Lexical Semantics

Lecture 8

VCU College of Engineering

Semantics

Focuses on the meaning of words and their relationships

Word?

Focuses on the meaning of words and their relationships

- Quantifying a word as a unit of meaning
  - school bus
  - bank
  - dog

Relationships between words

- Hononymy
- Polysemy
- Metonymy
- Synonyms

Homonymy

a word form that has more than one sense

Homonymy

a word form that has more than one sense

bank

Homonymy

a word form that has more than one sense

bank

bank^1 = financial institution

bank^2 = sloping mound by river

Polysemy

two sense are related semantically

Polysemy

two sense are related semantically

bank

bank^3 : building housing financial institution
bank^1 : financial institution

Polysemy

two sense are related semantically

bank

bank³ : building housing financial institution
bank¹ : financial institution

The *bank³* is on the corner of 1st street

I put my money in the *bank¹*

Metonymy

using one aspect of a concept to refer to another aspect of the concept

Metonymy

using one aspect of a concept to refer to another aspect of the concept

WHITE HOUSE

PRESIDENT AND HIS ADMINISTRATION

Metonymy

using one aspect of a concept to refer to another aspect of the concept

WHITE HOUSE

PRESIDENT AND HIS ADMINISTRATION

The *White House* said Monday it has drafted legislation with the Justice Department

Synonymy

Terms that refer to the same concept

Synonym

Terms that refer to the same concept

couch/sofa

vomit/throwup

car/automobile

Antonymy

Terms that come from opposite concept

Antonymy

Terms that come from opposite concept

light and dark

hot and cold

black and white

Senses (or concepts)

Semantic representation of a term

Senses (or concepts)

Semantic representation of a term

dog

Senses (or concepts)

Semantic representation of a term

How do we know if a word has more than one sense?

Zeugma

technique for determining if two sense are distinct

Zeugma

technique for determining if two sense are distinct

Does American Airlines *serve* breakfast?
Does American Airlines *serve* Philadelphia?

The QUESTION: IS *SERVE* both of the same sense?

Zeugma

technique for determining if two sense are distinct

Does American Airlines *serve* breakfast?
Does American Airlines *serve* Philadelphia?

The QUESTION: IS *SERVE* both of the same sense?

Does American Airlines serve breakfast and Philadelphia?

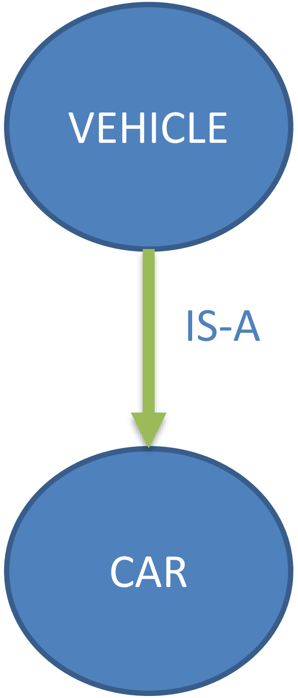
Zeugma

Concept
serve

Concept
serve

Relations between senses (concepts)

Hyponymy (is-a)

