Machine learning from a complexity point of view

Artemy Kolchinsky
SFI CSSS 2019
PART I: Overview

What is machine learning?

What are neural networks?

The rise of deep learning

PART II: Deep nets deep dive

Why do deep nets work so well?

Learning in deep nets, in the brain, and in evolution

Caveats of deep learning
(1) What is machine learning?
Artificial intelligence:
- General science of creating intelligent automated systems
  - Chess playing, robot control, automating industrial processes, etc.

Machine learning (ML):
- Subset of AI, aims to develop algorithms that can learn from data
- Strongly influenced by statistics
Example of ML problem

Given data, make model of how personal annual income depends on
- Age
- Gender
- Years in school
- Zip code
- ...

Example that’s not ML

“Traffic collision avoidance system” (TCAS)

if distance(plane1, plane2)<=1.0
    sound_alarm()
if altitude(plane1)>=altitude(plane2)
    alert(plane1, GO_UP)
else
    alert(plane2, GO_UP)
...

Example that’s not ML

“Traffic collision avoidance system” (TCAS)

if distance(plane1, plane2)<=1.0
    sound_alarm()
if altitude(plane1)>=altitude(plane2)
    alert(plane1, GO_UP)
else
    alert(plane2, GO_UP)
...

Example that’s not ML

“Traffic collision avoidance system” (TCAS)

if distance(plane1, plane2)<=1.0
    sound_alarm()
if altitude(plane1)>=altitude(plane2)
    alert(plane1, GO_UP)
else
    alert(plane2, GO_UP)
...
Supervised Learning

Training data set

→ “Cat”
→ “Dog”
→ “Cat”
→ “Cat”
→ “Cat”

Statistical model
Parameterized set of input-output maps:
{ Output = \( f_\theta (\text{Input}) \) }_{\theta}

Training algorithm
Chooses optimal parameter values \( \theta^* \)

“Trained Model” \( f_{\theta^*} \)

New input \( x \)
→ ???

Predictions \( f_{\theta^*}(x) \)
→ “Dog”

Example models/algorithms: logistic regression, support vector machines (SVMs), random forests, neural networks, “deep learning” (deep neural networks), etc.

Each algorithm has strengths and weaknesses. No “universally” best one for all domains / situations
A geometric view of supervised learning

Image can be represented digitally as a list of numbers specifying RGB color intensities at each pixel (a “vector”)

Each vector indicates a point in high-dimensional “data space” (# dimensions = 3 × # of pixels)

For conceptual simplicity, consider as coordinates in an abstract 2-D space

Cat

Dog

= <0.271, 0.543, 0.198, 0.362, …>

= <0.842, 0.527, 0.924, 0.421, …>

= <0.873, 0.321, 0.187, 0.011, …>

= <0.641, 0.874, 0.983, 0.232, …>
A geometric view of supervised learning

Choose parameters via
\[ \theta^* = \text{argmin}_\theta \text{Error}(\theta, \text{TrainData}) \]

Training algorithm selects parameters (i.e., twists “knobs”) to find the best separating surface
A geometric view of supervised learning

The separating surface splits “data space” into dog and cat regions
“Training error” on training dataset
Training adjusts parameters to minimize such errors

“Testing error”
Errors made on new data provided after training
A geometric view of supervised learning

“Underfitting”
Too few parameters
Doesn’t fit data well

“Overfitting”
Too many parameters
won’t generalize on new data
(i.e., “memorized” training data,
rather than learnt “the pattern”)

Good model

“Drifting”

Too many parameters
won’t generalize on new data
(i.e., “memorized” training data,
rather than learnt “the pattern’’

"Overfitting"
A geometric view of supervised learning

“Underfitting”

“cat” ×

“dog” ✓

“Overfitting”

“cat” ×
“Generalization performance”: ability of learning algorithm to do well on new data

How to select optimal number of parameters?

