DEEP LEARNING

David Kauchak
CS158 – Spring 2022

1

Admin

Assignment 8

2

Deep learning

WIKIPEDIA

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.

3

Deep learning

Key: learning better features that abstract from the "raw" data

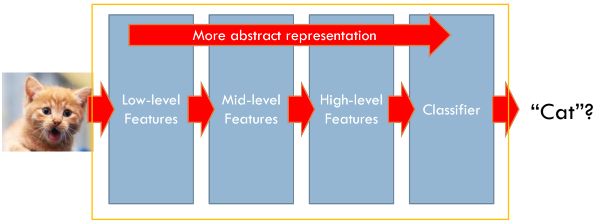
Using learned feature representations based on large amounts of data, generally unsupervised

Using classifiers with multiple layers of learning

4

Deep learning

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



More abstract representation

Low-level Features

Mid-level Features

High-level Features

Classifier

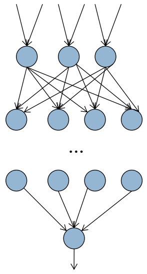
"Cat"?

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

Slide adapted from: Adam Coates

5

Deep learning for neural networks



Traditional NN models: 1-2 hidden layers

Deep learning NN models: 3+ hidden layers

6

Importance of features

Feature quality is critical to the performance of ML methods

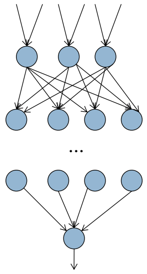
Normal process = hand-crafted features

Deep learning: find algorithms to automatically discover features from the data

7

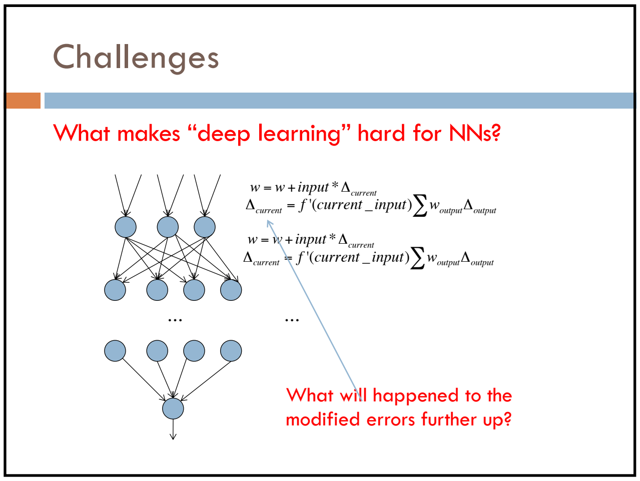
Challenges

What makes "deep learning" hard for NNs?



$$w = w + input * \Delta_{current}$$
$$\Delta_{current} = f'(current\_input) \sum w_{output} \Delta_{output}$$

8



Challenges

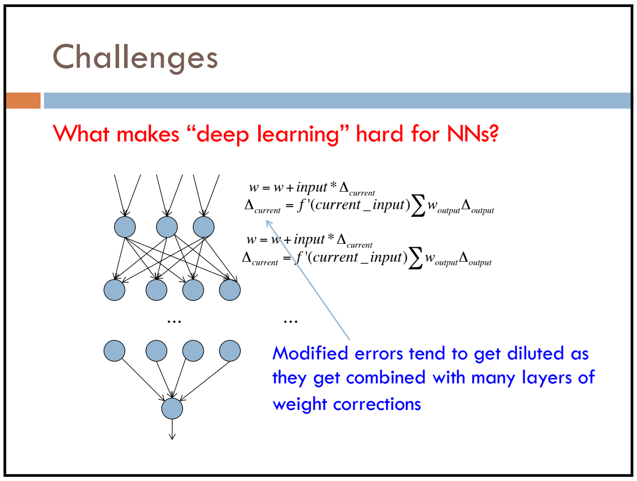
What makes "deep learning" hard for NNs?

$$w = w + input * \Delta_{current}$$
$$\Delta_{current} = f'(current\_input)\sum w_{output}\Delta_{output}$$
$$w = w + input * \Delta_{current}$$
$$\Delta_{current} = f'(current\_input)\sum w_{output}\Delta_{output}$$

…

What will happened to the modified errors further up?

9



Challenges

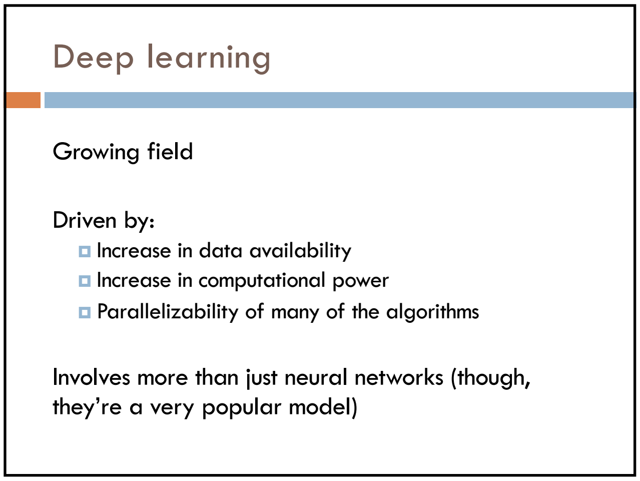
What makes "deep learning" hard for NNs?