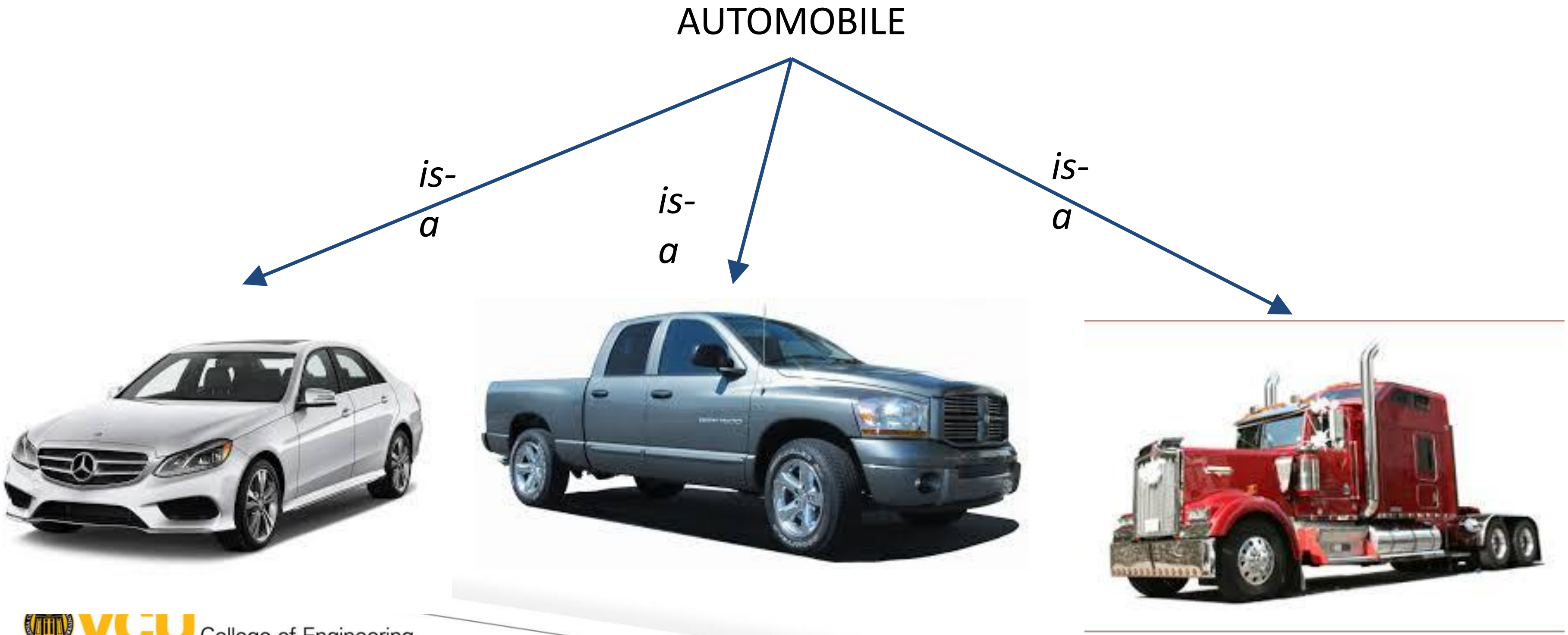


denotes subclass of the other

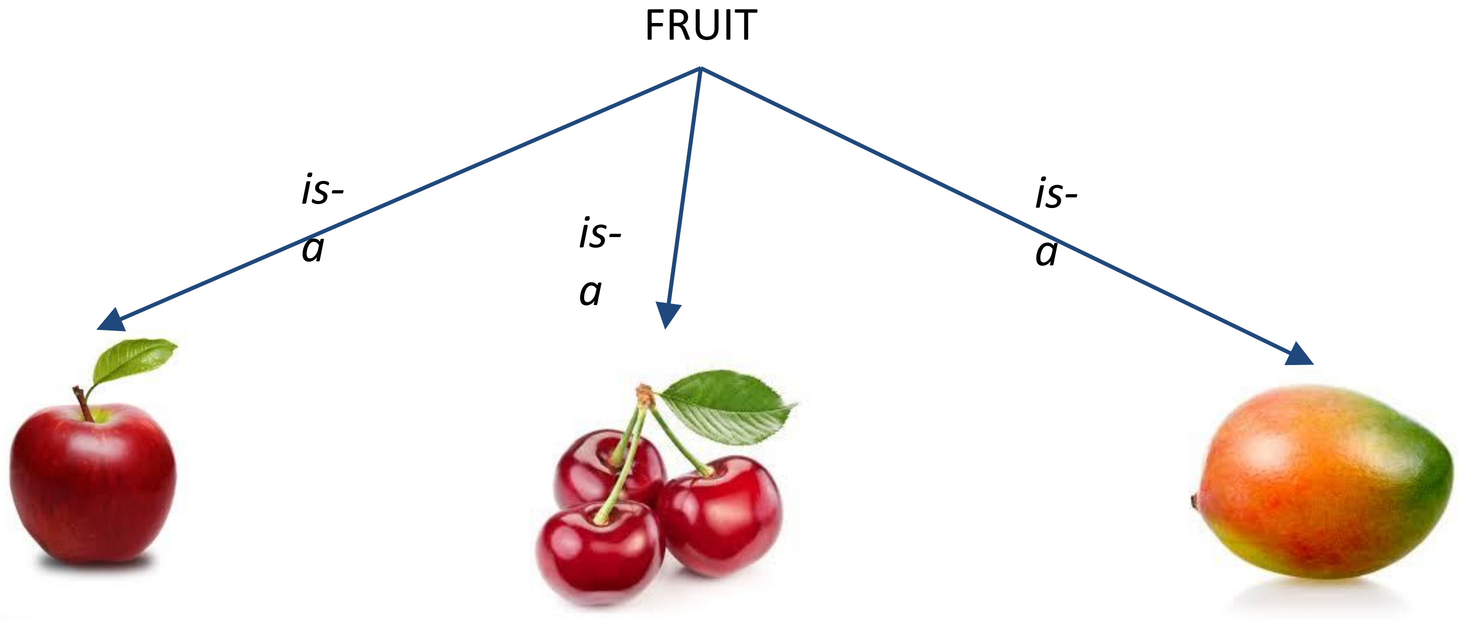
vehicle is a hypernym of *car*

car is a hyponym of *vehicle*

Taxonomy



Taxonomy



Lexical Database

Lexicon that contains relationships between the senses

Lexical Databases:

- general English domain
  - WordNet
- biomedical domain
  - Unified Medical Language System

WordNet

- developed by Fellbaum, 1998
- consists of three separate databases
  - nouns
  - verbs
  - adjectives/adverbs
- closed class words (e.g. ‘the’) are not included
- http://wordnetweb.princeton.edu/perl/webwn

WordNet 3.0

contains:

- 117,097 nouns
- 11,488 verbs
- 22,141 adjectives
- 4,601 adverbs

Ambiguity (# senses per word)

- noun: 1.23 senses
- verb: 2.16 senses

Senses of “bass” in Wordnet

Noun

- S: (n) **bass** (the lowest part of the musical range)
- S: (n) **bass**, bass part (the lowest part in polyphonic music)
- **S: (n) bass, basso (an adult male singer with the lowest voice)**
- S: (n) sea bass, **bass** (the lean flesh of a saltwater fish of the family Serranidae)
- S: (n) freshwater bass, **bass** (any of various North American freshwater fish with lean flesh (especially of the genus Micropterus))
- S: (n) **bass**, bass voice, basso (the lowest adult male singing voice)
- S: (n) **bass** (the member with the lowest range of a family of musical instruments)
- S: (n) **bass** (nontechnical name for any of numerous edible marine and freshwater spiny-finned fishes)

Adjective

- S: (adj) **bass**, deep (having or denoting a low vocal or instrumental range) *"a deep voice"; "a bass voice is lower than a baritone voice"; "a bass clarinet"*

“Synset”:

- POS
- Definition
- Synonym Terms
- Sentence containing the word

Western Science

VCU College of Engineering

Computational Lexical Semantics

How are terms and concepts related

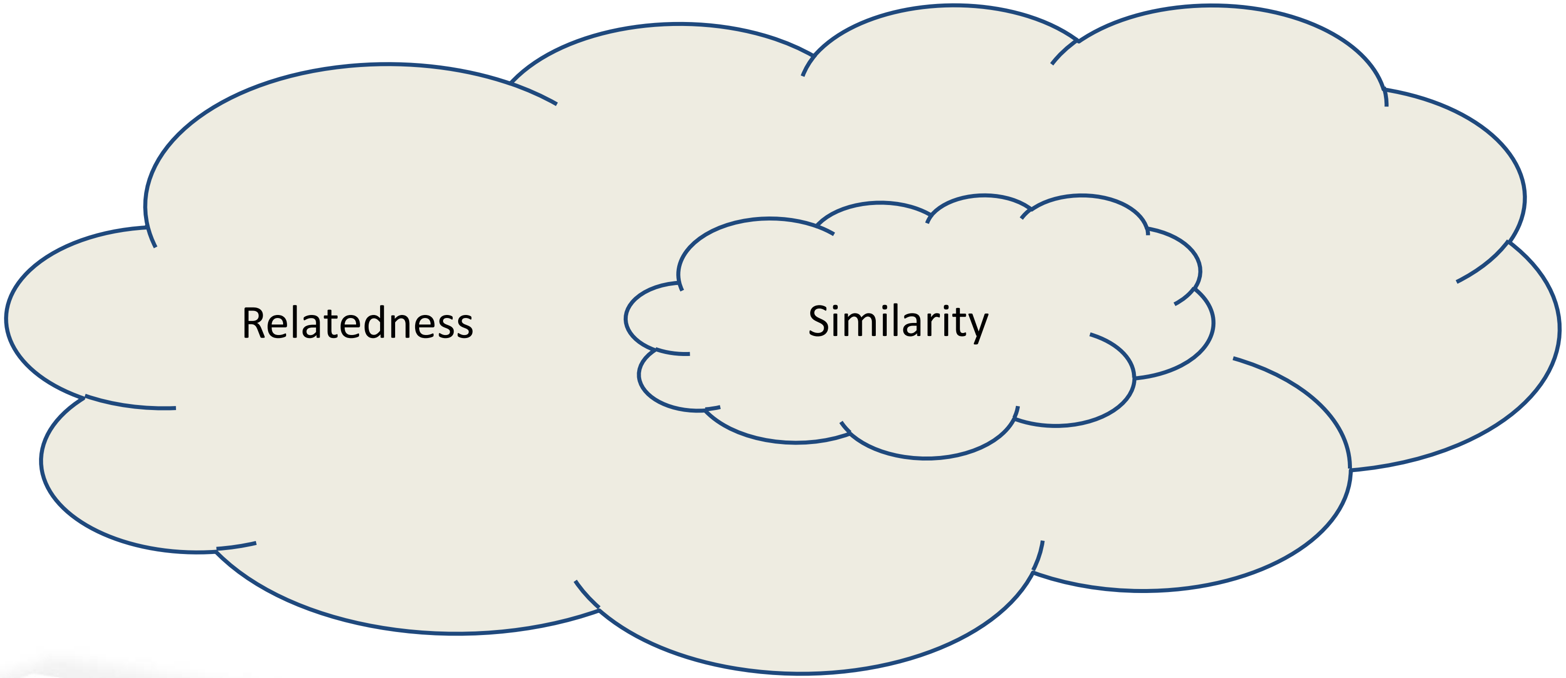
What terms are associated with what concepts

