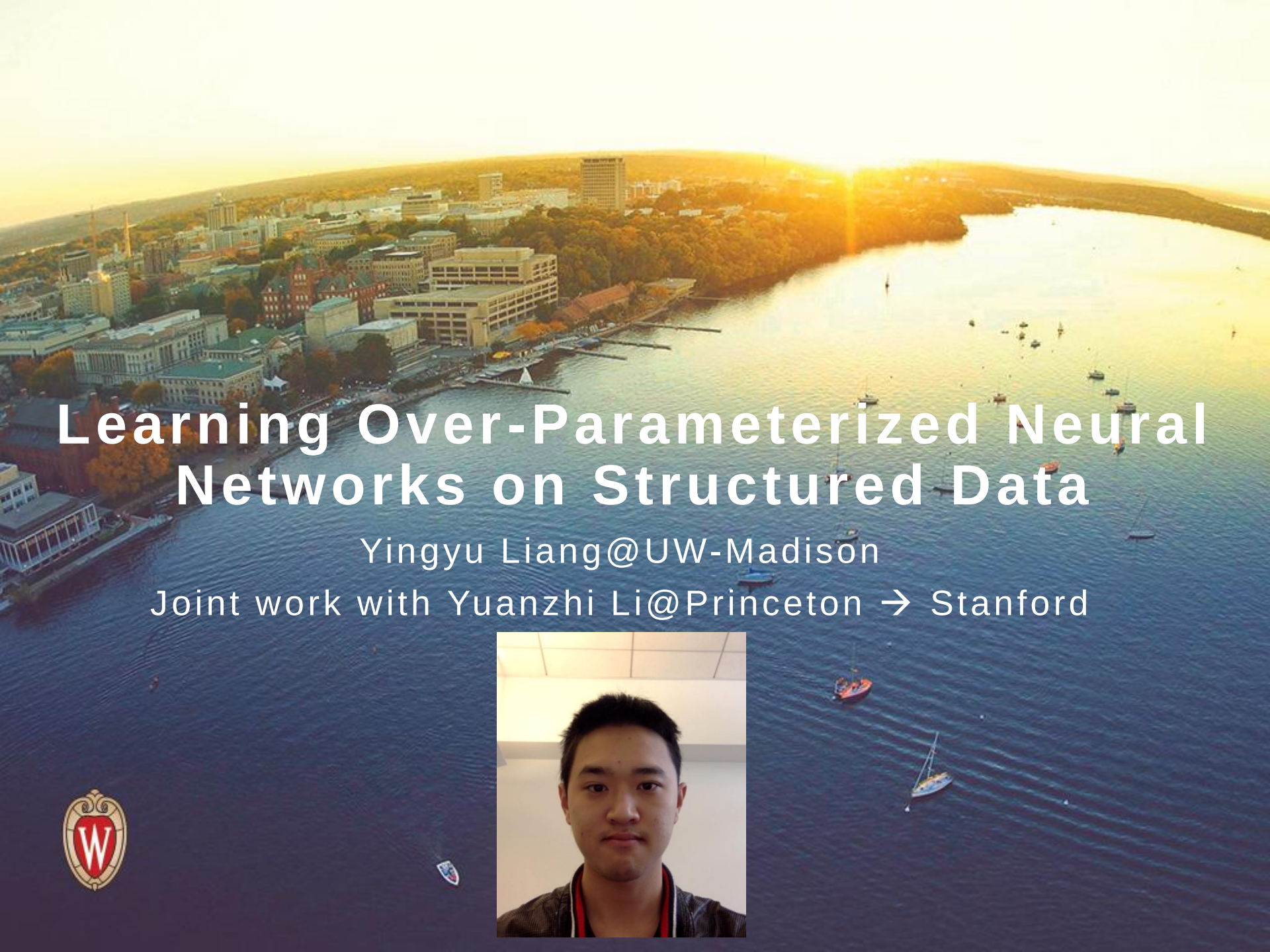
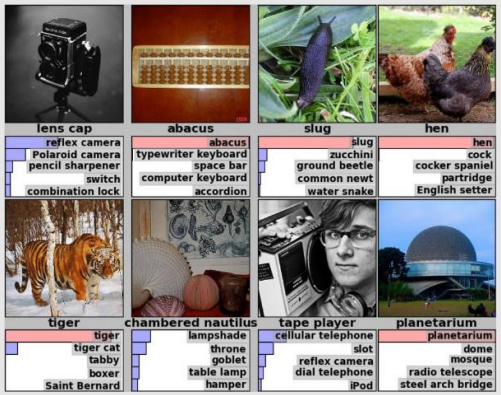