$$w = w + input * \Delta_{current}$$
$$\Delta_{current} = f'(current\_input)\sum w_{output}\Delta_{output}$$
$$w = w + input * \Delta_{current}$$
$$\Delta_{current} = f'(current\_input)\sum w_{output}\Delta_{output}$$

…

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

10



Deep learning

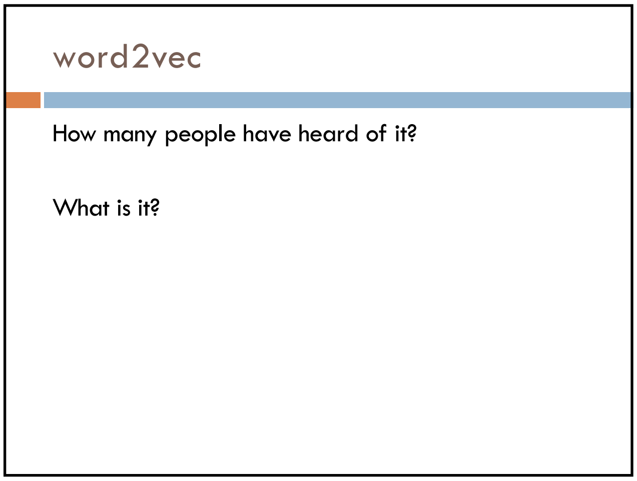
Growing field

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)

11

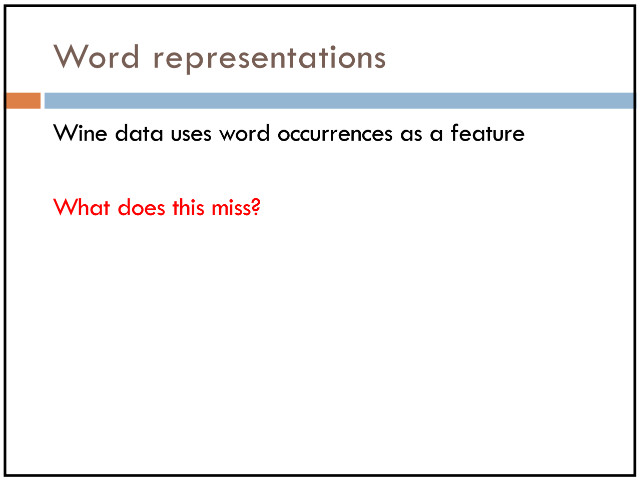


word2vec

How many people have heard of it?

What is it?

12

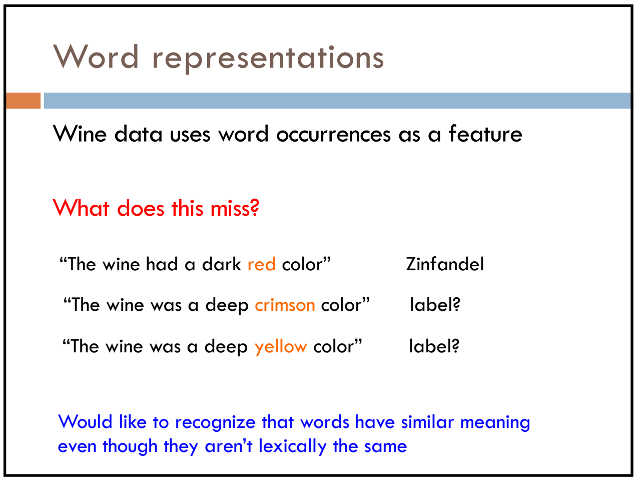


Word representations

Wine data uses word occurrences as a feature

What does this miss?

13



Word representations

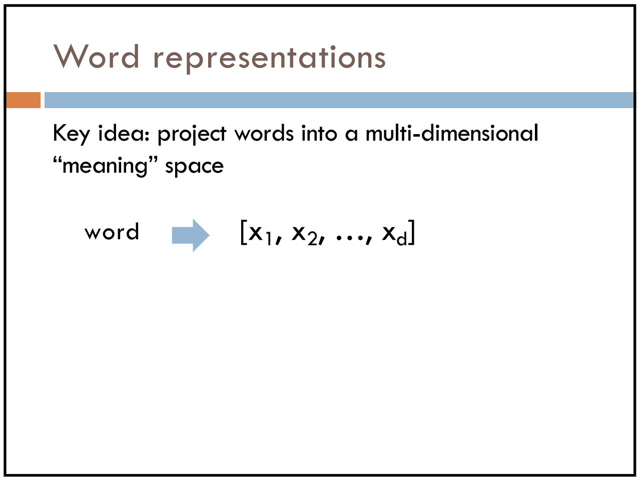
Wine data uses word occurrences as a feature

