The Mathematics of Deep Learning
Lecture 3: Analyzing the Training Algorithm

Gitta Kutyniok

(Ludwig-Maximilians-Universität München)

LMS Invited Lecture Series

University of Cambridge, February 28 – March 4, 2022

Brief Recap:

Statistical Learning Theory

A Bit More Formal....

Still Informal Definition:

Let $\mathcal{X}, \mathcal{Y}$, and $\mathcal{Z}$ be measurable spaces. In a learning task, one is given

- *data* in $\mathcal{Z}$ and
- a *loss function* $\mathcal{L}\colon \mathcal{M}(\mathcal{X}, \mathcal{Y}) \times \mathcal{Z} \to \mathbb{R}$.

The goal is to choose a *hypothesis set* $\mathcal{F} \subset \mathcal{M}(\mathcal{X}, \mathcal{Y})$ and construct a *learning algorithm*, i.e., a mapping

$$\mathcal{A}\colon \bigcup_{m \in \mathbb{N}} \mathcal{Z}^m \to \mathcal{F},$$

which uses training data $s = (z^{(i)})_{i=1}^m \in \mathcal{Z}^m$ to find a model $f_s = \mathcal{A}(s) \in \mathcal{F}$ that

1. performs well on the training data s and
2. generalizes to unseen data $z \in \mathcal{Z}$.

Here, *performance* is measured via the loss function $\mathcal{L}$ and the corresponding loss $\mathcal{L}(f_s, z)$.

Our Focus: Prediction Tasks

Definition:

In a *prediction task*, we have that $\mathcal{Z} := \mathcal{X} \times \mathcal{Y}$, i.e., we are given training data $s = ((x^{(i)}, y^{(i)}))_{i=1}^{m}$ that consist of input features $x^{(i)} \in \mathcal{X}$ and corresponding labels $y^{(i)} \in \mathcal{Y}$.

For *one-dimensional regression tasks with* $\mathcal{Y} \subset \mathbb{R}$, we consider the quadratic loss

$$\mathcal{L}(f, (x, y)) = (f(x) - y)^2$$

and, for *binary classification tasks with* $\mathcal{Y} = \{-1, 1\}$, we consider the 0-1 loss

$$\mathcal{L}(f, (x, y)) = \mathbb{1}_{(-\infty, 0)}(yf(x)).$$

We assume that our input features are in Euclidean space, i.e., $\mathcal{X} \subset \mathbb{R}^d$ with input dimension $d \in \mathbb{N}$.

Our Hypothesis Class

Hypothesis Sets of Neural Networks:

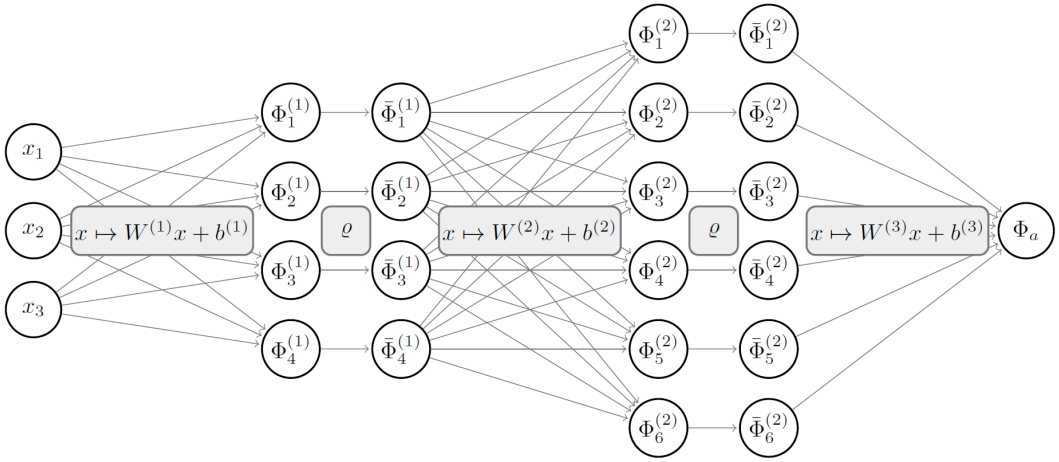
Let $a = (N, \varrho)$ be a neural network architecture with input dimension $N_0 = d$, output dimension $N_L = 1$, and measurable activation function ϱ.

For *regression tasks* the corresponding hypothesis set is given by

$$\mathcal{F}_a = \left\{\Phi_a(\cdot, \theta) \colon \theta \in \mathbb{R}^{P(N)}\right\}$$

and for *classification tasks* by

$$\mathcal{F}_{a,\mathrm{sgn}} = \left\{\mathrm{sgn}(\Phi_a(\cdot, \theta)) \colon \theta \in \mathbb{R}^{P(N)}\right\}, \quad \text{where} \quad \mathrm{sgn}(x) := \begin{cases} 1, & \text{if } x \geq 0, \\ -1, & \text{if } x < 0. \end{cases}$$



LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Learning Algorithm: Empirical Risk Minimization

Definition (Empirical Risk):
For training data $s = (z^{(i)})_{i=1}^{m} \in \mathcal{Z}^m$ and a function $f \in \mathcal{M}(\mathcal{X}, \mathcal{Y})$, we define the *empirical risk* by

$$\widehat{\mathcal{R}}_s(f) := \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(f, z^{(i)}).$$

...measures the average loss on the given training data.

Definition (ERM Learning Algorithm):
Given a hypothesis set $\mathcal{F}$, an *empirical risk minimization algorithm* $\mathcal{A}^{\mathrm{erm}}$ chooses for training data $s \in \mathcal{Z}^m$ a minimizer $\widehat{f}_s \in \mathcal{F}$ of the empirical risk in $\mathcal{F}$, i.e.,

$$\mathcal{A}^{\mathrm{erm}}(s) \in \operatorname*{argmin}_{f \in \mathcal{F}} \widehat{\mathcal{R}}_s(f).$$

Average Out-Of-Sample Performance of a Model

Assumption (Independent and Identically Distributed data):
We assume that $z^{(1)}, \ldots, z^{(m)}, z$ are realizations of i.i.d. random variables $Z^{(1)}, \ldots, Z^{(m)}, Z$.

Definition:
For a function $f \in \mathcal{M}(\mathcal{X}, \mathcal{Y})$, we define the *risk* by

$$\mathcal{R}(f) := \mathbb{E}\big[\mathcal{L}(f, Z)\big] = \int_{\mathcal{Z}} \mathcal{L}(f, z) \, \mathrm{d}\mathbb{P}_Z(z).$$

Defining $S := (Z^{(i)})_{i=1}^m$, the *risk of a model* $f_S = \mathcal{A}(S)$ is thus given by

$$\mathcal{R}(f_S) = \mathbb{E}\big[\mathcal{L}(f_S, Z) | S\big].$$

Regression and Classification Risk

Definition:

A function $f^* \in \mathcal{M}(\mathcal{X}, \mathcal{Y})$ achieving the smallest risk, the *Bayes risk*

$$\mathcal{R}^* := \inf_{f \in \mathcal{M}(\mathcal{X}, \mathcal{Y})} \mathcal{R}(f),$$

is called a *Bayes-optimal function*.

Lemma:

(1) For a *regression task* with $\mathbb{V}[Y] < \infty$, the risk can be decomposed into

$$\mathcal{R}(f) = \mathbb{E}\left[(f(X) - \mathbb{E}[Y|X])^2\right] + \mathcal{R}^*, \quad f \in \mathcal{M}(\mathcal{X}, \mathcal{Y}),$$

which is minimized by the *regression function* $f^*(x) = \mathbb{E}[Y|X = x]$.

(2) For a *classification task*, the risk can be decomposed into

$$\mathcal{R}(f) = \mathbb{E}\left[|\mathbb{E}[Y|X]| \mathbb{1}_{(-\infty, 0)}(\mathbb{E}[Y|X] f(X))\right] + \mathcal{R}^*, \quad f \in \mathcal{M}(\mathcal{X}, \mathcal{Y}),$$

which is minimized by the *Bayes classifier* $f^*(x) = \text{sgn}(\mathbb{E}[Y|X = x])$.

Error Decomposition

Let $f_{\mathcal{F}}^* \in \operatorname{argmin}_{f \in \mathcal{F}} \mathcal{R}(f)$ be a best approximation in $\mathcal{F}$, such that we can bound the error

$$\begin{aligned} & \mathcal{R}(f_S) - \mathcal{R}^* \\ = \;& \mathcal{R}(f_S) - \widehat{\mathcal{R}}_S(f_S) + \widehat{\mathcal{R}}_S(f_S) - \widehat{\mathcal{R}}_S(f_{\mathcal{F}}^*) + \widehat{\mathcal{R}}_S(f_{\mathcal{F}}^*) - \mathcal{R}(f_{\mathcal{F}}^*) + \mathcal{R}(f_{\mathcal{F}}^*) - \mathcal{R}^* \\ \leq \;& \varepsilon^{\text{opt}} + 2\varepsilon^{\text{gen}} + \varepsilon^{\text{approx}} \end{aligned}$$

by

- an *optimization error*

$$\varepsilon^{\text{opt}} := \widehat{\mathcal{R}}_S(f_S) - \widehat{\mathcal{R}}_S(\widehat{f}_S) \geq \widehat{\mathcal{R}}_S(f_S) - \widehat{\mathcal{R}}_S(f_{\mathcal{F}}^*),$$

- a (uniform) *generalization error*

$$\varepsilon^{\text{gen}} := \sup_{f \in \mathcal{F}} |\mathcal{R}(f) - \widehat{\mathcal{R}}_S(f)| \geq \max\{\mathcal{R}(f_S) - \widehat{\mathcal{R}}_S(f_S), \widehat{\mathcal{R}}_S(f_{\mathcal{F}}^*) - \mathcal{R}(f_{\mathcal{F}}^*)\},$$

- an *approximation error*

$$\varepsilon^{\text{approx}} := \mathcal{R}(f_{\mathcal{F}}^*) - \mathcal{R}^*.$$