Learning Over-Parameterized Neural Networks on Structured Data

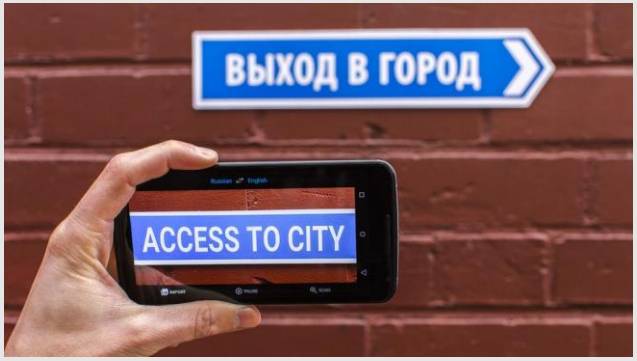
Yingyu Liang@UW-Madison

Joint work with Yuanzhi Li@Princeton → Stanford

Empirical Success of Deep Learning

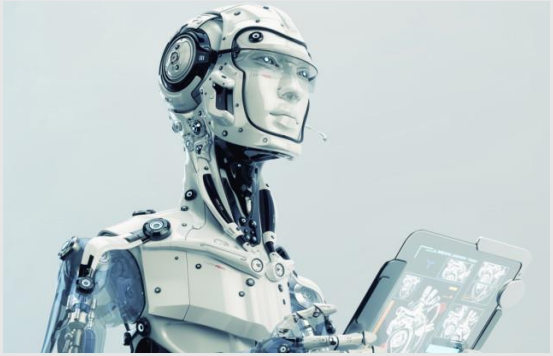


Computer vision



Machine translation

Game playing



Robots

Key Engine Behind the Success

- Deep Neural Networks: $y = f(x)$

Input x

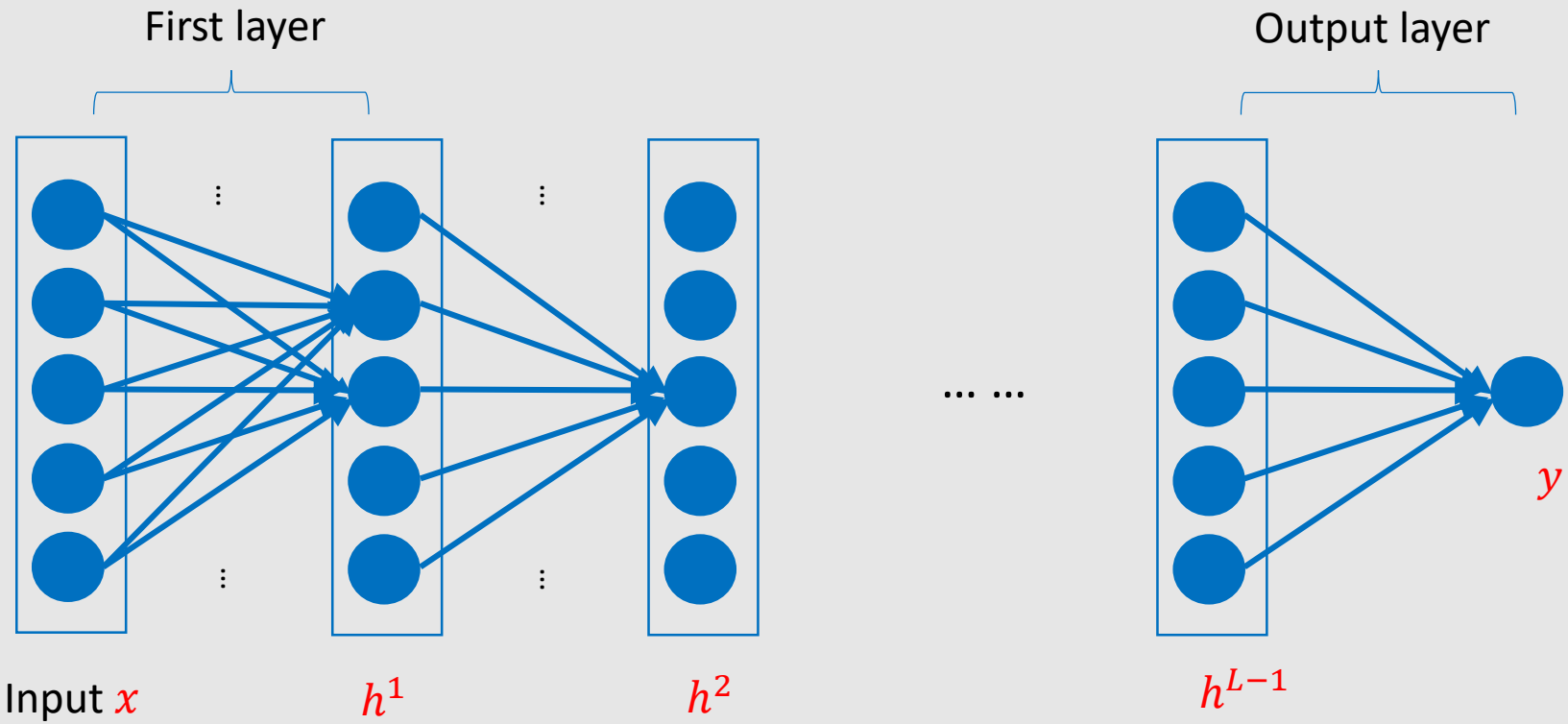
output y

Indoor

outdoor

Key Engine Behind the Success

- Deep Neural Networks: $y = f(x)$

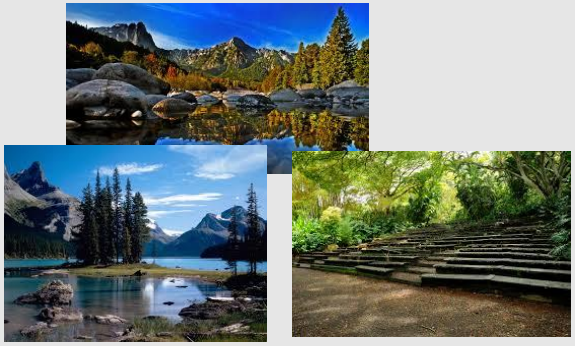


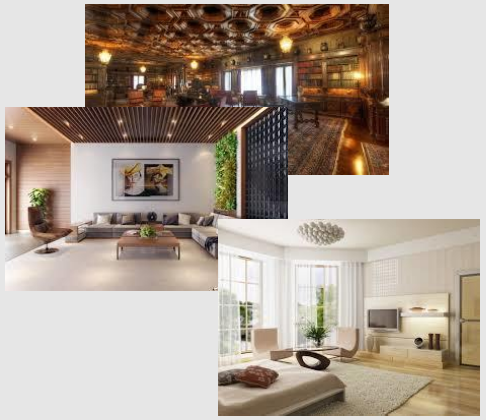
$h^i = \sigma(W_i h^{i-1})$, with activation $\sigma(z) = \max(0, z)$

