcal{H}$. A smaller hypothesis space introduces more bias into the inference process, meaning that $\mathcal{E}^*_{\mathcal{H}}$ may be greater than the best possible risk in a larger space. However, by restricting the complexity of the hypothesis space it becomes possible for an algorithm to produce more uniformly consistent functions. This trade-off leads to the concept of regularization.[2]

It is a theorem from VC theory that the following three statements are equivalent for a hypothesis space $\mathcal{H}$:

1. $\mathcal{H}$ is PAC-learnable.
2. The VC dimension of $\mathcal{H}$ is finite.
3. $\mathcal{H}$ is a uniform Glivenko-Cantelli class.

This gives a way to prove that certain hypothesis spaces are PAC learnable, and by extension, learnable.

An example of a PAC-learnable hypothesis space

$X=\mathbb{R}^d, Y=\{-1,1\}$, and let $\mathcal{H}$ be the space of affine functions on X, that is, functions of the form $x\mapsto\langle w,x\rangle+b$ for some $w\in\mathbb{R}^d, b\in\mathbb{R}$. This is the linear classification with offset learning problem. Now, note that four coplanar points in a square cannot be shattered by any affine function, since no affine function can be positive on two diagonally opposite vertices and negative on the remaining two. Thus, the VC dimension of $\mathcal{H}$ is $d+1$, so it is finite. It follows by the above characterization of PAC-learnable classes that $\mathcal{H}$ is PAC-learnable, and by extension, learnable.

Sample-complexity bounds

Suppose $\mathcal{H}$ is a class of binary functions (functions to $\{0,1\}$). Then, $\mathcal{H}$ is (ϵ,δ)-PAC-learnable with a sample of size: [3]

$$N=O\left(\frac{VC(\mathcal{H})+\ln\frac{1}{\delta}}{\epsilon}\right)$$

where $VC(\mathcal{H})$ is the VC dimension of $\mathcal{H}$. Moreover, any (ϵ, δ)-PAC-learning algorithm for $\mathcal{H}$ must have sample-complexity:[4]

$$N = \Omega\left(\frac{VC(\mathcal{H}) + \ln\frac{1}{\delta}}{\epsilon}\right)$$

Thus, the sample-complexity is a linear function of the VC dimension of the hypothesis space.

Suppose $\mathcal{H}$ is a class of real-valued functions with range in $[0, T]$. Then, $\mathcal{H}$ is (ϵ, δ)-PAC-learnable with a sample of size: [5][6]

$$N = O\left(T^2 \frac{PD(\mathcal{H})\ln\frac{T}{\epsilon} + \ln\frac{1}{\delta}}{\epsilon^2}\right)$$

where $PD(\mathcal{H})$ is Pollard's pseudo-dimension of $\mathcal{H}$.

Other Settings

In addition to the supervised learning setting, sample complexity is relevant to semi-supervised learning problems including active learning,[7] where the algorithm can ask for labels to specifically chosen inputs in order to reduce the cost of obtaining many labels. The concept of sample complexity also shows up in reinforcement learning,[8] online learning, and unsupervised algorithms, e.g. for dictionary learning.[9]

Efficiency in robotics

A high sample complexity means, that many calculations are needed for running a Monte Carlo tree search.[10] Its equal to a model free brute force search in the state space. In contrast, a high efficiency algorithm has a low sample complexity.[11] Possible techniques for reducing the sample complexity are metric learning[12] and model based reinforcement learning.[13]

References

1. Vapnik, Vladimir (1998), *Statistical Learning Theory*, New York: Wiley.
2. Rosasco, Lorenzo (2014), *Consistency, Learnability, and Regularization*, Lecture Notes for MIT Course 9.520.
3. Steve Hanneke (2016). "The optimal sample complexity of PAC learning" (https://www.jmlr.org/papers/v17/15-389.html). *J. Mach. Learn. Res.* **17** (1): 1319–1333.
4. Ehrenfeucht, Andrzej; Haussler, David; Kearns, Michael; Valiant, Leslie (1989). "A general lower bound on the number of examples needed for learning" (https://doi.org/10.1016%2F0890-5401%2889%2990002-3). *Information and Computation*. **82** (3): 247. doi:10.1016/0890-5401(89)90002-3 (https://doi.org/10.1016%2F0890-5401%2889%2990002-3).
5. Anthony, Martin; Bartlett, Peter L. (2009). *Neural Network Learning: Theoretical Foundations*. ISBN 9780521118620.
6. Morgenstern, Jamie; Roughgarden, Tim (2015). *On the Pseudo-Dimension of Nearly Optimal Auctions* (http://papers.nips.cc/paper/5766-on-the-pseudo-dimension-of-nearly-optimal-auctions). NIPS. Curran Associates. pp. 136–144. arXiv:1506.03684 (https://arxiv.org/abs/1506.03684).

7. Balcan, Maria-Florina; Hanneke, Steve; Wortman Vaughan, Jennifer (2010). "The true sample complexity of active learning" (https://doi.org/10.1007%2Fs10994-010-5174-y). *Machine Learning*. **80** (2–3): 111–139. doi:10.1007/s10994-010-5174-y (https://doi.org/10.1007%2Fs10994-010-5174-y).
8. Kakade, Sham (2003), *On the Sample Complexity of Reinforcement Learning* (http://www.ias.tu-darmstadt.de/uploads/Research/NIPS2006/SK.pdf) (PDF), PhD Thesis, University College London: Gatsby Computational Neuroscience Unit.
9. Vainsencher, Daniel; Mannor, Shie; Bruckstein, Alfred (2011). "The Sample Complexity of Dictionary Learning" (http://www.jmlr.org/papers/volume12/vainsencher11a/vainsencher11a.pdf) (PDF). *Journal of Machine Learning Research*. **12**: 3259–3281.
10. Kaufmann, Emilie and Koolen, Wouter M (2017). *Monte-carlo tree search by best arm identification*. Advances in Neural Information Processing Systems. pp. 4897–4906.
11. Fidelman, Peggy and Stone, Peter (2006). *The chin pinch: A case study in skill learning on a legged robot*. Robot Soccer World Cup. Springer. pp. 59–71.
12. Verma, Nakul and Branson, Kristin (2015). *Sample complexity of learning mahalanobis distance metrics*. Advances in neural information processing systems. pp. 2584–2592.
13. Kurutach, Thanard and Clavera, Ignasi and Duan, Yan and Tamar, Aviv and Abbeel, Pieter (2018). "Model-ensemble trust-region policy optimization". arXiv:1802.10592 (https://arxiv.org/abs/1802.10592) [cs.LG (https://arxiv.org/archive/cs.LG)].
