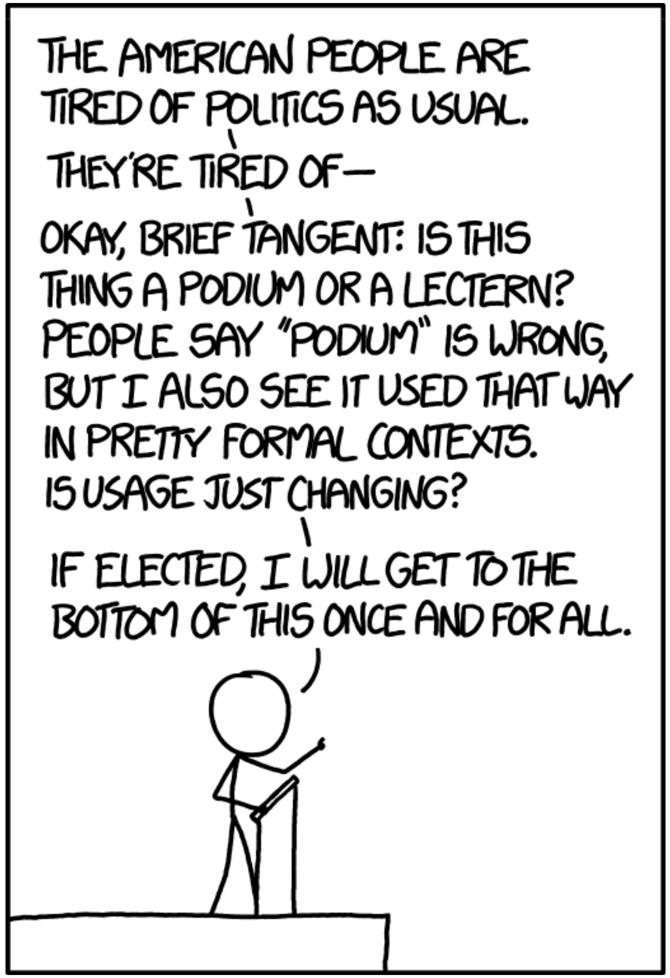
COMP 4446 / 5046
Natural Language Processing

Lecture 2: Word Embeddings and Representation

Dr. Jonathan Kay Kummerfeld

Semester 1, 2023
School of Computer Science,
University of Sydney

Podium



[BREAKING: Senator's bold pro-podium stand leads to primary challenge from prescriptivist base.]

Source: https://xkcd.com/1661/

0 LECTURE PLAN

Lecture 2: Word Embeddings and Representation

1. **Lab Info**
2. Previous Lecture Review
    1. One-Hot Vectors
    2. Bag of Words and TF-IDF
3. Prediction based Word Representation
    1. Introduction to the concept ‘Prediction’
    2. Word2Vec
    3. FastText
    4. GloVe
4. Next Week Preview

0 LECTURE PLAN

Consultative Group

Thanks to volunteers!

Email to come this week to schedule a time

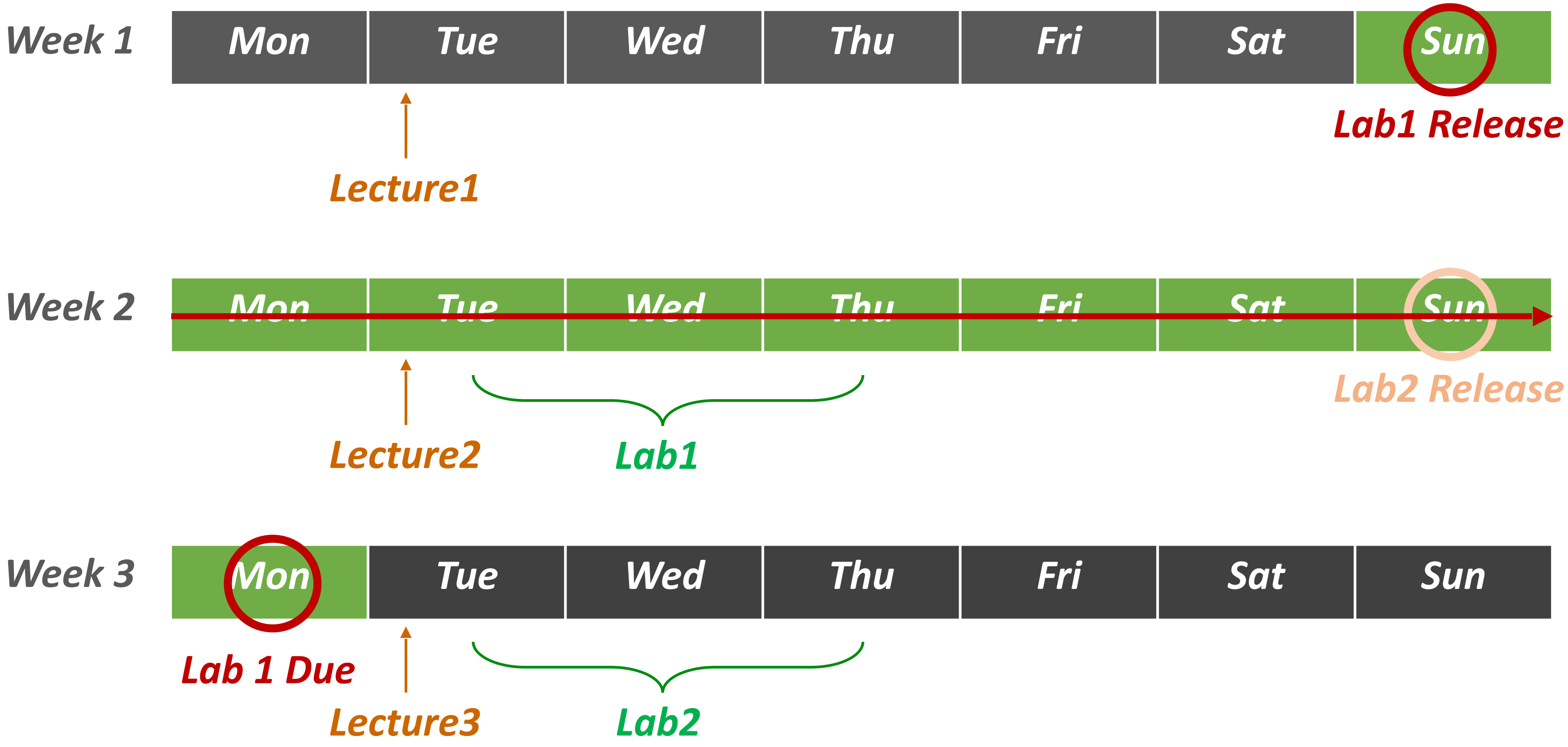
1 Info: Lab Exercise

How to Submit

Submit an **“ipynb” file** to Canvas.

When and Where to Submit

Submit Lab 2 (for Week 2) by ***Week 3 Monday 11:59PM.***



0 LECTURE PLAN

Lecture 2: Word Embeddings and Representation

1. Lab Info
2. **Count-based Word Representation**
   1. One-Hot Vectors
   2. Bag of Words and TF-IDF
3. Prediction based Word Representation
   1. Introduction to the concept ‘Prediction’
   2. Word2Vec
   3. FastText
   4. GloVe
4. Next Week Preview

2 WORD REPRESENTATION

motel = [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 … 0]

hotel = [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 … 0]

Inn   = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 … 1]

**Unicode (utf-8)*

2 COUNT based WORD REPRESENTATION

Problem: No word similarity representation

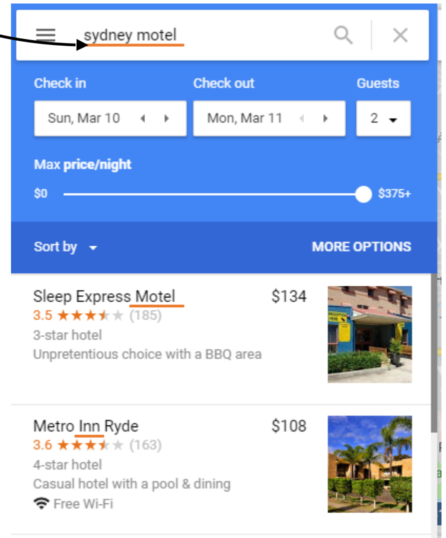
Example: in web search, if the user searches for “Sydney motel”, we would like to match documents containing “Sydney Inn”

hotel *motel* *Inn*

motel = [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 … 0]

hotel = [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 … 0]

Inn = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 … 1]



There is no natural notion of similarity for one-hot vectors!

2 COUNT based WORD REPRESENTATION

Bag of Words (BOW)

How are you

How have you been

Nice to see you

Have a nice day

2 COUNT based WORD REPRESENTATION

Term Frequency-Inverse Document Frequency

- Term Frequency-Inverse Document Frequency (TF-IDF) is a way of representing ***how important a word*** *is to a document in a collection or corpus*.

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

$w_{i,j}$ = weight of term i in document j

$tf_{i,j}$ = number of occurrences of term i in document j

N = total number of documents

df_i = number of documents containing term i

- The ***Term Frequency*** is a count of how many times a word occurs in a given document
- The ***Document Frequency*** is the number of times a word occurs in a corpus of documents

Limitations of Term Frequency Inverse Document Frequency

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

$\leftarrow$ *1+df*$_i$

w$_{i,j}$ *= weight of term i in document j*

tf$_{i,j}$ *= number of occurrences of term i in document j*

N = total number of documents

df$_i$ *= number of documents containing term i*

- It computes document similarity directly in the word-count space, which may be slow for large vocabularies (though it is easy to parallelise).
- It assumes that the counts of different words provides independent evidence of similarity.
- It makes no use of ***semantic similarities between words***.

2 COUNT based WORD REPRESENTATION

Sparse Representation

With **COUNT based word representation**, linguistic information was represented with **sparse representations** (ie., most of the dimensions are 0)

hotel *motel* *Inn*

motel = [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 … 0]

hotel = [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 … 0]

Inn = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 … 1]

2 COUNT based WORD REPRESENTATION

Sparse Representation

With **COUNT based word representation**, linguistic information was represented with **sparse representations** (ie., most of the dimensions are 0)

hotel *motel* *Inn*

motel = [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 … 0]

hotel = [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 … 0]

Inn = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 … 1]

Can we do better?

1. Can we get a low-dimensional vector representation?
2. Can the vectors better represent word similarity?

Can we use <u>a list of fixed numbers (properties)</u> to represent the word?

maybe a low-dimensional vector?

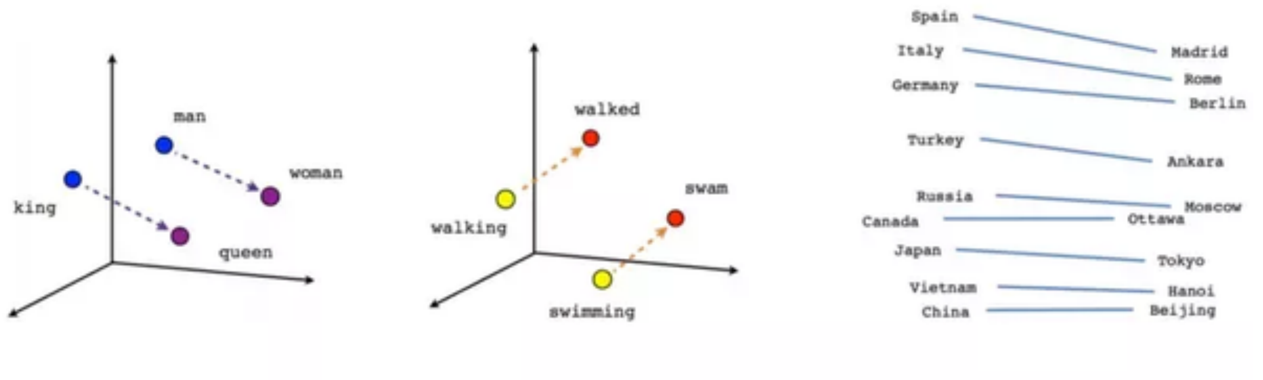
0 LECTURE PLAN

Lecture 2: Word Embeddings and Representation

1. Lab Info
2. Previous Lecture Review
    1. One-Hot Vectors
    2. Bag of Words and TF-IDF
3. **Prediction based Word Representation**
    1. Word Embedding
    2. Word2Vec
    3. FastText
    4. Glove
4. Next Week Preview

3 Prediction based Word representation

Dense vectors and how they capture similarity!



Male-Female　　Verb tense　　Country-Capital

Word Algebra

Enter all three words, the first two, or the last two and see the words that result.

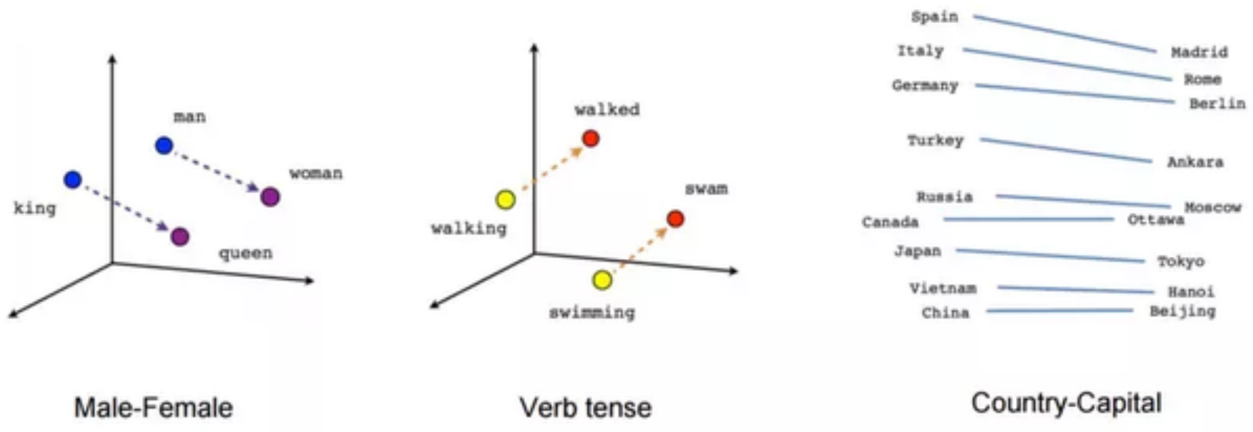
shanghai + (australia - sydney) = Get result

china 0.7477672216910414

Reference: http://turbomaze.github.io/word2vecjson/

3 Prediction based Word representation

Where do these word representations come from?



We want to…

1. Have a fixed low-dimensional vector representation
2. That represent the word similarity

maybe a low-dimensional vector?

What if we use <u>a list of fixed numbers (properties)</u> to represent the word?

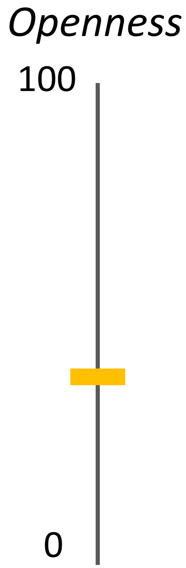
3 Prediction based Word representation

Let’s get familiar with using vectors to represent things

Assume that you are taking a personality test (the Big Five Personality Traits test)

1)Openness, 2)Agreeableness, 3)Conscientiousness, 4)Negative emotionality, 5)Extraversion

Jane



https://openpsychometrics.org/tests/IPIP-BFFM/

3 Prediction based Word representation

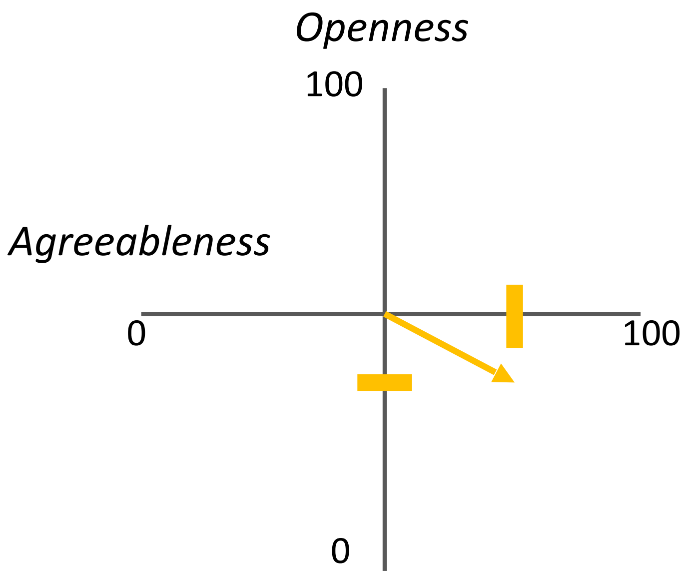
Let’s get familiar with using vectors to represent things

Assume that you are taking a personality test (the Big Five Personality Traits test)

1)Openness, 2)Agreeableness, 3)Conscientiousness, 4)Negative emotionality, 5)Extraversion