The Optimization Error

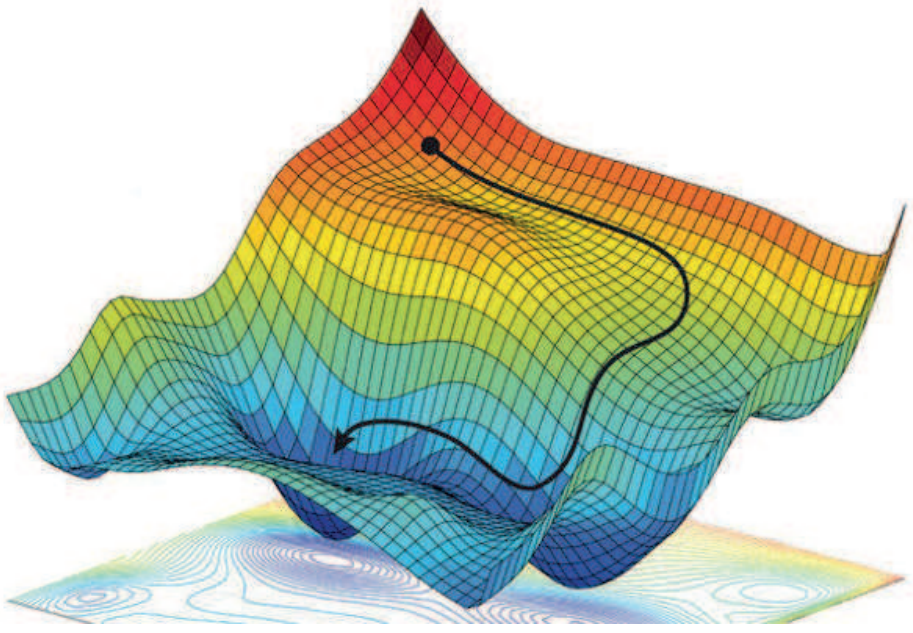
Remark:

- This error is primarily influenced by the numerical algorithm $\mathcal{A}$.
- We will focus on the setting where such an algorithm aims to approximately minimize the *empirical risk*

$$\widehat{\mathcal{R}}_s(f) := \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(f, z^{(i)}).$$

The most common are gradient-based methods!

Gradient Descent



The Optimization Error

Remark:

- This error is primarily influenced by the numerical algorithm $\mathcal{A}$.
- We will focus on the setting where such an algorithm aims to approximately minimize the *empirical risk*

$$\widehat{\mathcal{R}}_s(f) := \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(f, z^{(i)}).$$

The most common are gradient-based methods!

The Optimization Error

Remark:

- This error is primarily influenced by the numerical algorithm $\mathcal{A}$.
- We will focus on the setting where such an algorithm aims to approximately minimize the *empirical risk*

$$\widehat{\mathcal{R}}_s(f) := \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(f, z^{(i)}).$$

The most common are gradient-based methods!

Key Questions:

- How *fast* does the algorithm converges?
- Does it converge to a *"good" local minimum*?
- What are good *starting values*?

Stochastic Gradient Descent

Stochastic Gradient Descent

Core Algorithm (Robbins, Monro; 1951):

Input : Differentiable function $r\colon \mathbb{R}^p \to \mathbb{R}$

Sequence of step-sizes $\eta_k \in (0, \infty)$, $k \in [K]$

$\mathbb{R}^p$-valued random variable $\Theta^{(0)}$

Output: Sequence of $\mathbb{R}^p$-valued random variables $(\Theta^{(k)})_{k=1}^K$

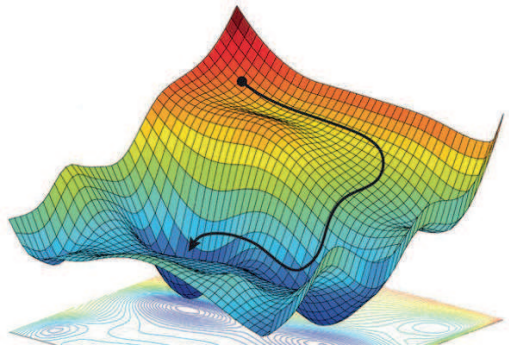
for $k = 1, \ldots, K$ **do**

Let $D^{(k)}$ be a random variable such that $\mathbb{E}[D^{(k)}|\Theta^{(k-1)}] = \nabla r(\Theta^{(k-1)})$;

Set $\Theta^{(k)} := \Theta^{(k-1)} - \eta_k D^{(k)}$;

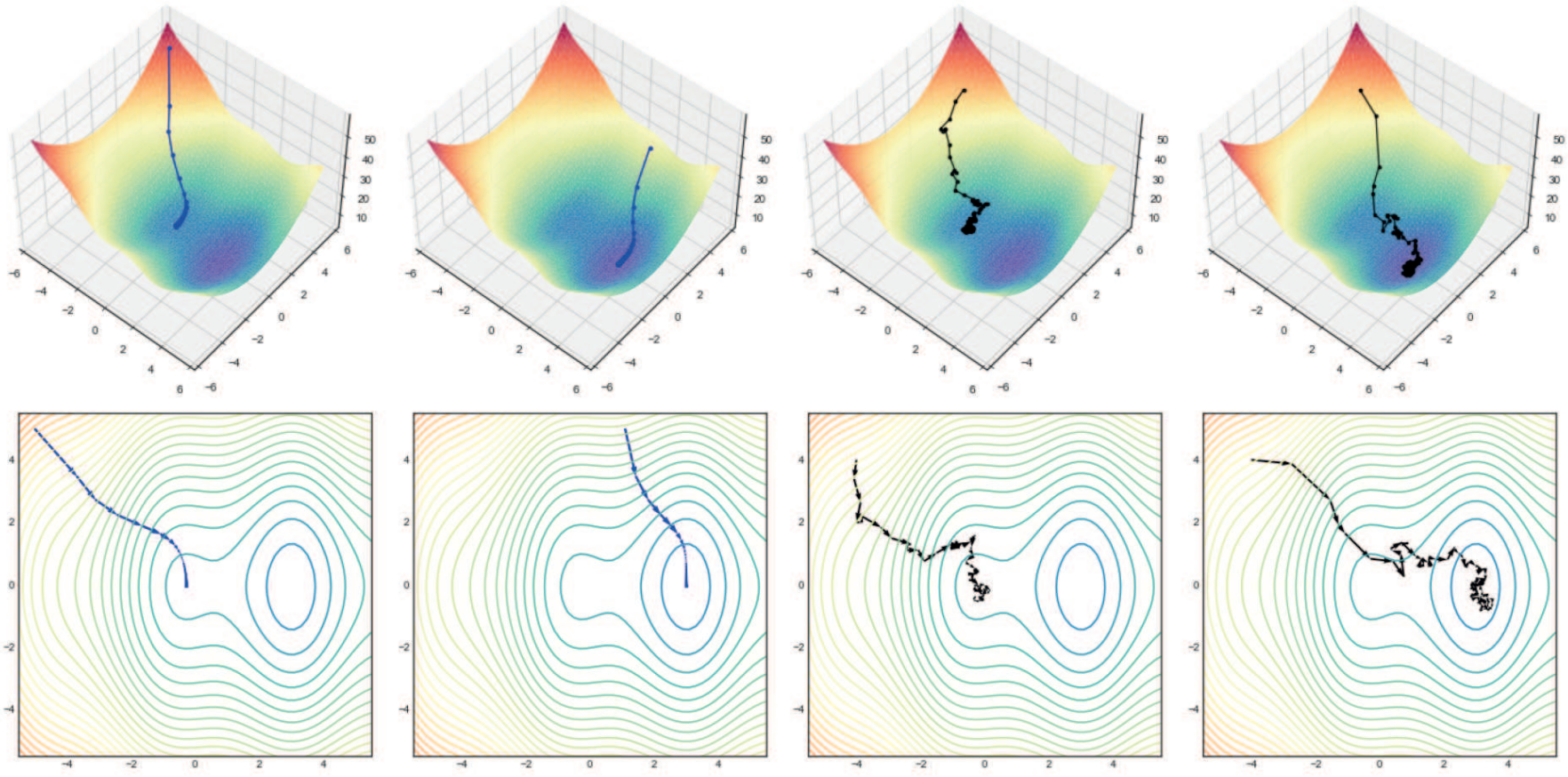
Gradient Descent:

- Choose $D^{(k)}$ is deterministically.



LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Gradient Descent versus Stochastic Gradient Descent



Source: Berner, Grohs, K, Petersen. The Modern Mathematics of Deep Learning. In: Mathematical Aspects of Deep Learning, Cambridge, 2022.

Stochastic Gradient Descent (Continued)

Minimizing the Empirical Loss:

- Choose $r\colon \mathbb{R}^{P(N)} \to \mathbb{R}$ as

$$r(\theta) = \widehat{\mathcal{R}}_s(\Phi_a(\cdot, \theta)).$$

- Choose a *batch-size* $m' \in \mathbb{N}$ with $m' \leq m$ and consider

$$\Theta^{(k)} := \Theta^{(k-1)} - \frac{\eta_k}{m'} \sum_{z \in S'} \nabla_\theta \mathcal{L}(\Phi_a(\cdot, \Theta^{(k-1)}), z).$$

- S' is a *mini-batch* of size $|S'| = m'$ chosen uniformly at random from the training data s.
- $(\eta_k)_{k \in \mathbb{N}}$ is called *learning rate*.

Stochastic Gradient Descent (Continued)

Minimizing the Empirical Loss:

- Choose $r\colon \mathbb{R}^{P(N)} \to \mathbb{R}$ as

$$r(\theta) = \widehat{\mathcal{R}}_s(\Phi_a(\cdot, \theta)).$$

- Choose a *batch-size* $m' \in \mathbb{N}$ with $m' \leq m$ and consider

$$\Theta^{(k)} := \Theta^{(k-1)} - \frac{\eta_k}{m'} \sum_{z \in S'} \nabla_\theta \mathcal{L}(\Phi_a(\cdot, \Theta^{(k-1)}), z).$$

- S' is a *mini-batch* of size $|S'| = m'$ chosen uniformly at random from the training data s.
- $(\eta_k)_{k \in \mathbb{N}}$ is called *learning rate*.

