UNDERSTANDING DEEP LEARNING REQUIRES RETHINKING GENERALIZATION

Aldo Lamarre [1]   Matthew C. Scicluna [2]

Feb 21 2018

[1]Département d'Informatique et de Recherche Opérationnelle
Université de Montréal

[2]Montréal Institute of Learning Algorithms
Université de Montréal

Table of contents

Introduction

Main Question

Main Question

What distinguishes Neural Networks that generalize well from those that don't?

- Capacity ?
- Regularization ?
- How we train the model?

Traditional View

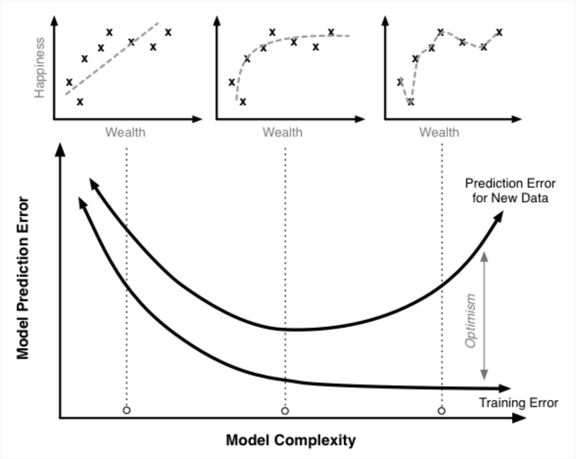

Figure 1: Traditional view of generalization. Image taken from [1]

Motivation

Why do we care about the problem?

- Make neural networks more interpretable
- May lead to more principled and reliable model architecture design

Background

Previous Approaches

We can bound the Generalization Error using measures of complexity such as:

- VC Dimension
- Rademacher Complexity
- Uniform Stability

Additionally, regularization can help (including Early Stopping)

Related Work

In 2016 Hardt et al. gives an Upper bound on Generalization error on model using SGD using uniform stability [2]

BUT

Uniform stability is a property of a learning algorithm and is not affected by the labelling of the training data.

Limitations

Main Message

Statistical Learning Theory is insufficient in that it cannot distinguish between neural networks with dramatically different generalization performance.

This is demonstrated in the paper [3]. The central finding:

Deep neural networks easily fit random labels

Results

Experiment

Setup: trained several standard architectures on the data with various modifications:

1. True labels $\rightarrow$ No modifications
2. Random labels $\rightarrow$ randomly changed some labels
3. shuffled pixels $\rightarrow$ apply some fixed permutation of pixels to all images
4. Random pixels $\rightarrow$ apply some random permutation of pixels to all images
5. Gaussian $\rightarrow$ Generate pixels for all images from a Gaussian

Main Results

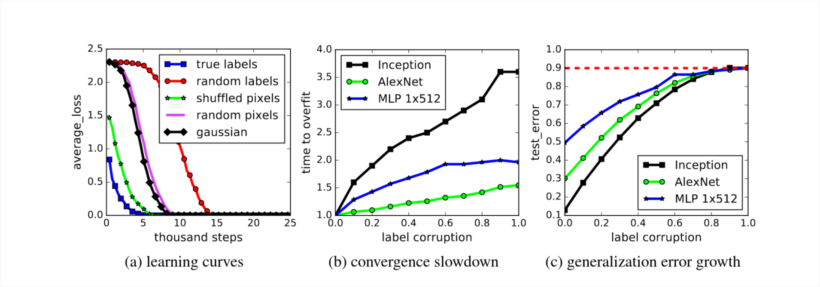

Figure 2: Fitting random labels and random pixels on CIFAR10.

Results

In most cases, the training error went to zero while test error was high

Notice:

the model capacity, hyperparameters, and the optimizer remained the same!

Results

Explicit regularization may improve generalization performance, but is neither necessary nor by itself sufficient for controlling generalization error

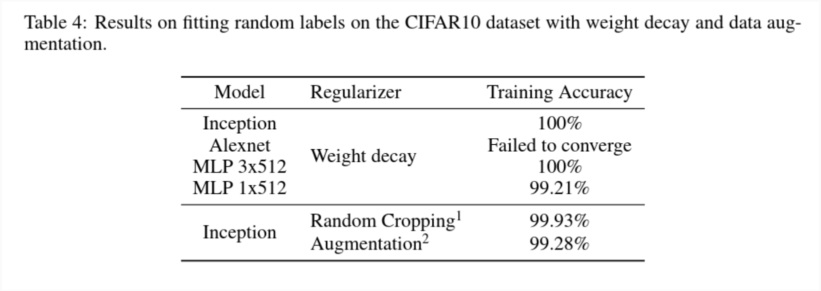
Table 4: Results on fitting random labels on the CIFAR10 dataset with weight decay and data augmentation.