Key Engine Behind the Success

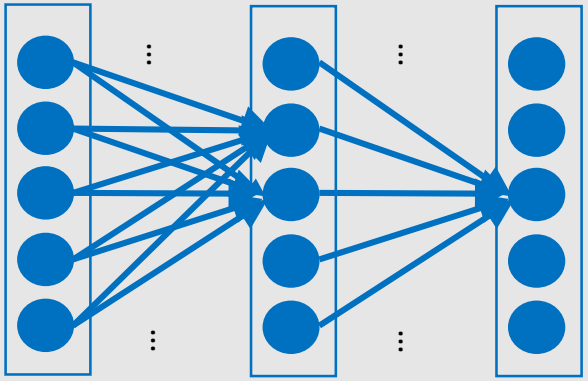
- Training Deep Neural Networks: $y = f(x; W)$
  - Given training data $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$
  - Try to find W such that the network fits the data

outdoor

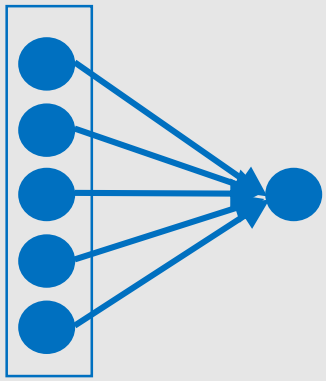




indoor



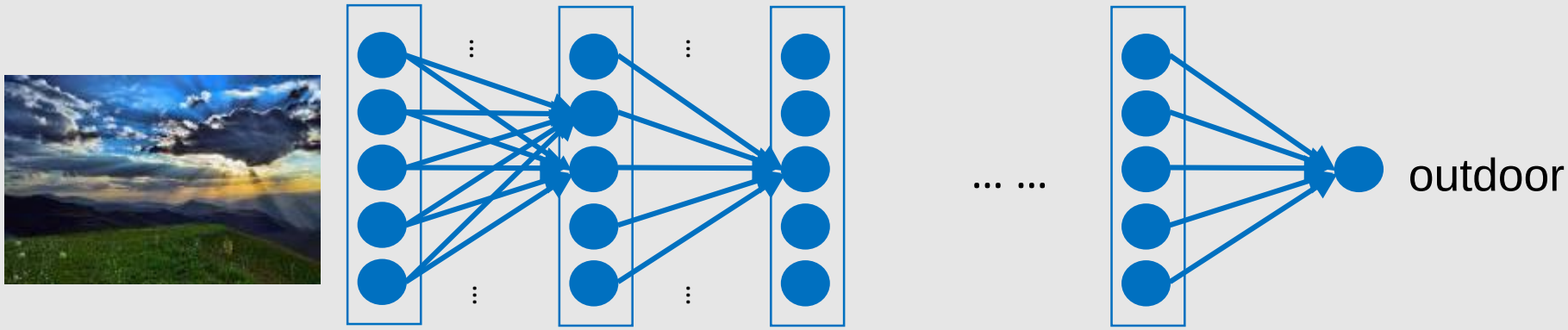
... ...



outdoor

Key Engine Behind the Success

- Using Deep Neural Networks: $y = f(x; W)$
  - Given a new test point x
  - Predict $y = f(x; W)$



Fundamental Questions

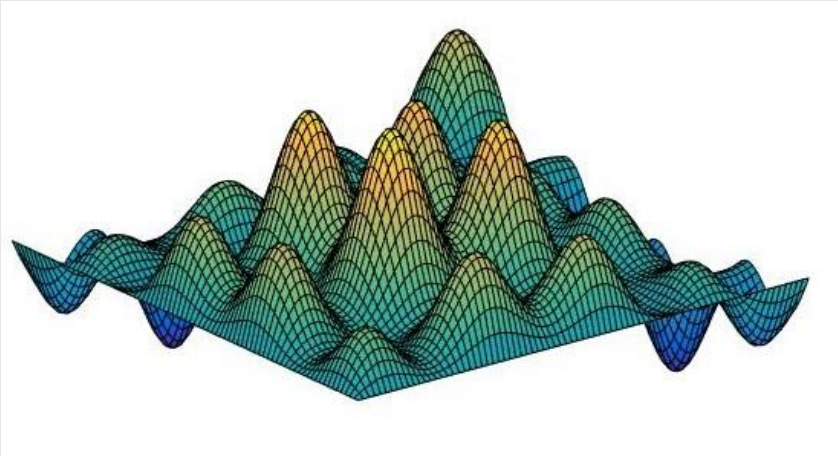
- Optimization:

  How to find W with good accuracy on training data?

- Generalization:

  Is the network also accurate on new test instances?

- Key challenge: the optimization is non-convex



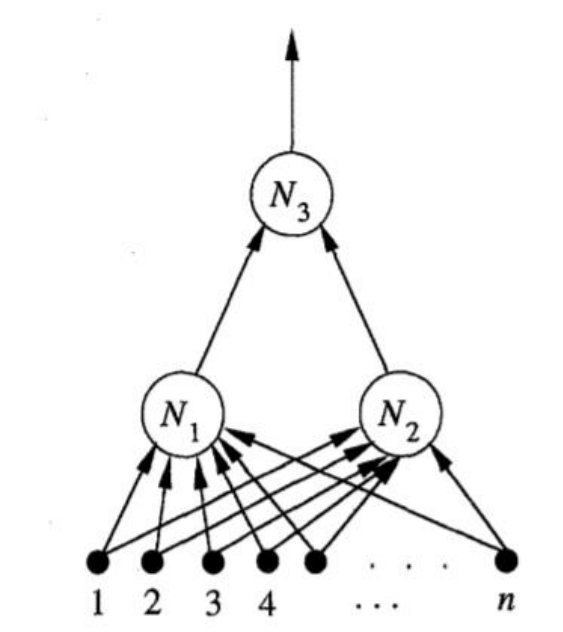
Open the Blackbox: Optimization

- Optimization has formed and shaped deep learning
  - Back-propagation
  - Momentum, Dropout, Batch-normalization, etc.