Output of Algorithm:

After K steps, this leads to

$$f_s = \mathcal{A}(s) = \Phi_a(\cdot, \bar{\theta}),$$

where $\bar{\theta}$ can be chosen as the realization of $\Theta^{(K)}$.

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Convergence of SGD

Theorem (Nemirovski, Juditsky, Lan, Shapiro; 2009):
Let $p, K \in \mathbb{N}$ and let $r\colon \mathbb{R}^p \supset B_1(0) \to \mathbb{R}$ be differentiable and *convex*. Further let $(\Theta^{(k)})_{k=1}^K$ be the output of stochastic gradient descent with initialization $\Theta^{(0)} = 0$, step-sizes $\eta_k = K^{-1/2}$, $k \in [K]$, and random variables $(D^{(k)})_{k=1}^K$ satisfying that $\|D^{(k)}\|_2 \leq 1$ almost surely for all $k \in [K]$. Then

$$\mathbb{E}[r(\bar{\Theta})] - r(\theta^*) \leq \frac{1}{\sqrt{K}},$$

where $\bar{\Theta} := \frac{1}{K}\sum_{k=1}^K \Theta^{(k)}$ and $\theta^* \in \operatorname{argmin}_{\theta \in B_1(0)} r(\theta)$.

Convergence of SGD

Theorem (Nemirovski, Juditsky, Lan, Shapiro; 2009):
Let $p, K \in \mathbb{N}$ and let $r\colon \mathbb{R}^p \supset B_1(0) \to \mathbb{R}$ be differentiable and *convex*. Further let $(\Theta^{(k)})_{k=1}^K$ be the output of stochastic gradient descent with initialization $\Theta^{(0)} = 0$, step-sizes $\eta_k = K^{-1/2}$, $k \in [K]$, and random variables $(D^{(k)})_{k=1}^K$ satisfying that $\|D^{(k)}\|_2 \leq 1$ almost surely for all $k \in [K]$. Then

$$\mathbb{E}[r(\bar{\Theta})] - r(\theta^*) \leq \frac{1}{\sqrt{K}},$$

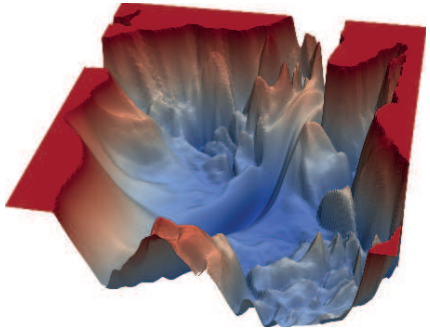
where $\bar{\Theta} := \frac{1}{K}\sum_{k=1}^K \Theta^{(k)}$ and $\theta^* \in \operatorname{argmin}_{\theta \in B_1(0)} r(\theta)$.

Remark: If r is *not convex*, then stochastic gradient descent may converge to a local, non-global minimum.

The Mystery

Observe:

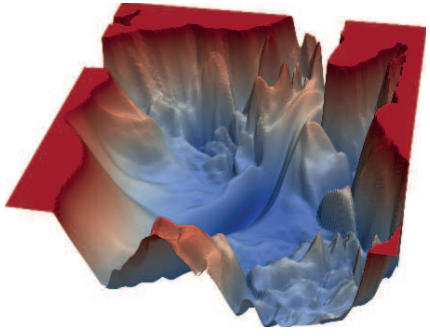
- *The empirical risk is severely nonconvex, may exhibit*
  - *(higher-order) saddle points,*
  - *seriously suboptimal local minima, and*
  - *wide flat areas where the gradient is very small.*
- *In applications, excellent performance of SGD is observed.*



The Mystery

Observe:

- *The empirical risk is severely nonconvex, may exhibit*
  - *(higher-order) saddle points,*
  - *seriously suboptimal local minima, and*
  - *wide flat areas where the gradient is very small.*
- *In applications, excellent performance of SGD is observed.*



True?

The trajectory of the optimization routine misses suboptimal critical points and other areas that may lead to slow convergence.

Analyzing the Loss Landscape

The Loss Landscape

Definition:

Let $\Phi(\cdot, \theta)$ be a neural network and let $s \in \mathcal{Z}^m$ be training data. Then the graph of the function $\theta \mapsto r(\theta) := \widehat{\mathcal{R}}_s(\Phi(\cdot, \theta))$ is called the *loss landscape*.

The Loss Landscape

Definition:
Let $\Phi(\cdot, \theta)$ be a neural network and let $s \in \mathcal{Z}^m$ be training data. Then the graph of the function $\theta \mapsto r(\theta) := \widehat{\mathcal{R}}_s(\Phi(\cdot, \theta))$ is called the *loss landscape*.

Idea: Analyze stochastic gradient descent through the shape of this high-dimensional surface.

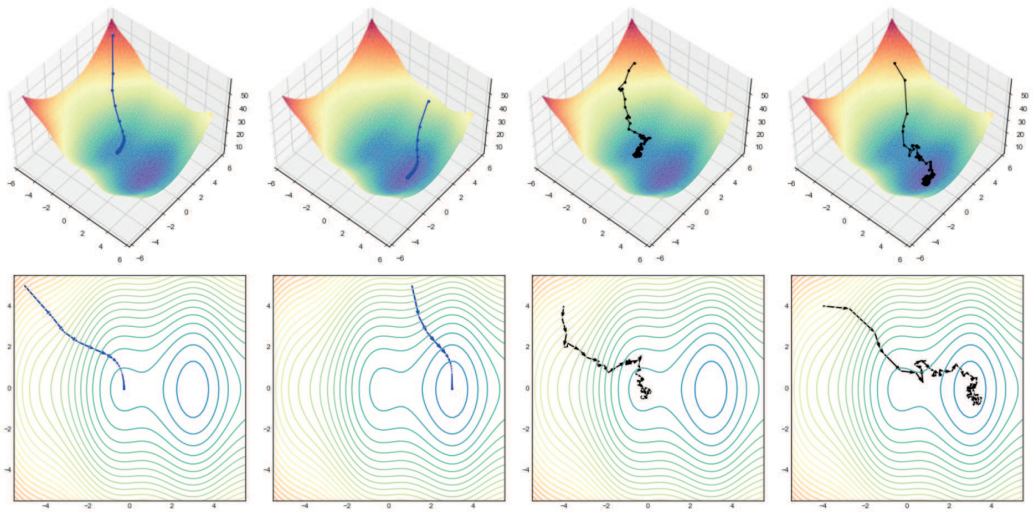
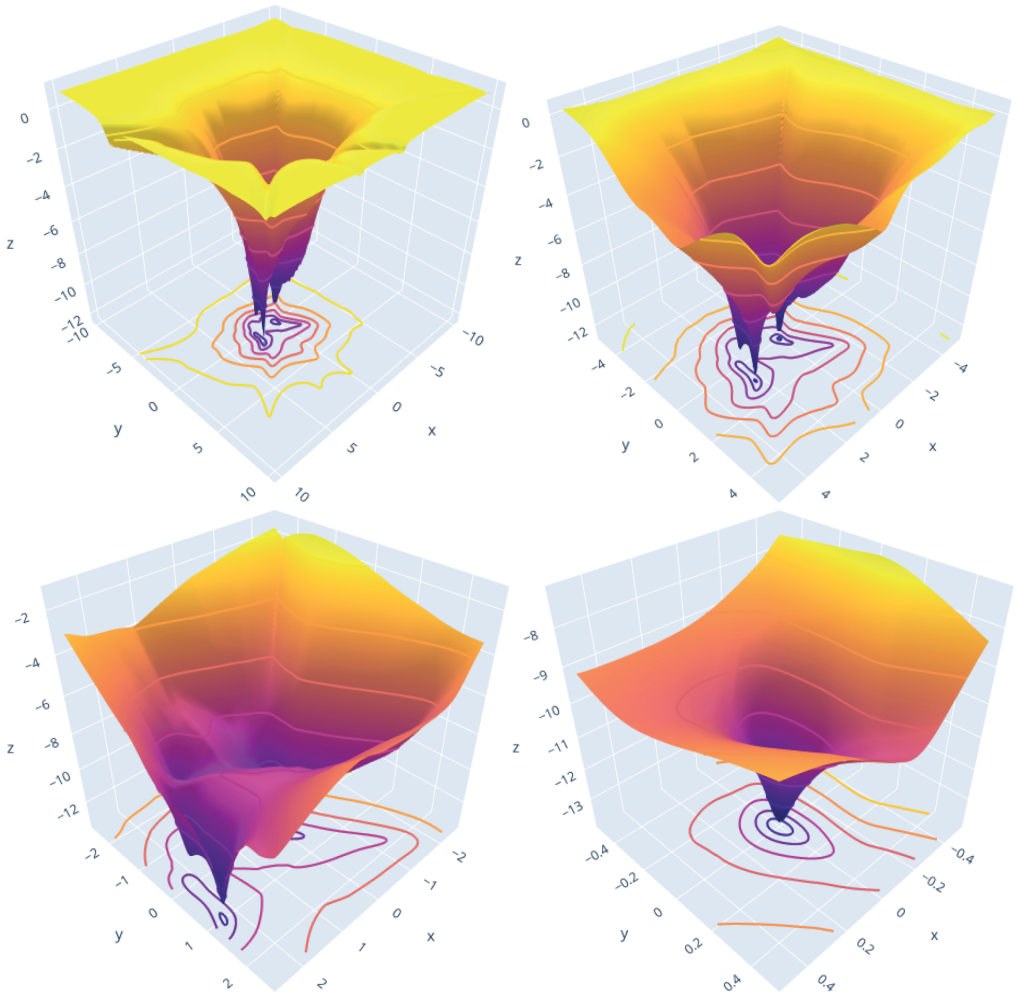


Illustration of the Loss Landscape



Source: Berner, Grohs, K, Petersen. The Modern Mathematics of Deep Learning. In: Mathematical Aspects of Deep Learning, Cambridge, 2022.