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

14

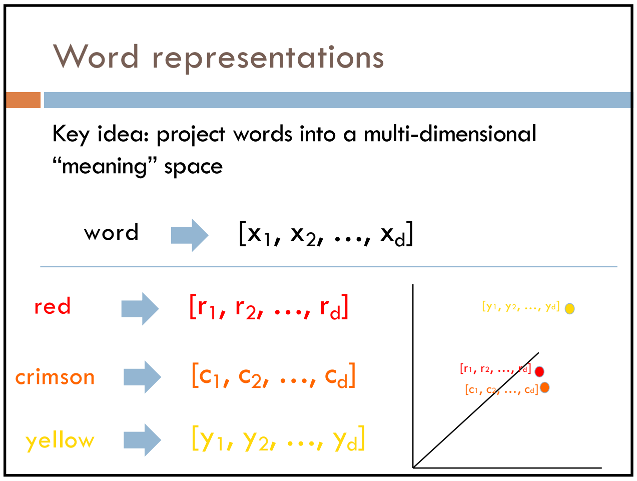


Word representations

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

word → $[x_1, x_2, \ldots, x_d]$

15



Word representations

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

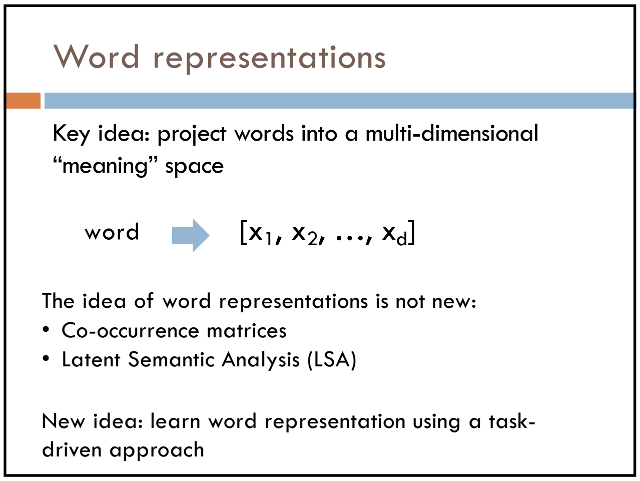
word → $[x_1, x_2, \ldots, x_d]$

red → $[r_1, r_2, \ldots, r_d]$

crimson → $[c_1, c_2, \ldots, c_d]$

yellow → $[y_1, y_2, \ldots, y_d]$

16



Word representations

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

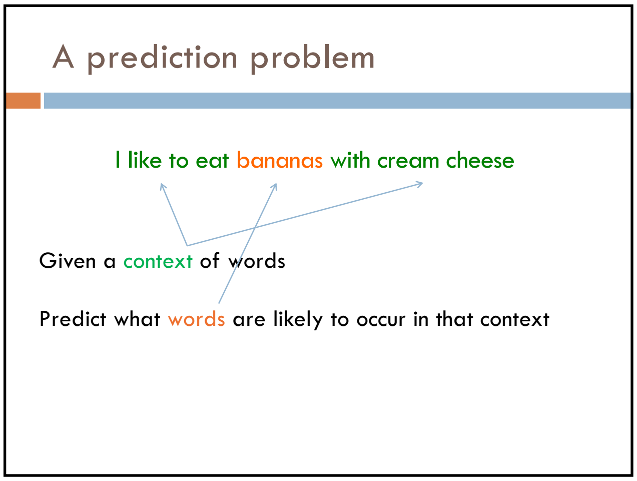
word ➡ $[x_1, x_2, \ldots, x_d]$

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

17



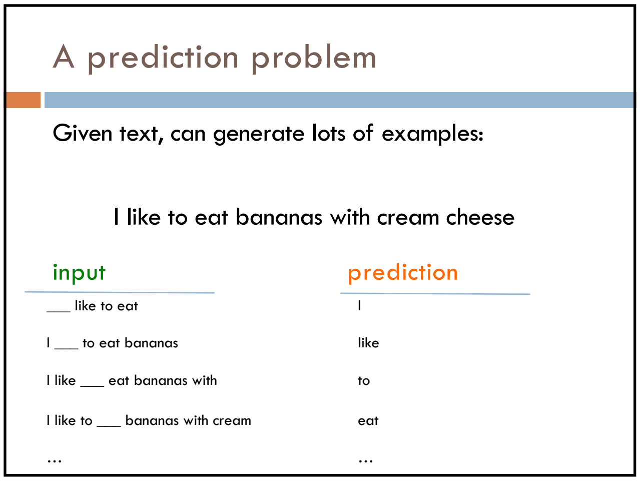
A prediction problem

I like to eat bananas with cream cheese

Given a context of words

Predict what words are likely to occur in that context

18



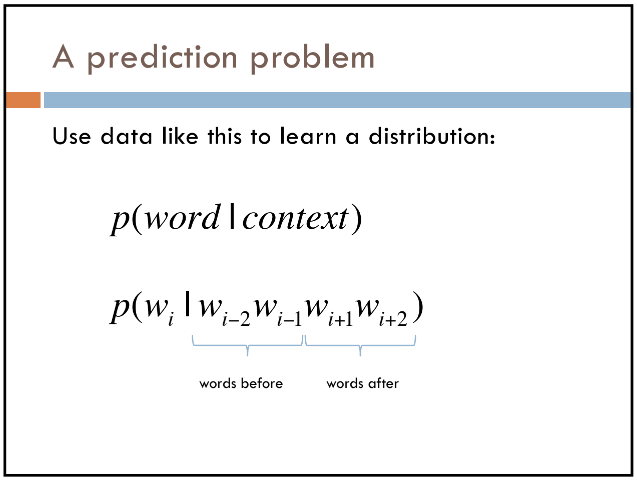
A prediction problem

Given text, can generate lots of examples:

I like to eat bananas with cream cheese

input	prediction
___ like to eat	I
I ___ to eat bananas	like
I like ___ eat bananas with	to
I like to ___ bananas with cream	eat
…	…

