Deep learning is a branch of machine learning based on a set of algorithms that attempt to model high level abstractions in data by using a deep graph with multiple processing layers, composed of multiple linear and non-linear transformations.

Deep learning is part of a broader family of machine learning methods based on learning representations of data.
Deep learning

- Train multiple layers of features/abstractions from data.
- Try to discover representation that makes decisions easy.

"Cat"?

Deep Learning: train layers of features so that classifier works well.

Slide adapted from: Adam Coates

Deep learning for neural networks

Traditional NN models: 1-2 hidden layers
Deep learning NN models: 3+ hidden layers

Importance of features

Feature quality is critical to the performance of ML methods

Normal process: hand-crafted features
Deep learning: find algorithms to automatically discover features from the data

Challenges

What makes "deep learning" hard for NNs?
Challenges

What makes “deep learning” hard for NNs?

\[ w = w + \text{input} \times \Delta \text{output} \]

\[ \Delta \text{output} = f'(\text{output}) \times \sum \Delta \text{input} \]

Modified errors tend to get diluted as they get combined with many layers of weight corrections

Deep learning

Growing field

Driven by:
- Increase in data availability
- Increase in computational power
- Parallelizability of many of the algorithms

Involves more than just neural networks (though, they’re a very popular model)

word2vec

How many people have heard of it?

What is it?

Word representations

Wine data uses word occurrences as a feature

What does this miss?
Word representations

Wine data uses word occurrences as a feature

What does this miss?

“The wine had a dark red color” Zinfandel
“The wine was a deep crimson color” label?
“The wine was a deep yellow color” label?

Would like to recognize that words have similar meaning even though they aren’t lexically the same

Word representations

Key idea: project words into a multi-dimensional “meaning” space

word $\rightarrow [x_1, x_2, \ldots, x_d]$

The idea of word representations is not new:
• Co-occurrence matrices
• Latent Semantic Analysis (LSA)

New idea: learn word representation using a task-driven approach
A prediction problem

Given text, can generate lots of examples:

I like to eat bananas with cream cheese

Input

Prediction

___ like to eat

I ___ to eat bananas

I like ___ eat bananas with

I like to ___ bananas with cream

Use data like this to learn a distribution:

\[ p(\text{word} \mid \text{context}) \]

\[ p(w_i \mid w_{i-2}w_{i-1}w_{i+1}w_{i+2}) \]

Train a neural network on this problem

Encoding words

How can we input a “word” into a network?

“One-hot” encoding

For a vocabulary of V words, have V input nodes

All inputs are 0 except the one corresponding to the word

apple
banana
zebra

V nodes

apple
banana
zebra

0
0
1

0
0
0

“One-hot” encoding

For a vocabulary of V words, have V input nodes

All inputs are 0 except the one corresponding to the word

banana

apple

0
0
1

0
0
0
Another view

Training: backpropagation
Word representation

The weights for each word provide an N dimensional mapping of the word. Words that predict similarly should have similar weights.

Why does this work?

Results

\[ \text{vector}(\text{word1}) - \text{vector}(\text{word2}) = \text{vector}(\text{word3}) - X \]

word1 is to word2 as word3 is to X
Results

2-Dimensional projection of the N-dimensional space

Visualized

https://projector.tensorflow.org/

Continuous Bag Of Words

Other models: skip-gram
word2vec
A model for learning word representations from large amounts of data
Has become a popular pre-processing step for learning a more robust feature representation
Models like word2vec have also been incorporated into other learning approaches (e.g. translation tasks)

word2vec resources
https://blog.acolyer.org/2016/04/21/the-amazing-power-of-word-vectors/
https://code.google.com/archive/p/word2vec/
https://deeplearning4j.org/word2vec

Image classification
Input: pixels of the image
Output: what's in the image
Ideally: have some features that identify “parts”

Challenge: many different features
Challenge: many different features

Image kernels

Idea: learn kernels
Traditional NN approach doesn't work

The information of image is the pixel

If we’re dealing with a 512x512 RGB image, we have $512 \times 512 \times 3 = 786,432$ inputs

How many weights will we have with 5 hidden nodes?

For example, a 512x512 RGB image has $512 \times 512 \times 3 = 786,432$ and therefore $786,432$ weights in the next layer per neuron

Traditional NN approach doesn’t work

The information of image is the pixel

If we’re dealing with a 512x512 RGB image, we have $512 \times 512 \times 3 = 786,432$ inputs

786,432 weights per neuron = ~4M weights!

Convolutional Neural Networks (CNNs)

We’ll draw layers as blocks of nodes/inputs, e.g., 512 x 512 x 3
Locally connected image features are usually local.

Reduce the fully-connected network to locally-connected network.

For example, if we set window size 5, we only need $5 \times 5 \times 3 = 75$.

Hidden nodes

Apply across entire image
How many weights assuming: 512x512x3 images, 5 x 5 x 3 locally connected, and 5 hidden nodes?

Too many weights!

Despite only locally-connected, there are still too many weights

512x512x5 neurons in the next layer, we have 5x5x3 local connections = 98 million weights

Share weights:

All weights to a given hidden node are the same for the locally-connected edges

During classification, we treat it like we have different edges, just with the same weight

During training, we update the weights as normal except we update the same weights for a given hidden node

Solves the positional issue!
Parameter sharing

We share parameter in the same depth

Now we only have $75 \times 5 = 375$ weights

We call these layers "convolution layers".

What is learned can be considered as the convolution filters (like a kernel).

Pool layers

Convolution layers are often followed by pool layers

Reduce the weights without losing too much information

Another example

https://adamharley.com/nn_vis/