1. **Cross-validation**: split training data into two chunks; train on one and validate on the other

2. **Regularization**: prevent overfitting by penalizing models that are “too flexible”
   
   E.g.: $\theta^* = \arg\min_{\theta} \text{TrainError}(\theta) + \lambda \|\theta\|_2$

**CAVEAT**: in Part II, we’ll see that recent research is putting much of the “common wisdom” about the above trade-off curve into question!
Supervised learning summary

Supervised learning uses training data to learn input-output mapping.

Many supervised learning algorithms exist, each with different strengths.

Goal is low testing error on new unseen data.

High testing error when model is too simple and underfits, or when model is too complex and overfits.
(2) What are neural nets?
1940s: Donald Hebb

Proposed that networks of simple interconnected units (aka “nodes” or “neurons”) using simple rules can learn to perform very complicated tasks.

The simplest rule: if two units are active at the same time, strengthen the connection between them (“Hebbian learning”).

Inspired by biological neurons.
Late 1950s: Perceptron

A computational model of learning by psychologist Frank Rosenblatt

The first neural network, along with a learning rule to minimize training error

Demonstrated that it could recognize simple patterns
Late 1950s: Perceptron

Weighted Sum

$\sum_{i} w_{i}x_{i}$

Output $y$ (either 0 or 1)

“Threshold Nonlinearity”:

0 if $\sum_{i} w_{i}x_{i} < b$

1 if $\sum_{i} w_{i}x_{i} \geq b$

Learning involves following a simple rule for changing the weights, so as to minimize training error

$w_{1}$ and $w_{2}$: connections “weights”, i.e., the parameters $\theta$
Late 1950s: Perceptron

Input 1 $x_1$

Input 2 $x_2$

$w_1$

$w_2$

Output $y$ (0 or 1)

Perceptron separating surface is a line

Has almost all the ingredients of a modern neural network
1969: Minsky & Papert, *Perceptrons*

Two AI pioneers analyzed mathematics of learning with perceptrons

Showed that a **single-layer perceptron** could never be taught to recognize some simple patterns

**Killed neural network research for 20 years**
Perceptron separating surface is a line

1969: Minsky & Papert, *Perceptrons*

x₁

w₁

x₂

w₂

Σ

Non-linearly separable problem:
1969: Minsky & Papert, *Perceptrons*

Non-linearly separable problem:

Linearly separable problem:

Perceptron **can** learn this

Perceptron **cannot** learn this
1986: Modern neural nets

Three crucial ingredients:

1. More layers
2. Differentiable activations and error functions
3. New training algorithm (“backpropagation”)

Learning representations by back-propagating errors

David E. Rumelhart*, Geoffrey E. Hinton† & Ronald J. Williams*

Nature, 1986
More layers

Input1 $x_1$

Input2 $x_2$

$\Sigma + \sum$

$\Sigma + \sum$

$\Sigma + \sum$

Out

“Intersection Nonlinearity”

0 if $\sum_i x_i < 2$

1 if $\sum_i x_i \geq 2$

Can solve non-linearly separable problems!
Differentiability

Learning by gradient descent:
$$\theta_{t+1} = \theta_t - \alpha \nabla L(\theta)$$

Threshold nonlinearity replaced by differentiable activation function
$$x_i = \phi \left( \sum_j w_{ji} x_j \right) \quad \text{E.g.: } \phi(x) = \frac{1}{1 + e^{-x}}$$

Differentiable error:
$$\text{E.g.: } L(\theta) = \sum_{x,y\in\text{Dataset}} (f_\theta(x) - y)^2$$
1986: Backpropagation

Learning by gradient descent:

\[ \theta_{t+1} = \theta_t - \alpha \nabla L(\theta) \]

Unfortunately, in general \( \nabla L(\theta) \) can hard to compute!

The backpropagation trick

\[ x^{(i+1)} = \phi(W^{(i)}x^{(i)}) \]

\[ \frac{\partial L}{\partial x^{(i)}} = \frac{\partial L}{\partial x^{(i+1)}} \frac{\partial x^{(i+1)}}{\partial x^{(i)}} \]

Error gradient in layer \( i \)

Error gradient in layer \( i+1 \)

Partial of layer \( i+1 \) w.r.t layer \( i \)

For prediction, activity flows forward layer-by-layer, from inputs to outputs

For learning, error gradients flow backwards layer-by-layer, from outputs to inputs
1989: Universal Approximation Theorem

Any continuous function \( f : \mathbb{R}^n \to \mathbb{R} \) can be computed by a neural network with one hidden layer, up to any desired accuracy \( \varepsilon > 0 \).