Jane

Openness	Agreeableness			
40	70			

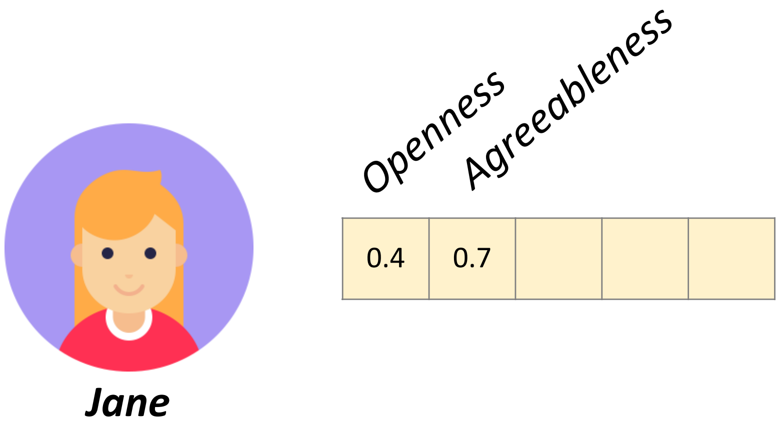


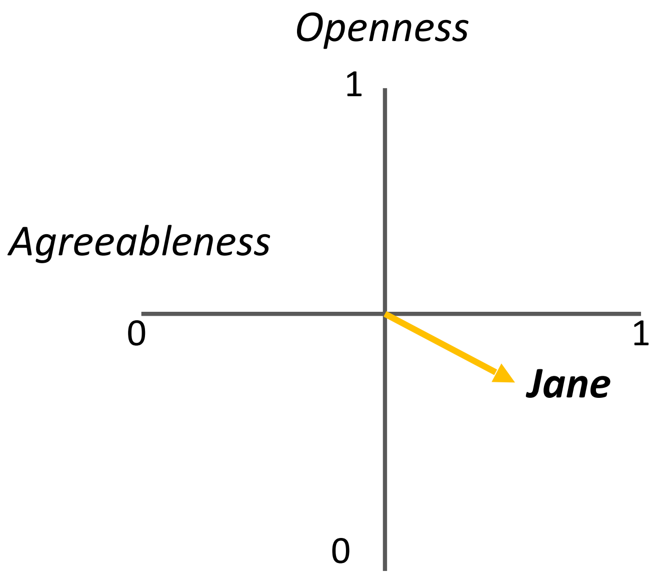
3 Prediction based Word representation

Let’s get familiar with using vectors to represent things

Assume that you are taking a personality test (the Big Five Personality Traits test)

1)Openness, 2)Agreeableness, 3)Conscientiousness, 4)Negative emotionality, 5)Extraversion





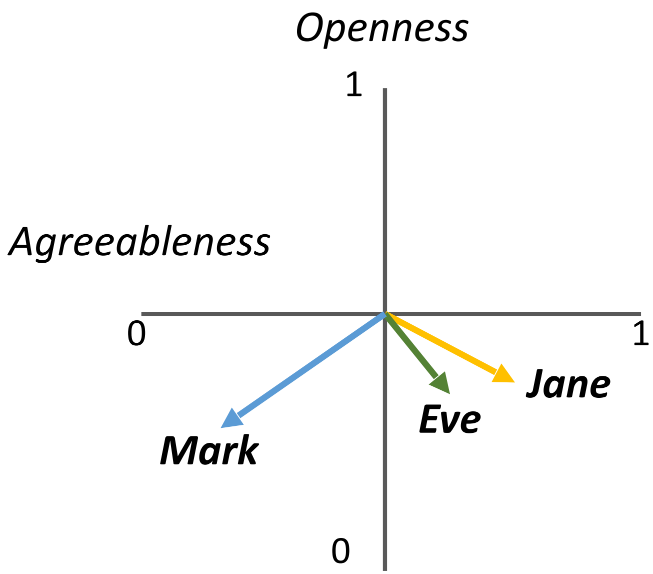
3 Prediction based Word representation

Let’s get familiar with using vectors to represent things

Assume that you are taking a personality test (the Big Five Personality Traits test)

1)Openness, 2)Agreeableness, 3)Conscientiousness, 4)Negative emotionality, 5)Extraversion

	Openness	*Agreeableness*			
Jane	0.4	0.7			
Mark	0.3	0.2			
Eve	0.4	0.6			



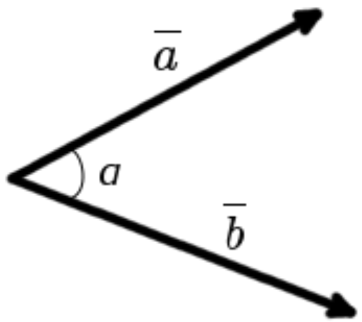
3 Prediction based Word representation

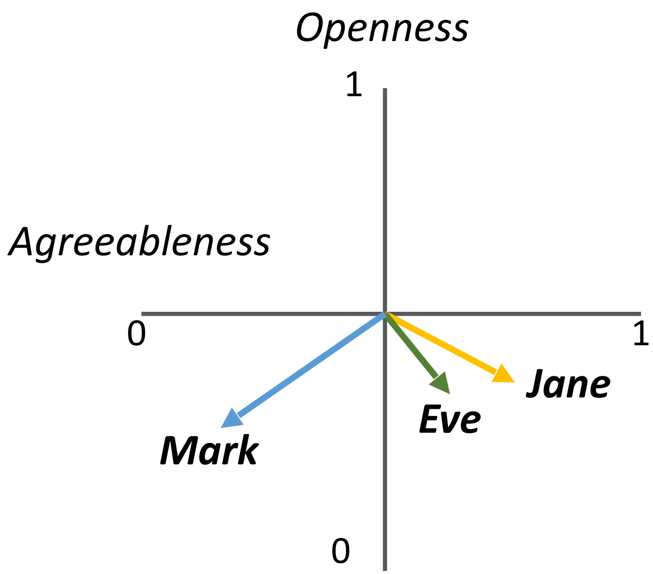
Let's get familiar with using vectors to represent things

Which of the other two people (Mark or Eve) is more similar to Jane?

Cosine Similarity

Measure of similarity between two vectors



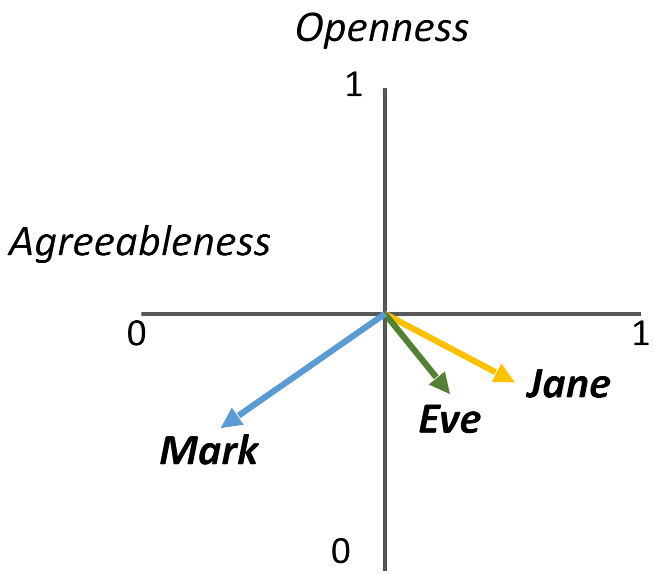


$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

Let’s get familiar with using vectors to represent things

Which of the other two people (Mark or Eve) is more similar to Jane?

	Openness	Agreeableness			
Jane	0.4	0.7			
Mark	0.3	0.2			
Eve	0.4	0.6			



$$\cos(\underbrace{[0.4,\ 0.7]}_{Jane},\ \underbrace{[0.3,\ 0.2]}_{Mark}) \approx 0.89$$

$$\cos(\underbrace{[0.4,\ 0.7]}_{Jane},\ \underbrace{[0.4,\ 0.6]}_{Eve}) \approx 0.99$$

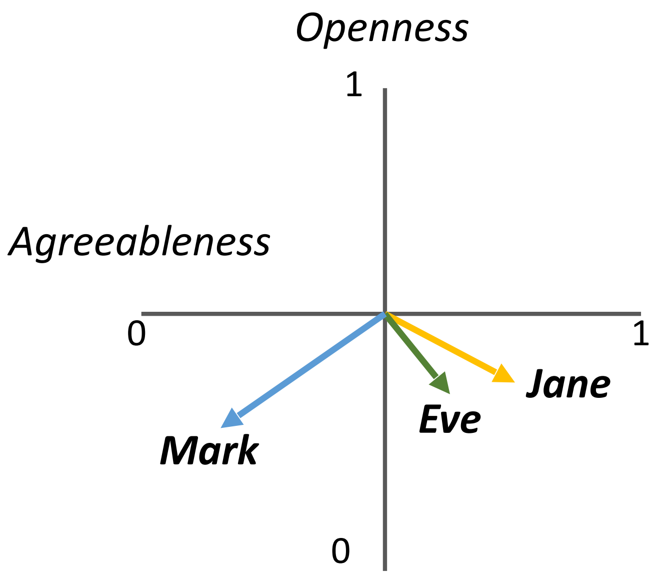
https://onlinemschool.com/math/assistance/vector/angl/
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.pairwise.cosine_similarity.html

3 Prediction based Word representation

Let’s get familiar with using vectors to represent things

We can extend the same idea from two dimensions to five

	Openness	Agreeableness	Conscientiousness	NE	Extraversion
Jane	0.4	0.7	0.5	0.2	0.1
Mark	0.3	0.2	0.3	0.7	0.2
Eve	0.4	0.6	0.4	0.3	0.5



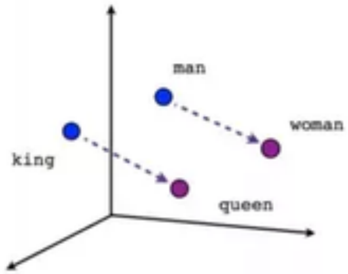
With these embeddings, we

1.Represent things as vectors of numbers!

2.Easily calculate the similarity between vectors

3 Prediction based Word representation

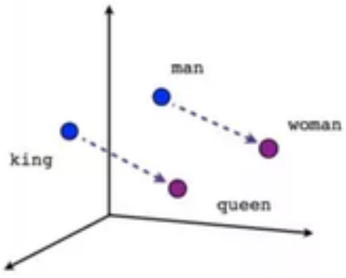
Remember? The Word2Vec Demo!



This is a word embedding for the word "king"

Prediction based Word representation

Remember? The Word2Vec Demo!



This is a word embedding for the word "king"

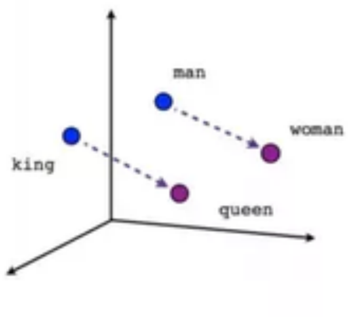
** Trained using Wikipedia Data, a 50-dimension Vector (with GloVe rathe than Word2Vec)*

king

[0.50451, 0.68607, -0.59517, -0.022801, 0.60046, 0.08813, 0.47377, -0.61798, -0.31012, -0.066666, 1.493, -0.034173, -0.98173, 0.68229, 0.812229, 0.81722, -0.51722, -744.5.4 1503, -0.55809, 0.66421, 0.1961, -0.1495, -0.033474, -0.30344, 0.41177, -2.223, -1.0756, -0.343554, 0.33505, 1.9927, -0.042434, -0.64519, 0.72519, 0.71419, 0.714319, 0.71419 9159, 0.16754, 0.34344, -0.25663, -0.8523, 0.1661, 0.40102, 1.1685, -1.0137, -0.2155, 0.78321, -0.91241, -1.6626, -0.64426, -0.542102]

3 Prediction based Word representation

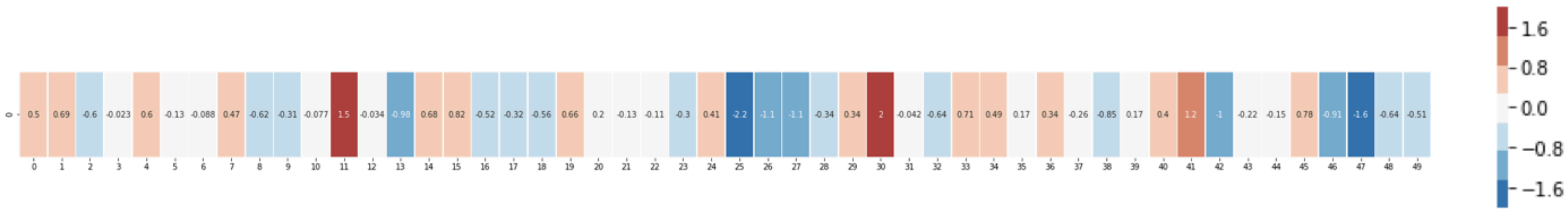
Remember? The Word2Vec Demo!



This is a word embedding for the word "king"

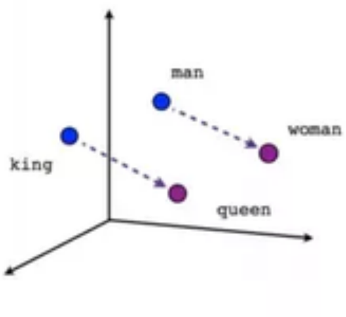
** Trained using Wikipedia Data, a 50-dimension Vector (with GloVe rathe than Word2Vec)*

king

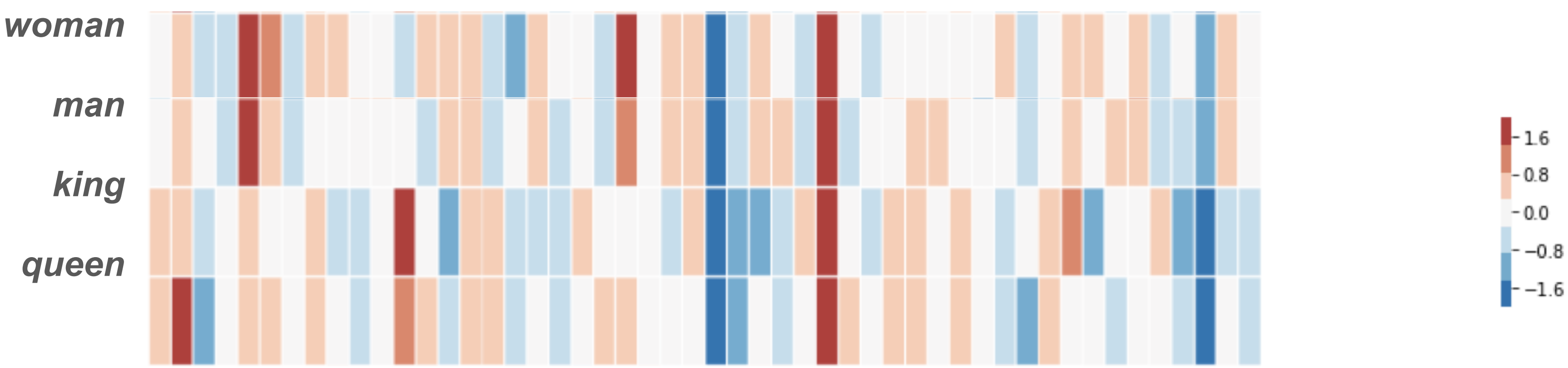


3 Prediction based Word representation

Remember? The Word2Vec Demo!

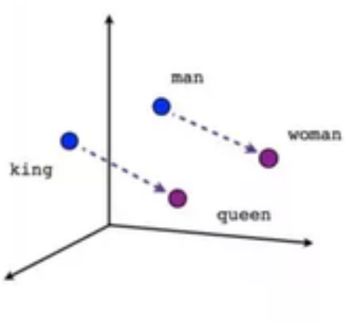


Compare with Woman, Man, King, and Queen

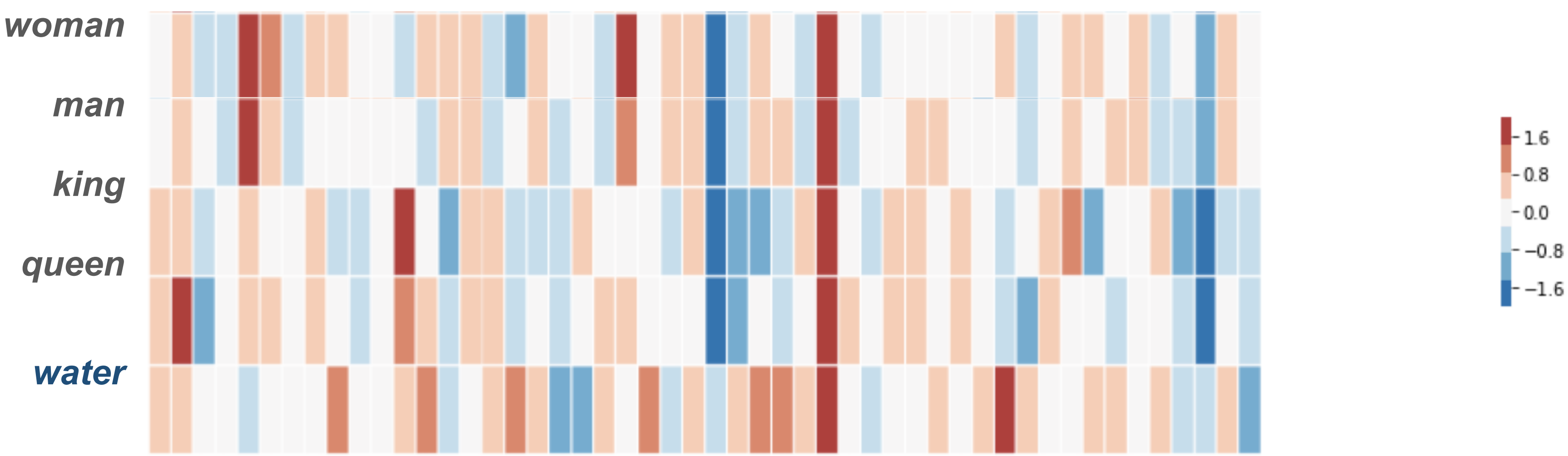


3 Prediction based Word representation

Remember? The Word2Vec Demo!