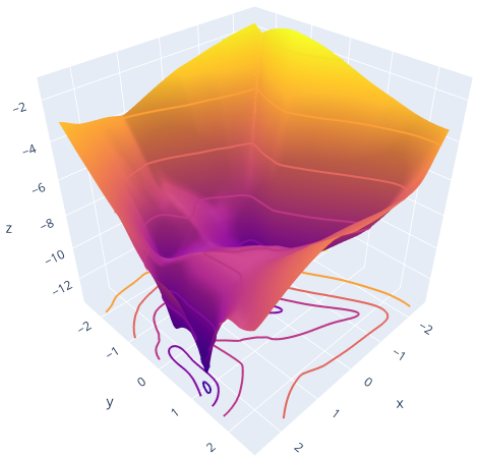
LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Some Approaches: Paths and Level Sets

Idea:

- Analyze paths through the parameter space.
- Focus on those, for which the associated empirical risks are monotone.
- Aim for paths of non-increasing empirical risk to the global minimum.

$\rightsquigarrow$ No such path can escape a minimum.

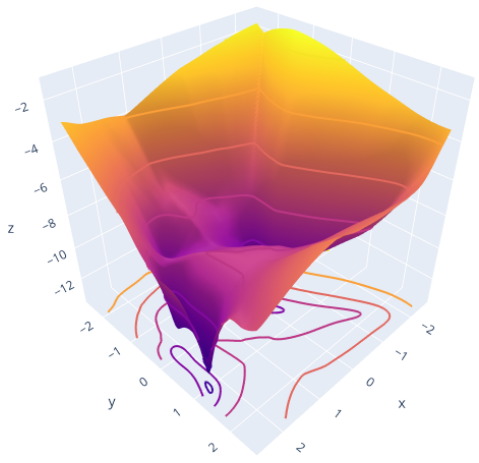


Some Approaches: Paths and Level Sets

Idea:

- Analyze paths through the parameter space.
- Focus on those, for which the associated empirical risks are monotone.
- Aim for paths of non-increasing empirical risk to the global minimum.

$\rightsquigarrow$ No such path can escape a minimum.



Some Results... *(Freeman, Bruna; '17) (Venturi, Bandeira, Bruna; '18)*
...about the presence or absence of *spurious valleys*, defined as connected components of sub-level sets that do not include a global minimum.

Some Approaches: Spin Glass Interpretation

"Definition": The Hamiltonian of the *spin glass model* is a random function on the $(n-1)$-dimensional sphere of radius $\sqrt{n}$.

Some Approaches: Spin Glass Interpretation

"Definition": The Hamiltonian of the *spin glass model* is a random function on the $(n-1)$-dimensional sphere of radius $\sqrt{n}$.

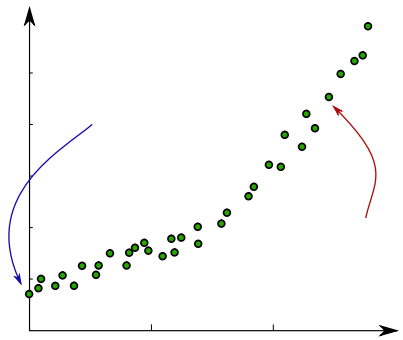
Theorem (Choromanska, Henaff, Mathieu, Arous, LeCun; 2015): "The loss of a neural network with random inputs can be considered as the Hamiltonian of a *spin glass model*, where the inputs of the model are the parameters of the neural network."

Some Approaches: Spin Glass Interpretation

"Definition": The Hamiltonian of the *spin glass model* is a random function on the $(n-1)$-dimensional sphere of radius $\sqrt{n}$.

Theorem (Choromanska, Henaff, Mathieu, Arous, LeCun; 2015):
"The loss of a neural network with random inputs can be considered as the Hamiltonian of a *spin glass model*, where the inputs of the model are the parameters of the neural network."

Implications:
The *set of critical points* leads to the relative number of directions in which the loss landscape has negative curvature.



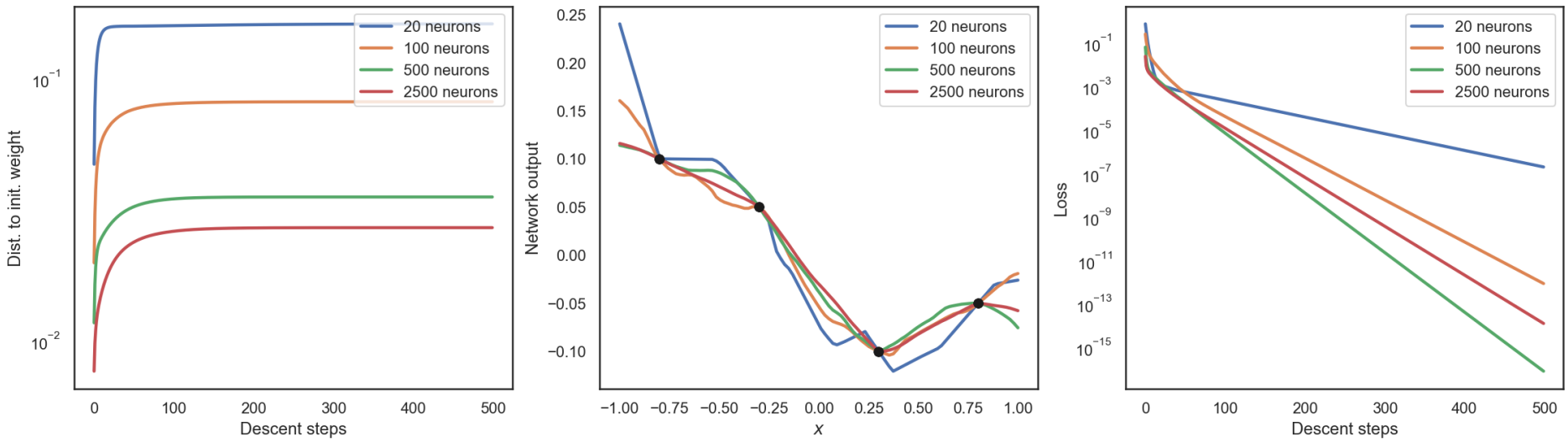
- Being further away from the optimal loss, then the critical points become more unstable.
- Being in a local minimum, implies that the loss is close to the global minimum.

Lazy Training

A Strange Effect

Observation:

During the training of highly overparametrized neural networks, the parameters seem to barely change.



Source: Berner, Grohs, K, Petersen. The Modern Mathematics of Deep Learning. In: Mathematical Aspects of Deep Learning, Cambridge, 2022.

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

A Simple Learning Model

Our Setting: Assume that...

- the *neural network model* is

$$\mathbb{R}^d \ni x \mapsto \Phi(x,\theta) := \sum_{j=1}^{n} \theta_j^{(2)} \varrho\big(\langle \theta_j^{(1)}, \begin{bmatrix} x \\ 1 \end{bmatrix} \rangle\big),$$

  where $\theta_j^{(1)} \in \mathbb{R}^{d+1}$ for $j \in [n]$, $\theta^{(2)} \in \mathbb{R}^n$ with a smooth activation function ϱ which is not affine linear.

- *training data* $s = ((x^{(i)}, y^{(i)}))_{i=1}^m \in (\mathbb{R}^d \times \mathbb{R})^m$, where $x_i \neq x_j$ for all $i \neq j$.

- the *empirical risk* is given by

$$r(\theta) = \widehat{\mathcal{R}}_s(\theta) = \frac{1}{m} \sum_{i=1}^{m} (\Phi(x^{(i)}, \theta) - y^{(i)})^2.$$

- for the *initialization* $\Theta = (\Theta^{(1)}, \Theta^{(2)})$, $\Theta_j^{(1)} \sim \mathcal{N}(0, 1/n)^{d+1}$, $j \in [n]$, and $\Theta_j^{(2)} \sim \mathcal{N}(0, 1/n)$, $j \in [n]$, are independent random variables.

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Introducing A Peculiar Kernel

Goal: Analyze the gradient $\nabla_\theta r(\Theta)$ over $\Theta = (\Theta^{(1)}, \Theta^{(2)})$.

Introducing A Peculiar Kernel

Goal: Analyze the gradient $\nabla_\theta r(\Theta)$ over $\Theta = (\Theta^{(1)}, \Theta^{(2)})$.

We obtain

$$\begin{aligned} \|\nabla_\theta r(\Theta)\|_2^2 &\geq \frac{4}{m^2} \Big\| \sum_{i=1}^{m} \nabla_{\theta^{(2)}} \Phi(x^{(i)}, \Theta)(\Phi(x^{(i)}, \Theta) - y^{(i)}) \Big\|_2^2 \\ &= \frac{4}{m^2} \big((\Phi(x^{(i)}, \Theta) - y^{(i)})_{i=1}^m\big)^T \bar{K}_\Theta (\Phi(x^{(j)}, \Theta) - y^{(j)})_{j=1}^m, \end{aligned}$$

where $\bar{K}_\Theta$ *is a random* $\mathbb{R}^{m \times m}$*-valued kernel* given by

$$(\bar{K}_\Theta)_{i,j} := \big(\nabla_{\theta^{(2)}} \Phi(x^{(i)}, \Theta)\big)^T \nabla_{\theta^{(2)}} \Phi(x^{(j)}, \Theta), \quad i, j \in [m].$$

Introducing A Peculiar Kernel

Goal: Analyze the gradient $\nabla_\theta r(\Theta)$ over $\Theta = (\Theta^{(1)}, \Theta^{(2)})$.