Model	Regularizer	Training Accuracy
Inception	Weight decay	100%
Alexnet		Failed to converge
MLP 3x512		100%
MLP 1x512		99.21%
Inception	Random Cropping[1]	99.93%
	Augmentation[2]	99.28%

Technical dive

Finite-sample expressivity

Theorem

There exists[1] a two-layer neural network with ReLU activations and $2n + d$ weights that can represent any function on a sample of size n in d dimensions.

[1]NOT all networks satisfy this

Proof

Lemma 1

For any two interleaving sequences of n real numbers $b_1 < x_1 < b_2 < x_2 \cdots < b_n < x_n$, the $n \times n$ matrix $A = [max\{x_i - b_j, 0\}]_{ij}$ has full rank. Its smallest eigenvalue is $min_i\{x_i - b_i\}$

Proof

For weight vectors $w, b \in R^n$ and $a \in R^d$, consider the function $c : R^n \to R$,

$$c(x) = \sum_{j=1} w_j \, max\{a^T x - b_j, 0\}$$

Proof

For weight vectors $w, b \in R^n$ and $a \in R^d$, consider the function $c : R^n \to R$,

$$c(x) = \sum_{j=1} w_j \, max\{a^T x - b_j, 0\}$$

- This can be done trivially with a depth 2 neural network with relu.

Proof

For weight vectors $w, b \in R^n$ and $a \in R^d$, consider the function $c : R^n \to R$,

$$c(x) = \sum_{j=1} w_j \, max\{a^T x - b_j, 0\}$$

- Now, fixing a sample $S = z_1, \ldots, z_n$ of size n and a target vector $y \in R_n$. We need to find weights a, b, w so that $y_i = c(z_i)$ for all $i \in \{1, \ldots, n\}$

Proof

For weight vectors $w, b \in R^n$ and $a \in R^d$, consider the function $c : R^n \to R$,

$$c(x) = \sum_{j=1} w_j \, max\{a^T x - b_j, 0\}$$

- First, choose a and b such that with $x_i = a_i^T z_i$ we have the interleaving property $b_1 < x_1 < b_2 < \cdots < b_n < x_n$ Next, consider the set of n equations in the n unknowns w,

  $$y_i = c(z_i), i \in \{1, \ldots, n\}$$

  We have $c(z_i) = Aw$, where $A = [max\{x_i b_i, 0\}]_{ij}$ is the matrix of Lemma 1.

Proof

For weight vectors $w, b \in R^n$ and $a \in R^d$, consider the function $c : R^n \to R$,

$$c(x) = \sum_{j=1} w_j \, max\{a^T x - b_j, 0\}$$

- Now, fixing a sample $S = z_1, \ldots, z_n$ of size n and a target vector $y \in R_n$. We need to find weights a, b, w so that $y_i = c(z_i)$ for all $i \in \{1, \ldots, n\}$
- We chose a and b so that the lemma applies and hence A has full rank. We can now solve the linear system $y = Aw$ to find suitable weights w.

Discussion

Some Thoughts...

1. Our favourite papers are the ones that shed light on truths that are taken for granted.
2. Its obvious that randomizing the labels would eliminate generalizability, but finding precise mathematical statements about this is not!
3. The paper doesn't really make many conclusions of its own.
4. Models used in practice have the capability of memorizing the training data. Is it somehow easier not to?
5. The interplay between generalization and ease of optimization seems like an interesting thing to explore...

Any Questions...

Any Questions?

References

- D. Sowinski, "What is generalization in machine learning?." Post.
- M. Hardt, B. Recht, and Y. Singer, "Train faster, generalize better: Stability of stochastic gradient descent," *CoRR*, vol. abs/1509.01240, 2015.
- C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," *CoRR*, vol. abs/1611.03530, 2016.