Computational Lexical Semantics

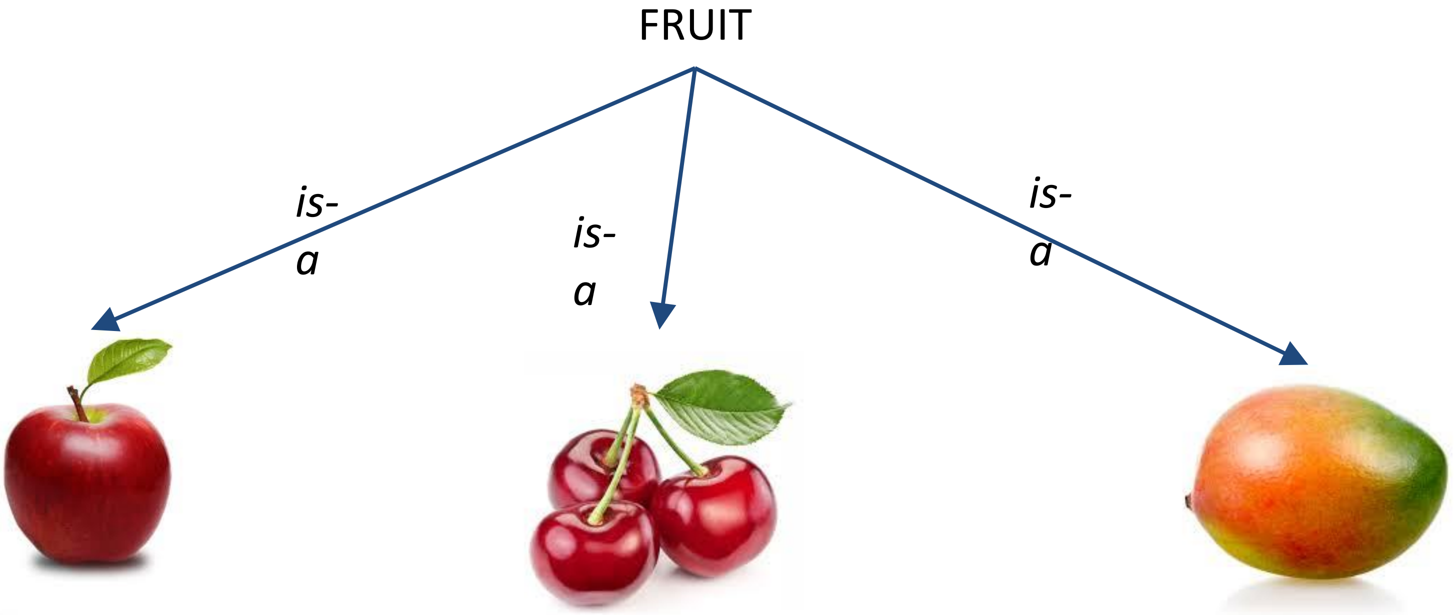
How are terms and concepts related

What terms are associated with what concepts

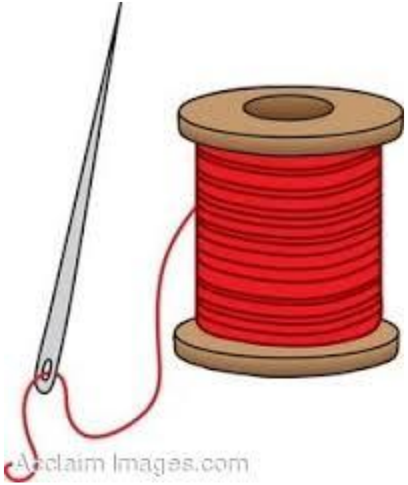
Semantic Similarity/Relatedness



Similarity: is-a relationships



Relatedness



Heart Disease

Quantify

the degree of similarity or relatedness between two concepts (or senses)

Quantify

the degree of similarity or relatedness between two concepts (or senses)

- Similarity
  - Path-based
  - Information-content based

- Relatedness
  - Gloss (or definition) based
  - Distributional Methods

Path-based measures

utilize *is-a* relations from lexical database
(e.g. WordNet; UMLS)

Path-based measures

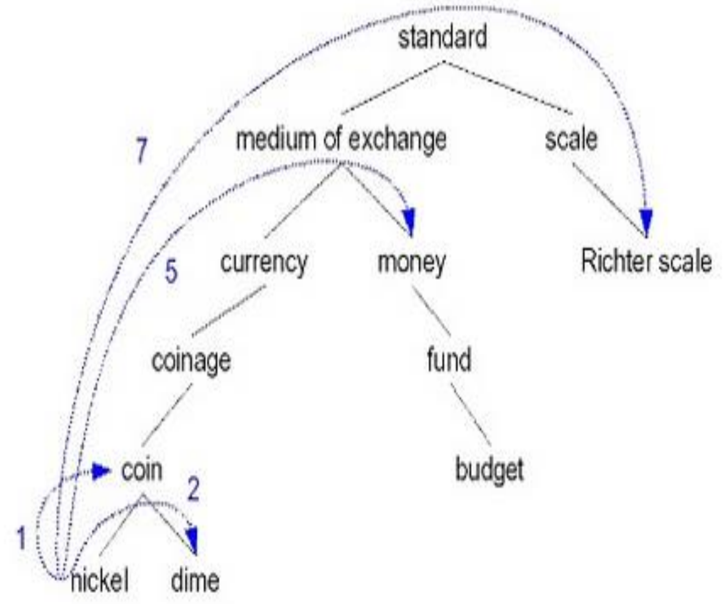
utilize *is-a* relations from lexical database
(e.g. WordNet)

- Path measure (path)
- Wu and Palmer (wup)
- Leacock and Chodorow (lch)

Path measure (path)

$$sim_{path}(c_1, c_2) = \frac{1}{minpath(c_1, c_2)}$$

$$sim_{path}(nickle, dime) = \frac{1}{2}$$

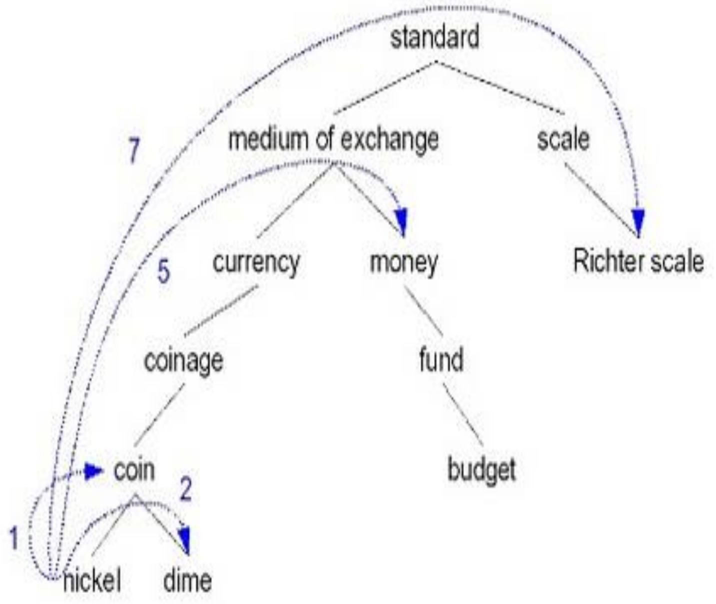


Leacock & Chodorow (lch)

$$sim_{lch}(c_1, c_2) = -\log(\frac{minpath(c_1, c_2)}{2 * D}$$

where D = depth of the taxonomy

$$sim_{lch}(nickle, dime) = -\log(\frac{2}{2 * D})$$

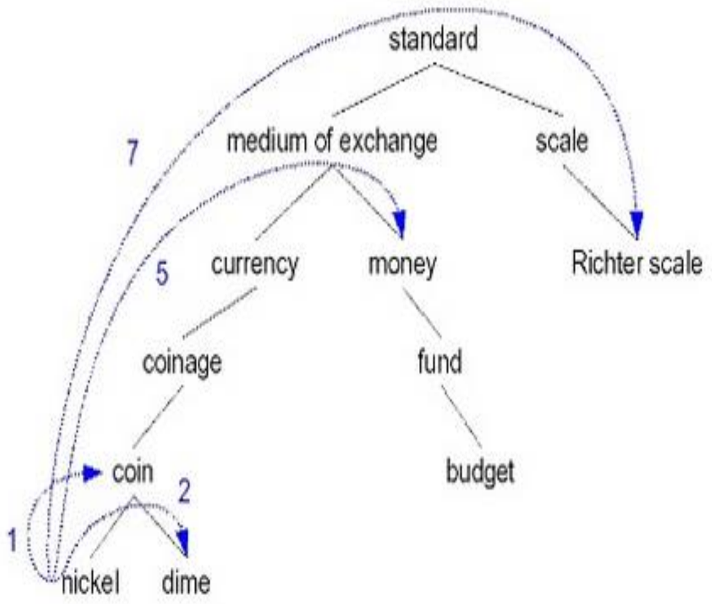


Wu & Palmer (wup)

$$sim_{wup}(c_1, c_2) = 2 * \left(\frac{depth(LCS(c_1, c_2))}{depth(c_1) + depth(c_2)}\right)$$

depth(c) = how deep it is in the taxonomy
so let standard be the root
of our taxonomy