We obtain

$$\begin{aligned} \|\nabla_\theta r(\Theta)\|_2^2 &\geq \frac{4}{m^2} \Big\| \sum_{i=1}^m \nabla_{\theta^{(2)}} \Phi(x^{(i)}, \Theta)(\Phi(x^{(i)}, \Theta) - y^{(i)}) \Big\|_2^2 \\ &= \frac{4}{m^2} \big((\Phi(x^{(i)}, \Theta) - y^{(i)})_{i=1}^m\big)^T \bar{K}_\Theta (\Phi(x^{(j)}, \Theta) - y^{(j)})_{j=1}^m, \end{aligned}$$

where $\bar{K}_\Theta$ *is a random* $\mathbb{R}^{m \times m}$*-valued kernel* given by

$$(\bar{K}_\Theta)_{i,j} := \big(\nabla_{\theta^{(2)}} \Phi(x^{(i)}, \Theta)\big)^T \nabla_{\theta^{(2)}} \Phi(x^{(j)}, \Theta), \quad i, j \in [m].$$

For our two-layer neural networks,

$$\big(\nabla_{\theta^{(2)}} \Phi(x, \Theta)\big)_k = \varrho\left(\left\langle \Theta_k^{(1)}, \begin{bmatrix} x \\ 1 \end{bmatrix} \right\rangle\right), \quad x \in \mathbb{R}^d, \ k \in [n].$$

Introducing A Peculiar Kernel

Goal: Analyze the gradient $\nabla_\theta r(\Theta)$ over $\Theta = (\Theta^{(1)}, \Theta^{(2)})$.

We obtain

$$\begin{aligned} \|\nabla_\theta r(\Theta)\|_2^2 &\geq \frac{4}{m^2} \Big\| \sum_{i=1}^m \nabla_{\theta^{(2)}} \Phi(x^{(i)}, \Theta)(\Phi(x^{(i)}, \Theta) - y^{(i)}) \Big\|_2^2 \\ &= \frac{4}{m^2} \big((\Phi(x^{(i)}, \Theta) - y^{(i)})_{i=1}^m\big)^T \bar{K}_\Theta (\Phi(x^{(j)}, \Theta) - y^{(j)})_{j=1}^m, \end{aligned}$$

where $\bar{K}_\Theta$ *is a random* $\mathbb{R}^{m\times m}$*-valued kernel* given by

$$(\bar{K}_\Theta)_{i,j} := \big(\nabla_{\theta^{(2)}} \Phi(x^{(i)}, \Theta)\big)^T \nabla_{\theta^{(2)}} \Phi(x^{(j)}, \Theta), \quad i,j \in [m].$$

For our two-layer neural networks,

$$\big(\nabla_{\theta^{(2)}} \Phi(x, \Theta)\big)_k = \varrho\left(\left\langle \Theta_k^{(1)}, \begin{bmatrix} x \\ 1 \end{bmatrix} \right\rangle\right), \quad x \in \mathbb{R}^d, \ k \in [n].$$

Thus,

$$\bar{K}_\Theta = \sum_{k=1}^n v_k v_k^T \quad \text{with } v_k = \left(\varrho\left(\left\langle \Theta_k^{(1)}, \begin{bmatrix} x^{(i)} \\ 1 \end{bmatrix} \right\rangle\right)\right)_{i=1}^m \in \mathbb{R}^m, \ k \in [n].$$

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Controlling the Gradient

Recall:

$$\bar{K}_\Theta = \sum_{k=1}^{n} v_k v_k^T \qquad \text{with} \quad v_k = \left(\varrho \left(\left\langle \Theta_k^{(1)}, \begin{bmatrix} x^{(i)} \\ 1 \end{bmatrix} \right\rangle \right) \right)_{i=1}^{m} \in \mathbb{R}^m, \quad k \in [n].$$

Controlling the Gradient

Recall:

$$\bar{K}_\Theta = \sum_{k=1}^{n} v_k v_k^T \qquad \text{with} \quad v_k = \left(\varrho \left(\left\langle \Theta_k^{(1)}, \begin{bmatrix} x^{(i)} \\ 1 \end{bmatrix} \right\rangle \right) \right)_{i=1}^{m} \in \mathbb{R}^m, \quad k \in [n].$$

Key Property of the Kernel:

For sufficiently large n, with high probability $\bar{K}_\Theta$ is a positive definite kernel with smallest eigenvalue $\lambda_{\min}(\bar{K}_\Theta)$ scaling linearly with n.

Controlling the Gradient

Recall:

$$\bar{K}_\Theta = \sum_{k=1}^{n} v_k v_k^T \qquad \text{with} \quad v_k = \left(\varrho \left(\left\langle \Theta_k^{(1)}, \begin{bmatrix} x^{(i)} \\ 1 \end{bmatrix} \right\rangle \right) \right)_{i=1}^{m} \in \mathbb{R}^m, \quad k \in [n].$$

Key Property of the Kernel:

For sufficiently large n, with high probability $\bar{K}_\Theta$ is a positive definite kernel with smallest eigenvalue $\lambda_{\min}(\bar{K}_\Theta)$ scaling linearly with n.

Controlling the Gradient:

With high probability,

$$\|\nabla_\theta r(\Theta)\|_2^2 \geq \frac{4}{m^2} \lambda_{\min}(\bar{K}_\Theta) \|(\Phi(x^{(i)}, \Theta) - y^{(i)})_{i=1}^m\|_2^2 \gtrsim \frac{n}{m} r(\Theta).$$

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Controlling the Gradient (Continued)

Recall: Controlling the Gradient:

With high probability,

$$\|\nabla_\theta r(\Theta)\|_2^2 \geq \frac{4}{m^2}\lambda_{\min}(\bar{K}_\Theta)\|(\Phi(x^{(i)},\Theta) - y^{(i)})_{i=1}^m\|_2^2 \gtrsim \frac{n}{m} r(\Theta).$$

Controlling the Gradient (Continued)

Recall: Controlling the Gradient:
With high probability,

$$\|\nabla_\theta r(\Theta)\|_2^2 \geq \frac{4}{m^2}\lambda_{\min}(\bar{K}_\Theta)\|(\Phi(x^{(i)},\Theta)-y^{(i)})_{i=1}^m\|_2^2 \gtrsim \frac{n}{m}r(\Theta).$$

Let $\bar{\theta} \in B_1(0)$. Then, with high probability,

$$\begin{aligned}
\|\nabla_\theta r(\Theta+\bar{\theta})\|_2^2 &\geq \frac{4}{m^2}\Big\|\sum_{i=1}^m \nabla_{\theta^{(2)}}\Phi(x^{(i)},\Theta+\bar{\theta})(\Phi(x^{(i)},\Theta+\bar{\theta})-y^{(i)})\Big\|_2^2 \\
&= \frac{4}{m^2}\Big\|\sum_{i=1}^m (\nabla_{\theta^{(2)}}\Phi(x^{(i)},\Theta)+\mathcal{O}(1))(\Phi(x^{(i)},\Theta+\bar{\theta})-y^{(i)})\Big\|_2^2 \\
&\gtrsim \frac{1}{m^2}(\lambda_{\min}(\bar{K}_\Theta)+\mathcal{O}(1))\|(\Phi(x^{(i)},\Theta+\bar{\theta})-y^{(i)})_{i=1}^m\|_2^2 \\
&\gtrsim \frac{n}{m}r(\Theta+\bar{\theta}).
\end{aligned}$$

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Exponential Convergence

Result of our Argumentation:

For sufficiently small step sizes η and $\|\Theta^{(k)} - \Theta\| \leq 1$ for all $k \in [K+1]$,

$$r(\Theta^{(K+1)}) \approx r(\Theta^{(K)}) - \eta \|\nabla_\theta r(\Theta^{(K)})\|_2^2 \leq \left(1 - \frac{c\eta n}{m}\right) r(\Theta^{(K)}) \lesssim \left(1 - \frac{c\eta n}{m}\right)^K,$$

for $c \in (0, \infty)$ so that $\|\nabla_\theta r(\Theta^{(k)})\|_2^2 \geq \frac{cn}{m} r(\Theta^{(k)})$ for all $k \in [K]$.

Exponential Convergence

Result of our Argumentation:
For sufficiently small step sizes η and $\|\Theta^{(k)} - \Theta\| \leq 1$ for all $k \in [K+1]$,

$$r(\Theta^{(K+1)}) \approx r(\Theta^{(K)}) - \eta \|\nabla_\theta r(\Theta^{(K)})\|_2^2 \leq \left(1 - \frac{c\eta n}{m}\right) r(\Theta^{(K)}) \lesssim \left(1 - \frac{c\eta n}{m}\right)^K,$$

for $c \in (0, \infty)$ so that $\|\nabla_\theta r(\Theta^{(k)})\|_2^2 \geq \frac{cn}{m} r(\Theta^{(k)})$ for all $k \in [K]$.

Extension:
If also $\|\nabla_\theta r(\Theta + \bar{\theta})\|_2^2 \lesssim \frac{n}{m} r(\Theta + \bar{\theta})$, then

$$\|\Theta^{(k)} - \Theta\|_2 \leq 1 \quad \text{for all } k \lesssim \sqrt{m/(\eta^2 n)}.$$

Also,

$$\left(1 - \frac{c\eta n}{m}\right)^K \leq e^{-c\sqrt{n/m}}.$$

Exponential Convergence

Theorem (Chizat, Oyallon, Bach; 2019):

(1) "Gradient descent converges with an exponential rate to an arbitrary small empirical risk if the width n is sufficiently large."

(ii) "The iterates of the descent algorithm stay in a small fixed neighborhood of the initialization during training."