19



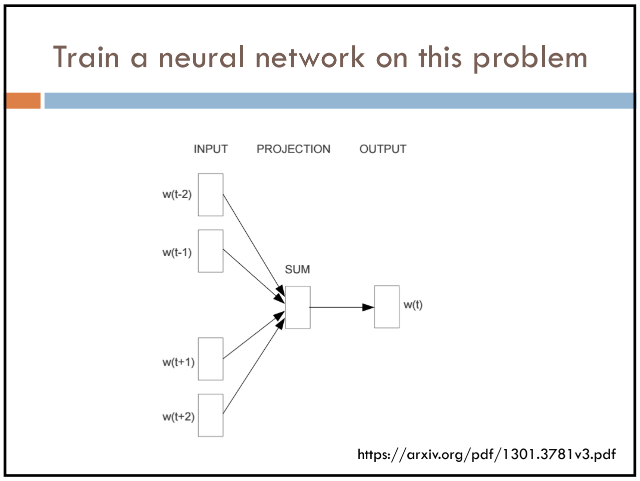
A prediction problem

Use data like this to learn a distribution:

$$p(word \mid context)$$

$$p(w_i \mid \underbrace{w_{i-2}w_{i-1}}_{\text{words before}}\underbrace{w_{i+1}w_{i+2}}_{\text{words after}})$$