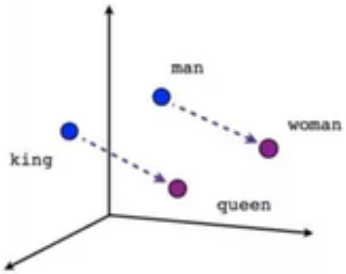


Compare with Woman, Man, King, Queen, and Water



3 Prediction based Word representation

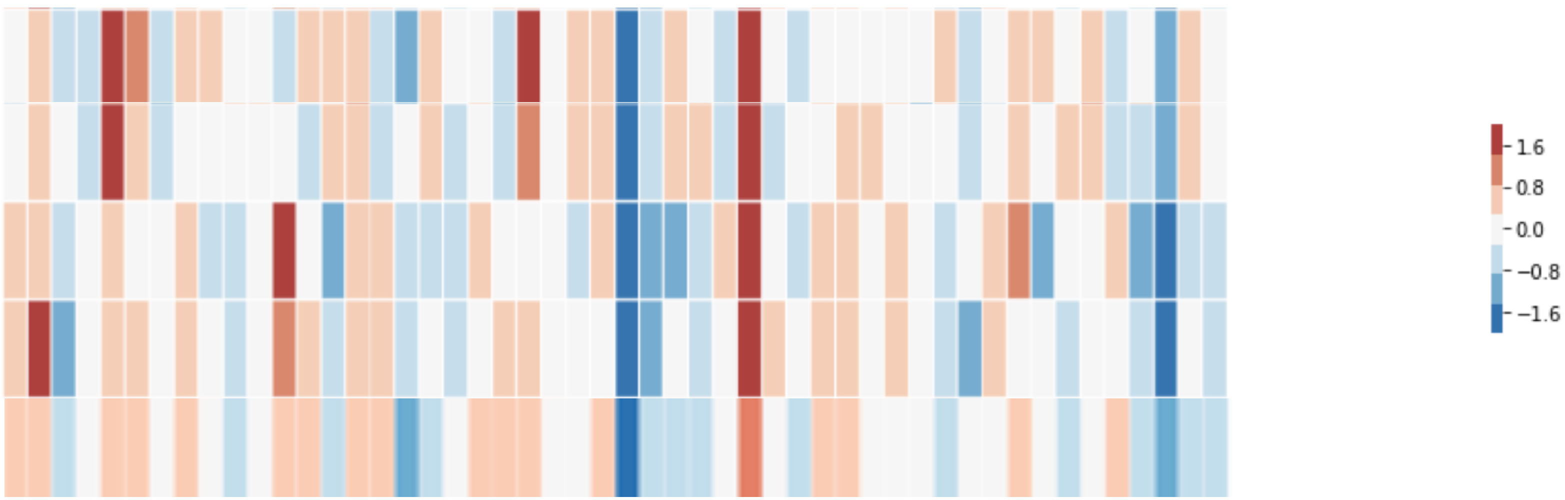
Remember? The Word2Vec Demo!



king – man + woman ≈ queen?

Word Algebra

woman
man
king
queen
king-man +woman



THE UNIVERSITY OF SYDNEY

How to make dense vectors for word representation

Prof. John Rupert Firth

Distributional Hypothesis

"You shall know a word by the company it keeps"

— (Firth, J. R. 1957:11)

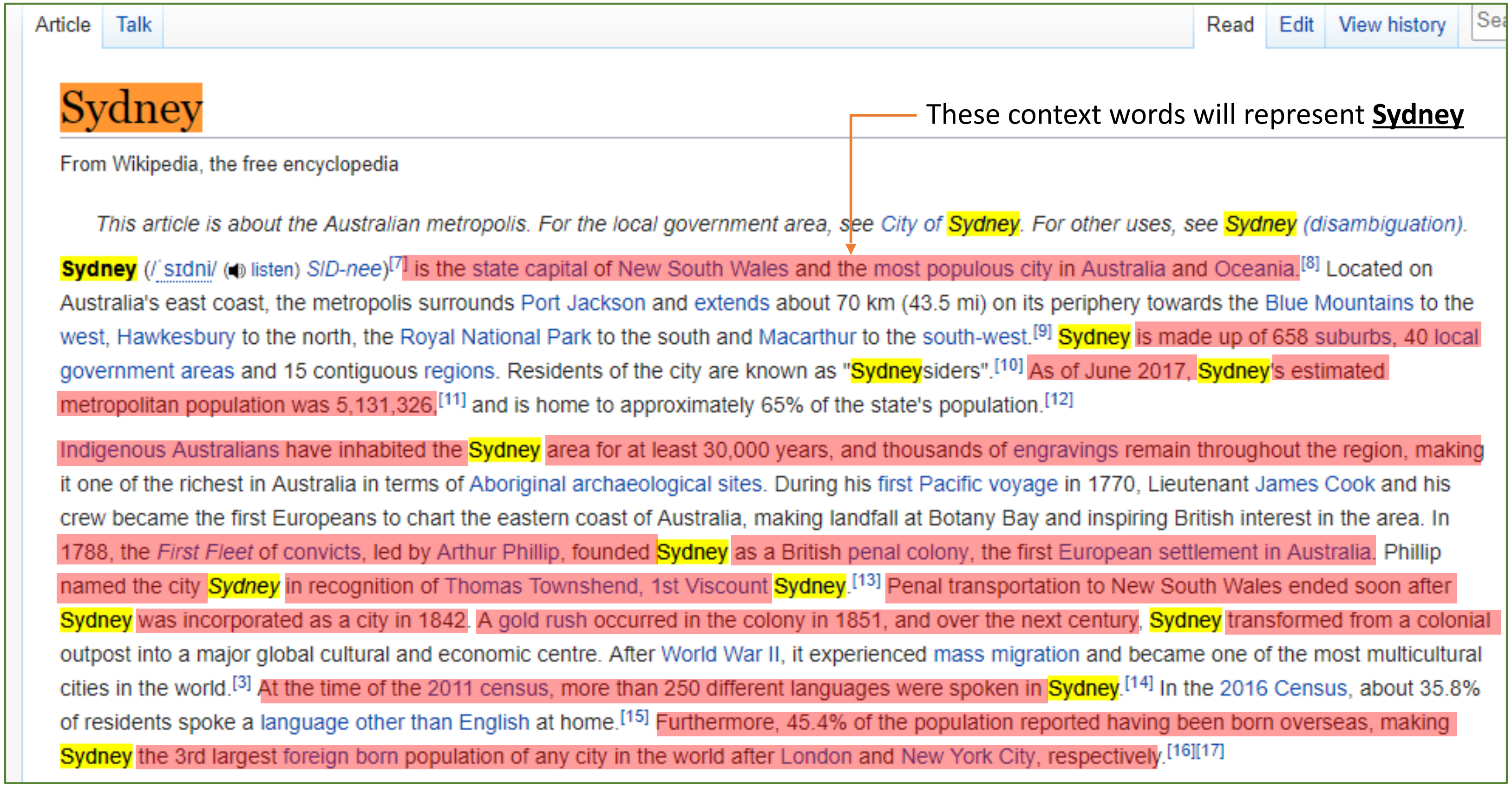
*Firth is noted for drawing attention to the context-dependent nature of meaning with his notion of '**context of situation**', and his work on **collocational meaning** is widely acknowledged in the field of **distributional semantics**.*

3 Prediction based Word representation

Word Representations based on context

When a *word w* appears in a text, its context is the set of words that *appear nearby*

- Use the surrounding context of w to create a representation of w


Article Talk Read Edit View history

Sydney

These context words will represent **Sydney**

From Wikipedia, the free encyclopedia

This article is about the Australian metropolis. For the local government area, see City of Sydney. For other uses, see Sydney (disambiguation).

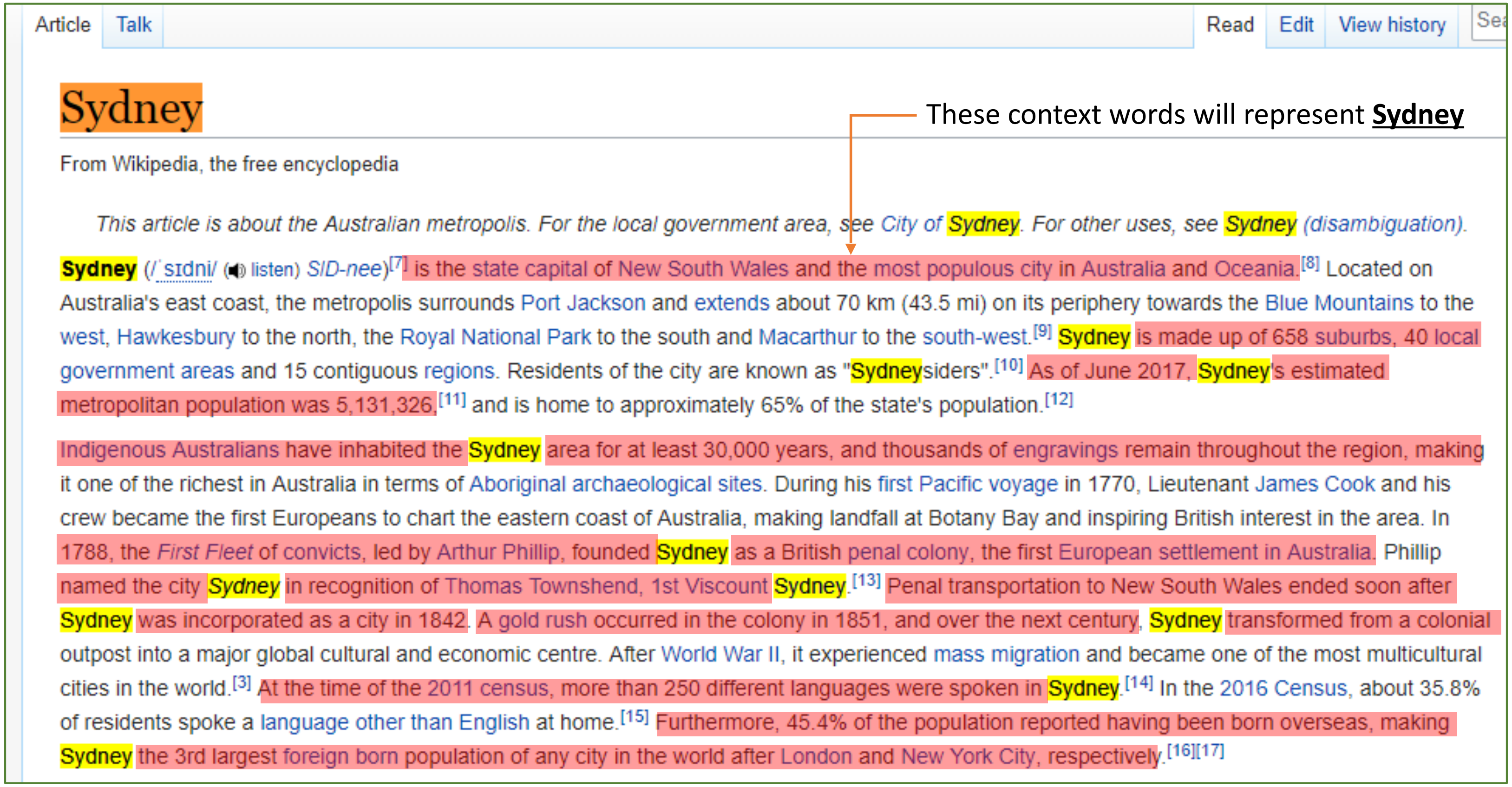
Sydney (/ˈsɪdni/ listen) *SID-nee*)[7] is the state capital of New South Wales and the most populous city in Australia and Oceania.[8] Located on Australia's east coast, the metropolis surrounds Port Jackson and extends about 70 km (43.5 mi) on its periphery towards the Blue Mountains to the west, Hawkesbury to the north, the Royal National Park to the south and Macarthur to the south-west.[9] Sydney is made up of 658 suburbs, 40 local government areas and 15 contiguous regions. Residents of the city are known as "Sydneysiders".[10] As of June 2017, Sydney's estimated metropolitan population was 5,131,326,[11] and is home to approximately 65% of the state's population.[12]

Indigenous Australians have inhabited the Sydney area for at least 30,000 years, and thousands of engravings remain throughout the region, making it one of the richest in Australia in terms of Aboriginal archaeological sites. During his first Pacific voyage in 1770, Lieutenant James Cook and his crew became the first Europeans to chart the eastern coast of Australia, making landfall at Botany Bay and inspiring British interest in the area. In 1788, the *First Fleet* of convicts, led by Arthur Phillip, founded Sydney as a British penal colony, the first European settlement in Australia. Phillip named the city *Sydney* in recognition of Thomas Townshend, 1st Viscount Sydney.[13] Penal transportation to New South Wales ended soon after Sydney was incorporated as a city in 1842. A gold rush occurred in the colony in 1851, and over the next century, Sydney transformed from a colonial outpost into a major global cultural and economic centre. After World War II, it experienced mass migration and became one of the most multicultural cities in the world.[3] At the time of the 2011 census, more than 250 different languages were spoken in Sydney.[14] In the 2016 Census, about 35.8% of residents spoke a language other than English at home.[15] Furthermore, 45.4% of the population reported having been born overseas, making Sydney the 3rd largest foreign born population of any city in the world after London and New York City, respectively.[16][17]

3 Prediction based Word representation

How can we we use a machine to train a word representation?

Neural Networks! (Machine Learning)



Article Talk Read Edit View history Sea

These context words will represent **Sydney**

Sydney

From Wikipedia, the free encyclopedia

This article is about the Australian metropolis. For the local government area, see City of Sydney. For other uses, see Sydney (disambiguation).

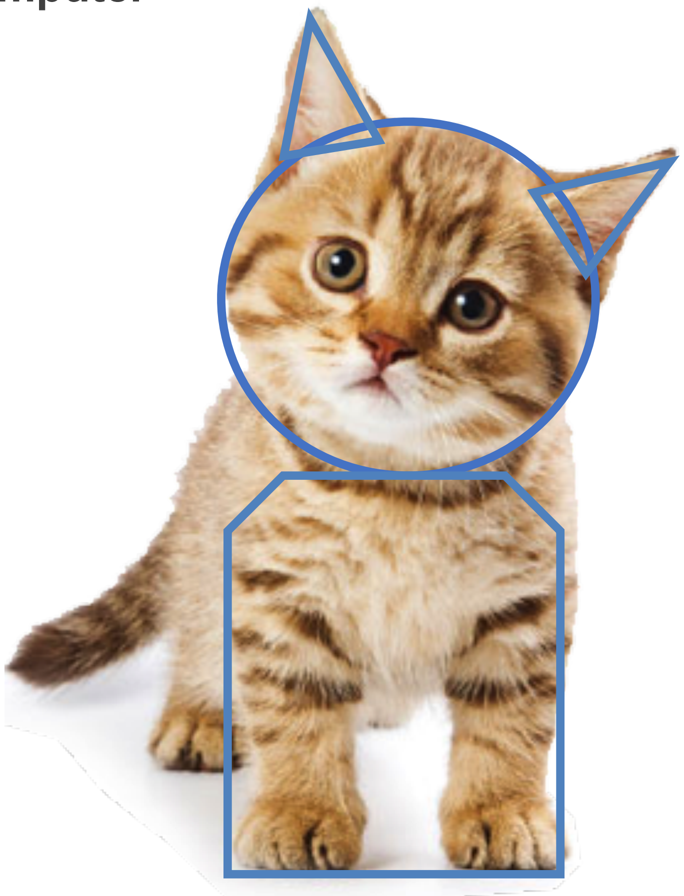
Sydney (/ˈsɪdni/ listen) *SID-nee*)[7] is the state capital of New South Wales and the most populous city in Australia and Oceania.[8] Located on Australia's east coast, the metropolis surrounds Port Jackson and extends about 70 km (43.5 mi) on its periphery towards the Blue Mountains to the west, Hawkesbury to the north, the Royal National Park to the south and Macarthur to the south-west.[9] Sydney is made up of 658 suburbs, 40 local government areas and 15 contiguous regions. Residents of the city are known as "Sydneysiders".[10] As of June 2017, Sydney's estimated metropolitan population was 5,131,326,[11] and is home to approximately 65% of the state's population.[12]

Indigenous Australians have inhabited the Sydney area for at least 30,000 years, and thousands of engravings remain throughout the region, making it one of the richest in Australia in terms of Aboriginal archaeological sites. During his first Pacific voyage in 1770, Lieutenant James Cook and his crew became the first Europeans to chart the eastern coast of Australia, making landfall at Botany Bay and inspiring British interest in the area. In 1788, the *First Fleet* of convicts, led by Arthur Phillip, founded Sydney as a British penal colony, the first European settlement in Australia. Phillip named the city *Sydney* in recognition of Thomas Townshend, 1st Viscount Sydney.[13] Penal transportation to New South Wales ended soon after Sydney was incorporated as a city in 1842. A gold rush occurred in the colony in 1851, and over the next century, Sydney transformed from a colonial outpost into a major global cultural and economic centre. After World War II, it experienced mass migration and became one of the most multicultural cities in the world.[3] At the time of the 2011 census, more than 250 different languages were spoken in Sydney.[14] In the 2016 Census, about 35.8% of residents spoke a language other than English at home.[15] Furthermore, 45.4% of the population reported having been born overseas, making Sydney the 3rd largest foreign born population of any city in the world after London and New York City, respectively.[16][17]

Brief intro to Machine Learning!

Machine Learning

How can your computer classify this?



Object: CAT

Brief intro to Machine Learning!