$\rightsquigarrow$ *Lazy Training!*

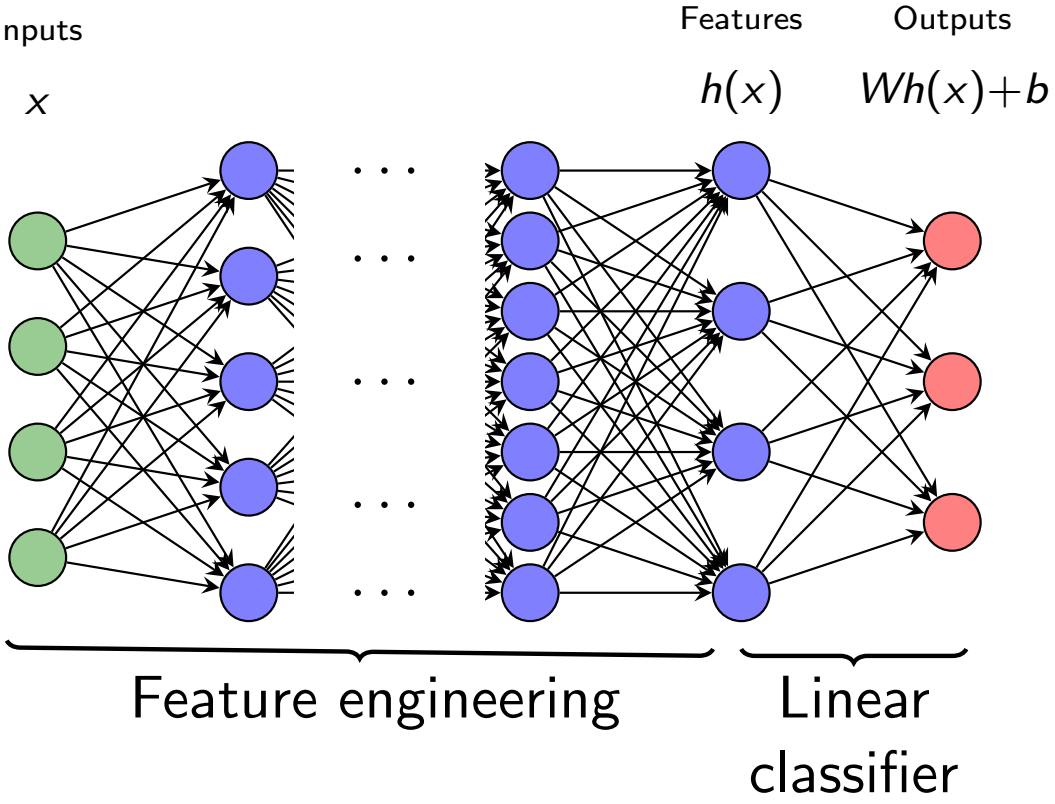
Neural Collapse

Single-Label Classification Problem

- **Goal:** Predict probabilities of classes $\{1, \ldots, N\}$ for inputs $x \in \mathcal{D}$

Single-Label Classification Problem

- **Goal:** Predict probabilities of classes $\{1, \ldots, N\}$ for inputs $x \in \mathcal{D}$
- Train network $\Phi : \mathbb{R}^d \to \mathbb{R}^N$ computing pre-softmax scores



Equiangular Tight Frame

Definition:

An *equiangular tight frame* is a family of vectors $\{x_i\}_{i=1}^n$ in $\mathbb{R}^d$ with

(1) $\|x_i\| = 1$ for all i,

(2) $|\langle x_i, x_j \rangle| = c$ for all $i \neq j$ and some constant c,

(3) $\frac{d}{n} \sum_{i=1}^n \langle x, x_i \rangle x_i = x$ for all $x \in \mathbb{R}^d$.

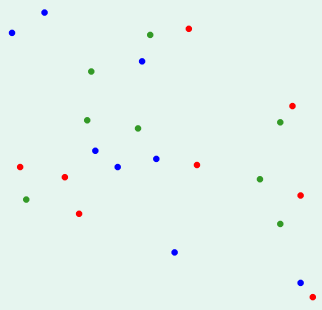
Remark:

An equiangular tight frame is a type of optimal packing of lines in Euclidean space.

Neural Collapse (Papyan, Han, Donoho; 2020)

Illustration

For 3 classes, features of training samples:



Initialization

Notation

- $h_n^{(1)}, \ldots, h_n^{(K)} :=$ features of samples in class n
- $h_n := \frac{1}{K} \sum_{k=1}^{K} h_n^{(k)}$ class-n mean
- $h := \frac{1}{N} \sum_{n=1}^{N} h_n$ global mean

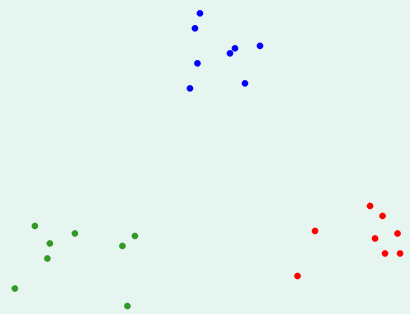
Neural Collapse Phenomena

(in the *terminal phase* of training)

Neural Collapse (Papyan, Han, Donoho; 2020)

Illustration

For 3 classes, features of training samples:



Notation

- $h_n^{(1)}, \ldots, h_n^{(K)}$:= features of samples in class n
- $h_n := \frac{1}{K} \sum_{k=1}^{K} h_n^{(k)}$ class-n mean
- $h := \frac{1}{N} \sum_{n=1}^{N} h_n$ global mean

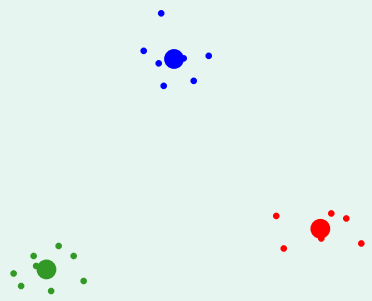
Neural Collapse Phenomena

(in the *terminal phase* of training)

Neural Collapse (Papyan, Han, Donoho; 2020)

Illustration

For 3 classes, features of training samples:



Notation

- $h_n^{(1)}, \ldots, h_n^{(K)}$:= features of samples in class n
- $h_n := \frac{1}{K} \sum_{k=1}^{K} h_n^{(k)}$ class-n mean
- $h := \frac{1}{N} \sum_{n=1}^{N} h_n$ global mean

Neural Collapse Phenomena

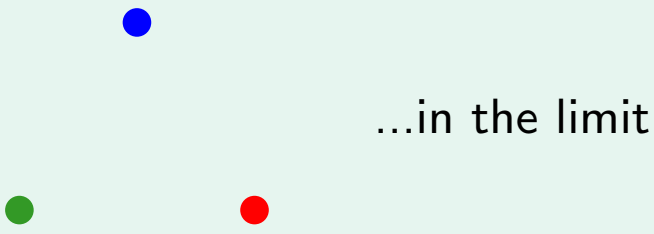
(in the *terminal phase* of training)

1. *Variability collapse:* $\|h_n^{(k)} - h_n\|_2 \to 0$

Neural Collapse (Papyan, Han, Donoho; 2020)

Illustration

For 3 classes, features of training samples:



Notation

- $h_n^{(1)}, \ldots, h_n^{(K)} :=$ features of samples in class n
- $h_n := \frac{1}{K} \sum_{k=1}^{K} h_n^{(k)}$ class-n mean
- $h := \frac{1}{N} \sum_{n=1}^{N} h_n$ global mean

Neural Collapse Phenomena

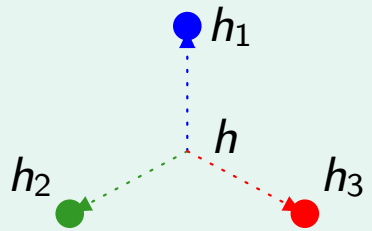
(in the *terminal phase* of training)

1. *Variability collapse:* $\|h_n^{(k)} - h_n\|_2 \to 0$

Neural Collapse (Papyan, Han, Donoho; 2020)

Illustration

For 3 classes, features of training samples:



...in the limit

Notation

- $h_n^{(1)}, \ldots, h_n^{(K)} :=$ features of samples in class n
- $h_n := \frac{1}{K} \sum_{k=1}^{K} h_n^{(k)}$ class-n mean
- $h := \frac{1}{N} \sum_{n=1}^{N} h_n$ global mean

Neural Collapse Phenomena

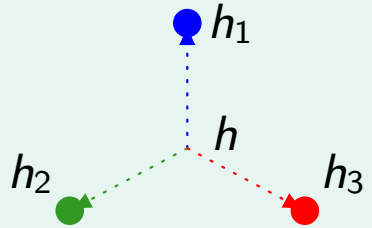
(in the *terminal phase* of training)

1. *Variability collapse:* $\|h_n^{(k)} - h_n\|_2 \to 0$
2. *Simplex equiangular tight frame (ETF) configuration:*
   - $\|h_n - h\|_2 - \|h_m - h\|_2 \to 0$
   - $\cos \measuredangle(h_n - h, h_m - h) \to -\frac{1}{N-1}$

Neural Collapse (Papyan, Han, Donoho; 2020)

Illustration

For 3 classes, features of training samples:



...in the limit

Notation

- $h_n^{(1)}, \ldots, h_n^{(K)} :=$ features of samples in class n
- $h_n := \frac{1}{K} \sum_{k=1}^{K} h_n^{(k)}$ class-n mean
- $h := \frac{1}{N} \sum_{n=1}^{N} h_n$ global mean

Neural Collapse Phenomena

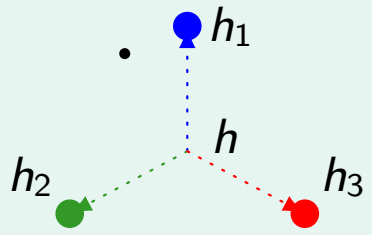
(in the *terminal phase* of training)

1. *Variability collapse:* $\|h_n^{(k)} - h_n\|_2 \to 0$
2. *Simplex equiangular tight frame (ETF) configuration:*
   - $\|h_n - h\|_2 - \|h_m - h\|_2 \to 0$
   - $\cos \measuredangle(h_n - h, h_m - h) \to -\frac{1}{N-1}$
3. *Duality:* $\|h_n - h - Cw_n\|_2 \to 0$ for some $C \in \mathbb{R}$

Neural Collapse (Papyan, Han, Donoho; 2020)

Illustration

For 3 classes, features of training samples:



...in the limit

Notation

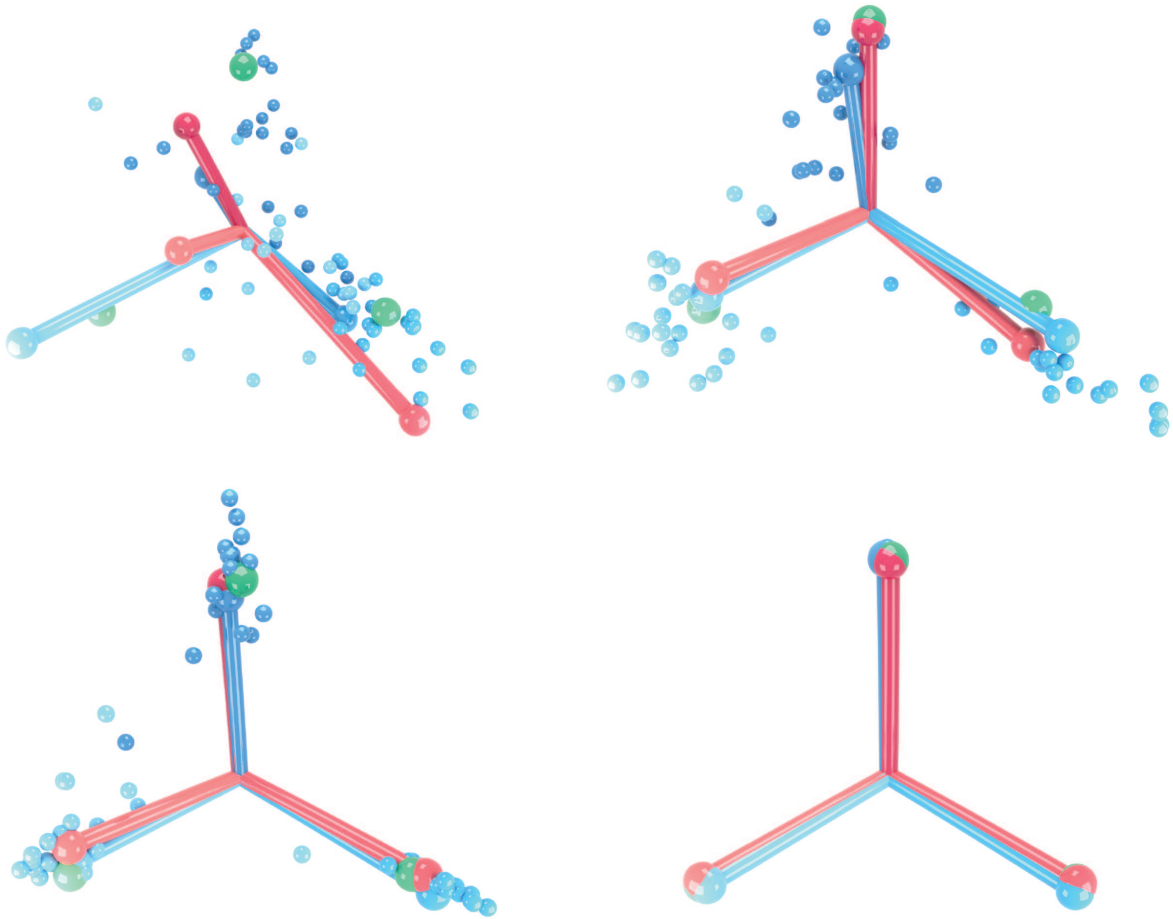
- $h_n^{(1)}, \ldots, h_n^{(K)} :=$ features of samples in class n
- $h_n := \frac{1}{K} \sum_{k=1}^{K} h_n^{(k)}$ class-n mean
- $h := \frac{1}{N} \sum_{n=1}^{N} h_n$ global mean

Neural Collapse Phenomena

(in the *terminal phase* of training)

1. *Variability collapse:* $\|h_n^{(k)} - h_n\|_2 \to 0$
2. *Simplex equiangular tight frame (ETF) configuration:*
   - $\|h_n - h\|_2 - \|h_m - h\|_2 \to 0$
   - $\cos \measuredangle(h_n - h, h_m - h) \to -\frac{1}{N-1}$
3. *Duality:* $\|h_n - h - Cw_n\|_2 \to 0$ for some $C \in \mathbb{R}$
4. *Nearest class center behavior*

Neural Collapse



Source: Papyan, Han, Donoho. Prevalence of Neural Collapse during the terminal phase of deep learning training. PNAS 117 (2020), 24652–24663.

Considerations

Terminal Phase of Training:

- The last-layer features are not only linearly separable, but actually collapsed to an *equi-angular tight frame*.
- The last-layer classifier is behaviorally equivalent to the *Nearest Class-Center decision rule*.

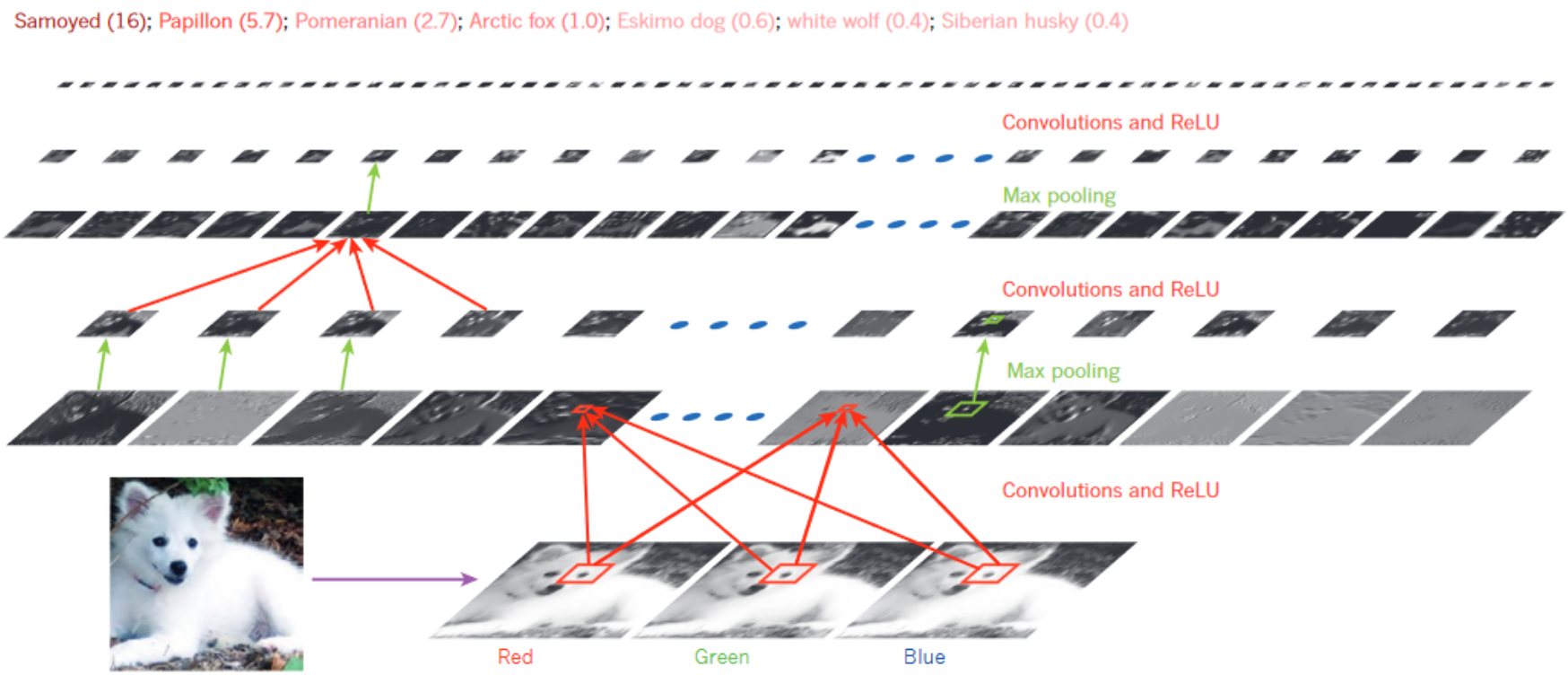
Additional Work:

- Mixon, Parshall, Pi; 2020
- Nguyen, Levie, K, Bruna; 2021
- Kornblith, Chen, Lee, Norouzi; 2021
- ...

Is Training Necessary?

Convolutional Neural Networks (CNNs)

Schematic Illustration:



Operation in each Layer:

Input $\rightarrow$ Convolution $\rightarrow$ Activation $\rightarrow$ Pooling $\rightarrow$ Output