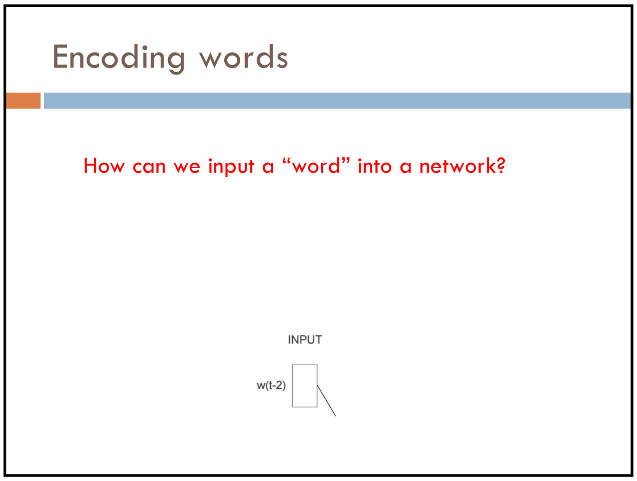
20



Train a neural network on this problem

https://arxiv.org/pdf/1301.3781v3.pdf

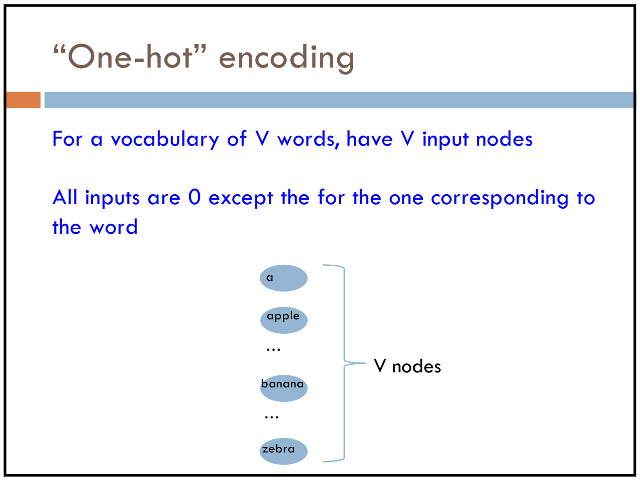
21



Encoding words

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

22

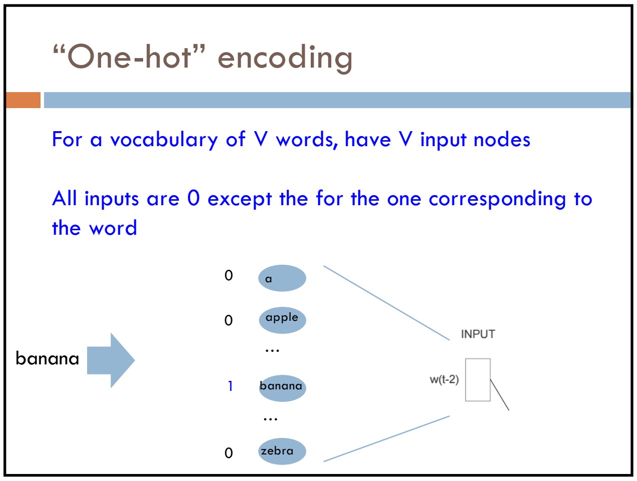


“One-hot” encoding

For a vocabulary of V words, have V input nodes

All inputs are 0 except the for the one corresponding to the word

23

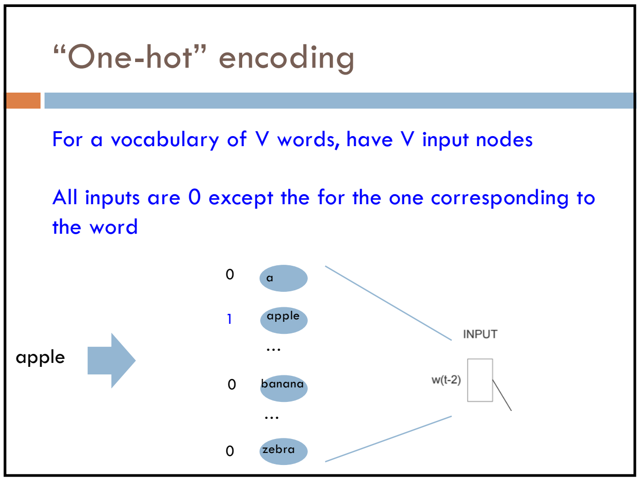


“One-hot” encoding

For a vocabulary of V words, have V input nodes

All inputs are 0 except the for the one corresponding to the word

24

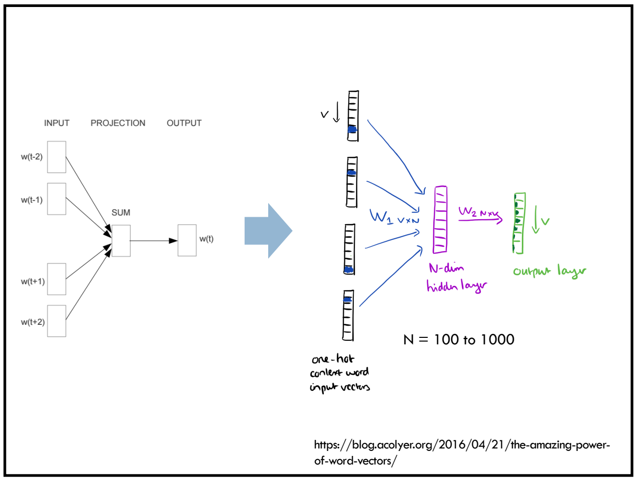


“One-hot” encoding

For a vocabulary of V words, have V input nodes

All inputs are 0 except the for the one corresponding to the word

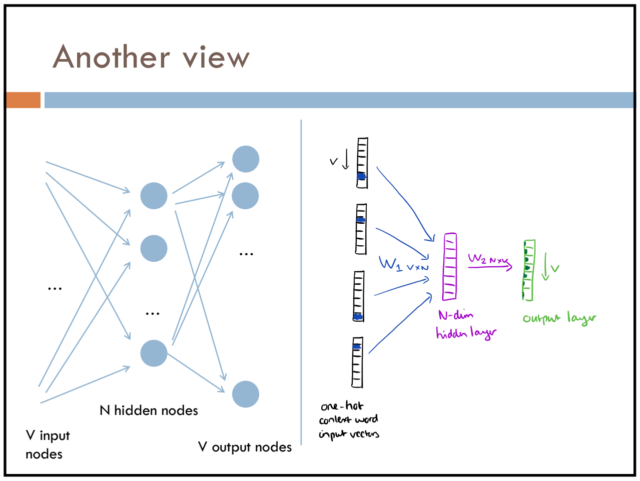
25



N = 100 to 1000

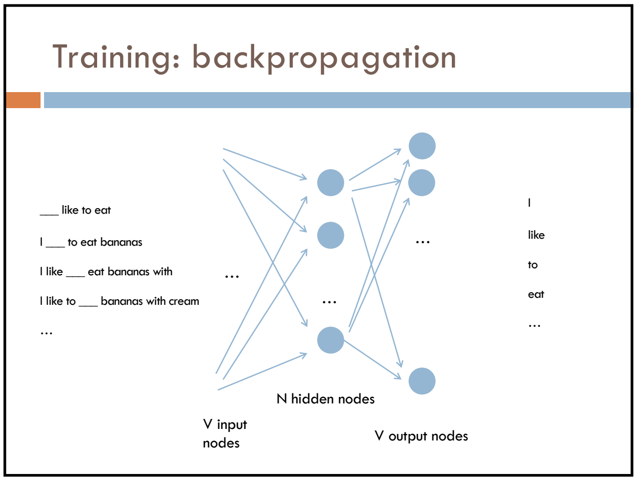
https://blog.acolyer.org/2016/04/21/the-amazing-power-of-word-vectors/

26



Another view

27



Training: backpropagation

___ like to eat

I ___ to eat bananas

I like ___ eat bananas with

I like to ___ bananas with cream

…

I

like

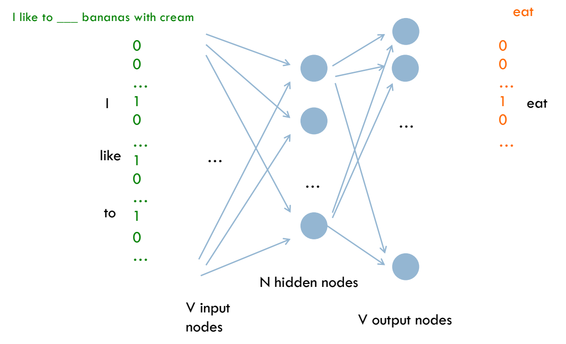
to

eat

…

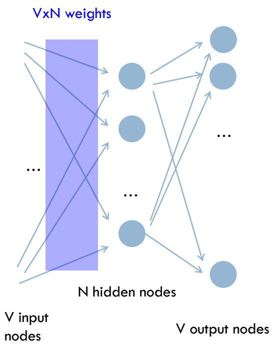
28

Training: backpropagation



29

Word representation



The weights for each word provide an N dimensional mapping of the word

Words that predict similarly should have similar weights

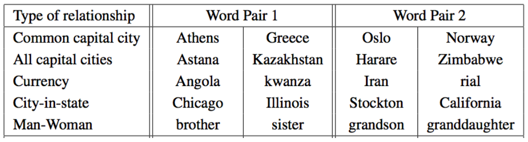
Why does this work?