Computer System

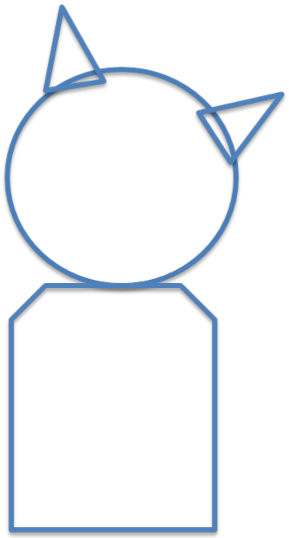
Data

def prediction(image as input):

*...**program**...*

return result

Result

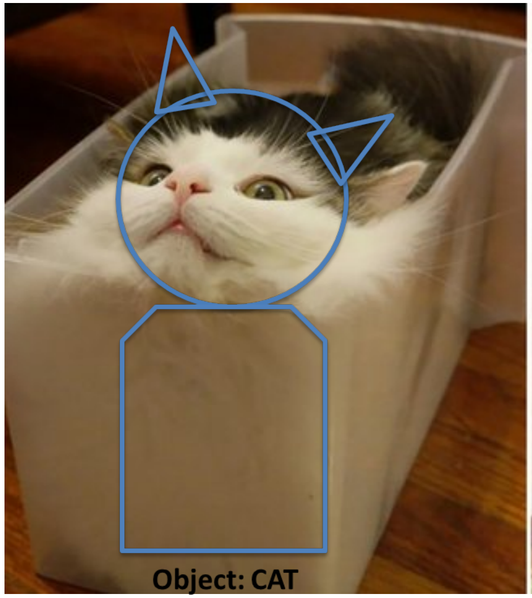


Object: CAT

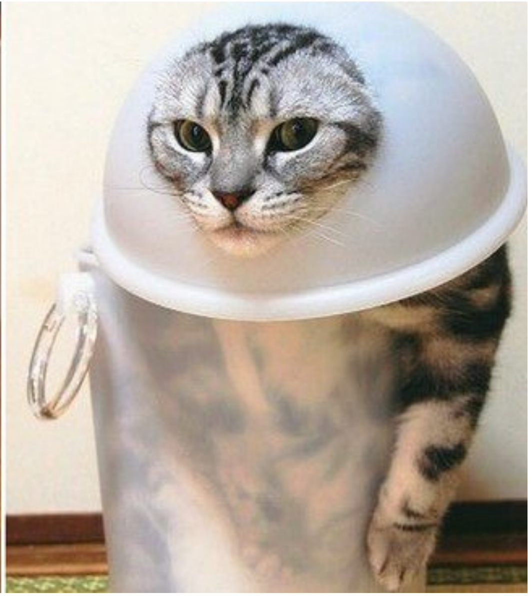
Object: CAT

Brief intro to Machine Learning!

Does it generalise to all examples?



Object: ???



Object: ???

Object: ???

Brief intro to Machine Learning!

Computer System VS Machine Learning

Computer System

Data

def prediction(image as input):
...program...
return result

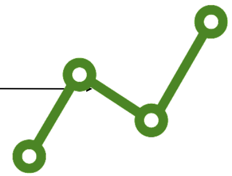
Result

Machine Learning

Data+Result

Data: Result
Image 1: Dog
Image 2: Cat
Image 3: Dog
Image 4: Cat
Image 5: Dog
...

training



Pattern

$\{x_i, y_i\}^N_{i=1}$

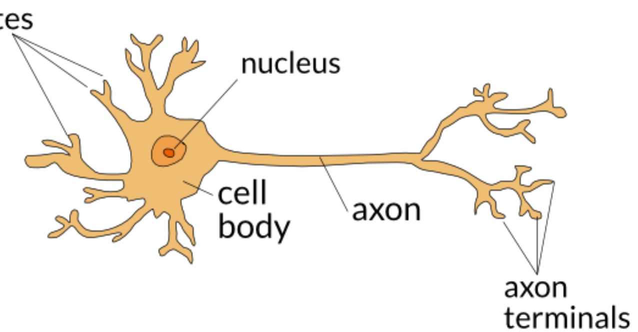
x_i	Input	words (indices or vectors), sentences, documents, etc.
y_i	class	What we try to classify/predict

\# Brief in Machine Learning!

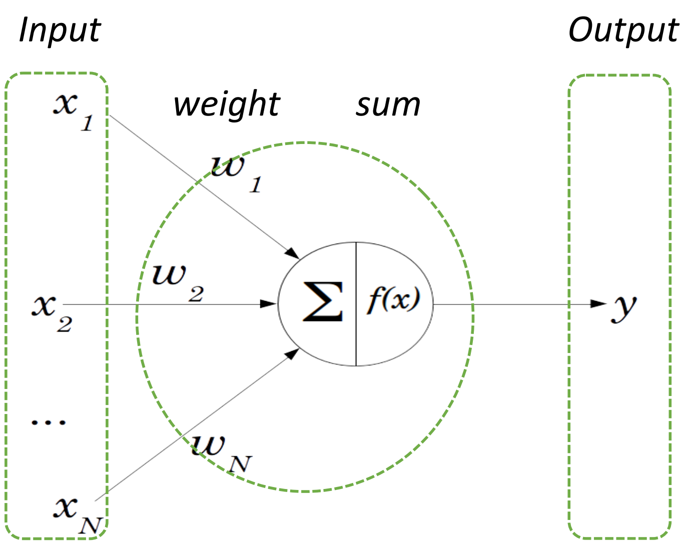
Neural Network and Deep Learning

Neuron and Perceptron

Neuron



Perceptron



NOTE: Neural networks and deep learning will be covered in Lecture 3

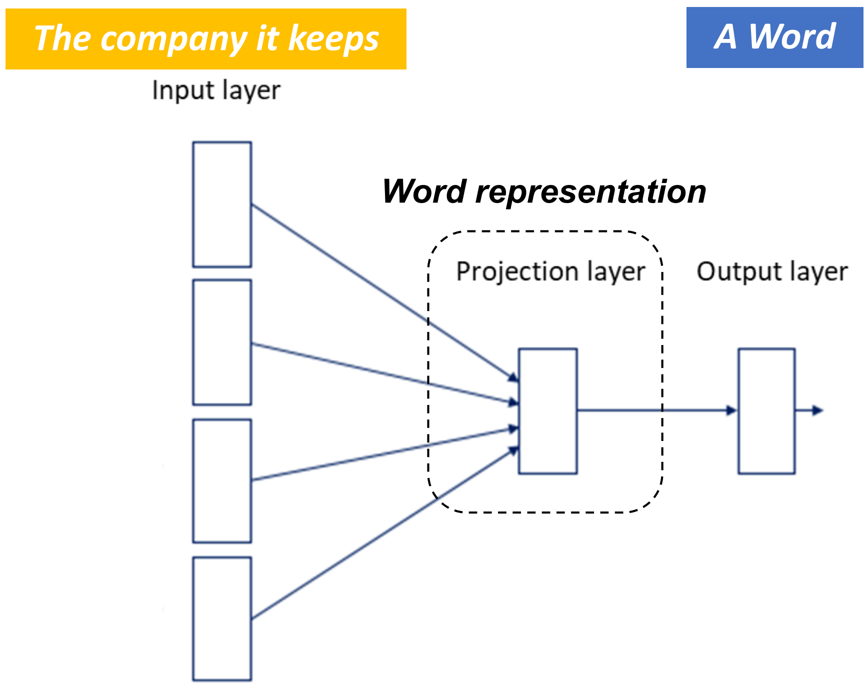
THE UNIVERSITY OF SYDNEY

Neural Networks and Deep Learning in Word Representations

"You shall know a word by the company it keeps" *(Firth, J. R. 1957:11)*

Why don't we <u>represent a word</u> by the company it keeps?

Why don't we <u>train a word representation</u> by the company the word keeps?



3 Prediction based Word representation

Neural Networks and Deep Learning in Word Representations

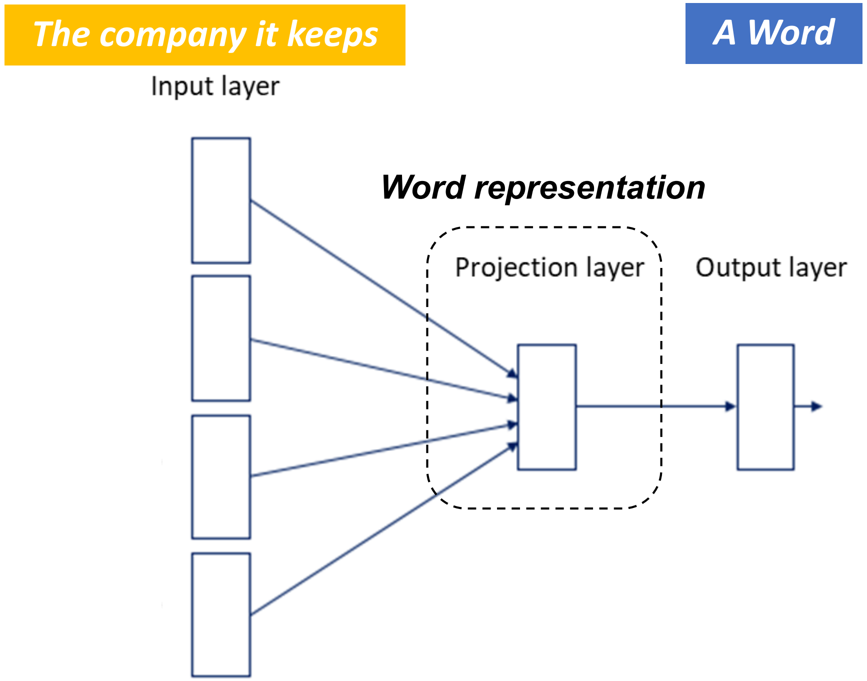
Wikipedia: “Sydney is the state capital of NSW…”


Sydney

From Wikipedia, the free encyclopedia

This article is about the Australian metropolis. For the local government

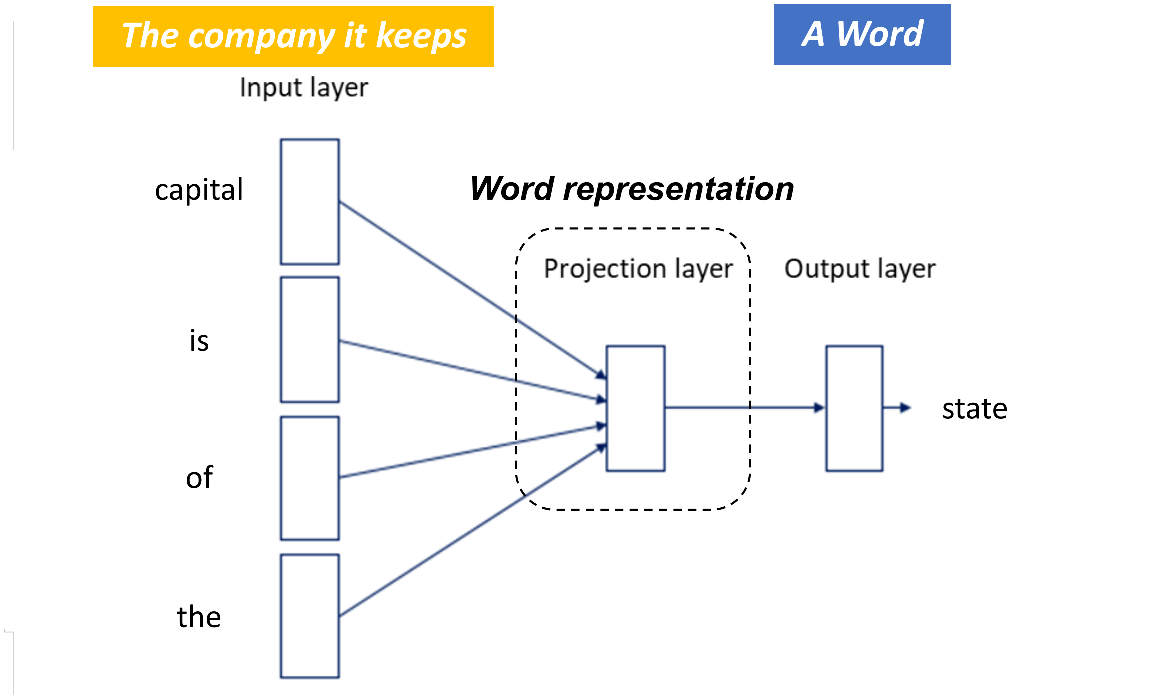
Sydney (/ˈsɪdni/ listen) *SID-nee*)[7] is the state capital of New South Wales



3 Prediction based Word representation

Neural Networks and Deep Learning in Word Representations

Wikipedia: "Sydney is the state capital of NSW…"



3 Prediction based Word representation

Neural Networks and Deep Learning in Word Representations

Wikipedia: "Sydney is the state capital of NSW…"

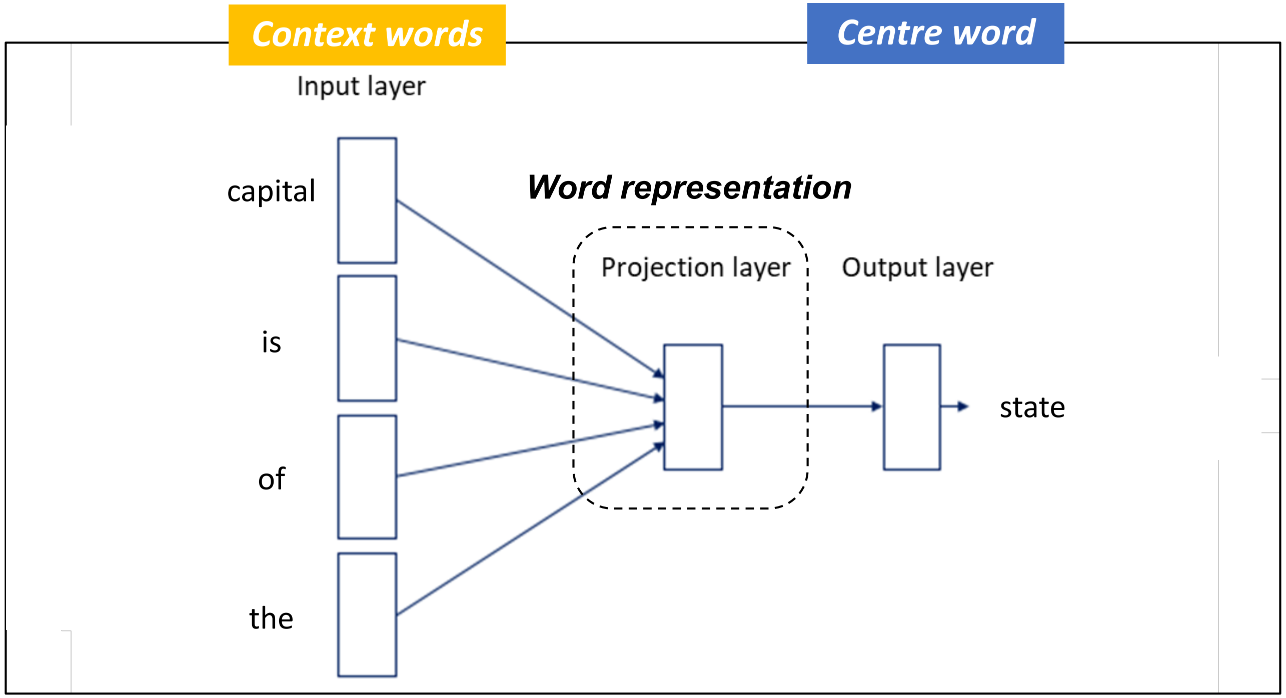

Sydney

From Wikipedia, the free encyclopedia

This article is about the Australian metropolis. For the local government

Sydney (/ˈsɪdni/ listen) *SID-nee*)[7] is the state capital of New South Wales

Word2Vec



40

Break

s/keyboard/leopard/

[Problem Exists Between Leopard And Chair]

Source: https://xkcd.com/1031/

THE INTERNET GOT 100 TIMES BETTER WHEN, THANKS TO AN EXTENSION WITH A TYPO'D REGEX, MY BROWSER STARTED REPLACING THE WORD "KEYBOARD" WITH "LEOPARD".

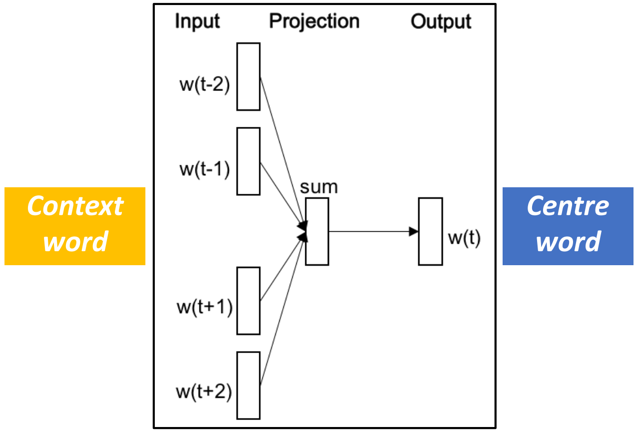
3 Prediction based Word representation

Word2Vec

Word2vec is a library with two models / algorithms that create distributed representations of words:

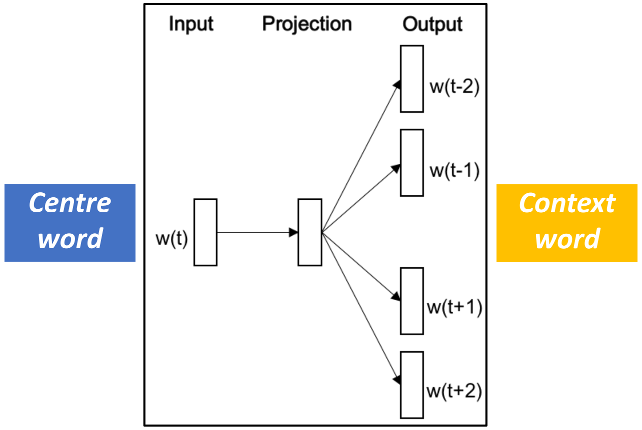
1. Continuous Bag of Words (CBOW)

Predict the **center word** from a (bag of) **context words**



2. Continuous Skip-gram

Predict **context (“outside”) words** **given** **center word**



Word2Vec with Continuous Bag of Words (CBOW)

