Chapter 1

Deep Learning Theory

Notes based on blog by Desh Raj https://desh2608.github.io/.

1.1 Optimization

Neural networks can be viewed as trying to minimize a loss function. The reason this is difficult is because the loss function is non-convex. Non-convex functions can have many local minima, saddle points, flat regions, and varying curvature which makes non-convex optimization at least NP-Hard (https://www.cs.cornell.edu/courses/cs6787/2017fa/Lecture7.pdf). If this is the case, why can deep networks find approximately reasonable solutions?

Almost all neural networks are trained by moving the parameters in the direction of a non-zero gradient. The goals of such descent are to find a critical point $\nabla = 0$ and to find local optimum $\nabla = 0$ and $\nabla^2 > 0$ (aka Hessian is positive semi-definite).

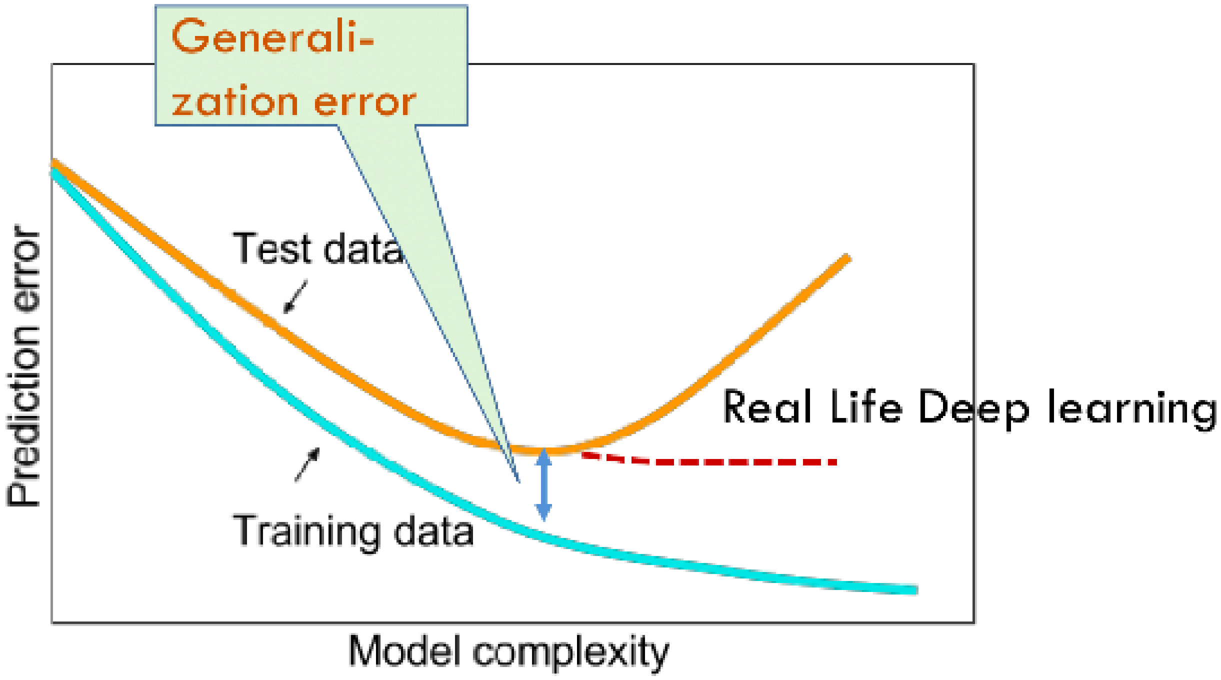
Critical Points Parameter θ, usually update in the following form: $\theta_{n+1} = \theta_n - \eta \nabla f(\theta_n)$. The hyperparameter in the update equation is the learning rate, η. We know that we want it to be small but how do we know that η is small enough? We utilize the Hessian aka ∇^2. Suppose there exists some β such that $-\beta I \leq \nabla^2 f(\theta) \leq \beta I$. Then a higher β means ∇^2 varies more so the learning rate should be lower. It can be more formally proved that in $O(\frac{\beta}{\epsilon^2})$, θ will arrive at a critical point.