- Optimization lies in the center of many mysteries
  - Empirical success v.s. theoretical hardness
  - Overparameterized networks still good, contrast to classical theory

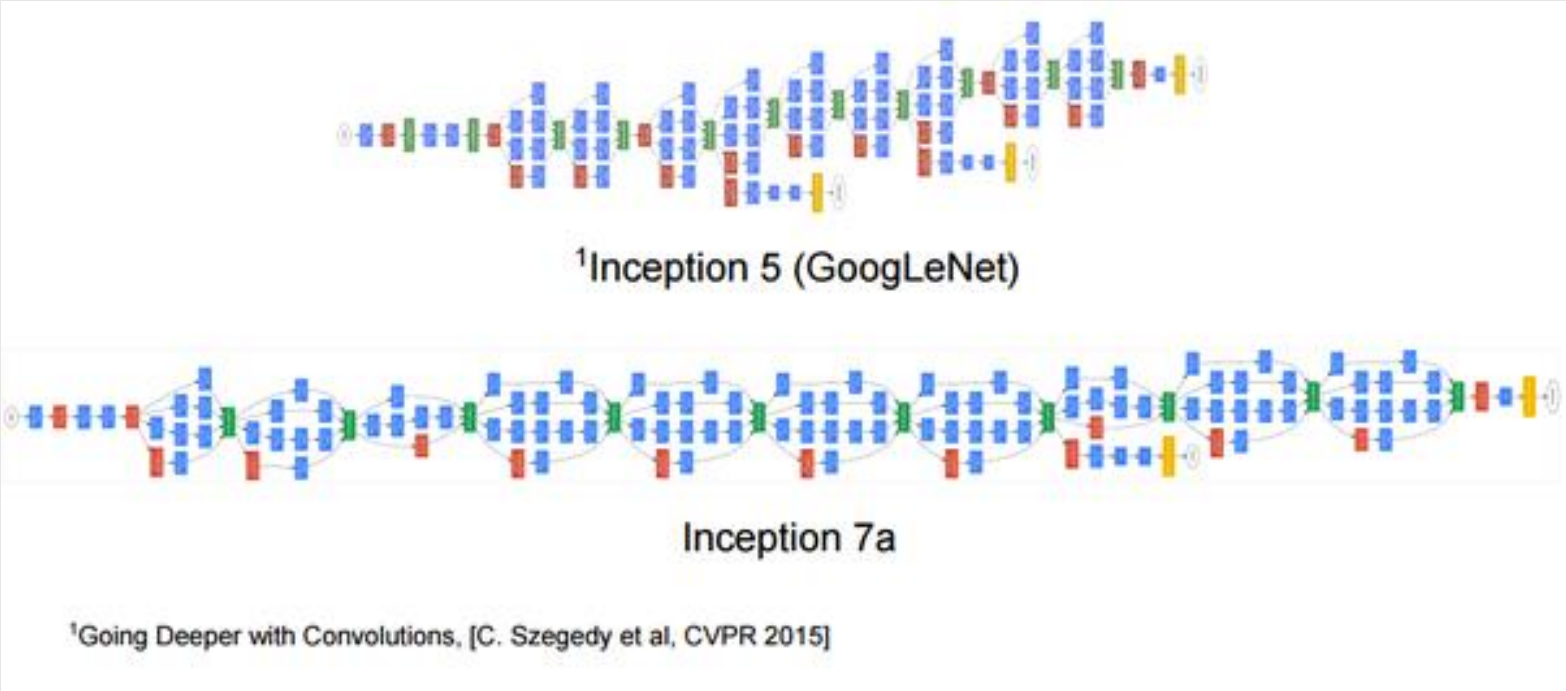
Empirical Success v.s. Theoretical Hardness

- Theoretically hard
  - Training a 3-Node Neural Network is NP-Complete [Blum & Rivest, 93]



Empirical Success v.s. Theoretical Hardness

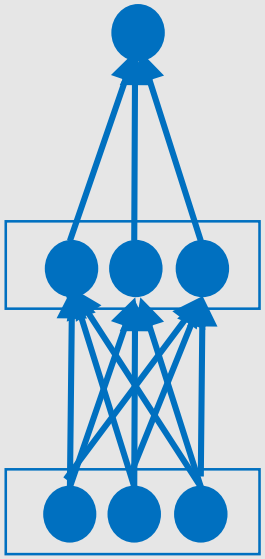
- Practically quite feasible
  - Simple algorithms like SGD often find good solutions
  - Practical networks are often very large and deep: hundreds of layers, thousands of nodes per layer



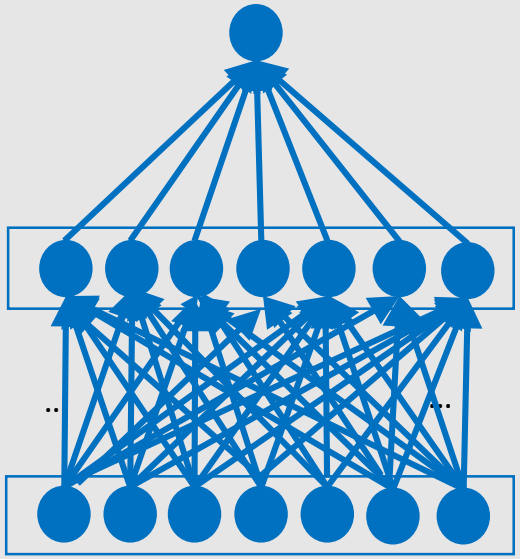
Over-Parameterization Helps Optimization

- Empirical observation: easier to train larger networks

- First generate synthetic data from a ground-truth network,
- Then train a larger network on the data



Ground truth



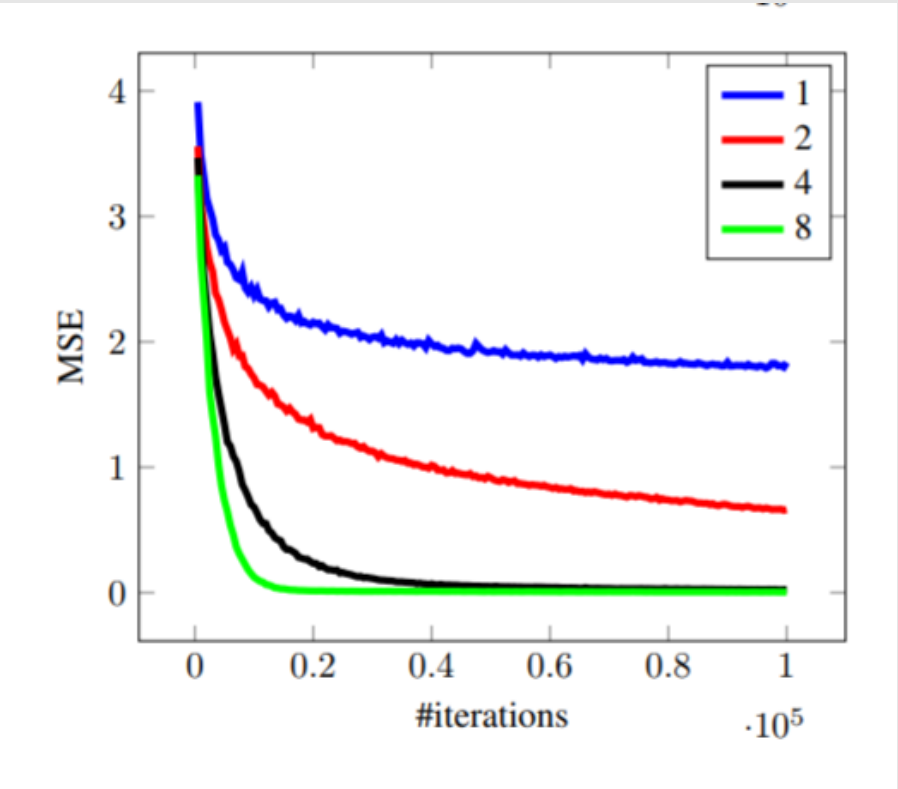
Train a larger network

On the Computational Efficiency of Training Neural Networks. Roi Livni, Shai Shalev-Shwartz, Ohad Shamir. NIPS 2014.

Over-Parameterization Helps Optimization

- Empirical observation: easier to train larger networks

- Faster convergence with larger networks

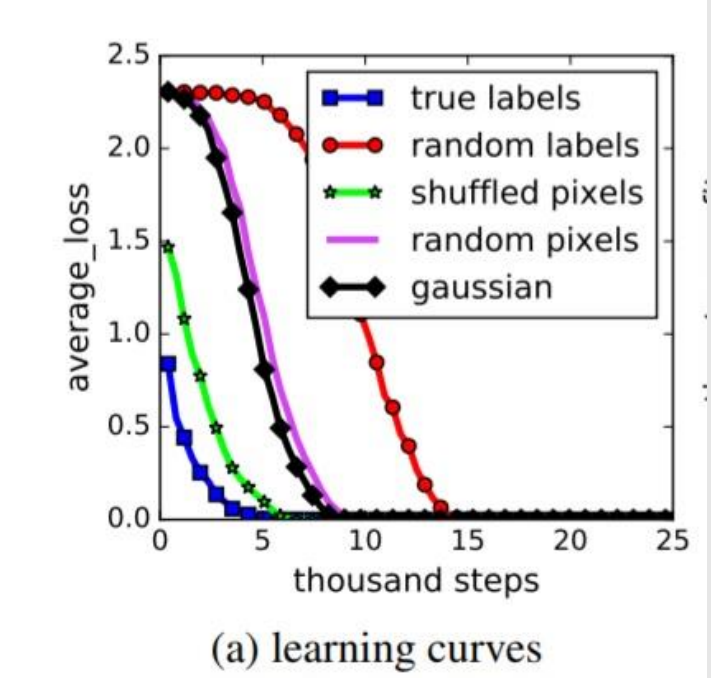


On the Computational Efficiency of Training Neural Networks. Roi Livni, Shai Shalev-Shwartz, Ohad Shamir. NIPS 2014.

DNNs Easily Fit Random Labels

- Empirical observation: practical DNNs easily fit random labels

- First replace the training labels with random labels
- Then train with net architectures and methods used in practice



Understanding deep learning requires rethinking generalization. Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, Oriol Vinyals. ICLR 2017.

DNNs Easily Fit Random Labels

- Empirical observation: practical DNNs easily fit random labels

Surprising implications:

1. Practical DNNs are overparameterized
   - Sufficient to fit random labels → sufficient to fit labels with structure

DNNs Easily Fit Random Labels

- Empirical observation: practical DNNs easily fit random labels

Surprising implications:

1. Practical DNNs are overparameterized
2. Even optimization on random labels remains easy
   - Simple methods (variants of SGD) can converge to 0 (global optima)

DNNs Easily Fit Random Labels

- Empirical observation: practical DNNs easily fit random labels

Surprising implications:

1. Practical DNNs are overparameterized
2. Even optimization on random labels remains easy
3. Optimization automatically adapts to the structure of the data
   - With random labels, it fits the training labels by memorization (no generalization)
   - With practical labels with structure, it learns the underlying structure without memorization (good generalization)

DNNs Easily Fit Random Labels

- Empirical observation: practical DNNs easily fit random labels

Surprising implications:

1. Practical DNNs are overparameterized
2. Even optimization on random labels remains easy
3. Optimization automatically adapts to the structure of the data

- Appear to contradict traditional theory!

DNNs Easily Fit Random Labels

- Empirical observation: practical DNNs easily fit random labels

Surprising implications:

1. Practical DNNs are overparameterized
2. Even optimization on random labels remains easy
3. Optimization automatically adapts to the structure of the data

- Appear to contradict traditional theory!

Is there a simple theoretical explanation?

Our work: Yes for two-layer NN on clustered data!

DNNs Easily Fit Random Labels

- Empirical observation: practical DNNs easily fit random labels

Surprising implications:

1. Practical DNNs are overparameterized
2. Even optimization on random labels remains easy
3. Optimization automatically adapts to the structure of the data

- Appear to contradict traditional theory!

Is there a simple theoretical explanation?

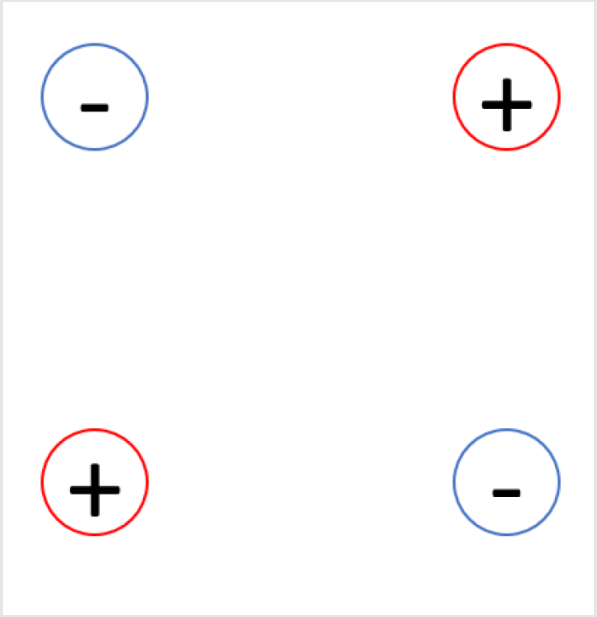
Our subseq. work: Yes for deeper NN, general data

A Simple Setting

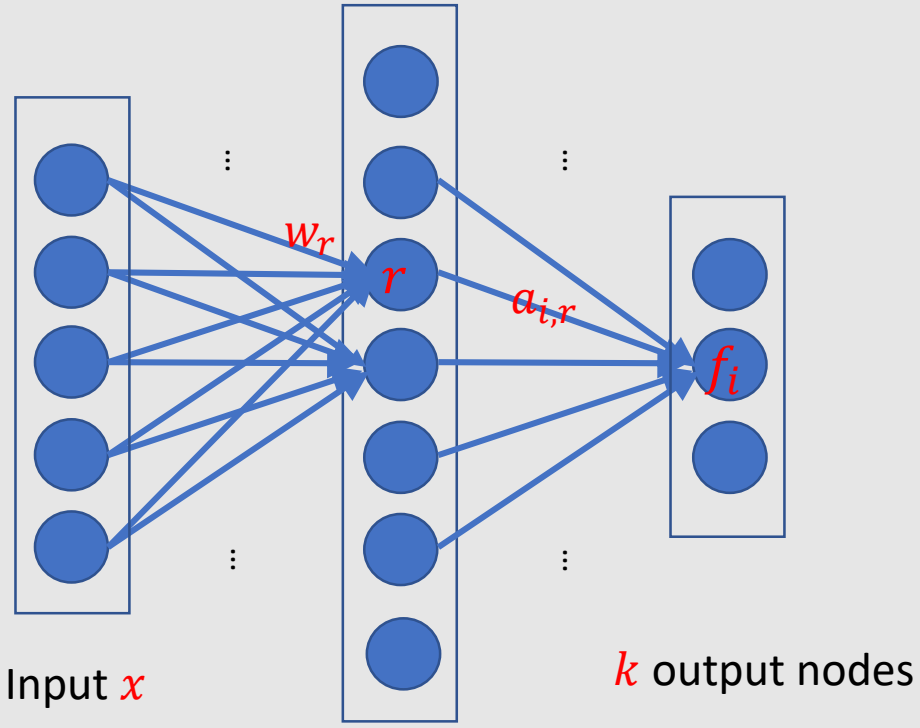
- Two layer networks
- Simple data, yet with rich structure (inspired by MNIST data)

Assumption About the Data

- k classes, each with l components.
- Min distance between two components of different classes $\geq \delta$,
  Diameter of each component $< \delta/8l$



Assumption About the Learning Method



Learned by 2-layer NN:

$$f_i(x) = \sum_{r=1}^{m} a_{i,r} \mathbf{ReLU}(\langle w_r, x \rangle)$$

$$\mathbf{ReLU}(z) = \max\{0, z\}$$

Random initialization:

$$w_r^{(0)} \sim \mathcal{N}(0, \sigma^2 \mathbf{I}),\ a_{i,r} \sim \mathcal{N}(0,1), \text{ with } \sigma = \frac{1}{m^{1/2}}$$

Training: SGD on cross-entropy loss

Our Results [NeurIPS'18]

- Theorem (informal): For every $\epsilon > 0$ there exists M such that: When #hidden neurons $> M$, SGD learns a network with at most ϵ generalization error.

Our Results [NeurIPS'18]

- Theorem (informal): For every $\epsilon > 0$ there exists M such that: When #hidden neurons $> M$, SGD learns a network with at most ϵ generalization error.

Remarks:

1. Optimization indeed easy for overpara. networks
   - In fact, the analysis shows that it's close to convex

Our Results [NeurIPS'18]

- Theorem (informal): For every $\epsilon > 0$ there exists M such that: When #hidden neurons $> M$, SGD learns a network with at most ϵ generalization error.

Remarks:

1. Optimization indeed easy for overpara. networks
2. One can provably train a good overparameterized network
   - Not just easy optimization, but good generalization

Our Results [NeurIPS'18]

- Theorem (informal): For every $\epsilon > 0$ there exists M such that: When #hidden neurons $> M$, SGD learns a network with at most ϵ generalization error.
- $M = polynomial(k, l, 1/\delta, 1/\epsilon)$
- #training data $polynomial(k, l, 1/\delta, 1/\epsilon, \log \#\text{hidden neurons})$

Our Results [NeurIPS'18]

- Theorem (informal): For every $\epsilon > 0$ there exists M such that: When #hidden neurons $> M$, SGD learns a network with at most ϵ generalization error.
- $M = polynomial(k, l, 1/\delta, 1/\epsilon)$
- #training data $polynomial(k, l, 1/\delta, 1/\epsilon, \log \#\text{hidden neurons})$

Remarks:

3. Dimension free! Only depend on the structure of the data.

   #training data only depend on the logarithm of #hidden neurons

Our Results [NeurIPS'18]

- Theorem (informal): For every $\epsilon > 0$ there exists M such that: When #hidden neurons $> M$, SGD learns a network with at most ϵ generalization error.