**Caveat 1**: The number of hidden neurons may be exponentially large.

**Caveat 2**: We can represent any function. But that doesn’t guarantee that we can learn any function (even given infinite data!)
1990s - 2010s

Neural nets attract attention from cognitive scientists and psychologists.

However, their was not competitive for most applications.

A neural network winter lasts for two decades.
Neural networks summary

- Neural nets: **supervised learning algorithms** consisting of multiple layers of interconnected “neurons”, with nonlinear transformations

- Connection strengths (“**weights**”) are the learnable parameters

- Trained using **backpropagation**, a clever trick for efficient **gradient descent**

- Foundational neural net ideas begin in the 40s-50s; appeared in their modern form by the mid-1980s
(3) The rise of deep learning
2012: Deep net wins ImageNet (a major ML competition)

“Deep neural network” did so much better that a breakthrough moment in AI was immediately recognized

Deep neural net: 15% error

Next best (w/ hand-coded features): 25% error

Krizhevsky, Sutskever, Hinton 2012
Traditional neural network

Deep neural network

GoogLeNet (image recognition)
Deep learning now dominates most areas of ML

- Image processing
- Voice recognition
- Video games
- Board games
- Translation
- Medical diagnosis
Machine learning phases of matter

Juan Carrasquilla¹* and Roger G. Melko¹,²

“…we show that modern machine learning architectures, such as fully connected and convolutional neural networks, can be trained to detect non-trivial states with no conventional order, directly from raw state configurations sampled with Monte Carlo simulations.”
Why do deep networks do so well?

Mystery 1: On the surface, deep networks only marginally different from previous neural network approaches. Why do they do qualitatively better?

Mystery 2: Neural networks do not work well in highly-structured domains, like language translation and rule-driven board-games.

Mystery 3: Deep nets tend to have millions/billions of parameters. Traditional “statistical learning theory” suggests they should overfit horribly.
Adversarial examples

Panda: 77.7% + ∇ = Schoolbus: 99.3%
Generative adversarial networks (GANs)

Generator network tunes parameters to fool discriminator network (increase its error)

Discriminator network tunes parameters to distinguish training images from fake images made by generator net

Training dataset of unlabelled images

Random noise

Generator

Fake image

Discriminator

Real
Fake

Example of deep nets for unsupervised generative modeling

https://skymind.ai/images/wiki/GANs.png
GANs: Auto-generated faces
GANs: Auto-generated anime characters

Figure 7: Generated samples

"glasses"), color attributes are easier to learn. Notice that the boundary between similar colors like "white hair", "silver hair", "gray hair" is not clear enough. Sometimes people may have troubles to classify those confusing colors. This phenomenon lead to low precision scores for those attributes in our test.

Surprisingly, some rare color attributes like "orange eyes", "aqua hair", "aqua eyes" have a relative high precision even though samples containing those attributes are less than 1% in the training dataset. We believe visual concepts related to colors are simple enough for the generator to get well learned with a extremely small number of training samples.

On contrast, complex attribute like "hat", "glasses", "drill hair" are worst behaved attributes in our experiments. When conditioned on those labels, generated images are often distorted and difficult to identify. Although there are about 5% training samples assigned with those attributes, the complicated visual concept they implied are far more accessible for the generator to get well learned.

5.3.2 FID Evaluation

One possible quantitative evaluation method for GAN model is Fréchet Inception Distance (FID) proposed by Heusel et al.\cite{Heusel2017}. To calculate the FID, they use a pre-trained CNN (Inception model) to extract vision-relevant features from both real and fake samples. The real feature distribution and the
GANs: Auto-generated ML papers

Figure 8. Randomly generated samples of good papers.
These random samples capture the gestalt of a good paper: illustrative figures upfront, colorful images, a balanced layout of texts/math/tables/plots.