Predict the centre word from a bag of context words

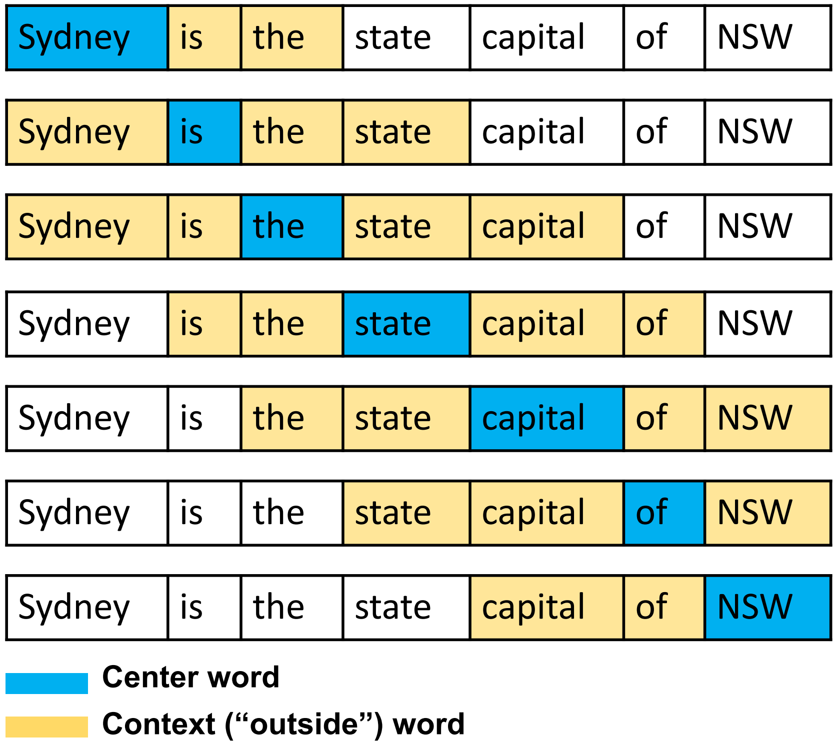
Sentence: "Sydney is the state capital of NSW"

Aim

- Predict the centre word

Setup

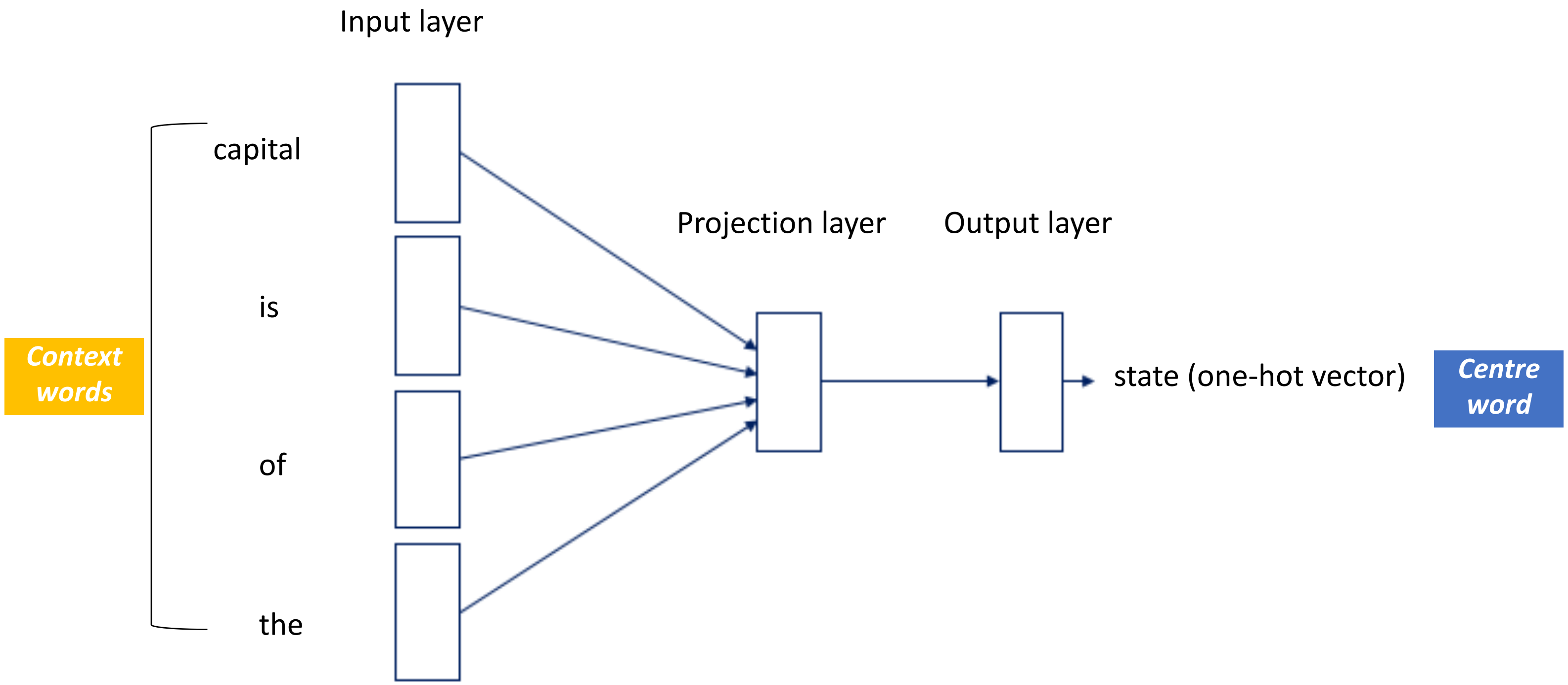
- Window size, 2 in this case



CBOW – Neural Network Architecture

Predict the centre word from a bag of context words

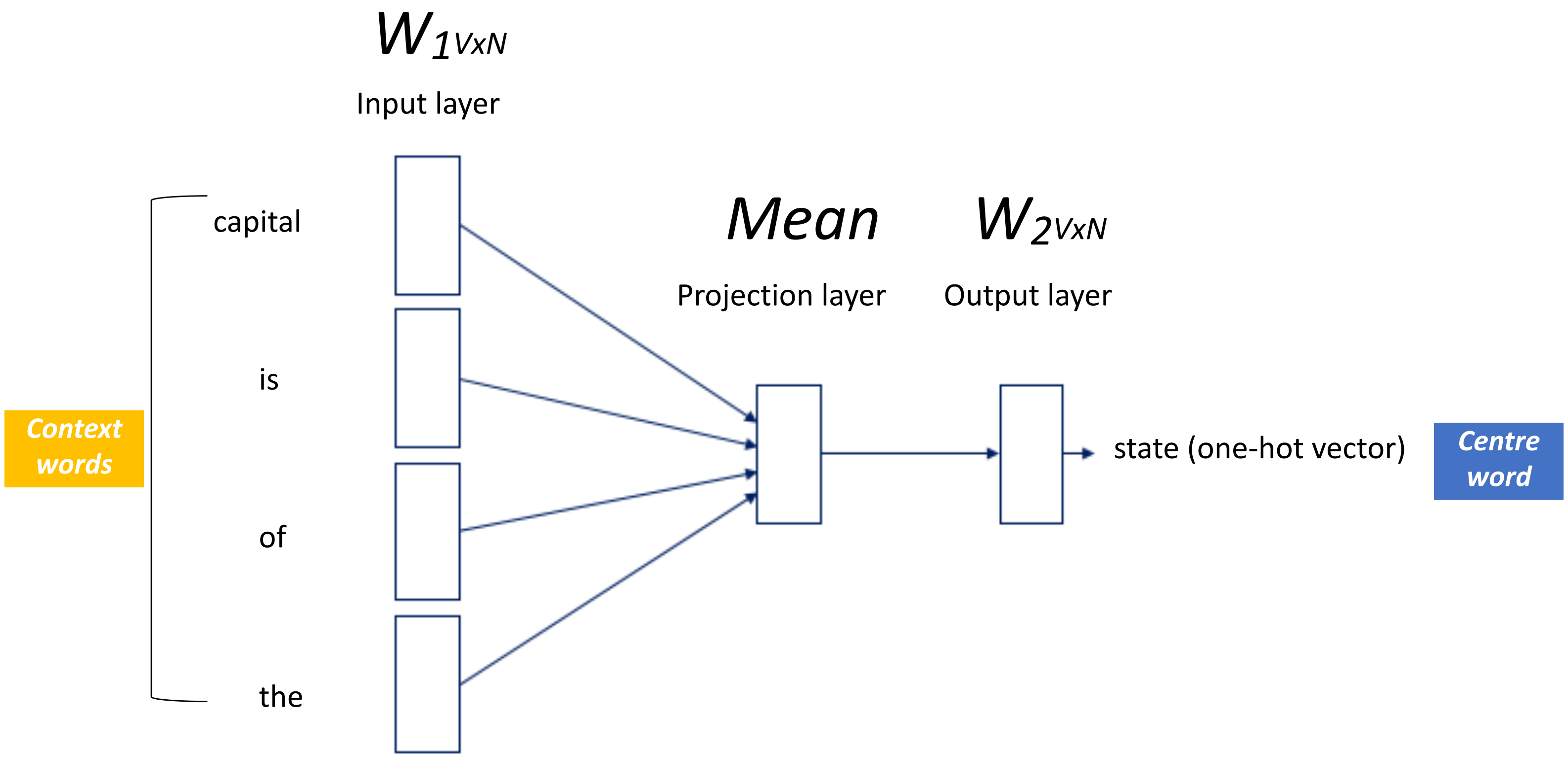
Sentence: "Sydney is the state capital of NSW"



CBOW – Neural Network Architecture

Predict the centre word from a bag of context words

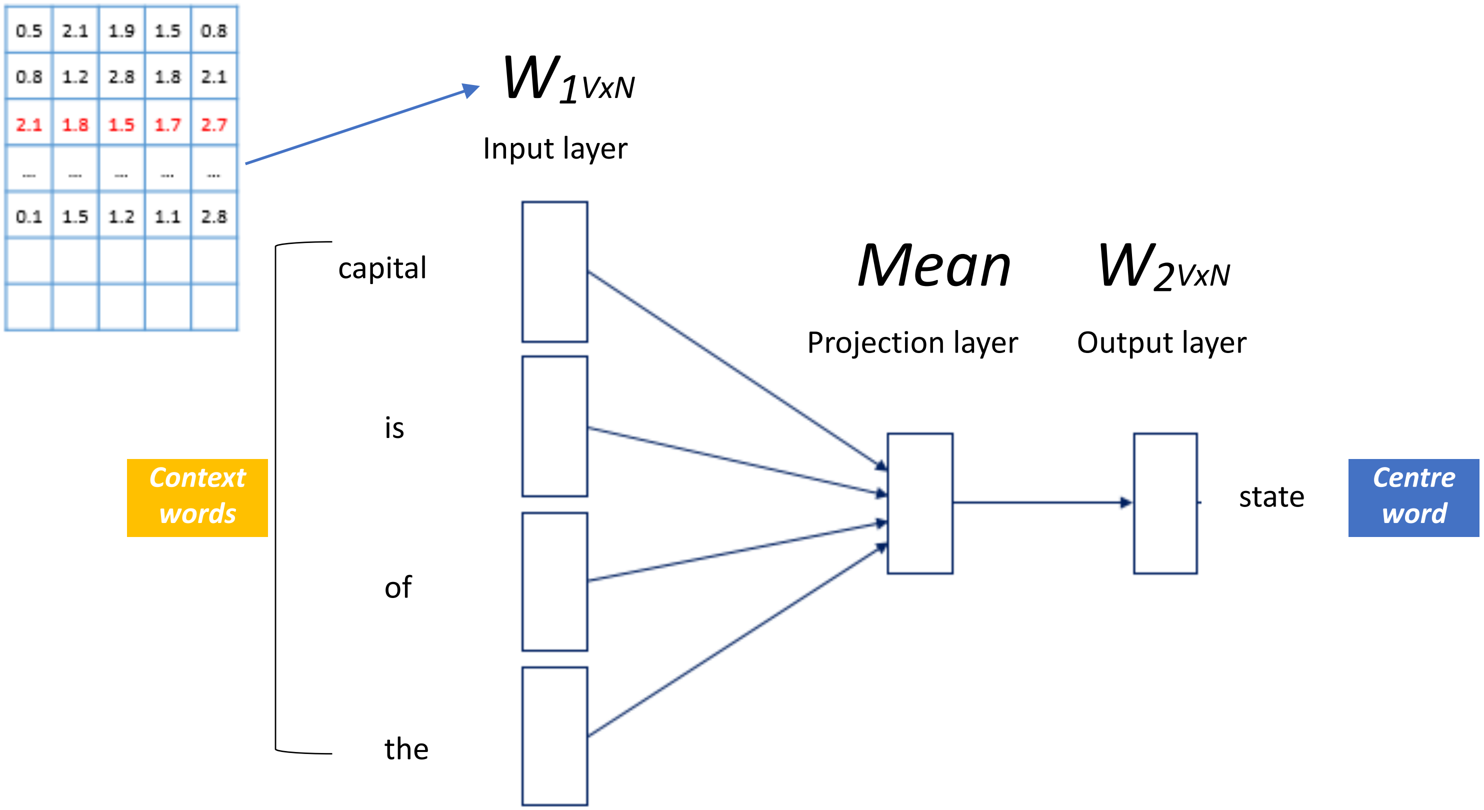
Sentence: "Sydney is the state capital of NSW"



CBOW – Neural Network Architecture

Predict the centre word from a bag of context words

Sentence: "Sydney is the state capital of NSW"



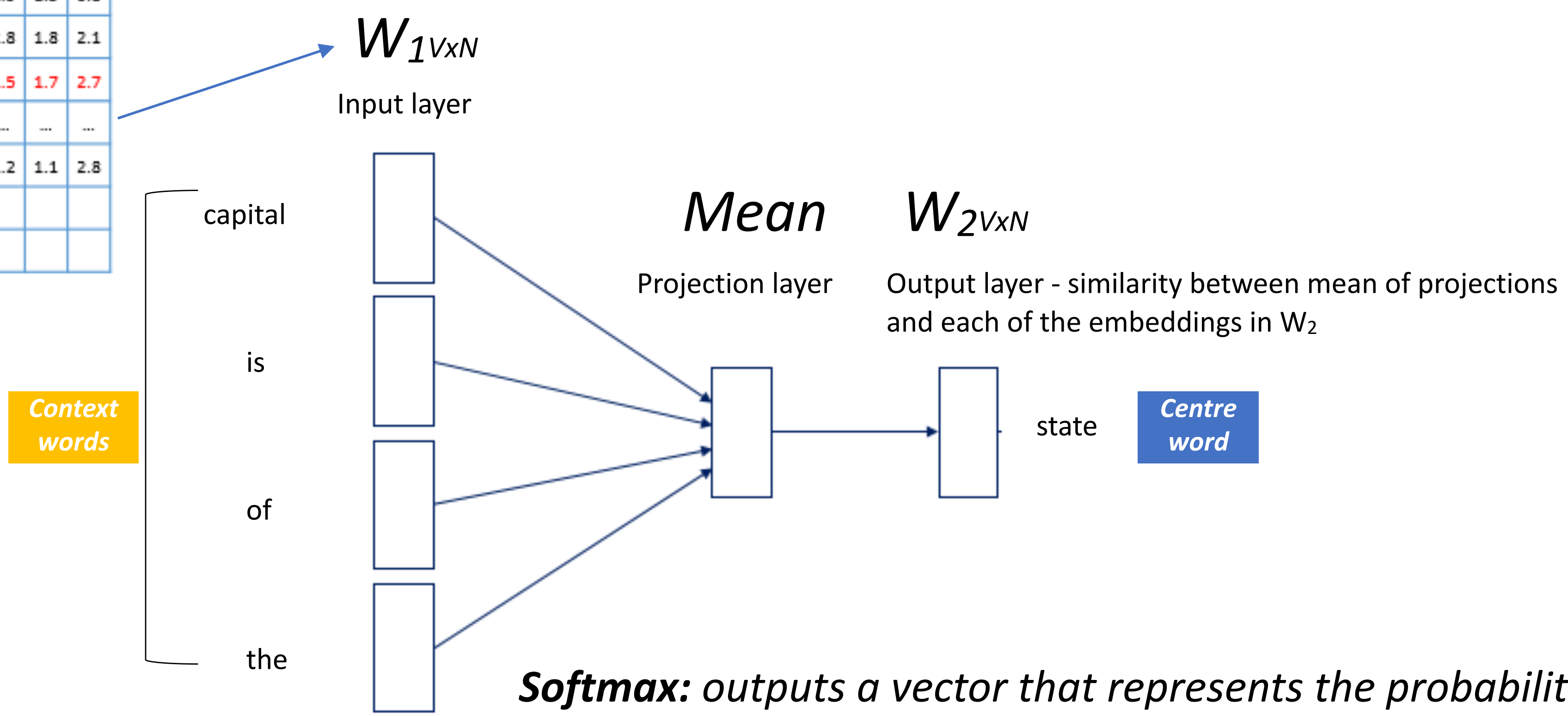
3 Prediction based Word representation

CBOW – Neural Network Architecture

Predict the centre word from a bag of context words

Sentence: "Sydney is the state capital of NSW"

0.5	2.1	1.9	1.5	0.8
0.8	1.2	2.8	1.8	2.1
2.1	1.8	1.5	1.7	2.7
...	...	...	...	...
0.1	1.5	1.2	1.1	2.8