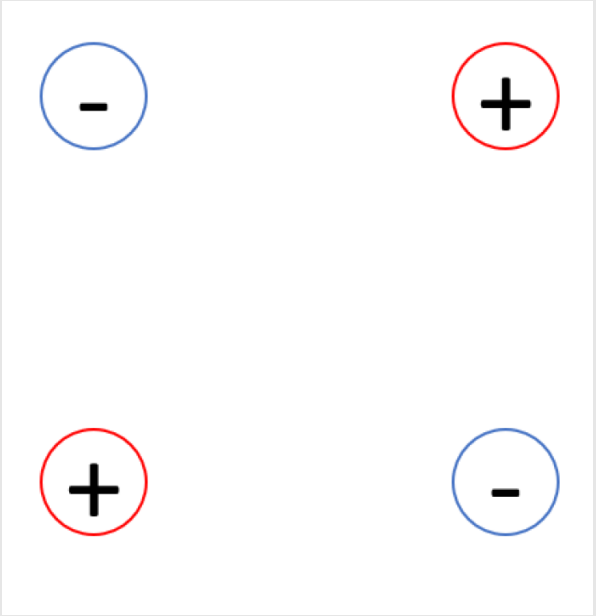
What about random labels?

Yes, we can also explain that!

Assumption About the Data

Recall

- k classes, each with l components. Points have norm 1.
- Min distance between two components of different classes $> \delta$,
  Diameter of each component $\leq \delta/8l$



Holds for random labeled data: each point is a component, δ=min distance between points

Our Results [NeurIPS'18]

- Theorem (informal): For every $\epsilon > 0$ there exists M such that: When #hidden neurons $> M$, SGD learns a network with at most ϵ generalization error.

Remarks:

4. On any training data: if $M = polynomial(\#\text{points}, 1/\delta, 1/\epsilon)$, then can always get at most ϵ training error

Yes, can fit random labels with sufficient overpara!

Our Results [NeurIPS'18]

- Theorem (informal): For every $\epsilon > 0$ there exists M such that: When #hidden neurons $> M$, SGD learns a network with at most ϵ generalization error.

Remarks:

5. If $M = polynomial(\#\text{points}, 1/\delta, 1/\epsilon)$,
   - Can fit random labels (without generalization)
   - But will first find a generalizable network when data has structures

Optimization automatically adapts to the structure of the data!

Our Results [NeurIPS'18]

- Theorem (informal): For every $\epsilon > 0$ there exists M such that: When #hidden neurons $> M$, SGD learns a network with at most ϵ generalization error.

- The intuitions (combined with new techniques) lead to several subsequent works:
  - Convergence result (i.e. 0 training error) for deep neural networks and RNN [ALS'18a, ALS'18b, DLLWZ'18, ZCZG'18]
  - Learning guarantees (both training and generalization) of two/three layer networks, assuming data from a grounth-truth network [ALL'18]

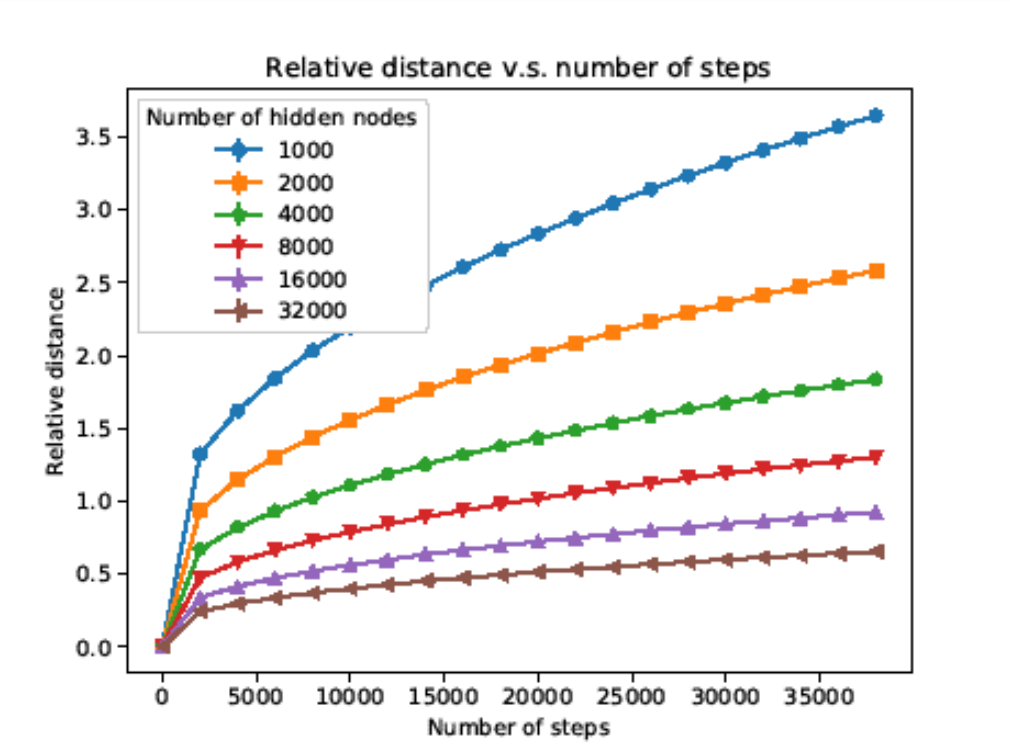
Intuition

- In a neighborhood around the random initialization, the optimization landscape is nice
  - Error is large → Gradient is large
- More overpara., larger such a neighborhood (relatively)
- Within it, SGD will find a solution
  - Closeness → good generalization

Not contradictory to traditional theory!