Figure 9. Paper enhancement using CycleGAN [25]. The trained bad-to-good paper model can be used as a suggestive tool for translating a bad paper into a good one. Typical suggestions include adding teaser figure upfront, making the figures more colorful, and filling up the last page so that it appears like a well-polished paper. This figure contains animated images flipping back and forth between the original bad paper and the translated good paper (best viewed using Adobe Acrobat Reader).


GANs: Style transfer

Karras et al 2019
youtube.com/watch?v=bIVU8UuHPKJ
Deep learning summary

- Deep neural nets are similar to existing neural networks, but with more layers and more structure.

- Since 2012, they have dominated most areas of machine learning.

- We don’t know exactly why they work so well (topic of Part II).

- They have been used to build very powerful generative models.
PART II

Deep nets deep dive
1. Why do deep networks work so well?

2. Learning in deep nets, in the brain, and in evolution

3. Caveats of deep learning
(1) Why do deep nets work so well?
Why do deep nets work so well?

Mystery 1: On the surface, deep nets are only marginally different from previous neural net approaches. Why do they do so much better?

Mystery 2: Neural networks are not supposed to work well in highly-structured domains, like language translation and rule-driven board-games.

Mystery 3: Deep nets have millions/billions of parameters. Traditional “statistical learning theory” suggests they should overfit horribly.
Why do deep nets work so well?

Reason 1: **Huge training datasets** and **computational power** (GPUs)

Unlike other algorithms, **deep nets** seem to “keep getting better” with more and more **training data** (*but when datasize is small, they often do worse than others!*).
Why do deep nets work so well?

Reason 2: **Noisy training regimes**

**Stochastic gradient descent (SGD):** gradient computed on random subsets of training data

**Dropout:** 50% of neurons randomly disabled during training

Random noise during training improves performance during testing

**Regularization:** controlling overfitting in a data-dependent manner, prevent algorithm from being “too flexible”

\[
\theta^* = \arg\min_\theta \text{Error}(\theta, \text{TrainingData}) + \lambda \|\theta\|_2^2
\]

\[
\theta^* = \arg\min_\theta \text{Error}(\theta, \text{TrainingData}) + \text{Noise}
\]
Why do deep nets work so well?

Reason 3: **Novel architectures**: deeper and more structured
Why do deep nets work so well?

Reason 3: **Novel architectures: deeper and more structured**

- Traditional connectivity pattern
- Connectivity pattern of ImageNet 2012 winner

“Convolutional layers”, which have highly structured, repeating weight patterns (reminiscent of **receptive fields** in our **visual system**).
Inductive bias: implicit or explicit assumptions built into a learning algorithm

No Free Lunch Theorem (slightly simplified) (Wolpert, 1996)
Let $\Omega$ be the set of all possible functions mapping inputs to outputs. On average across $\Omega$, no supervised learning algorithm can do better than random guessing.

Example assumptions

“Output is a linear function of input”

“Output is a smooth function of input”

“Input-output mapping has a low complexity”

….
Inductive bias of deep nets:
more layers may reflect greater hierarchy
Why does inductive bias matter?

The bigger the red region (the “haystack”), the more training data is needed by the learning algorithm to find $f$ (the “needle”).
Why Does Deep and Cheap Learning Work So Well?

Henry W. Lin1 · Max Tegmark2 · David Rolnick3

Received: 3 December 2016 / Accepted: 27 June 2017
© Springer Science+Business Media, LLC 2017

Abstract
We show how the success of deep learning could depend not only on mathematics but also on physics: although well-known mathematical theorems guarantee that neural networks can approximate arbitrary functions well, the class of functions of practical interest can frequently be approximated through “cheap learning” with exponentially fewer parameters than generic ones. We explore how properties frequently encountered in physics such as symmetry, locality, compositionality, and polynomial log-probability translate into exceptionally simple neural networks. We further argue that when the statistical process generating the data is of a certain hierarchical form prevalent in physics and machine learning, a deep neural network can be more efficient than a shallow one. We formalize these claims using information theory and discuss the relation to the renormalization group. We prove various “no-flattening theorems” showing when efficient linear deep networks cannot be accurately approximated by shallow ones without efficiency loss; for example, we show that $n$ variables cannot be multiplied using fewer than $2^n$ neurons in a single hidden layer.