30

Results

vector(word1) – vector(word2) = vector(word3) - X

word1 is to word2 as word3 is to X



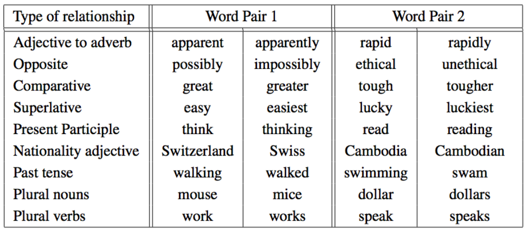
Type of relationship	Word Pair 1		Word Pair 2	
Common capital city	Athens	Greece	Oslo	Norway
All capital cities	Astana	Kazakhstan	Harare	Zimbabwe
Currency	Angola	kwanza	Iran	rial
City-in-state	Chicago	Illinois	Stockton	California
Man-Woman	brother	sister	grandson	granddaughter

31

Results

vector(word1) – vector(word2) = vector(word3) - X

word1 is to word2 as word3 is to X



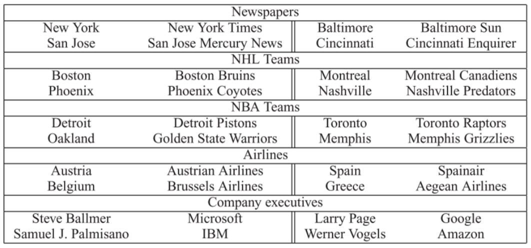
Type of relationship	Word Pair 1		Word Pair 2	
Adjective to adverb	apparent	apparently	rapid	rapidly
Opposite	possibly	impossibly	ethical	unethical
Comparative	great	greater	tough	tougher
Superlative	easy	easiest	lucky	luckiest
Present Participle	think	thinking	read	reading
Nationality adjective	Switzerland	Swiss	Cambodia	Cambodian
Past tense	walking	walked	swimming	swam
Plural nouns	mouse	mice	dollar	dollars
Plural verbs	work	works	speak	speaks

32

Results

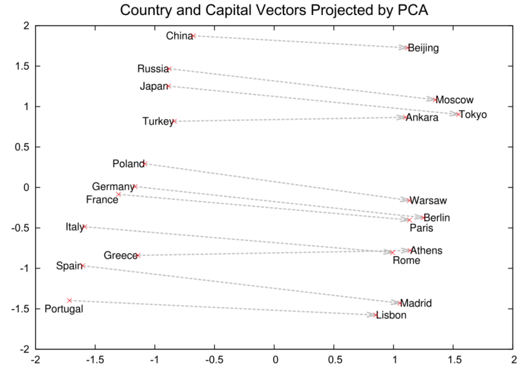
vector(word1) – vector(word2) = vector(word3) - X

word1 is to word2 as word3 is to X



Newspapers			
New York San Jose	New York Times San Jose Mercury News	Baltimore Cincinnati	Baltimore Sun Cincinnati Enquirer
NHL Teams			
Boston Phoenix	Boston Bruins Phoenix Coyotes	Montreal Nashville	Montreal Canadiens Nashville Predators
NBA Teams			
Detroit Oakland	Detroit Pistons Golden State Warriors	Toronto Memphis	Toronto Raptors Memphis Grizzlies
Airlines			
Austria Belgium	Austrian Airlines Brussels Airlines	Spain Greece	Spainair Aegean Airlines
Company executives			
Steve Ballmer Samuel J. Palmisano	Microsoft IBM	Larry Page Werner Vogels	Google Amazon

33



2-Dimensional projection of the N-dimensional space

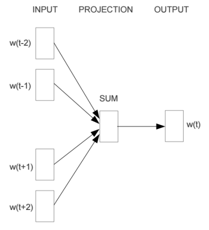
34

Visualized

https://projector.tensorflow.org/

35

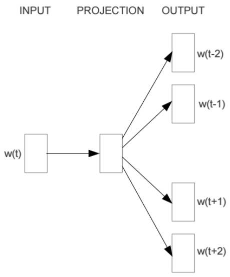
Continuous Bag Of Words



CBOW

36

Other models: skip-gram



37

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)

38

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

https://arxiv.org/pdf/1301.3781v3.pdf

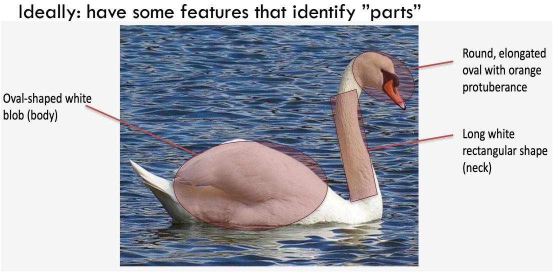
39

Image classification

Input: pixels of the image

Output: what's in the image

Ideally: have some features that identify "parts"