Saddle Points One issue that such a descent may run into is the issue of saddle points. Several results (http://proceedings.mlr.press/v40/Ge15.pdf, https://arxiv.org/pdf/1602.04915.pdf, https://arxiv.org/pdf/1703.00887.pdf) show that gradient descent can be done by evading saddle points. The main ideas from these papers are that there are many saddle points but it is hard to converge to one and while Hessians can be used to avoid the saddle points, we do not actually have to because a noisy form of gradient descent converges to a local optimum in a polynomial number of steps. The last paper discusses how perturbed gradient descent does a better job of escaping saddle points than regular gradient descent.

1.2 Generalization

What is impressive of most modern deep learning frameworks is that many generalize very well to test cases even when the number of parameters is greater than the number of training samples. This is counterintuitive because one would think that deep neural networks would overfit with a small number of samples.

A common 'folklore' experiment (discussed in cs.princeton.edu/~rlivni/files/papers/LivnComputational.pdf) 'is to show that for sufficiently over-specified networks, global optima are ubiquitous and in general



computationally easy to find.' To understand this conceptually, think of fixing a 2 layer neural network with n hidden nodes. We provide it random inputs and collect the outputs. We then use these input/output pairs to train randomly initialized (2 layer) neural network. With only 2 layers, this would be quite difficult but it becomes easier when we increase the number of hidden nodes.

Capacity The capacity of a learning model can be viewed as the complexity of training samples that it can fit. A quadratic regression will have more capacity than linear regression. It can be proved that the following is true:

Test Loss - training loss $\leq \sqrt{\frac{N}{m}}$ where N is the effective capacity and m is the number of training samples. There are other measures such as Rademacher complexity and VC dimension that relate error to capacity. However, these fail for deep neural networks because the upper bounds from these measures are greater than 1.

Excess Capacity From the figure we see that in general deep neural networks are able to generalize relatively well. For it was believed that the performance of deep neural networks did not necessarily mean high capacity and that a combination of stochastic gradient descent and regularization eliminates the excess capacity of a neural network. However, this was proved wrong in a paper from 2017 (`https://arxiv.org/abs/1611.03530`). The experimental findings of the paper include that a classic convolutional neural net, like Alexnet, could be trained with random labels and still achieve high accuracy on training data. The paper proved that 'there exists a two-layer neural network with RELu activations and $2n + d$ weights that can represent any function on a sample size n in d dimensions. Recent work has shown that excess capacity can also exist in Kernel methods.

1.3 Other notions of Generalizability

There are some other correlations of good generalizability with flat minima and noise stability implying compressibility. The idea of flat minima is an old one but noise stability is relatively new. The idea of noise stability starts with measuring capacity using Gaussian noise. What happens is that zero-mean Gaussian noise is added at an intermediate output of a neural network which is reduced in higher layers. The capacity of the network to fit this random noise can be measured. If a network has the capacity for the noise then it is considered to have noise stability. Noise stability implies compressibility, which is defined as 'given a network C with N parameters and some training loss, compression means obtaining a new network C that has fewer parameters and the training loss effectively remains the same which would mean that the new network would have better generalization capability. There is a more formal definition of noise capacity involving the distribution of singular values when noise is added to a neural-network.

1.4 Bounding Capacity

PAC-Bayes The PAC-Bayes theorem informally bounds expected error of a classifier chosen from a distribution Q in terms of KL-Divergence from a priori fixed distribution P. KL-Divergence is a metric that compares the similarity between probability distributions:

$$KL(p||q) = \mathbb{E}(logp(x) - logq(x)) = \sum_{i=1}^{N} p(x_i)(logp(x_i) - logq(x_i))$$

PAC-Bayes relates to the following bound:

$$KL(\hat{e}(Q, S_m)||e(q)) \leq \frac{KL(Q||P) + log\frac{m}{\delta}}{m-1}$$

where $\hat{e}(Q, S_m)$ is the empirical loss of Q with respect to some i.i.d sample S_m and $e(Q)$ is the expected loss. PAC-Bayes bounds the test error of a binary variant of MNIST in 16-22% where the actual bound is around 3%.

Compressibility Arora et al introduced boundaries using compressions where the loss is bounded by q parameters having at most r discrete values.

1.5 Depth of Neural Networks

The proofs for these are relatively involved but it can be shown that 'radial functions can be approximated by depth-3 networks but not with depth-2 networks and that functions expressible by $\theta(k^3)$-depth networks of constant width cannot be approximated by $O(k)$-depth networks with polynomial width. Recent work might also suggest that increasing depth can speed up optimization.