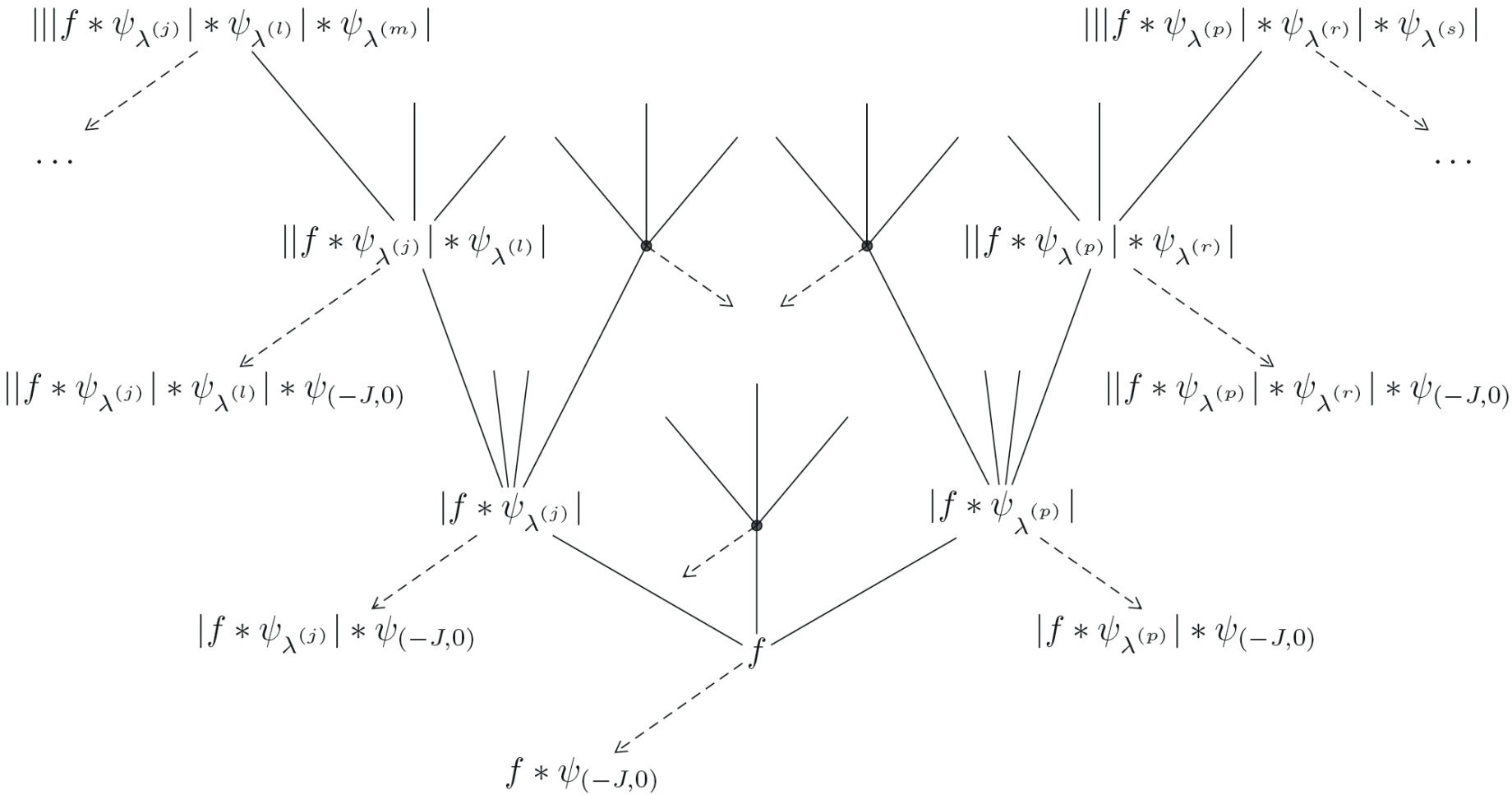
A Mathematical Approach

A Very Nice Idea...

The *scattering transform* (Mallat, 2014) is a special convolutional neural network:

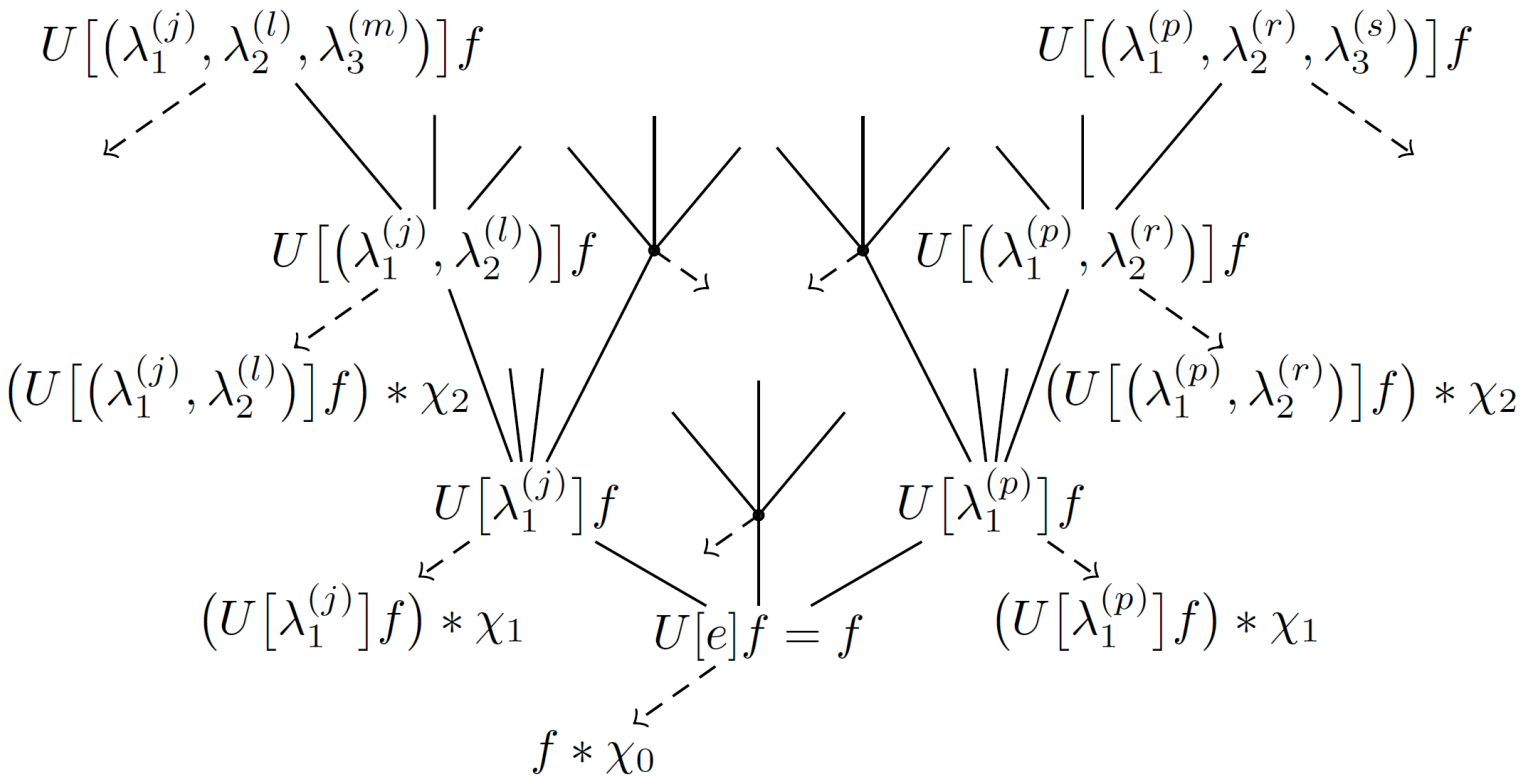
- It uses fixed predefined (wavelet) filters.
- It performs almost as good as a trained neural network in some applications.
- It is more accessible to theoretical analysis.
- There exists a continuous as well as discrete theory.

Scattering Transform



LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Scattering Transform



Scattering Transform

Definition: Let

- $\Psi_n = \{\psi_{\lambda_n}\}_{\lambda_n \in \Lambda_n}, \psi_{\lambda_n} \in L^1(\mathbb{R}^d) \cap L^2(\mathbb{R}^d)$ with

$$\sum_{\lambda_n \in \Lambda_n} \|f * \psi_{\lambda_n}\|_2^2 \leq B_n \|f\|_2^2 \quad \text{for all } f \in L^2(\mathbb{R}),$$

- For $R_n \geq 1$ the subsampling factor, let

$$(U_n[\lambda_n]f)(x) = R_n^2 |f * \psi_{\lambda_n}|(R_n x), \quad \lambda \in \mathbb{R}^d,$$

- For a path of index sets $q = (\lambda_1, \ldots \lambda_n)$, $\lambda_i \in \Lambda_i$ let

$$U[q]f = U_n[\lambda_n](U_{n-1}[\lambda_{n-1}] - (U_1[\lambda_1]f)),$$

- $\chi_{n-1} := \psi_{\lambda_n}$ for every $n \in \mathbb{N}$.

The associated *scattering transformation* Φ_Ω is defined by

$$f \mapsto \Phi_\Omega(f) := \underbrace{\bigcup_{n=0}^{\infty} \{U[q]f * \chi_{n-1}\}_{q=(\lambda_1, \ldots \lambda_n)}}_{\text{Interpretation: Feature vector}}.$$

Translation Invariance of the Scattering Transform

Theorem (Mallat; 2014)(Wiatowski at al.; 2016):
Let Φ_Ω be a scattering transformation with $R_n := 1$ for all n. Then Φ_n is *translation invariant*, i.e.

$$\Phi_\Omega(T_t f) = T_t \Phi_\Omega(f)$$

for all $t \in \mathbb{R}^d$ with $(T_t f)(x) = f(x - t), x \in \mathbb{R}^d$, in particular,

$$U[q](T_t f) * \chi_{n-1} = T_t(U[q]f * \chi_{n-1})$$

for all $t \in \mathbb{R}^d$.

Deformation Stability of the Scattering Transform

Theorem (Mallat; 2014)(Wiatowski at al.; 2016):
Let Φ_Ω be a scattering transformation with $\max_{n\in\mathbb{N}} \max\{B_n, B_n L_n^2\} \leq 1$. Then for any $K > 0$, the scattering transformation Φ_Ω is *stable on* $\mathcal{E}_s^2(\mathbb{R}^d)$ *with respect to deformations*.
This means that for every $K > 0$, there exists $C_K > 0$ such that for all $f \in \mathcal{E}_s^2(\mathbb{R}^d)$ and $\tau \in \mathcal{C}^1(\mathbb{R}^d, \mathbb{R}^d)$ with

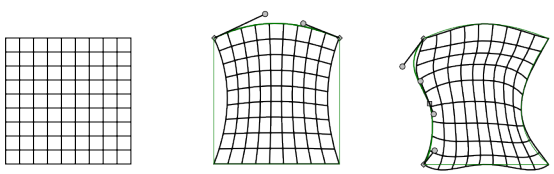
$$\|\tau\|_\infty \leq \frac{1}{2} \quad \text{and} \quad \|D\tau\|_\infty \leq \frac{1}{2d},$$

we have

$$|||\Phi_\Omega(F_\tau f) - \Phi_\Omega(f)||| \leq C_K \|\tau\|_\infty^{\frac{1}{2}},$$

where

$$(F_\tau f)(x) = f(x - \tau(x)).$$



LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Some Final Thoughts...

Conclusions

Optimization:

- *Stochastic gradient descent* is the typical choice.
- Due to the severe nonconvexity, it is a *mystery* why "good" local minima are found.

Unraveling the Mystery:

- Analyzing the loss landscape
- Lazy training
- Neural Collapse

Is Training Necessary?

- Scattering Transform

THANK YOU!

References available at:

`www.ai.math.lmu.de/kutyniok`

Survey Paper (arXiv:2105.04026):

Berner, Grohs, K, Petersen, *The Modern Mathematics of Deep Learning*.

Check related information on Twitter at:

`@GittaKutyniok`

Upcoming Book:

- P. Grohs and G. Kutyniok, eds.
  Mathematical Aspects of Deep Learning
  Cambridge University Press, to appear.

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN