Introduction

Theoretical Deep Learning

Eugene Golikov

MIPT, spring 2019

Neural Networks and Deep Learning Lab., MIPT

Intro

Levels of research which studies NNs:

Intro

Levels of research which studies NNs:

1. Competition level:
    - Try almost random things; do not invent anything truly novel;
    - Give explanation based on faint intuition;
    - Objective: SOTA on popular datasets;
    - Success is achieved mostly by careful fine-tuning.

Intro

Levels of research which studies NNs:

1. Competition level:
    - Try almost random things; do not invent anything truly novel;
    - Give explanation based on faint intuition;
    - Objective: SOTA on popular datasets;
    - Success is achieved mostly by careful fine-tuning.
2. Invention level:
    - Provide a novel idea based on surmise or intuition;
    - Objective: robust improvement on popular datasets;
    - Fine-tuning can still be crucial for success;
    - Explanation is still mostly intuitive;
    - Examples: Batchnorm, Resnet, Attention, Dropout papers.

Intro

Levels of research which studies NNs:

3. Physics level:
    - Treat NNs as physical objects; make experiments, develop a theory;
    - Experiment (mostly) in simple setups;
    - Objective: gain understanding of how things work;
    - Examples: empirical research of loss surfaces, ability to fit random labels, learning phases.

Intro

Levels of research which studies NNs:

3. Physics level:
    - Treat NNs as physical objects; make experiments, develop a theory;
    - Experiment (mostly) in simple setups;
    - Objective: gain understanding of how things work;
    - Examples: empirical research of loss surfaces, ability to fit random labels, learning phases.
4. Math level:
    - Treat NNs as mathematical objects; prove theorems in simplified setups;
    - Use empirical observations as source of hypotheses;
    - Almost no practical outcome;
    - Objective: "solve a puzzle";
    - Examples: GD achieves 100% train accuracy as long as NN is overparametrized; there exist local minima for sufficiently wide NNs.

Intro

Supervised learning objective:

$$\mathcal{L}_{train}(W) = \mathbb{E}_{x,y\sim\mathcal{D}_{train}} L(y, \hat{y}(x, W)) \to \min_W,$$

where W – network weights, $\hat{y}$ – network response, $\mathcal{D}_{train}$ – train data distribution, L – loss function.
Dimension of $W > 10^4$ (typically $10^6 \div 10^8$).
Optimize with (stochastic) gradient descent.

Intro

Supervised learning objective:

$$\mathcal{L}_{train}(W) = \mathbb{E}_{x,y\sim\mathcal{D}_{train}} L(y, \hat{y}(x, W)) \to \min_W,$$

where W – network weights, $\hat{y}$ – network response, $\mathcal{D}_{train}$ – train data distribution, L – loss function.
Dimension of $W > 10^4$ (typically $10^6 \div 10^8$).
Optimize with (stochastic) gradient descent.

Main puzzles:

1. **A non-overfitting puzzle**
2. **A local optimization puzzle**

A non-overfitting puzzle

Basic theorem of generalization theory:

$$\|R_{test}(W) - R_{train}(W)\| \leq O\left(\sqrt{\frac{N}{m}}\right),$$

where R is the empirical risk (i.e. classification error), m is the number of training examples, and N is the complexity measure.

A non-overfitting puzzle

Basic theorem of generalization theory:

$$\|R_{test}(W) - R_{train}(W)\| \leq O\left(\sqrt{\frac{N}{m}}\right),$$

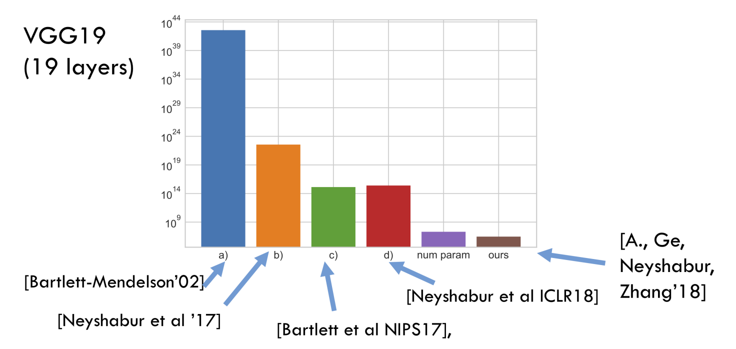
where R is the empirical risk (i.e. classification error), m is the number of training examples, and N is the complexity measure.

Problem:

Existing complexity measures lead to vacuous bounds.

A non-overfitting puzzle

Some existing complexity measures[1]:



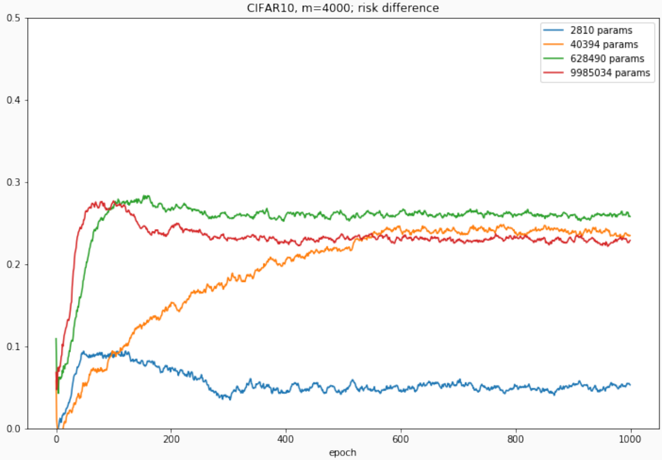
[1] http://www.offconvex.org/2018/02/17/generalization2/

A non-overfitting puzzle

Does network overfits more as the number of parameters grows?

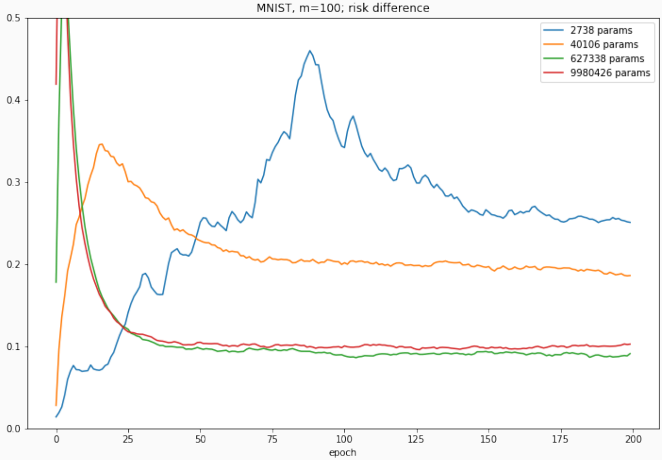
A non-overfitting puzzle

Does network overfits more as the number of parameters grows?



A non-overfitting puzzle

Does network overfits more as the number of parameters grows?



A local optimization puzzle

Learning objective:

$$\mathcal{L}_{train}(W) = \mathbb{E}_{x,y \sim \mathcal{D}_{train}} L(y, \hat{y}(x, W)) \to \min_{W} .$$

A local optimization puzzle

Learning objective:

$$\mathcal{L}_{train}(W) = \mathbb{E}_{x,y \sim \mathcal{D}_{train}} L(y, \hat{y}(x, W)) \to \min_{W}.$$

This optimization problem is proved to be NP-complete.

A local optimization puzzle

Learning objective:

$$\mathcal{L}_{train}(W) = \mathbb{E}_{x,y \sim \mathcal{D}_{train}} L(y, \hat{y}(x, W)) \to \min_{W} .$$

This optimization problem is proved to be NP-complete.

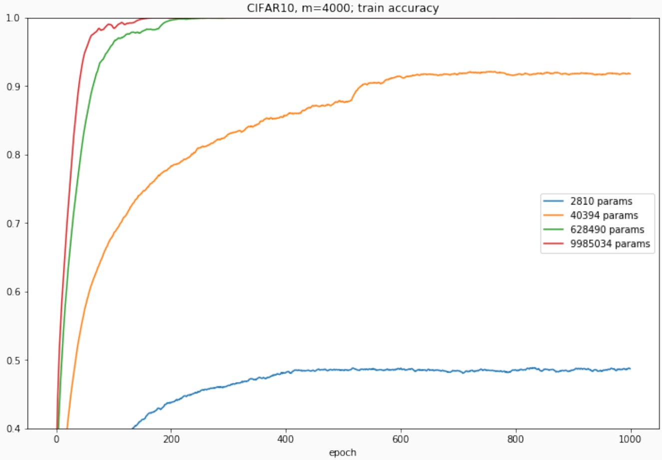
However, empirically as long as the number of parameters is large enough, we can achieve a near-global optimum with gradient descent — a local search method!

A local optimization puzzle

Optimization becomes easier as number of parameters grows:

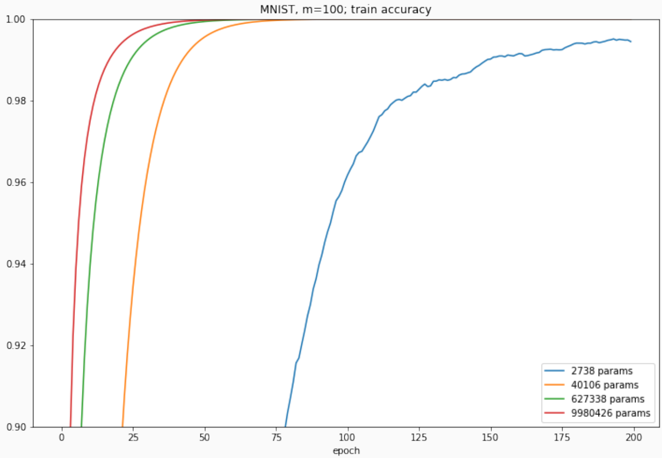
A local optimization puzzle

Optimization becomes easier as number of parameters grows:



A local optimization puzzle

Optimization becomes easier as number of parameters grows:



Other topics

Extra topics:

- Signal propagation in deep and wide nets
- Information bottleneck

Will not be present at the course:

- Expressivity
- Why does Batchnorm / Resnet / Attention / Dropout / Other popular stuff work
- Theory of convolutional neural networks
- Unsupervised learning

Homeworks

Labs ($\sim 20\%$ of final grade):

- We use pytorch[2]
- GPU is desirable

Theoretical assignments ($\sim 30\%$ of final grade)

Based on papers mentioned in lectures

Oral exam ($\sim 50\%$ of final grade)

In the form of interview

E-mail to send homeworks: tdl_course_mipt@protonmail.com

[2] `https://pytorch.org/`