$LCS(c_1, c_2)$ = Least Common Subsumer

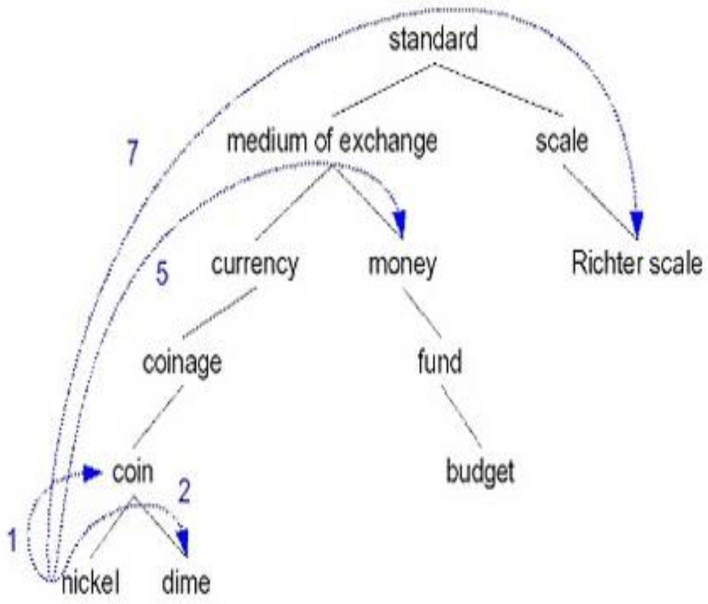


Wu & Palmer (wup)

$$sim_{wup}(c_1, c_2) = 2 * \left(\frac{depth(LCS(c_1, c_2))}{depth(c_1) + depth(c_2)}\right)$$

depth(nickle) = 6

$LCS(nickle, dime)$ = *coin*



Path-based measures

Path-based measures use the path information between the concepts to quantify their similarity

So their similarity is based on their co-location to each other in the taxonomy

What I want you to remember about path-based similarity measures!!

Information content (IC) measures

Utilize path information **but also incorporate the probability of the concept occurring in text**

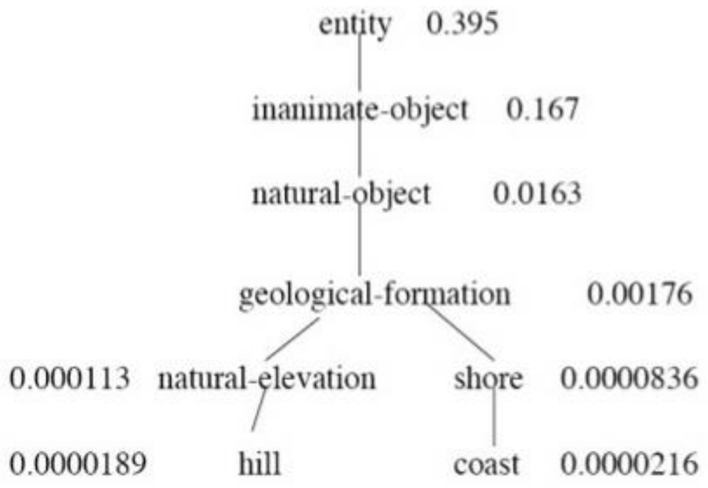
Information Content

The lower a node is in the hierarchy, the lower the probability of seeing it in text

IC(c) = -log(P(c))

IC(hill) = -log(0.0000189)

WordNet hieararchy augmented with probabilities P(C)

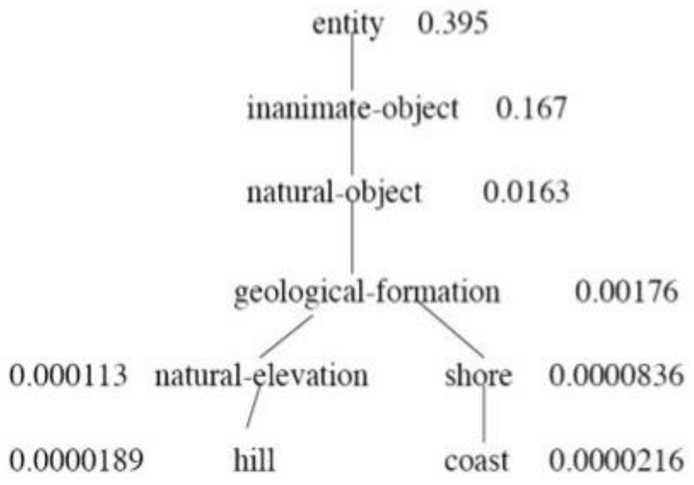


Information content (IC) measures

Utilize path information but also
incorporate probability of the concept occurring in text

- Resnik
- Lin
- Jiang & Conrath

Resnik (res)



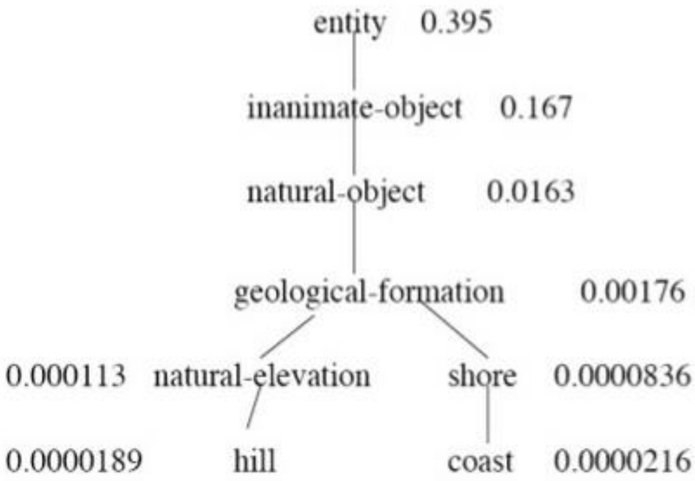
$$sim_{res}(c_1, c_2) = -\log(P(LCS(c_1, c_2)))$$

$$\begin{aligned} sim_{res}(hill, coast) &= -\log(P(LCS(hill, coast)) \\ &= -\log(P(LCS(geological\ formation)) \\ &= -\log(P(0.00179)) \end{aligned}$$

Lin (lin) & Jiang & Conrath (jcn)

$$sim_{lin}(c_1, c_2) = \frac{-\mathrm{IC}(\mathrm{LCS}(c_1,c_2))}{\mathrm{IC}(c_1) + \mathrm{IC}(c_2)}$$

$$sim_{jcn}(c_1, c_2) = \frac{1}{\mathrm{IC}(c_1) + \mathrm{IC}(c_2) - 2(\mathrm{IC}(\mathrm{LCS}(c_1,c_2)))}$$