Oval-shaped white blob (body)
Round, elongated oval with orange protuberance
Long white rectangular shape (neck)

https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac

40

Challenge: many different features

Round, elongated head with orange or black beak

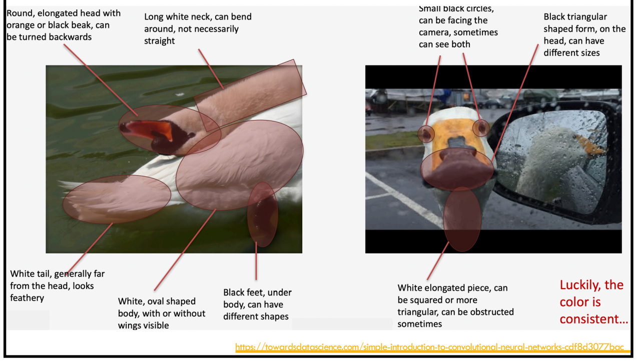
Long white neck, square shape

Oval-shaped white body with or without large white symmetric blobs (wings)

https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac

41

Challenge: many different features



Round, elongated head with orange or black beak, can be turned backwards

Long white neck, can bend around, not necessarily straight

Small black circles, can be facing the camera, sometimes can see both

Black triangular shaped form, on the head, can have different sizes

White tail, generally far from the head, looks feathery

White, oval shaped body, with or without wings visible

Black feet, under body, can have different shapes

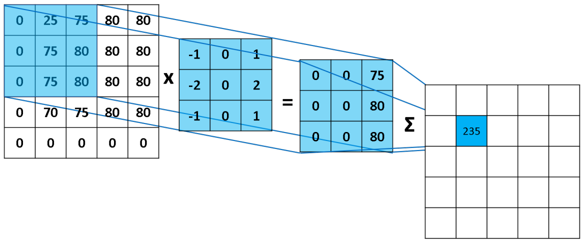
White elongated piece, can be squared or more triangular, can be obstructed sometimes

Luckily, the color is consistent…

https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac

42

Image kernels



0	25	75	80	80
0	75	80	80	80
0	75	80	80	80
0	70	75	80	80
0	0	0	0	0

x

-1	0	1
-2	0	2
-1	0	1

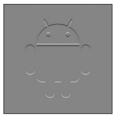
=

0	0	75
0	0	80
0	0	80

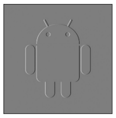
Σ

	235			

43

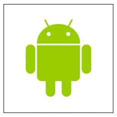


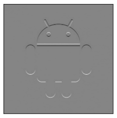
-1	-2	-1
0	0	0
1	2	1



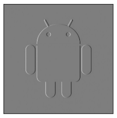
-1	0	1
-2	0	2
-1	0	1

44





Idea: learn kernels

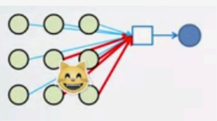


-1	-2	-1
0	0	0
1	2	1

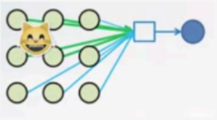
-1	0	1
-2	0	2
-1	0	1

45

Traditional NN approach doesn't work



In this case, the red weights will be modified to better recognize cats



In this case, the green weights will be modified.

https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac

46

Traditional NN approach doesn't work

The information of image is the pixel

If we're dealing with a 512x512 RGB image, we have 512x512x3 = 786,432 inputs

How many weights will we have with 5 hidden nodes?

For example, a 512x512 RGB image has 512x512x3 = 786,432 and therefore 786,432 weights in the next layer **per neuron**

47

Traditional NN approach doesn't work

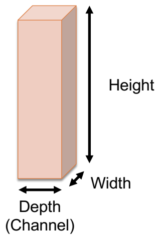
The information of image is the pixel

If we're dealing with a 512x512 RGB image, we have 512x512x3 = 786,432 inputs

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

48

Convolutional Neural Networks (CNNs)



We'll draw layers as blocks of nodes/inputs, e.g., 512 x 512 x 3

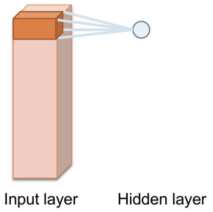
Locally connected

image features are usually local

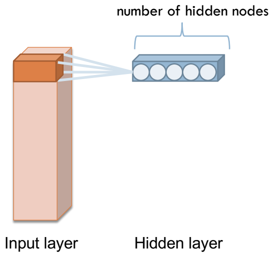
reduce the fully-connected network to locally-connected network.

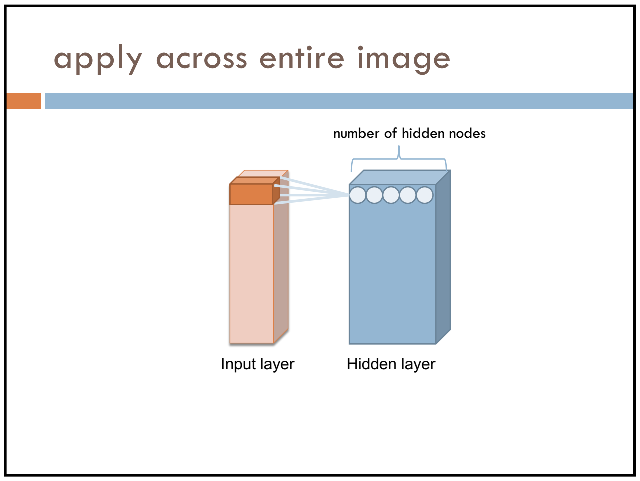
For example, if we set window size 5, we only need 5x5x3 = 75