Keywords
Artificial neural networks · Deep learning · Statistical physics

1 Introduction
Deep learning works remarkably well, and has helped dramatically improve the state-of-the-art in areas ranging from speech recognition, translation and visual object recognition to drug discovery, genomics and automatic game playing [1,2]. However, it is still not fully understood why deep learning works so well. In contrast to GOFAI (“good old-fashioned AI”) algorithms that are hand-crafted and fully understood analytically, many algorithms...
The inductive bias of convolutional deep nets

Typical image output by a randomly initialized deep net

“Fixed”/natural-looking versions of target image

Corrupted “target” image

Optimization trajectory

\[ \theta^* = \arg\min_{\theta} \text{Error}(\theta, \text{CorruptedImage}) \]
\[ \text{s.t. } \text{Dist}(\theta, \theta_{\text{init}}) < c \]

Ulyanov et al 2017
The inductive bias of convolutional deep nets

Ulyanov et al. 2017
Why do deep nets work so well?

Reason 4: High-dimensional optimization is weird
Why do deep nets work so well?

Reason 4: High-dimensional optimization is weird

Multi-layer neural nets have non-convex error surfaces

In high-dimensions, most critical points are saddle points

Modern deep nets can have $10^8$-$10^{10}$ parameters
Why do deep nets work so well?

Reason 4: High-dimensional optimization is weird

In high-dimensions, most critical points are actually saddle points

Local minima are not a problem for deep nets: they are rare, and close to global minima in terms of error (Dauphin 2014, Kawaguchi 2016, Du 2019)

“Mode Connectivity”
Not only are all local minima also global minima, but all minima are connected by simple, low-error paths (Garipov 2018, Draxler 2019, Kuditipudi 2019)
Why don’t deep nets overfit?
Why don’t deep nets overfit?

Modern deep nets can have $10^8$-$10^{10}$ parameters. Should overfit horribly.

Classical regime vs. Non-classical over-parameterized regime.

Testing error vs. Training error.
Why don’t deep nets overfit?

Explanation 1: effective # of parameters is low

• Most directions in parameter space do not matter much, so intrinsic dimension is low (Li 2018)

• Higher layers can be set to fixed random weights (Zhang 2019)

• Noise during training allows deep nets to be highly compressible (Arora 2018)

Explanation 2: "Lottery ticket hypothesis” (Frankle 2019)

Why don’t deep nets overfit?

Explanation #3: Overparameterization allows optimization to find “smoother” functions

Low-dimensional parameter space

High-dimensional parameter space

Set of all functions that fit training data perfectly

Zhang 2017, Belkin 2018
Why don’t deep nets overfit?

Modern deep nets can have $10^8-10^{10}$ parameters. They should overfit horribly.

Classical regime

Testing error

Training error

Non-classical over-parameterized regime

Zhang 2017, Belkin 2018
Summary: Why do deep nets work so well?

- More data and more computational power
- Lots of noise during training
- The right assumptions about the world (inductive bias)
- Advantages to optimization in high-dimensional spaces
- Deep nets have a nonclassical complexity vs. error-tradeoff
(2) Learning in deep nets, in the brain, and in evolution
Are deep nets like the brain?

**DEEP NETWORKS**
- Simple neurons
- Layered architecture, some structured connectivity
- Activity propagates forward, learning signals backward
- Supervised learning
- Noisy training
- Disembodied

**BRAINS**
- Complex spiking neurons, many other cell types
- Layered architecture, very complex connectivity
- Activity and learning signals propagate both ways
- Supervised + unsupervised
- Noisy training and operation
- Embodied
Are deep nets like the brain? Internal representations

Receptive field of early neural net layers resemble V1 receptive fields in the brain

Macaque V1
(Zylberberg, DeWeese 2013)

Deep neural net

We don’t know if higher-level representations are similar
Are deep nets like the brain?
Backpropagation

Learning by \textbf{gradient descent}:
\[
\theta_{t+1} = \theta_t - \alpha \nabla L(\theta)
\]

Unfortunately, in general \( \nabla L(\theta) \)
can hard to compute!