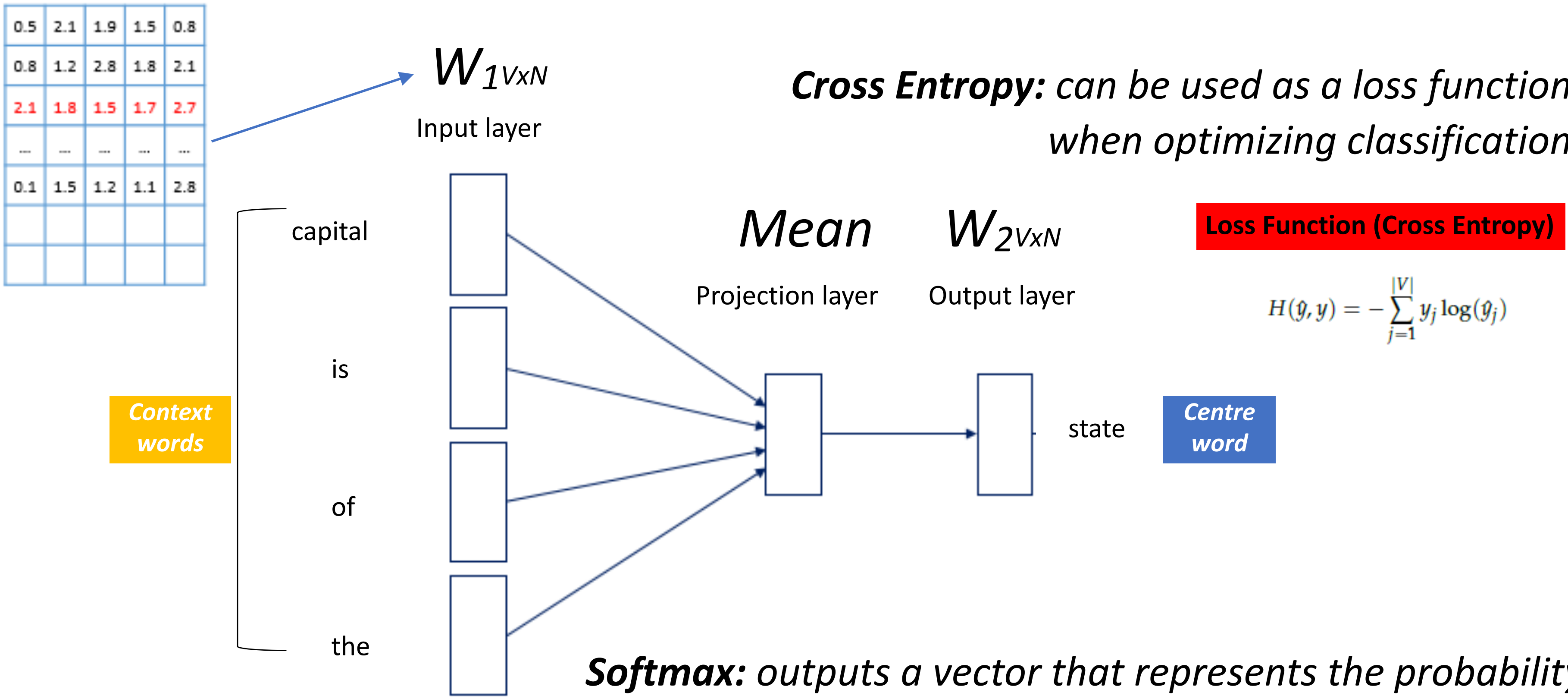
Softmax: *outputs a vector that represents the probability distributions (sum to 1) of a list of potential outcome*

CBOW – Neural Network Architecture

Predict the centre word from a bag of context words

Sentence: "Sydney is the state capital of NSW"

Cross Entropy: *can be used as a loss function when optimizing classification*



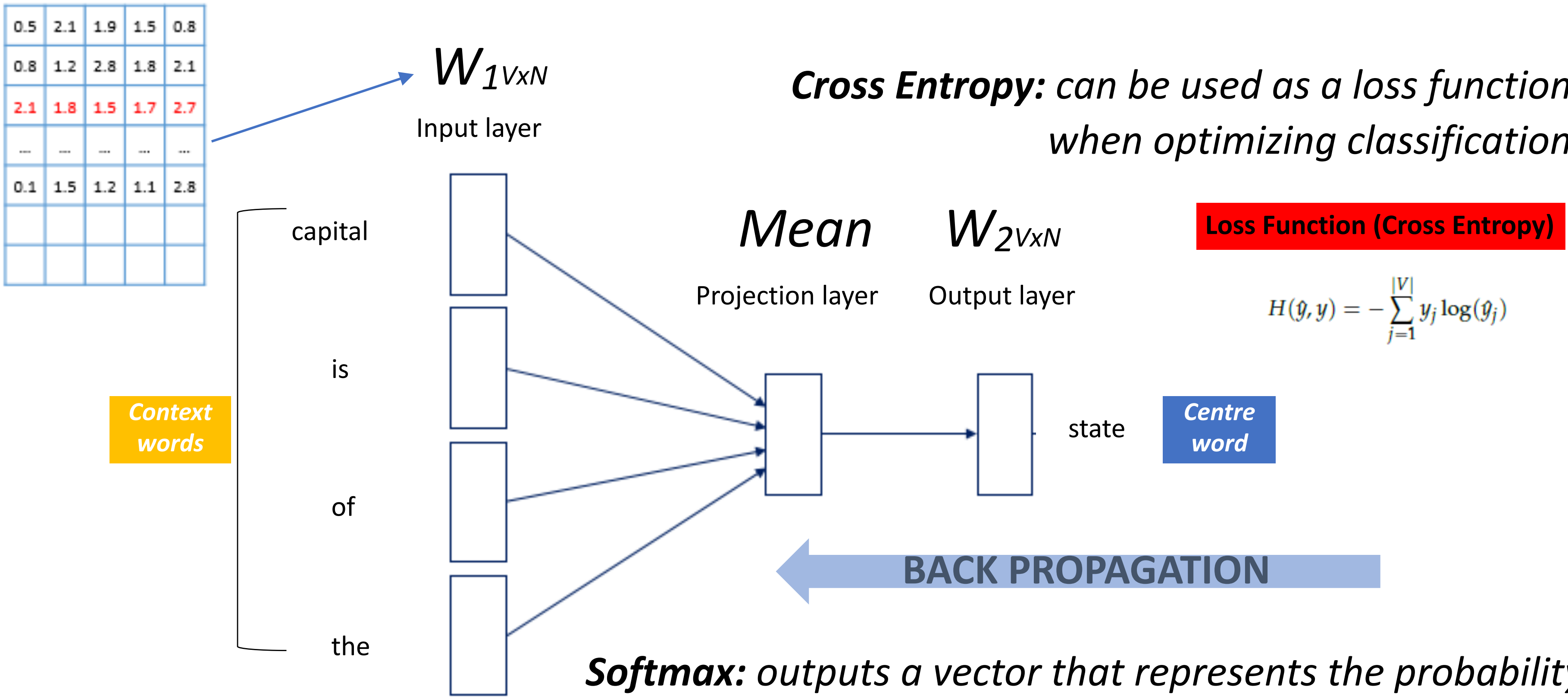
Softmax: *outputs a vector that represents the probability distributions (sum to 1) of a list of potential outcome*

CBOW – Neural Network Architecture

Predict the centre word from a bag of context words

Sentence: "Sydney is the state capital of NSW"

Cross Entropy: *can be used as a loss function when optimizing classification*



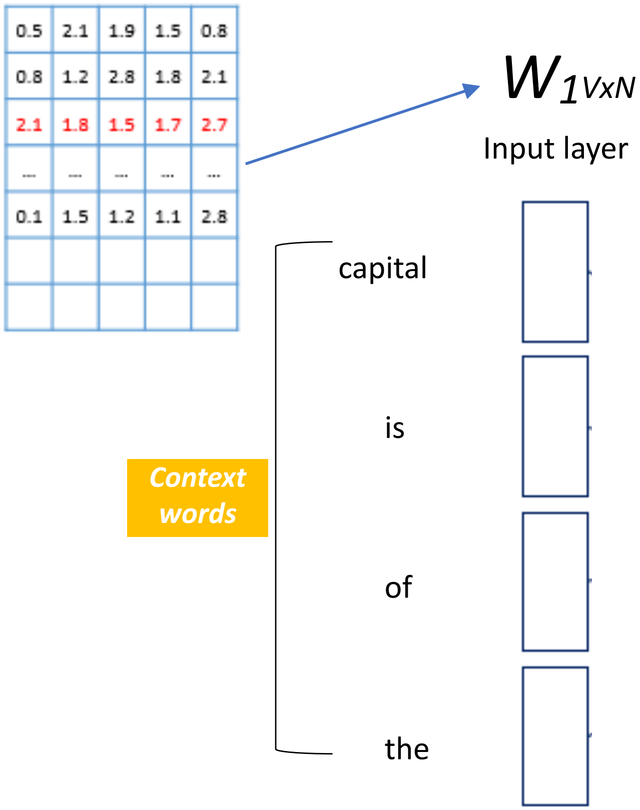
Softmax: *outputs a vector that represents the probability distributions (sum to 1) of a list of potential outcome*

3 Prediction based Word representation

CBOW – Neural Network Architecture

Predict the centre word from a bag of context words

Sentence: "Sydney is the state capital of NSW"



One hot or not? - Equivalent in practise

Option (1)
Take a one-hot vector and multiply it by the W_1 matrix

Option (2)
Look up a vector in the W_1 matrix

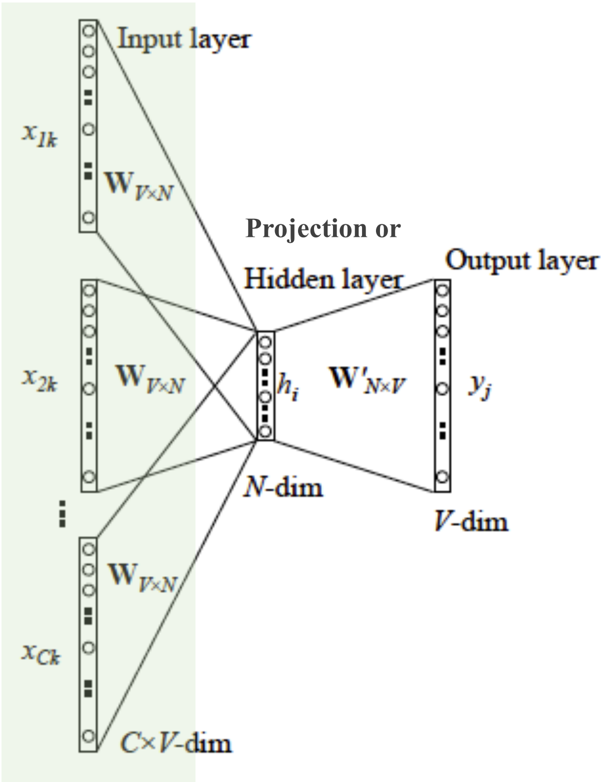
$$\begin{bmatrix}0\\0\\1\\\dots\\0\\0\\0\end{bmatrix}^T \times \begin{bmatrix}0.5 & 2.1 & 1.9 & 1.5 & 0.8\\0.8 & 1.2 & 2.8 & 1.8 & 2.1\\2.1 & 1.8 & 1.5 & 1.7 & 2.7\\\dots & \dots & \dots & \dots & \dots\\0.1 & 1.5 & 1.2 & 1.1 & 2.8\\ & & & & \\ & & & & \end{bmatrix} = \begin{bmatrix}2.1\\1.8\\1.5\\1.7\\1.7\end{bmatrix}$$

3 Prediction based Word representation

CBOW – Neural Network Architecture

Predict the centre word from a bag of context words.

Summary of CBOW Training (Review your understanding with equations)



Uses context words (2m, based on window size =m) as input of the Word2Vec-CBOW model.

$(\boldsymbol{x}^{c-m}, \boldsymbol{x}^{c-m+1}, \ldots, \boldsymbol{x}^{c-1}, \boldsymbol{x}^{c+1}, \ldots, \boldsymbol{x}^{c+m-1}, \boldsymbol{x}^{c+m}) \in \mathbb{R}^{|V|}$

Has two Parameter Matrices:

1) Parameter Matrix (from Input Layer to Hidden/Projection Layer)
$\mathbf{W} \in \mathbb{R}^{V\text{x}N}$

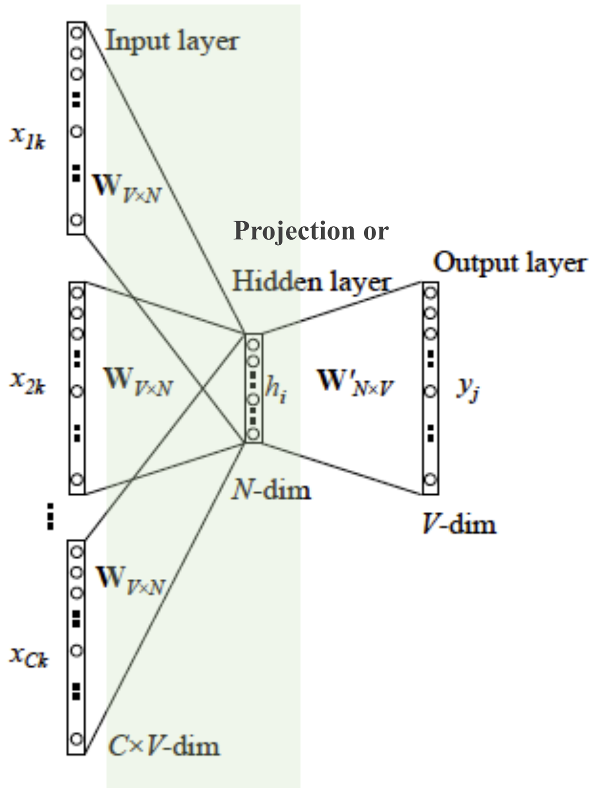
2) Parameter Matrix (to Output Layer)
$\mathbf{W}' \in \mathbb{R}^{N\text{x}V}$

3 Prediction based Word representation

CBOW – Neural Network Architecture

Predict the centre word from a bag of context words.

Summary of CBOW Training (Review your understanding with equations)



Initial words are looked up in $\mathbf{W}_{\mathbf{V x N}}$ to get a 1 x N (embedded word) vector.

$$(\boldsymbol{v}_{\boldsymbol{c-m}} = \mathrm{W}x^{c-m}, \ldots, \boldsymbol{v}_{\boldsymbol{c+m}} = \mathrm{W}x^{c+m}) \in \mathbb{R}^{\boldsymbol{n}}$$

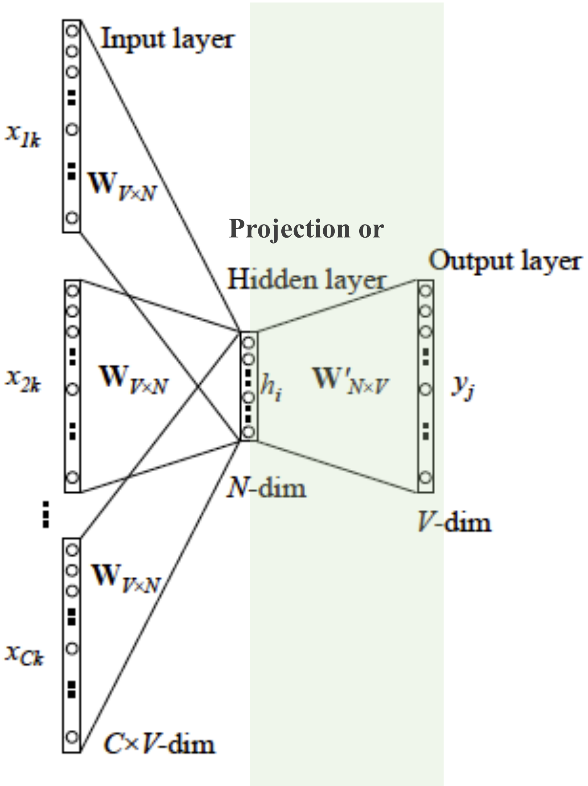
Average those 2m embedded vectors to calculate a single vector.

$$\hat{v} = \frac{\boldsymbol{v}_{\boldsymbol{c-m}} + \boldsymbol{v}_{\boldsymbol{c-m+1}} + \ \ldots + \boldsymbol{v}_{\boldsymbol{c+m}}}{2m}$$

CBOW – Neural Network Architecture

Predict the centre word from a bag of context words.

Summary of CBOW Training (Review your understanding with equations)



Calculate the similarity between the mean vector and each of the vectors in W2

$$z = \mathbf{W}'\hat{v} \in \mathbb{R}^{|V|}$$

Convert this to a probability using Softmax

$$\hat{y} = softmax\,(z) \in \mathbb{R}^{|V|}$$

Train the parameter matrix using objective function.

$$H\left(\hat{y},\ y\right) = -\sum_{j=1}^{|V|} y_j\, log(\hat{y}_j)$$

* We are minimising the value

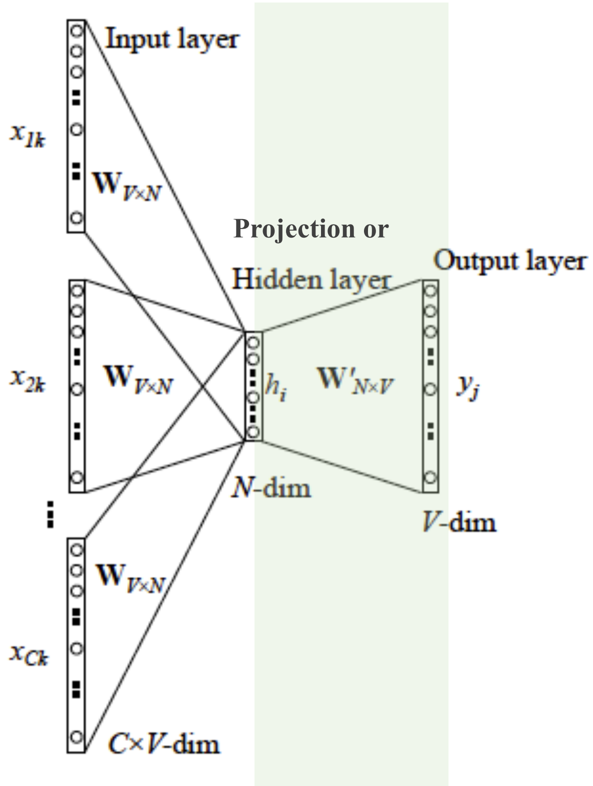
Only one term is non-zero, the one for the true word:

$$H\left(\hat{y},\ y\right) = -y_j\, log(\hat{y}_j)$$

CBOW – Neural Network Architecture

Predict the centre word from a bag of context words.