fully connected -> locally connected



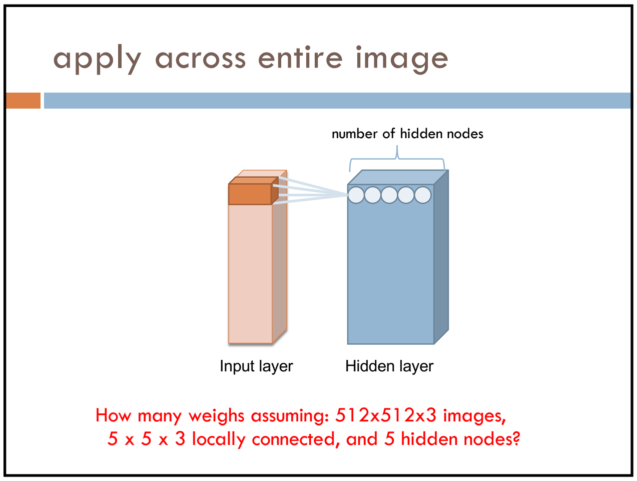
hidden nodes





apply across entire image

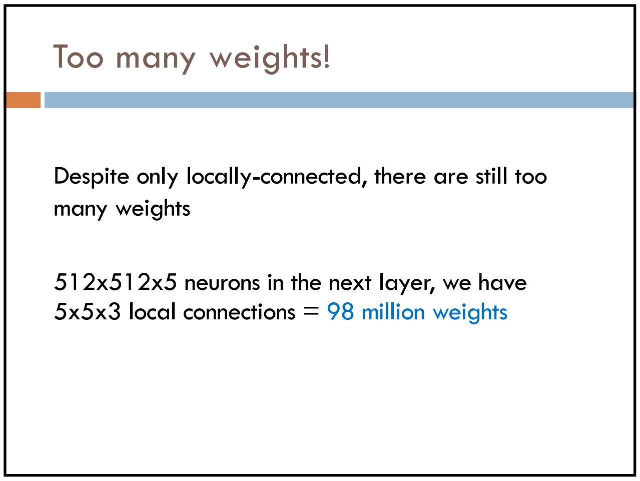
53



apply across entire image

How many weighs assuming: 512x512x3 images,
5 x 5 x 3 locally connected, and 5 hidden nodes?

54

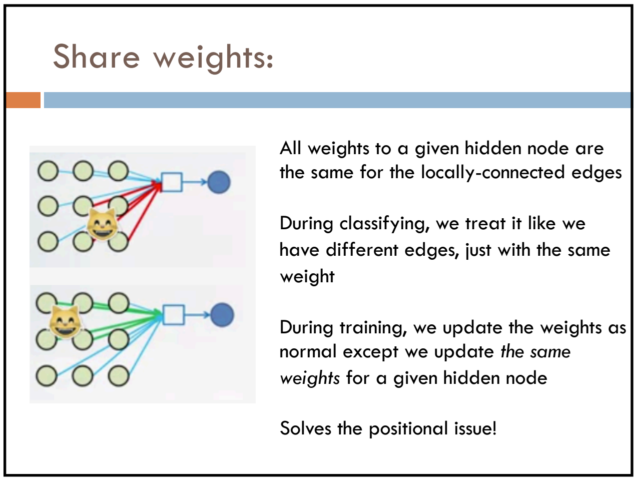


Too many weights!

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

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

55



Share weights:

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

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

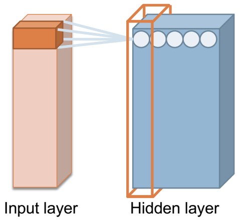
During training, we update the weights as normal except we update *the same weights* for a given hidden node

Solves the positional issue!

56

Share weights

We share parameter in the same depth.



57

Parameter sharing

We share parameter in the same depth

Now we only have 75x5=375 weights

We call these layers "convolution layers".

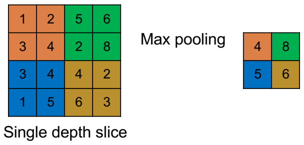
What is learned can be considered as the convolution filters (like a kernel)

58

Pool layers

Convolution layers are often followed by pool layers

Reduce the weights without losing too much information



1	2	5	6
3	4	2	8
3	4	4	2
1	5	6	3

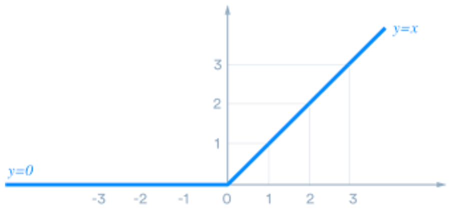
Max pooling

4	8
5	6

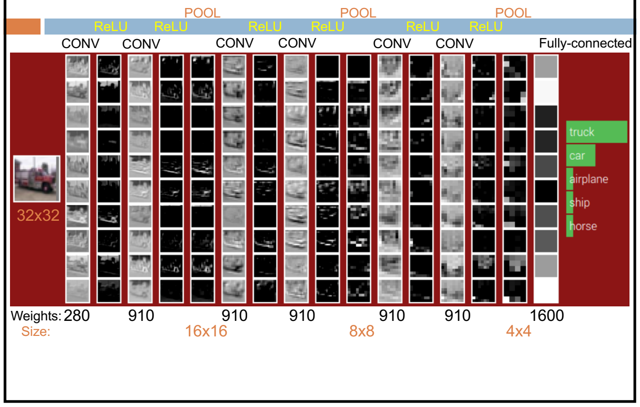
Single depth slice

59

Rectified Linear Unit (ReLU)



60



61

Another example

http://scs.ryerson.ca/~aharley/vis/conv/

62

Next class

63