The backpropagation trick
\[
x^{(i+1)} = \phi(W^{(i)}x^{(i)})
\]
\[
\text{↓ chain rule of calculus}
\]
\[
\frac{\partial L}{\partial x^{(i)}} = \frac{\partial L}{\partial x^{(i+1)}} \frac{\partial x^{(i+1)}}{\partial x^{(i)}}
\]
\[
= \frac{\partial L}{\partial x^{(i+1)}} \phi'(x^{(i+1)}) W^{(i)}
\]

\begin{itemize}
\item \textbf{Error gradient in layer} \( i \)
\item \textbf{Error gradient in layer} \( i+1 \)
\item \textbf{Partial of layer} \( i+1 \) \textbf{w.r.t layer} \( i \)
\end{itemize}

This weight matrix can be replaced by a \textbf{fixed random matrix}, and learning still works!
Is deep learning like evolution?

**EVOLUTION WITH NATURAL SELECTION**

- **Stochastic hill climbing** on fitness landscape (similar to SGD on loss surface)
- Very **high dimensional** spaces

**Evolution and speciation on holey adaptive landscapes**

Sergey Gavrilets

“Properties of multidimensional adaptive landscapes are very different from those of low dimension. … a theoretical challenge in a low-dimensional case might be a trivial problem in a multidimensional context and vice versa.”

**Extradimensional bypass**

Peter A. Cariani *

Eaton Peabody Laboratory of Auditory Physiology, Department of Otoology and Laryngology, Harvard Medical School, Massachusetts Eye and Ear Infirmary, 243 Charles Street, Boston, MA 02114, USA

Received 3 May 2001; accepted 20 September 2001

“…Many local maxima may become saddle points in the higher dimensional space, such that gradient ascent can continue unimpeded”

https://msu.edu/~ostman/landscapes.html
Deep nets, brain, evolutions: a recipe for learning

- Gradient-descent (or something like it)
- Regularization by randomness
- High dimensional search spaces
- Lots of trials/computation

Powerful results
(3) Caveats of deep learning
Despite successes, much remains to be done

1. **Natural language understanding** (e.g.: summarizing a long story)
2. **Causal reasoning**
3. **Common sense reasoning**
4. Learning from **small data** (a.k.a. “zero-shot” or “single-shot” learning)
5. **Learning with less computation**
6. **Open-ended domains** (e.g.: autonomous cars)
7. **Motor control / embodiment**
8. **Transfer learning**
9. Can be very **brittle**
Is a deep learning winter coming?

- “Deep nets are over-hyped”
- “Deep nets don’t have causal reasoning / common sense / one-shot learning, etc.”
- “Deep nets are good for solving games, but not other real-world applications”
- Etc.

Deep nets work **much better** than expected.

In the past **7 years**, they solved many **very hard** problems (image recognition, voice recognition, Go, generative modeling, etc.)

They have many real-world applications, from **Siri to science to surveillance**

Most weaknesses are acknowledged, and actively being researched in ML
**RESOURCES**

**Foundations**

PDF: tinyurl.com/y6b8z5qv

Classic (2006)

PDF: tinyurl.com/y6khzl9e

**Deep learning**

Some good blogs:

- [distill.pub](http://distill.pub)
- [ai.googleblog.com](http://ai.googleblog.com)
- [deepmind.com/blog](http://deepmind.com/blog)
- [openai.com/blog](http://openai.com/blog)
- [lilianweng.github.io/lil-log](http://lilianweng.github.io/lil-log)
- [machinelearningmastery.com/blog](http://machinelearningmastery.com/blog)
- [offconvex.org](http://offconvex.org)

**State of the art**

Online:

- [www.deeplearningbook.org](http://www.deeplearningbook.org)

Arxiv Sanity Preserver

[arxiv-sanity.com](http://arxiv-sanity.com)

Top recent machine learning papers from arXiv
Thanks!

Artemy Kolchinsky
artemy@santafe.edu