IC-based measures

IC-based measures use the path information between the concepts to quantify their similarity
and
the probability of the concept occurring

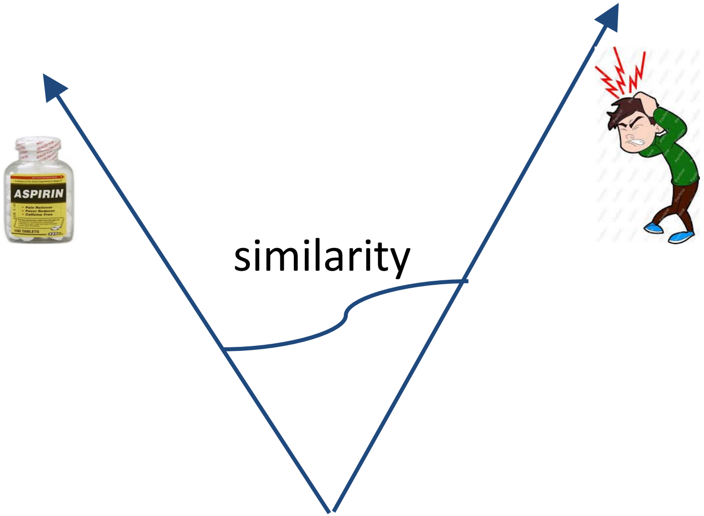
What I want you to remember about IC-based similarity measures!!

Relatedness measures

Use the context in which the terms occur to quantify their relatedness

Relatedness measures

Use the context in which the terms occur to quantify their relatedness

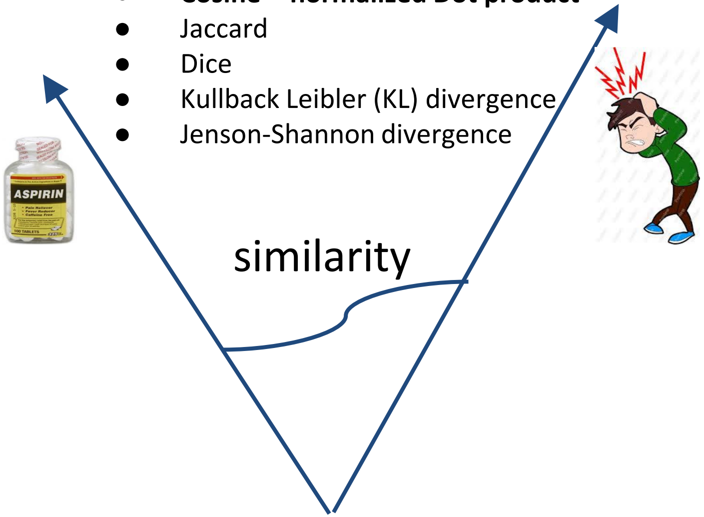


Relatedness measures

Use the context in which the terms occur to quantify their relatedness

Vector Metrics:

- Manhatten DistanceEuclidean Distance
- **Cosine - normalized Dot product**
- Jaccard
- Dice
- Kullback Leibler (KL) divergence
- Jenson-Shannon divergence



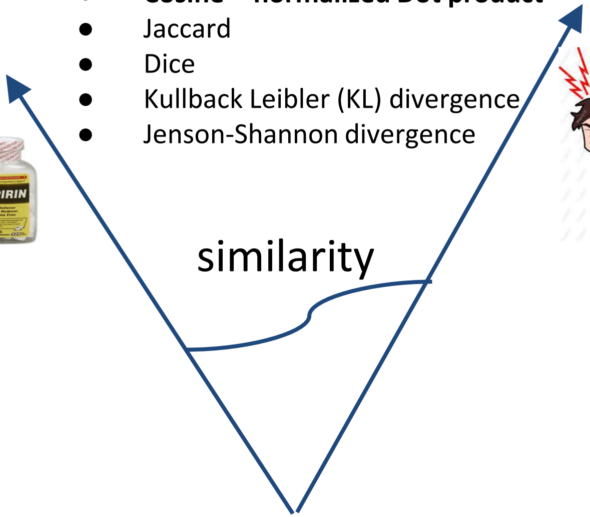
Relatedness measures

Use the context in which the terms occur to quantify their relatedness

How to represent the terms in vector form?

Vector Metrics:

- Manhatten DistanceEuclidean Distance
- **Cosine - normalized Dot product**
- Jaccard
- Dice
- Kullback Leibler (KL) divergence
- Jenson-Shannon divergence



Cosine

Normalized dot product

$$Cosine(\overrightarrow{x}\,\overrightarrow{y}) = \frac{\overrightarrow{x} \cdot \overrightarrow{y}}{|\overrightarrow{x}||\overrightarrow{y}|}$$

Cosine

Normalized dot product

$$Cosine(\overrightarrow{x}\,\overrightarrow{y}) = \frac{\overrightarrow{x} \cdot \overrightarrow{y}}{|\overrightarrow{x}||\overrightarrow{y}|}$$

$$dotproduct(\overrightarrow{x}, \overrightarrow{y}) = \overrightarrow{x} \cdot \overrightarrow{y} = \sum_{i=1}^{n} x_i \times y_i$$

Adding the product of what they have in common

Cosine

Normalized dot product

$$Cosine(\overrightarrow{x}, \overrightarrow{y}) = \frac{\overrightarrow{x} \cdot \overrightarrow{y}}{|\overrightarrow{x}||\overrightarrow{y}|}$$

$$vectorlength = |\overrightarrow{x}| = sqrt(\sum_{i=1}^{n} x_i^2)$$

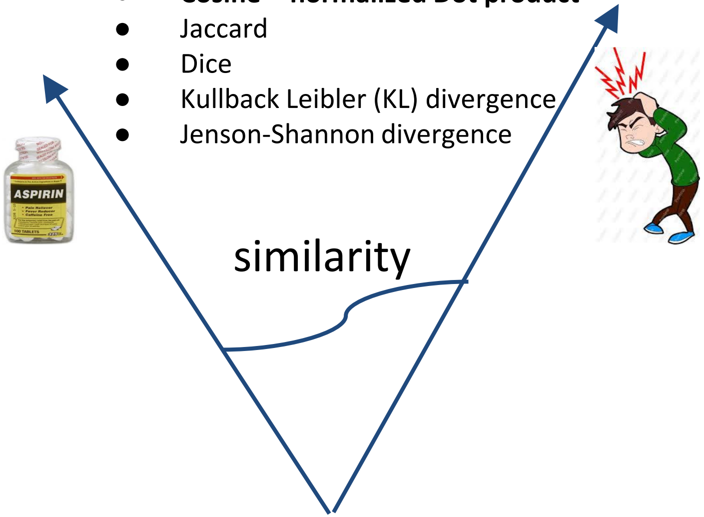
Dividing by the total number of items

Relatedness measures

Use the context in which the terms occur to quantify their relatedness

Vector Metrics:

- Manhatten DistanceEuclidean Distance
- **Cosine - normalized Dot product**
- Jaccard
- Dice
- Kullback Leibler (KL) divergence
- Jenson-Shannon divergence



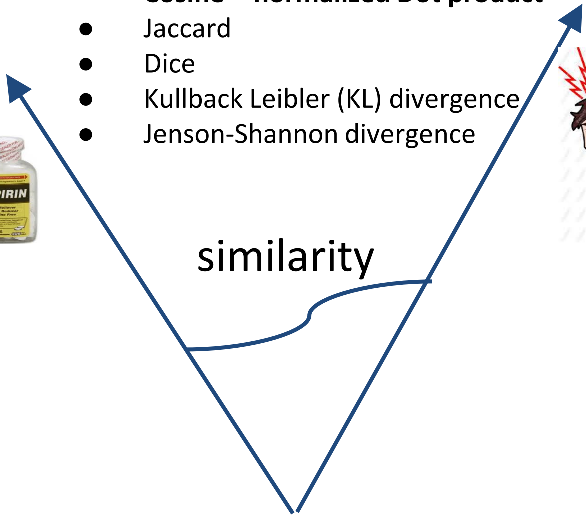
Relatedness measures

Use the context in which the terms occur to quantify their relatedness

Where does this context come from?

