Machine Learning
– an introduction –

Summer School on Emergent Phenomena in Quantum Materials
Cornell, May 2017

Simon Trebst
University of Cologne
In computer science, machine learning is concerned with algorithms that allow for **data analytics**, most prominently **dimensional reduction** and **feature extraction**.

Many computer programs/apps have machine learning algorithms built-in already.
Applications of machine learning techniques are booming and poised to enter our daily lives.

digital assistants  self-driving cars
Machines at play

Machine learning techniques can make **computers play**.

A computer at play is probably one of the most striking realization of **artificial intelligence**.

1996: G. Kasparow vs. IBM’s deep blue

2016: L. Sedol vs. Google’s AlphaGo
How do machines learn?

What is it that they can learn?

Can we control what they learn?

How can we benefit from machine learning?

biological neural network

artificial neural network
Artificial neural networks and deep learning

Recommended introduction: http://neuralnetworksanddeeplearning.com by Michael Nielsen
Artificial neural networks mimic biological neural networks (albeit at a much smaller scale).

They allow for an implicit knowledge representation, which is infused in supervised or unsupervised learning settings.
artificial neural networks

artificial neurons

(binary) input

\[ x_1 \rightarrow w_1 \]
\[ x_2 \rightarrow w_2 \]
\[ x_3 \rightarrow w_3 \]

output

\[ b \rightarrow \theta(z) \]

\[ \Theta(z) \]

\[ \Theta(\bar{w} \cdot \bar{x} - b) \]

example – Should I skip the first talk?

free coffee?

deep learning?

no topology?

[4] sleep in

[+3]

[+2]

[-4]
Artificial neural networks are pretty powerful.

Like circuits of NAND gates artificial neural networks can encode arbitrarily complex logic functions, thus allowing for universal computation.

But the power of neural networks really comes about by varying the weights such that one obtains some desired functionality.
neural network architectures

Neural networks with **multiple hidden layers** have been popularized as “**deep learning**” networks.
How to train a neural network?

- **quadratic cost function**

\[
C(\vec{w}, \vec{b}) = \frac{1}{2n} \sum_x ||y(x) - a(x)||^2
\]

Small adjustments on the level of a single neuron should result in small changes of the cost function.
How to train a neural network?

• **quadratic cost function**

\[
C(\vec{w}, \vec{b}) = \frac{1}{2n} \sum_{x} \| y(x) - a(x) \|^2
\]

• **back propagation** algorithm


extremely efficient way to calculate *all* partial derivatives

\[
\frac{\partial C}{\partial w}, \quad \frac{\partial C}{\partial b}
\]

needed for a **gradient descent** optimization.
Three flavors of machine learning

* thanks to Giuseppe Carleo (ETH Zurich) for some of the slides
Supervised Learning

- training with **labeled data**

\[
\{ (x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N) \}
\]

\[
C(w, b) = \frac{1}{2n} \sum_{i} C[y_i, F(x_i, w, b)]
\]

- stochastic **gradient descent**

\[
(w, b)' = (w, b) - \eta \cdot \nabla C[y_i, F(x_i, w, b)]
\]

converges to global minimum

(\sim\text{Langevin dynamics})
Example: digit recognition

Some 60 lines of code (Python/Julia) will do this for you with >95% accuracy.

![Example of neural network architecture](image)

- Input image: 28x28 pixels
- Input layer: 784 neurons
- Hidden layer: 100 neurons
- Output layer: 10 neurons
- Output: predicted digit

Labeled data for training

9 3 1 0 6
5 0 9 3 7

© Simon Trebst
Physics: phase classification

After training, 100% of configurations shown are correctly identified. Try to do it by eye instead ...

Unsupervised Learning

• training with unlabeled data

\[ \{ x_1, x_2, \ldots, x_N \} \]

Data point uniformly drawn from (unknown) distribution \( P(x) \)

\[ F(x, w, b) \sim P(x) \]

Goal: Find an approximation for the data distribution (to find correlations etc.)

• typical cost function

\[ D_{KL}(P||F) = \sum_i P(x_i) \log \frac{P(x_i)}{F(x_i)} \]

Kullback-Leibler divergence

Normalized probability (intractable)

\[ \nabla D_{KL}(P||F) = \langle G(x) \rangle_P - \langle G(x) \rangle_F \]

Gradient is difference between two expectation values (tractable with sampling, no need to know \( P \))
Example: forging hand writing

Cornell is the best place for a summer school.

Machine learning is fascinating.

Physicists are always skeptical.

Does it really work?

http://www.cs.toronto.edu/~graves/handwriting.cgi

unsupervised learning on different hand-writing styles

arbitrary sentences using different hand-writing styles
Physics: improve Monte Carlo moves

• unsupervised training

\[ P(x) \]

We want to sample efficiently from this probability distribution.

\[ F(x, w, b) \sim P(x) \]

We can learn \( P \), and perform standard cluster updates on \( F \).

• transition probabilities

\[ A(x \to x') = \min \left( 1, \frac{P(x')}{P(x)} \cdot \frac{F(x)}{F(x')} \right) \]

Use samples from machine as proposed configurations.

For perfectly learned \( F=P \), one always accepts move.
Reinforcement Learning

- generate data, obtain feedback, come up with strategy

\[ S[F] \]

“Scoring function” is a functional of the network

\[ \min_{F'} S[F] \]

Network generates/harvests data by some “strategy”

The best “strategy” obtains the best score

- learning

**Produce** meaningful input/output with \( F \)

**Feedback** from \( S \) (reinforcement stimulus)

**Adapt** the network accordingly
Example: game playing

after a short training period

after a few hours of training

Silver et al., Nature 529, 484 (2016)
(AlphaGo)
Preprocessing
Convolutional neural networks preprocess data by first looking for recurring patterns using small filters (and then sending it into a neural network).
Convolutional neural networks look for **recurring patterns** using small filters.
Convolutional neural networks look for **recurring patterns** using small filters.

*Slide filters* across image and create new image based on how well they fit.
Convolutional neural networks look for **recurring patterns** using small filters.
Convolutional neural networks have proved to be some of the most powerful ingredients for **pattern recognition**/machine learning.

![Convolutional Neural Networks Example](image)

**Discussion**

Our results show that a large, deep convolutional neural network is capable of achieving record-breaking results on a highly challenging dataset using purely supervised learning. It is notable that our network’s performance degrades if a single convolutional layer is removed. For example, removing any of the middle layers results in a loss of about 2% for the top-1 performance of the network. So the depth really is important for achieving our results.

To simplify our experiments, we did not use any unsupervised pre-training even though we expect that it will help, especially if we obtain enough computational power to significantly increase the size of the network without obtaining a corresponding increase in the amount of labeled data. Thus far, our results have improved as we have made our network larger and trained it longer but we still have many orders of magnitude to go in order to match the infero-temporal pathway of the human visual system. Ultimately we would like to use very large and deep convolutional nets on video sequences where the temporal structure provides very helpful information that is missing or far less obvious in static images.
Quantum loop topography is a physics preprocessor allowing to identify features associated with topological order in quantum many-body systems.

Quantum loop = sample of two-point operators that form loops.

\[ \tilde{P}_{jk} \tilde{P}_{kl} \tilde{P}_{lj} \]

\[ \tilde{P}_{jk} \equiv \langle c_j^\dagger c_k \rangle_{\alpha} \]
Quantum loop topography is a physics preprocessor allowing to identify features associated with topological order in quantum many-body systems.

- Training set

<table>
<thead>
<tr>
<th></th>
<th>( \kappa )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trivial</td>
<td>0.1</td>
</tr>
<tr>
<td>Topological</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Yi (Frank) Zhang and Eun-Ah Kim, PRL (2017)
Need for Speed
GPUs & open-source codes
Thanks!
Let’s take a break.