Summary of CBOW Training (Review your understanding with equations)



Where does the optimisation function on the last slide come from?

$$
\begin{aligned}
minimize J &= -\log P(w_c | w_{c-m}, \ldots, w_{c+m}) \\
&= -\log P(u_c | v) \\
&= -\log \frac{exp(u_c^\intercal \hat{v})}{\sum_{j=1}^{|V|} exp(u_j^\intercal \hat{v})} \\
&= -u_c^{intercal} \hat{v} + \log \sum_{j=1}^{|V|} exp(u_j^\intercal \hat{v})
\end{aligned}
$$

*This optimisation objective will be discussed in more detail in lecture 3.

3 Prediction based Word representation

Skip Gram

Predict context ("outside") words (position independent) given centre word

Sentence: "Sydney is the state capital of NSW"

$P(w_{t+2} | w_t)$

$P(w_{t-1} | w_t)$ $P(w_{t+1} | w_t)$

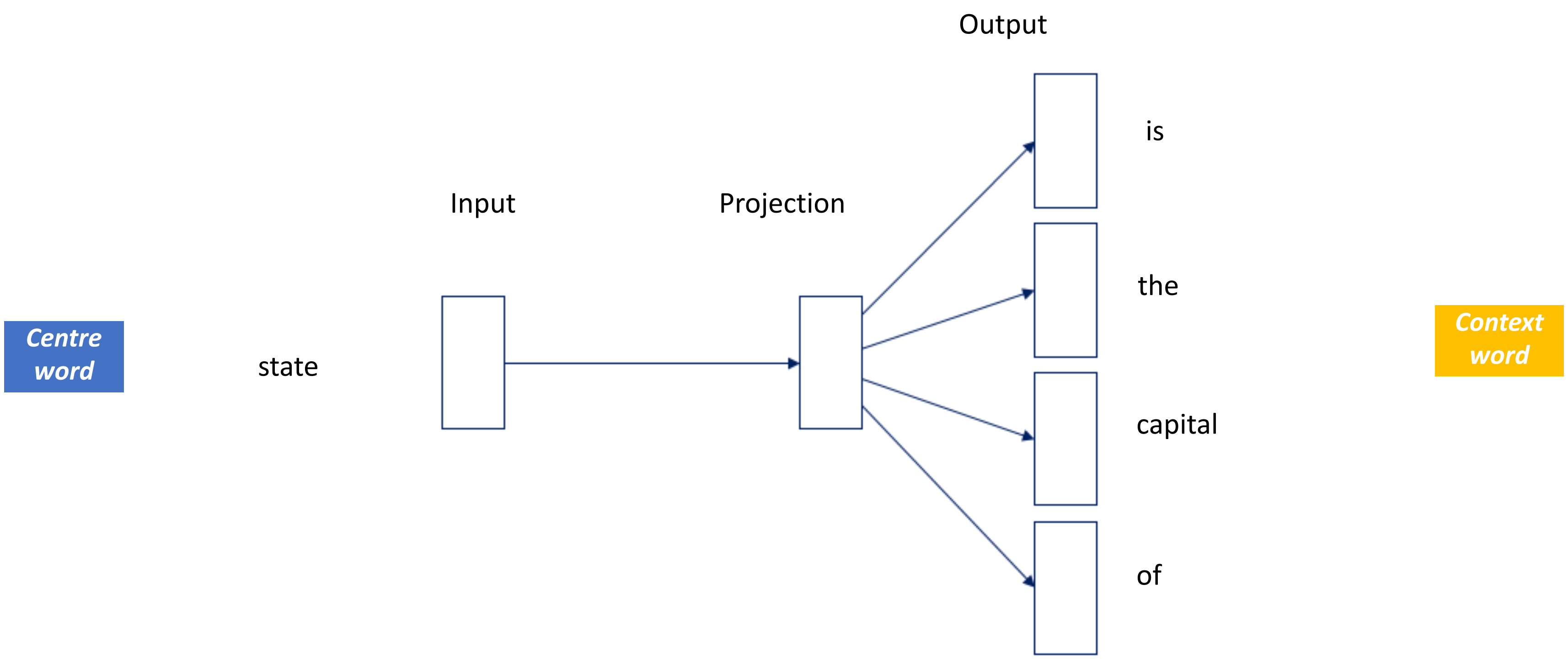
… Sydney is the ← **state** → capital → of NSW …

3 Prediction based Word representation

Skip Gram

Predict context ("outside") words (position independent) given centre word

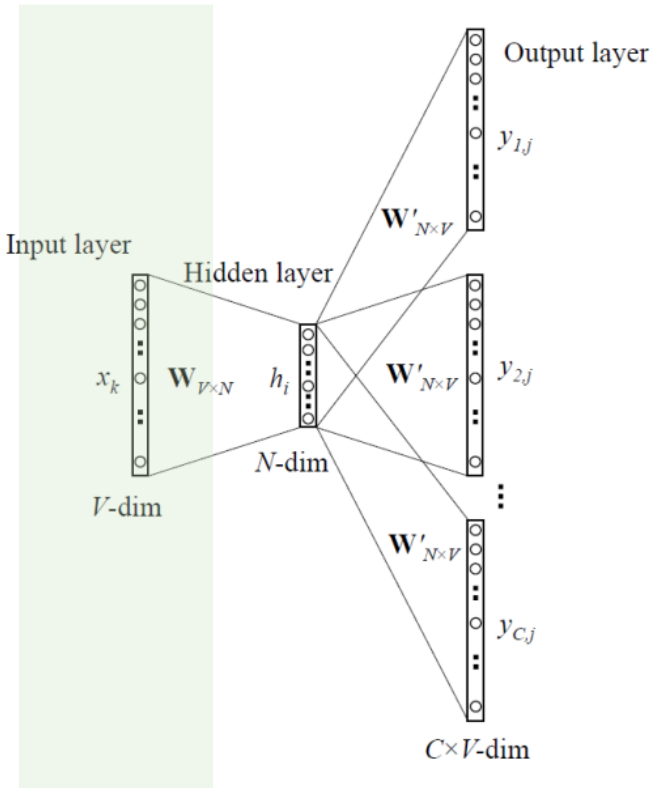
Sentence: "Sydney is the state capital of NSW"



Skip Gram – Neural Network Architecture

Predict context ("outside") words (position independent) given centre word

Summary of Skip Gram Training (Review your understanding with equations)



Has two Parameter Matrices:

1) Parameter Matrix (Input)

$\mathbf{W} \in \mathbb{R}^{V\text{x}N}$

2) Parameter Matrix (Output)

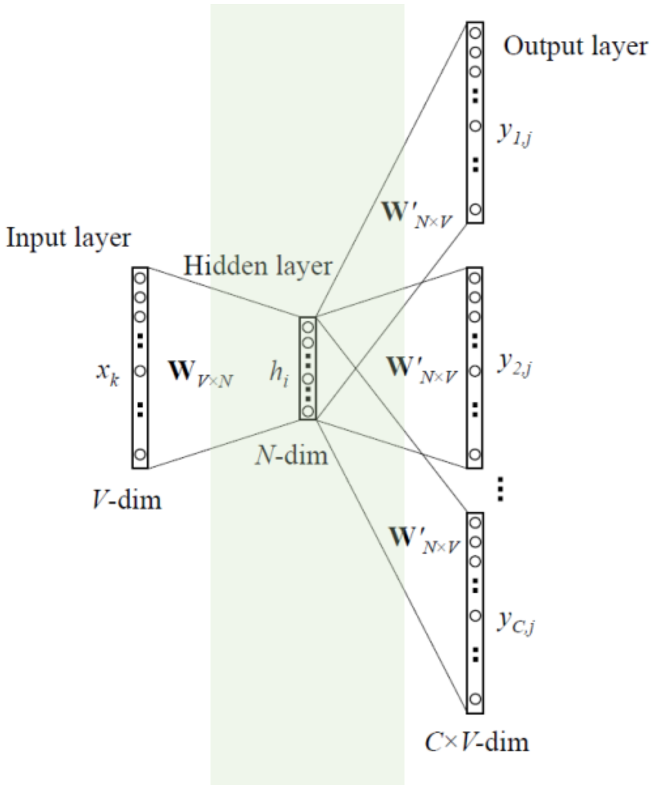
$\mathbf{W}' \in \mathbb{R}^{N\text{x}V}$

3 Prediction based Word representation

Skip Gram – Neural Network Architecture

Predict context (“outside”) words (position independent) given centre word

Summary of Skip Gram Training (Review your understanding with equations)



Initial words are looked up in $\mathbf{W}_{\mathbf{V}\mathrm{x}\mathbf{N}}$ to get a 1 x N (embedded word) vector.

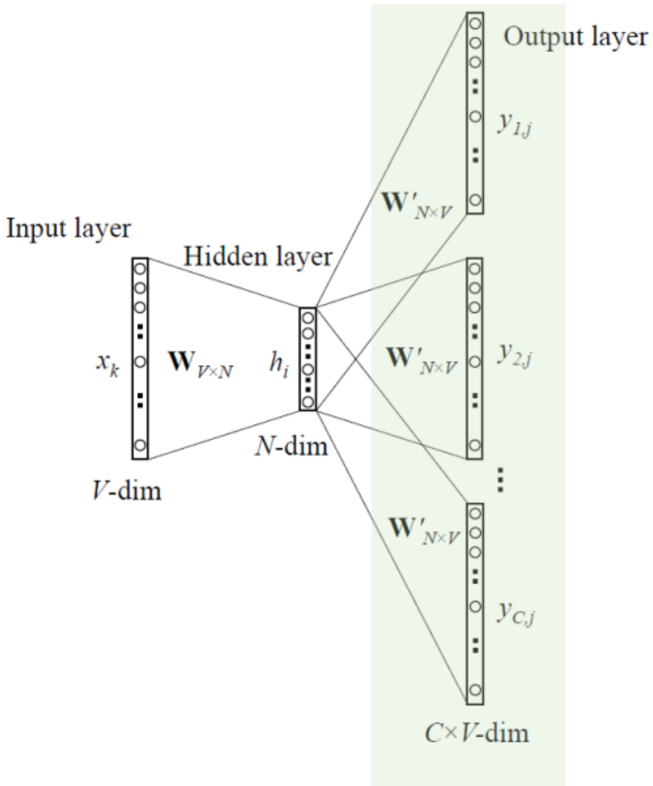
$\boldsymbol{v_c} = \mathbf{W}_{\boldsymbol{x}} \in \mathbb{R}^{\boldsymbol{n}}$ (as there is only one input)

Calculate the score value for the output layer by multiplying by the parameter matrix W’
$\boldsymbol{z} = \mathbf{W}'_{\boldsymbol{v_c}}$

Skip Gram – Neural Network Architecture

Predict context ("outside") words (position independent) given centre word

Summary of Skip Gram Training (Review your understanding with equations)



Calculate the probability using softmax
$\hat{y} = softmax\ (\boldsymbol{z})$

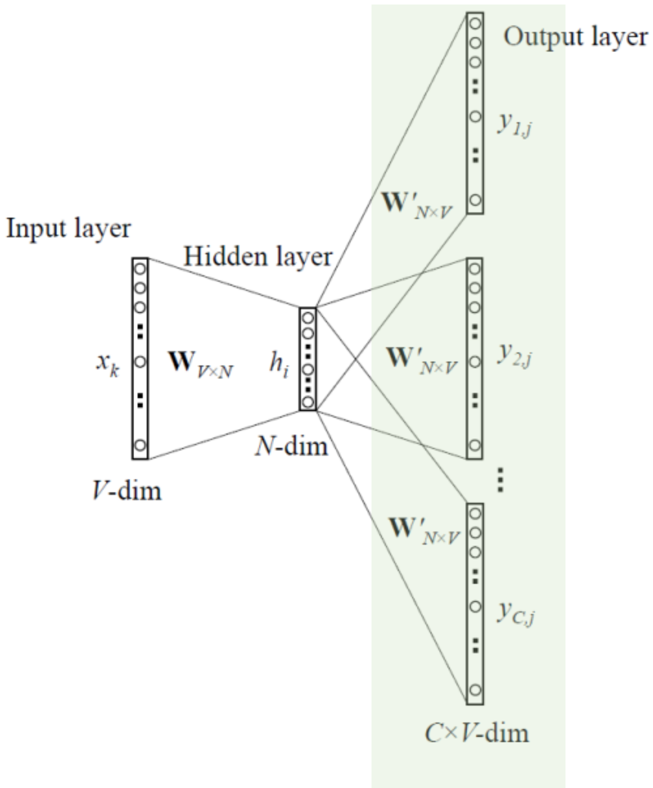
Calculate 2m probabilities as we need to predict 2m context words.
$\hat{y}_{\boldsymbol{c-m}},\ \dots, \hat{y}_{\boldsymbol{c-1}}, \hat{y}_{\boldsymbol{c+1}},\ \dots, \hat{y}_{\boldsymbol{c+m}}$

and compare with the ground truth
$y^{(c-m)}, \dots, y^{(c-1)},\ y^{(c+1)}, \dots, y^{(c+m)}$

Skip Gram – Neural Network Architecture

Predict context (“outside”) words (position independent) given centre word

Summary of Skip Gram Training (Review your understanding with equations)



As in CBOW, use an objective function for us to evaluate the model. A key difference here is that we invoke a Naïve Bayes assumption to break out the probabilities. It is a strong naïve conditional independence assumption. Given the centre word, all output words are completely independent.

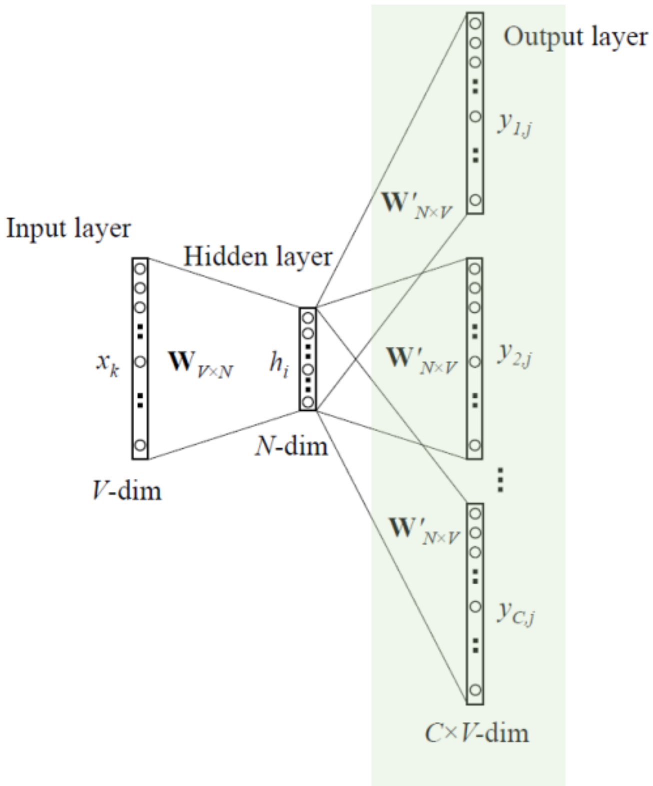
$$\begin{aligned}
\text{minimize } J &= -\log P(w_{c-m}, \ldots, w_{c-1}, w_{c+1}, \ldots, w_{c+m} | w_c) \\
&= -\log \prod_{j=0, j\neq m}^{2m} P(w_{c-m+j} | w_c) \\
&= -\log \prod_{j=0, j\neq m}^{2m} \frac{\exp(u_{c-m+j}^\top v_c)}{\sum_{k=1}^{|V|} \exp(u_k^\top v_c)} \\
&= -\sum_{j=0, j\neq m}^{2m} u_{c-m+j}^\top v_c + 2m \log \sum_{k=1}^{|V|} \exp(u_k^\top v_c)
\end{aligned}$$

*This optimisation objective will be discussed in more detail in lecture 3.

Skip Gram – Neural Network Architecture

Predict context ("outside") words (position independent) given centre word

Summary of Skip Gram Training (Review your understanding with equations)



With this objective function, we can compute the gradients with respect to the unknown parameters and at each iteration update them via Stochastic Gradient Descent

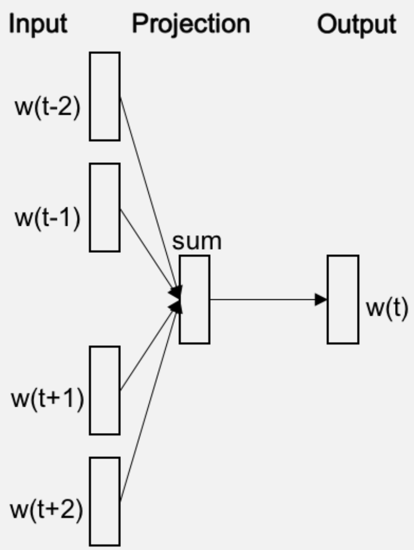
$$J = -\sum_{j=0, j\neq m}^{2m} \log P(u_{c-m+j}|v_c)$$
$$= \sum_{j=0, j\neq m}^{2m} H(\hat{y}, y_{c-m+j})$$

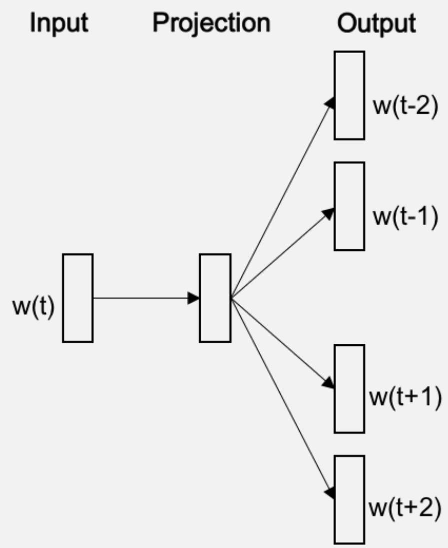
* Stochastic Gradient Descent will be discussed in more detail in lecture 3.