Vector Metrics:

- Manhatten DistanceEuclidean Distance
- **Cosine - normalized Dot product**
- Jaccard
- Dice
- Kullback Leibler (KL) divergence
- Jenson-Shannon divergence

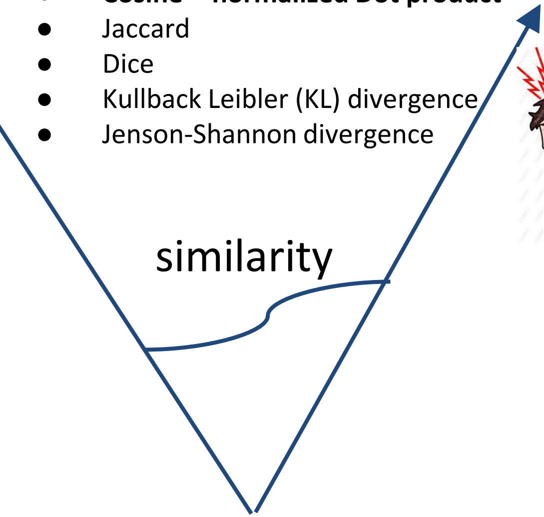


Relatedness measures

Use the context in which the terms occur to quantify their relatedness

Vector Metrics:

- Manhatten DistanceEuclidean Distance
- **Cosine - normalized Dot product**
- Jaccard
- Dice
- Kullback Leibler (KL) divergence
- Jenson-Shannon divergence



Relatedness measures

Use the context in which the terms occur to quantify their relatedness

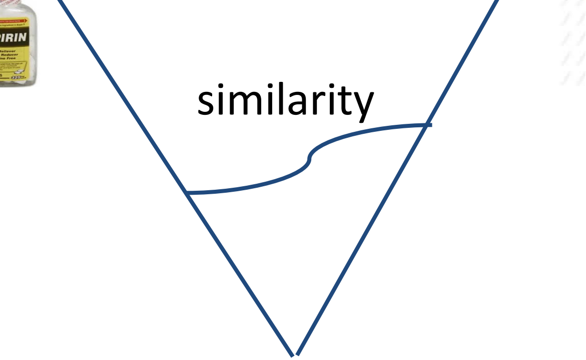
PRINCETON UNIVERSITY

WordNet

A lexical database for English

Vector Metrics:

- Manhatten DistanceEuclidean Distance
- **Cosine - normalized Dot product**
- Jaccard
- Dice
- Kullback Leibler (KL) divergence
- Jenson-Shannon divergence



Term representation

- First order co-occurrence vectors
- Second order co-occurrence vectors
- Word embeddings
- Contextualized word embeddings
- Singular Value Decomposition