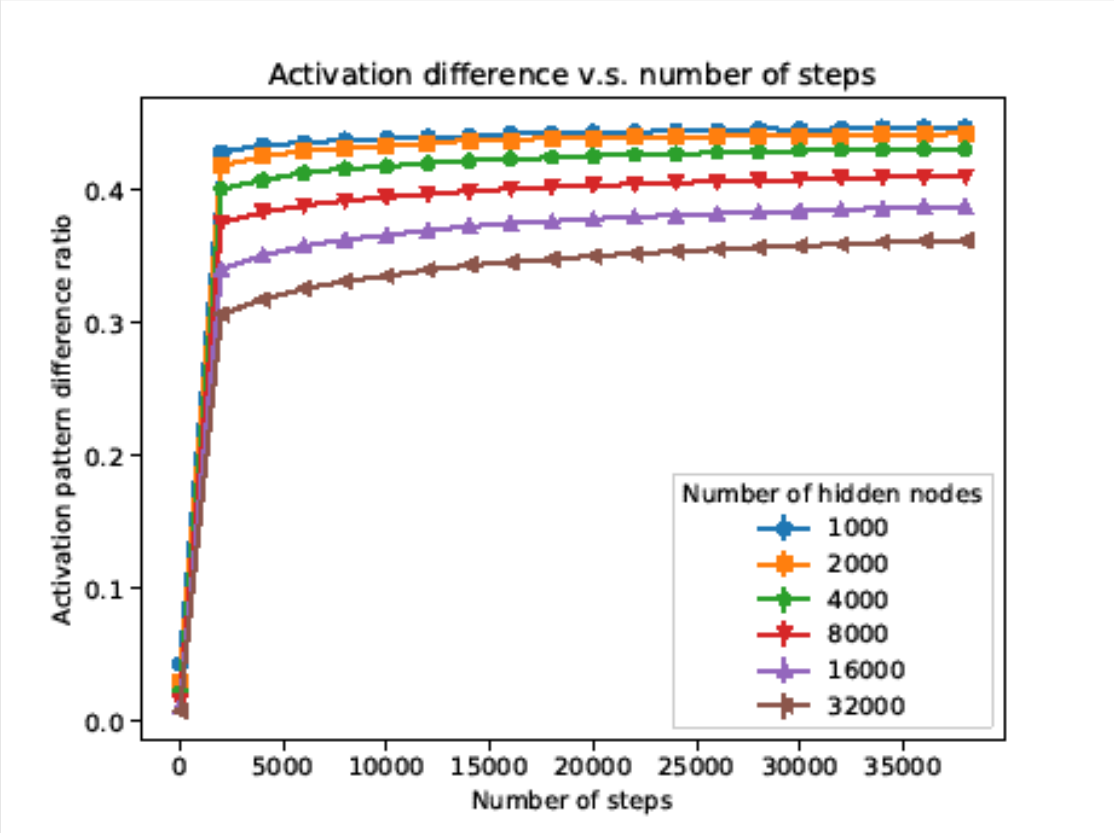
Surprising property: good networks are almost everywhere!

Empirical verification



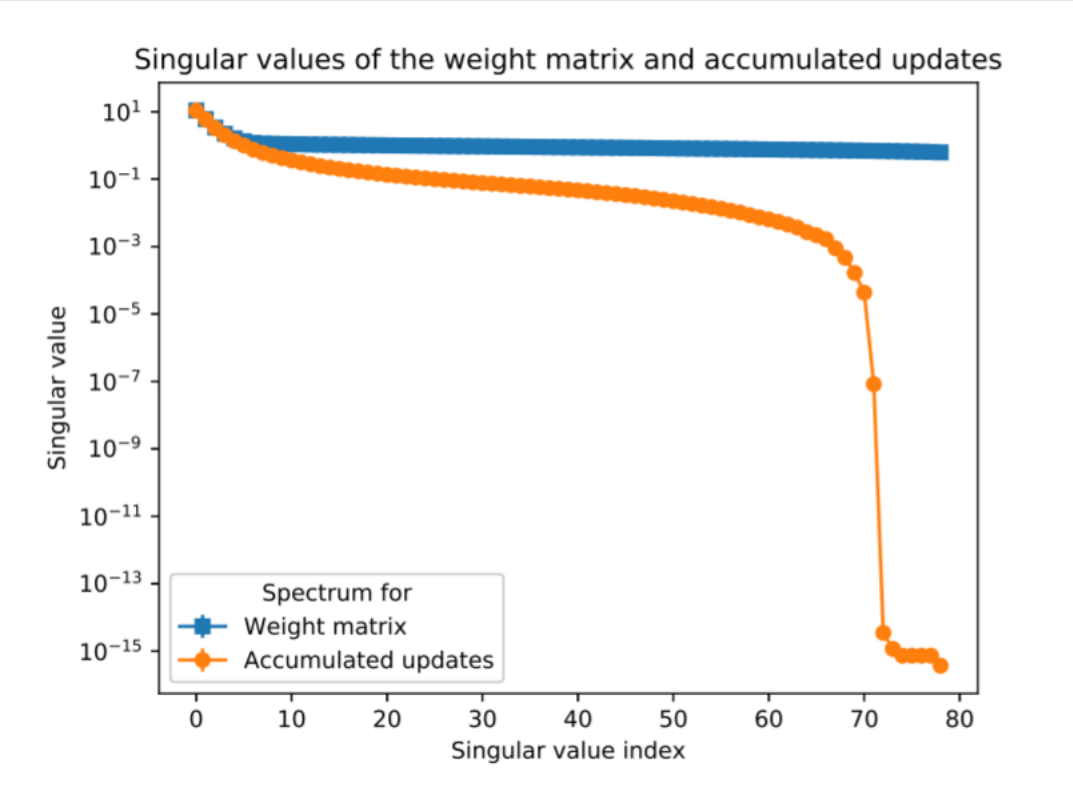
MNIST data: relative distance to the initialization is small

Empirical verification



MNIST data: majority of activation patterns remain the same

Empirical verification



MNIST data: Spectrum of the weight matrix W and $W - W^{(0)}$


THANK YOU!