3 Prediction based Word representation

CBOW vs Skip Gram Overview

CBOW	***Skip-gram***
Predict centre word from (bag of) context words	***Predict context words given centre word***





3 Prediction based Word representation

Key Parameter (1) for Training methods: Window Size

Different tasks are served better by different window sizes.

Smaller window sizes (2-5) lead to embeddings that are syntactically similar

Larger window sizes (5+) lead to embeddings that are semantically similar

Key Parameter (2) for Training methods: Negative Samples

The number of negative samples is another factor of the training process.

Negative samples in our dataset – samples of words that are not neighbours

Negative sample: 2

Input word	*Output word*	*Target*
eat	mango	1
eat	exam	0
eat	tobacco	0

1 = Appeared

0 = Did Not Appear

Negative sample: 5

Input word	*Output word*	*Target*
eat	mango	1
eat	exam	0
eat	tobacco	0
eat	pool	0
eat	supervisor	0

The original paper prescribes **5-20 as being a good number of negative samples**. It also states that **2-5 seems to be enough when you have a large enough dataset**.

THE UNIVERSITY OF SYDNEY

Key Parameter (2) for Training methods: Negative Samples

The number of negative samples is another factor of the training process.

Negative samples to our dataset – samples of words that are not neighbors

Negative sample: 2

Input word	*Output word*	*Target*
eat	mango	1
eat	exam	0
eat	tobacco	0

1 = Appeared

0 = Did Not Appear

Negative sample: 5

Input word	*Output word*	*Target*
eat	mango	1
eat	exam	0
eat	tobacco	0
eat	pool	0
eat	supervisor	0

How to select Negative Samples?

The "negative samples" are selected using a "unigram distribution", where more frequent words are more likely to be selected as negative samples.

$$P(w_i) = \frac{f(w_i)}{\sum_{j=0}^{n}(f(w_j))}$$

The probability of picking the word (w_i) would be equal to the number of times (w_i) appears in the corpus, divided the total number of words in the corpus.

Word2Vec Overview

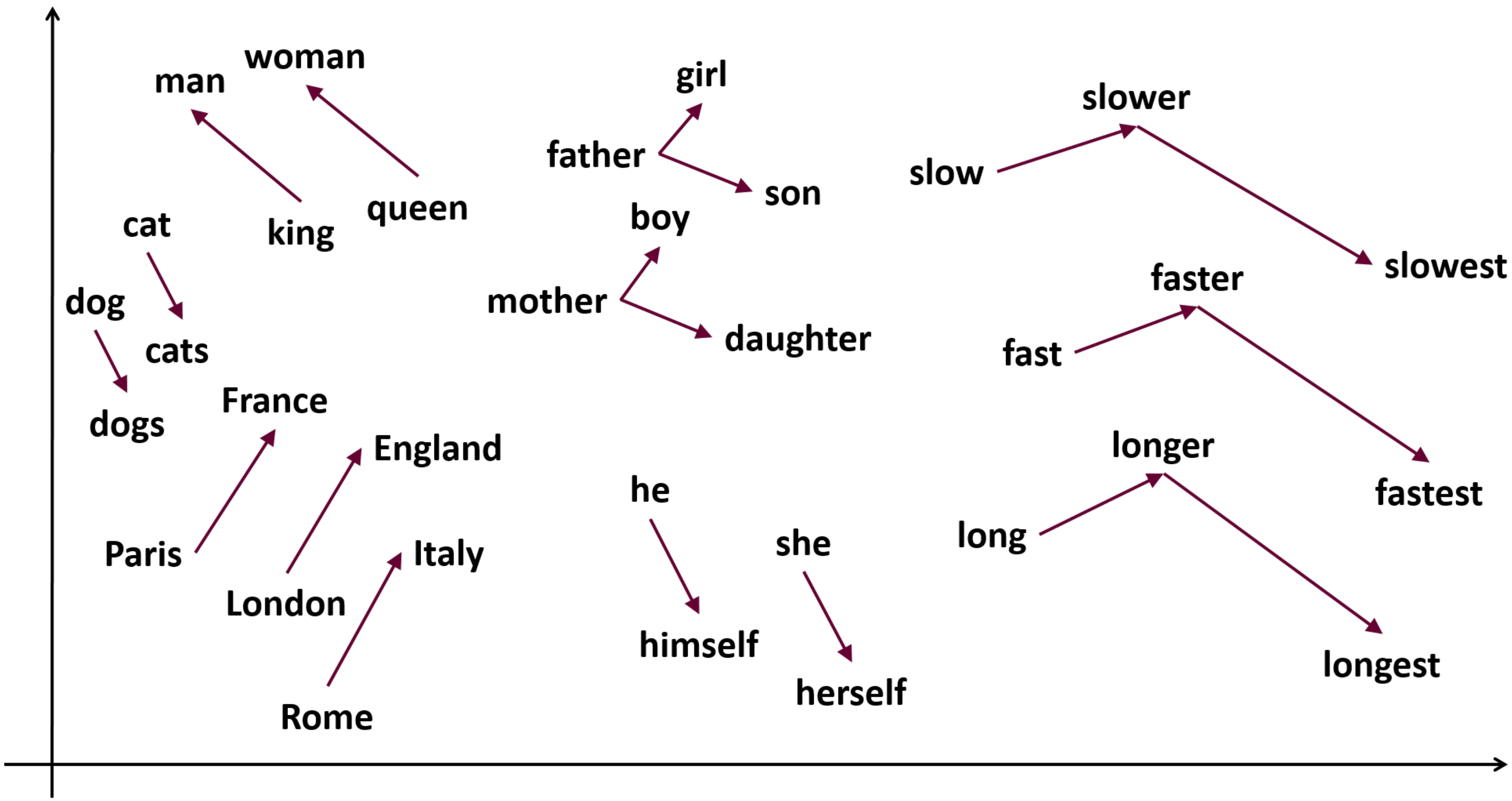
Word2vec (Mikolov et al. 2013) is a framework for learning word vectors

Idea:

- Have a large corpus of text
- Every word in a fixed vocabulary is represented by a vector
- Go through each position *t* in the text, which has a centre word *c* and context ("outside") words *o*
- Use the similarity of the word vectors for c and o to calculate the probability of o given c (or vice versa)
- Keep adjusting the word vectors to maximise this probability

3 Prediction based Word representation

Let's try some Word2Vec!



Gensim: https://radimrehurek.com/gensim/models/word2vec.html

Resources: https://wit3.fbk.eu/

https://github.com/3Top/word2vec-api#where-to-get-a-pretrained-models

Limitation of Word2Vec

Issue#1: Cannot cover morphological similarity

- Word2vec represents every word as an independent vector, even though many words are morphologically similar, like: teach, teacher, teaching

Issue#2: Hard to create embeddings for rare words

- Word2vec is based on the Distribution hypothesis. Works well with the frequent words but does not embed the rare words.

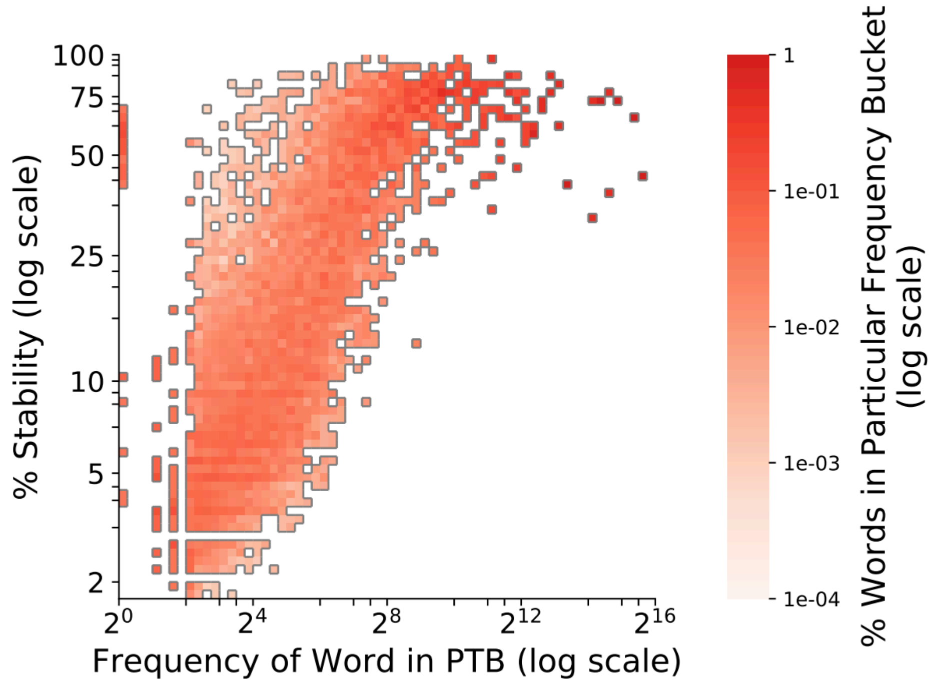
Limitation of Word2Vec

Issue#1: Cannot cover morphological similarity

- Word2vec represents every word as an independent vector, even though many words are morphologically similar, like: teach, teacher, teaching

Issue#2: Hard to create embeddings for rare words

- Word2vec is based on the Distribution hypothesis. Works well with the frequent words but does not embed the rare words.



https://aclanthology.org/N18-1190.pdf

Limitation of Word2Vec

Issue#1: Cannot cover <u>morphological similarity</u>

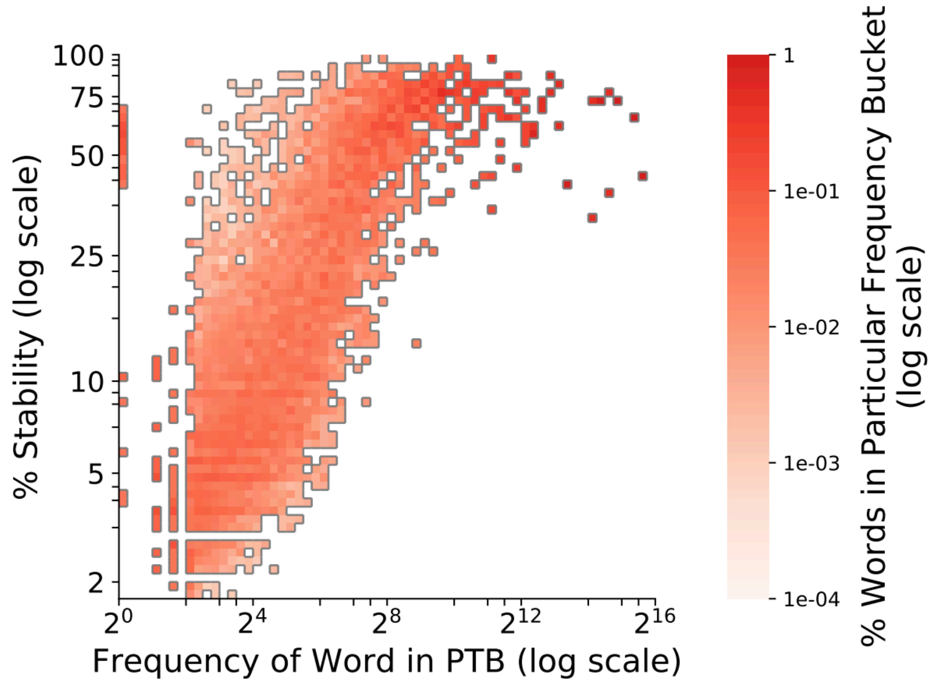
- Word2vec represents every word as an independent vector, even though many words are morphologically similar, like: teach, teacher, teaching

Issue#2: Hard to create embeddings for <u>rare words</u>

- Word2vec is based on the Distribution hypothesis. Works well with the frequent words but does not embed the rare words.

Issue#3: Cannot handle <u>Out-of-Vocabulary (OOV) words</u>

- Word2vec does not work at all if the word is not included in the Vocabulary



https://aclanthology.org/N18-1190.pdf

THE UNIVERSITY OF SYDNEY

FastText

- Deal with these Word2Vec Limitations
- Another Way to transfer *WORDS* to *VECTORS*

fastText

- FastText is a library for learning of word embeddings and text classification created by Facebook's AI Research lab. The framework provides a way to usean unsupervised learning or supervised learning algorithm for obtaining vector representations for words.

- Extension to Word2Vec
  - Instead of feeding individual words into the Neural Network, FastText breaks words into several n-grams (sub-words)

https://fasttext.cc/

FastText with N-gram Embeddings

- N-grams are simply all combinations of adjacent words or letters of length n that you can find in your source text. For example, given the word *apple*, the 2-grams (or "bigrams") are ***ap, pp, pl**,* and ***le***

- The tri-grams (n=3) for the word apple are ***app, ppl**,* and ***ple***. The word embedding vector for apple will be the sum of vectors for all of these n-grams.

- After training the Neural Network (either with skip-gram or CBOW), we will have word embeddings for all the n-grams given the training dataset.

- Rare words can now be properly represented since it is highly likely that some of their n-grams also appears in other words.

https://fasttext.cc/

Word2Vec VS FastText

Find synonym with Word2vec

```
from gensim.models import Word2Vec
cbow_model = Word2Vec(sentences=result, size=100, window=5, min_count=5, workers=4, sg=0)

a=cbow_model.wv.most_similar("electrofishing")
pprint.pprint(a)
```

Find synonym with FastText

```
from gensim.models import FastText
FT_model = FastText(sentences=result, size=100, window=5, min_count=5, workers=4, sg=0)

a=FT_model.wv.most_similar("electrofishing")
pprint.pprint(a)
```

electrofishing

https://fasttext.cc/

3 Prediction based Word representation

Global Vectors (GloVe)

*"Methods like skip-gram may do better on the analogy task, but they poorly utilize the statistics of the corpus since they train on separate local context windows instead of on **global co-occurrence counts**."*

(PeddingLon et al., 2014)

Global optimisation using matrix factorisation

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

e.g. P(k l i) *k=context words, i =centre words*

https://nlp.stanford.edu/projects/glove/

3 Prediction based Word representation

Limitation of Prediction based Word Representation

- I like ________

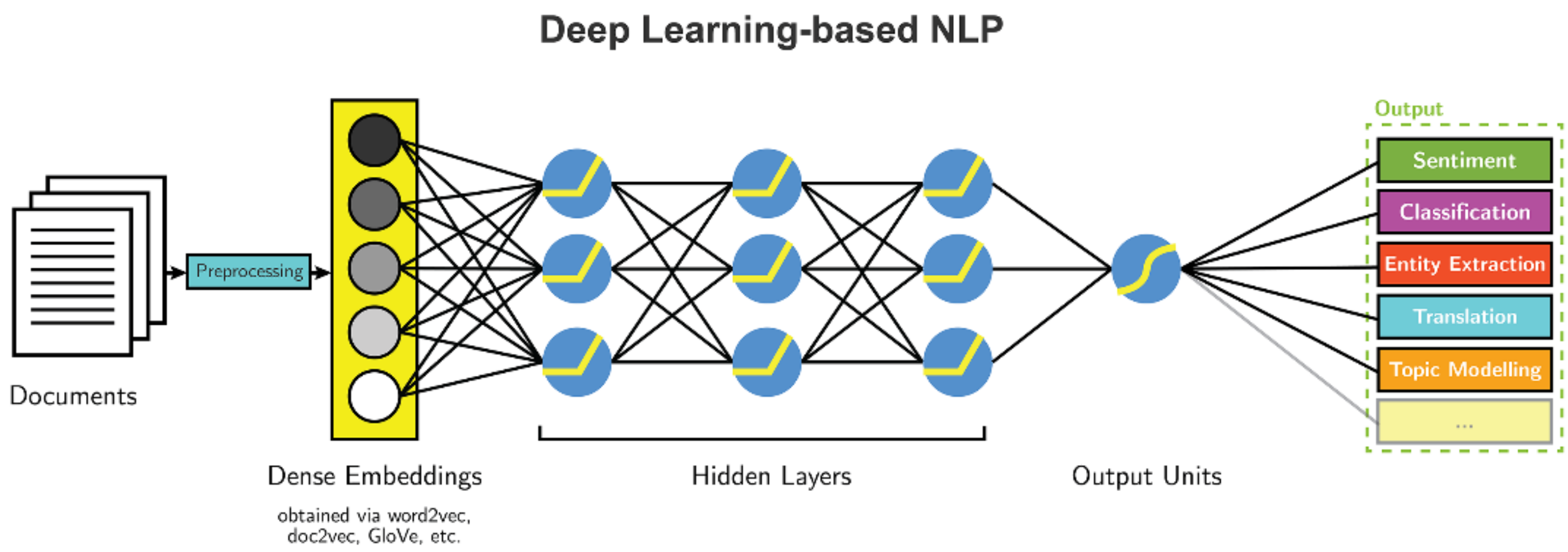
  apple *banana* *fruit*

- Training dataset impacts the word representations
  - The word similarity of the word ‘software’ for a model learned using the Google News corpus can be different from the one from Twitter.

https://nlp.stanford.edu/projects/glove/

4 NEXT WEEK PREVIEW…

Machine Learning/ Deep Learning for Natural Language Processing



Reference

Reference for this lecture

- Deng, L., & Liu, Y. (Eds.). (2018). Deep Learning in Natural Language Processing. Springer.
- Rao, D., & McMahan, B. (2019). Natural Language Processing with PyTorch: Build Intelligent Language Applications Using Deep Learning. " O'Reilly Media, Inc.".
- Manning, C. D., Manning, C. D., & Schütze, H. (1999). Foundations of statistical natural language processing. MIT press.
- Manning, C 2017, Introduction and Word Vectors, Natural Language Processing with Deep Learning, lecture notes, Stanford University
- Images: http://jalammar.github.io/illustrated-word2vec/
- Goldberg, Lewis R. 1992, "The development of markers for the Big-Five factor structure." Psychological assessment 4.1: 26.

Word2vec

- Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.
- Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119).

FastText

- Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics, 5, 135-146.
- Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., & Joulin, A. (2017). Advances in pre-training distributed word representations. arXiv preprint arXiv:1712.09405.