First order word vectors

first order context vectors

	W_1	W_2	W_3	...	W_n
W_1	0	2	5	...	11
W_2	1	0	4	...	13
W_m	6	1	0	...	9

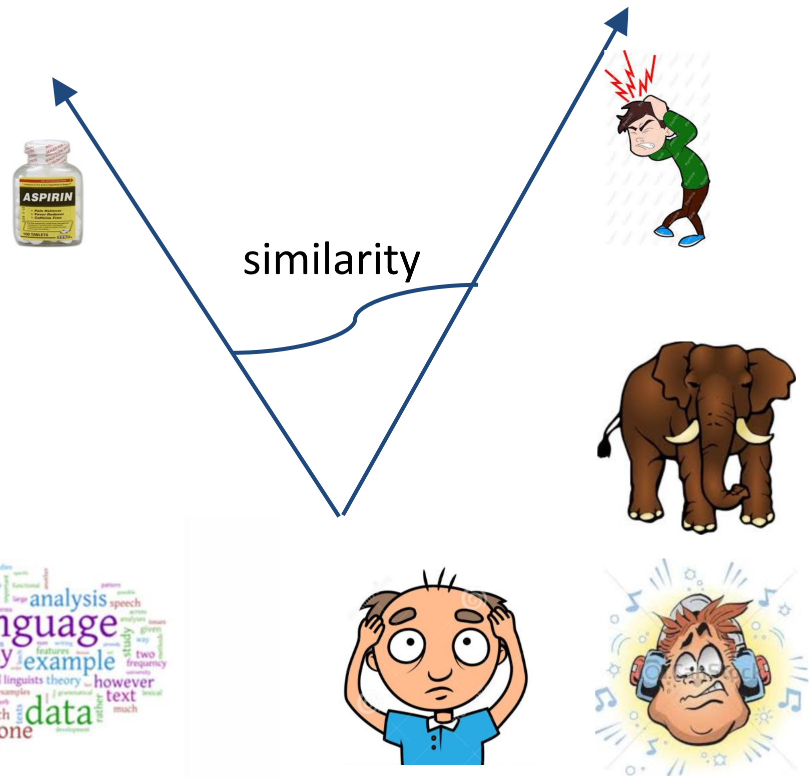
n x m co-occurrence matrix

First order word vectors

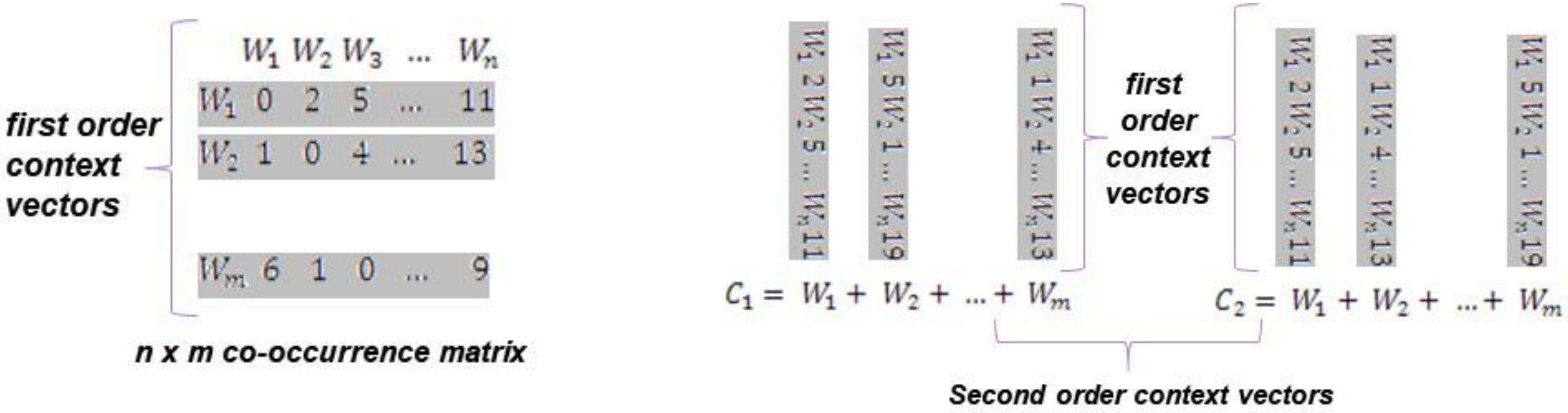
first order context vectors

	W_1	W_2	W_3	...	W_n
W_1	0	2	5	...	11
W_2	1	0	4	...	13
W_m	6	1	0	...	9

n x m co-occurrence matrix

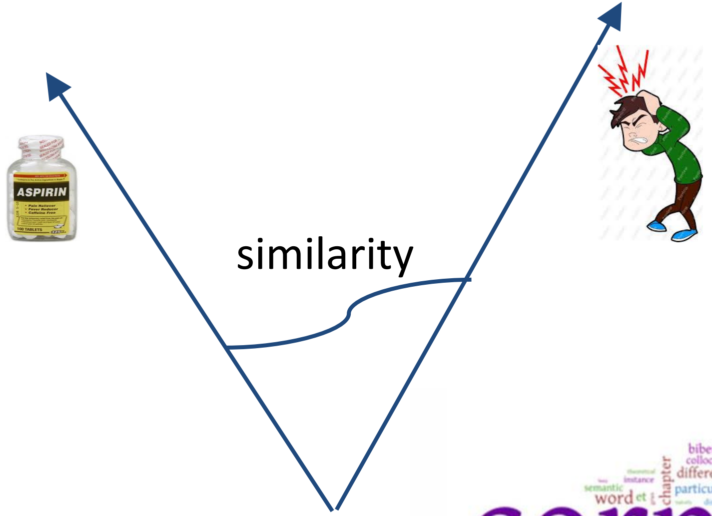


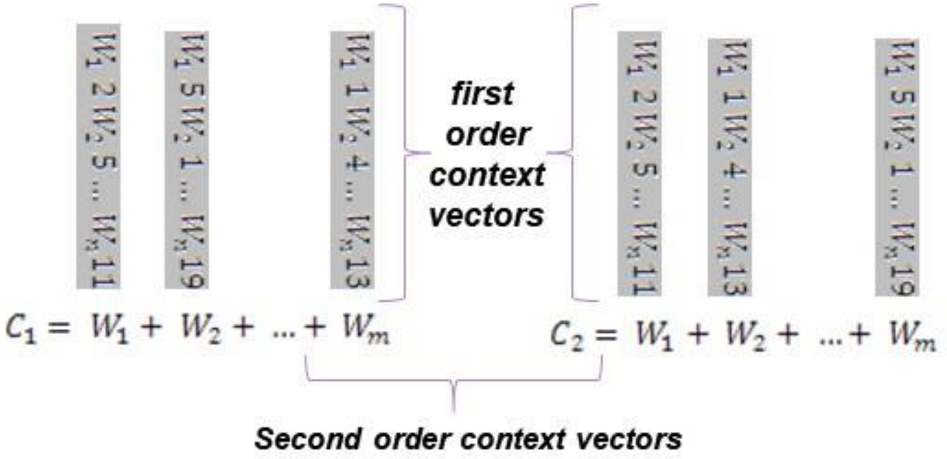
Second order word vectors



PRINCETON UNIVERSITY
WordNet
A lexical database for English

Second order word vectors





PRINCETON UNIVERSITY
WordNet
A lexical database for English

Relatedness based measures

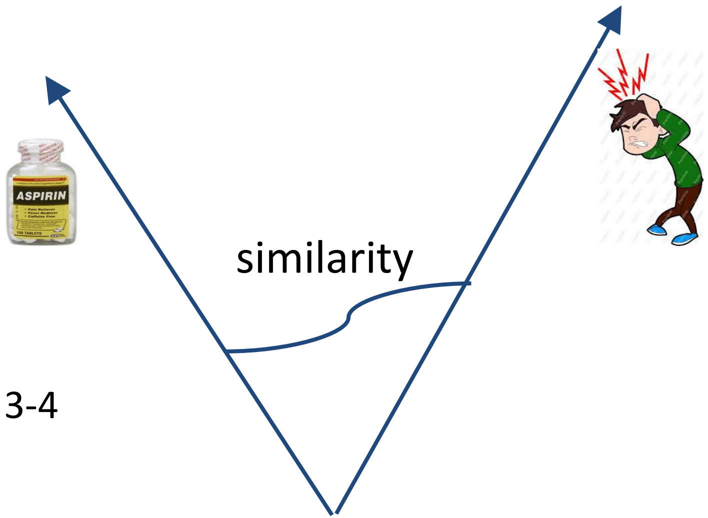
They are all using the context in which words appear together to quantify their relatedness

What I want you to remember about relatedness measures!!

Evaluation of Similarity and Relatedness Measures

WORD SIMILARITY

- Dataset:
  - Rubenstein & Goodenough (1965)
    - 51 human subjects
    - 65 word Pairs
    - evaluation scale: 0-4
  - MILLER & CHARLES (1991)
    - 38 human subjects
    - 30 pairs from the original 65
      - 10 from 0-1, 10 from 1-3, 10 from 3-4
    - evaluation scale: 1-3



Why would we use this?

Information retrieval

retrieve documents whose words have similar meanings to the query words

Summarisation

can we substitute one sentence for another in a particular context

Word Sense Disambiguation

can we substitute one sentence for another in a particular context

Step back to feature representations

To address the question: What is SVD?

Features Representation

An instance consists of an n-dimensional array where each element in the array represents a feature.

What are these features?

Lexical
Syntactic
Semantic
Domain Knowledge

Lexical : Bag of words

instance	Word 1	Word 2	Word 3	Word 4	Word 5	…	Word m
1	0	0	1	1	0		1
2	0	0	1	0	6		0
3	0	0	0	1	0		0
4	1	1	0	0	0		0
5	0	0	0	0	0		1
…							-
n	0	0	0	1	0		0

Lexical : Bag of words

instance	Word 1	Word 2	Word 3	Word 4	Word 5	…	Word m
1	0	0	1	1	0		1
2	0	0	1	0	6		0
3	0	0	0	1	0		0
4	1	1	0	0	0		0
5	0	0	0	0	0		1
…							-
n	0	0	0	1	0		0

Each instance is represented based on its row

1	0	0	1		0		6

Lexical : Bag of words

instance	ant	ankle	ardvark	art	ask	…	zebra
1	0	0	1	1	0		1
2	0	0	1	0	1		0
3	0	0	0	1	0		0
4	1	1	0	0	0		0
5	0	0	0	0	0		1
…							-
n	0	0	0	1	0		0

Each instance is represented based on its row

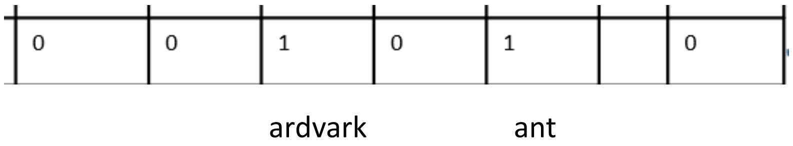
0	0	1	1	0		1

ardvark ant zebra

Lexical : Bag of words

instance	ant	ankle	ardvark	art	ask	…	zebra
1	0	0	1	1	0		1
2	0	0	1	0	1		0
3	0	0	0	1	0		0
4	1	1	0	0	0		0
5	0	0	0	0	0		1
…							-
n	0	0	0	1	0		0

Each instance is represented based on its row



Lexical : Bag of words

instance	ant	ankle	ardvark	art	ask	…	zebra
1	0	0	1	1	0		1
2	0	0	1	0	1		0
3	0	0	0	1	0		0
4	1	1	0	0	0		0
5	0	0	0	0	0		1
…							-
n	0	0	0	1	0		0

this can be a binary vector

Lexical : Bag of words

instance	ant	ankle	ardvark	art	ask	…	zebra
1	0	0	2	1	0		4
2	0	0	1	0	1		0
3	0	0	0	7	0		0
4	2	1	0	0	0		0
5	0	0	0	0	0		8
…							-
n	0	0	0	3	0		0

this can be a frequency vector

Lexical : Bag of words

instance	ant	ankle	ardvark	art	ask	…	zebra
1	0	0	2	1	0		4
2	0	0	1	0	1		0
3	0	0	0	7	0		0
4	2	1	0	0	0		0
5	0	0	0	0	0		8
…							-
n	0	0	0	3	0		0

this can be a frequency vector

0	0	0	0	0		8

zebra

Lexical : Bag of words

instance	ant	ankle	ardvark	art	ask	…	zebra
1	0	0	1.8	.5	0		4
2	0	0	.3	0	.3		0
3	0	0	0	5.6	0		0
4	1.5	.6	0	0	0		0
5	0	0	0	0	0		7.6
…							-
n	0	0	0	2.3	0		0

this can be a TF-IDF vector

$$TF - IDF_{i=} tf_i \ * \ \log(\frac{N}{df_i})$$

TF-IDF

Word	df	idf
Romeo	1	1.57
salad	2	1.27
Falstaff	4	0.967
forest	12	0.489
battle	21	0.246
wit	34	0.037
fool	36	0.012
good	37	0
sweet	37	0

$$TF-IDF_{i=}\, tf_i \, * \, \log(\frac{N}{df_i})$$

Corpus = Shakespeare's 37 plays

TF-IDF

Word	df	idf
Romeo	1	1.57
salad	2	1.27
Falstaff	4	0.967
forest	12	0.489
battle	21	0.246
wit	34	0.037
fool	36	0.012
good	37	0
sweet	37	0

$$TF - IDF_{i=} tf_i * \log(\frac{N}{df_i})$$

Corpus = Shakespeare's 37 plays

TF-IDF

Word	df	idf
Romeo	1	1.57
salad	2	1.27
Falstaff	4	0.967
forest	12	0.489
battle	21	0.246
wit	34	0.037
fool	36	0.012
good	37	0
sweet	37	0

$$TF - IDF_{i=} tf_i \; * \; \log(\frac{N}{df_i})$$

Corpus = Shakespeare's 37 plays

Feature Vector

instance	ant	ankle	ardvark	art	ask	…	zebra
1	0	0	1	1	0		1
2	0	0	1	0	1		0
3	0	0	0	1	0		0
4	1	1	0	0	0		0
5	0	0	0	0	0		1
…							-
n	0	0	0	1	0		0

Lexical
Syntactic
Semantic
Domain Knowledge

Feature Vector

instance	NOUN	ADJ	VERB	ADV	ask	…	zebra
1	0	0	1	1	0		1
2	0	0	1	0	1		0
3	0	0	0	1	0		0
4	1	1	0	0	0		0
5	0	0	0	0	0		1
…							-
n	0	0	0	1	0		0

Lexical
Syntactic
Semantic
Domain Knowledge

Feature Vector

instance	WN SYNSET	WN SYNSET	WN SYNSET	art	ask	…	zebra
1	0	0	1	1	0		1
2	0	0	1	0	1		0
3	0	0	0	1	0		0
4	1	1	0	0	0		0
5	0	0	0	0	0		1
…							-
n	0	0	0	1	0		0

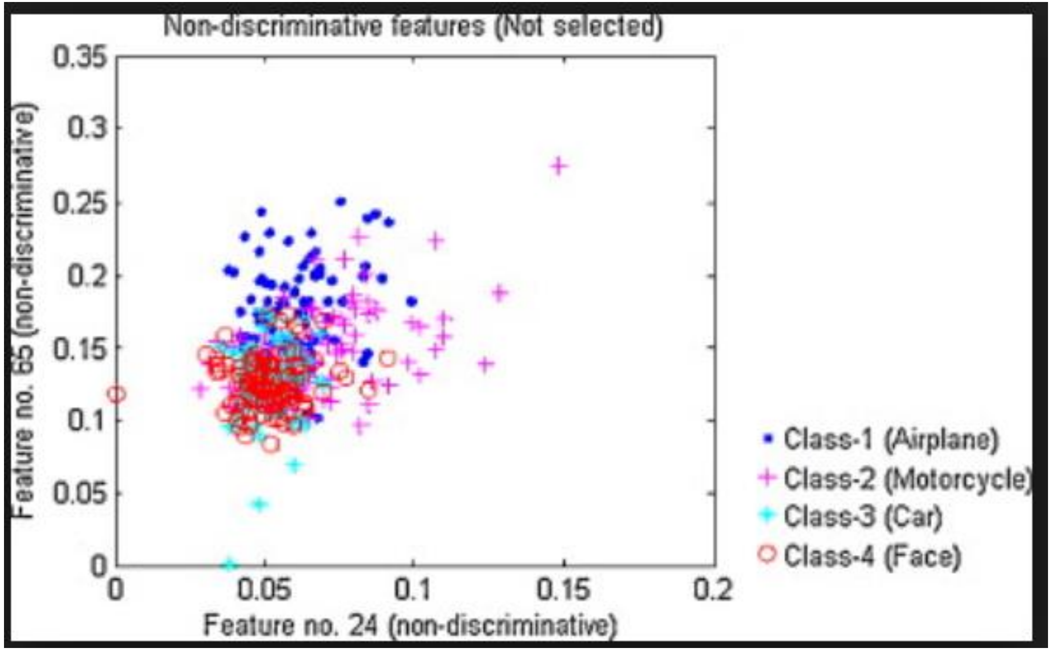
Lexical
Syntactic
Semantic
Domain Knowledge

Feature Vector

instance	DISEASE	DRUG	ADE	art	ask	…	zebra
1	0	0	1	1	0		1
2	0	0	1	0	1		0
3	0	0	0	1	0		0
4	1	1	0	0	0		0
5	0	0	0	0	0		1
…							-
n	0	0	0	1	0		0

Lexical
Syntactic
Semantic
Domain Knowledge

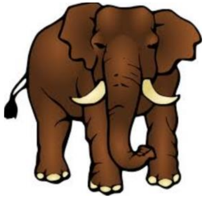
Discriminative Features

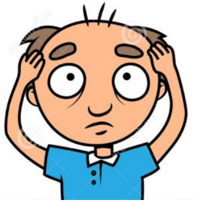


Difficulty: Large Sparse and Noisy

- Feature Extraction
  - Second order co-occurrence vectors [Schütze 1998]
  - Singular Value Decomposition (SVD) [Deerwester, et al 1990]
    - aka: Latent Semantic Indexing (LSI)
    - aka: Latent Semantic Analysis (LSA)

  - Principle Component Analysis (PCA)
  - Word embeddings [Mikolov, et al 2013]
  - Contextualized word embeddings (2019)





1st order co-occurrence vectors

instanc e	abandon	ability	able	…	ant	…	zone
1	0	0	1		0		6
2	0	0	5		6		0
3	0	1	1		0		2
4							
5	0	0	0		0		7
…							
n	2	0	6		0		5

Each instance is represented based on its row

1	0	0	1		0		6

2st order co-occurrence vectors

	abandon	ability	able	…	ant	…	zone
abandon	0	3	4		0		10
ability	0	0	2		6		0
able	0	1	0		0		8
…							
ant	0	1	0		0		2
…							
zone	1	0	4		0		0

instance	**abandon**	**ability**	**able**	**…**	**ant**	**…**	**zone**
1	0	0	1		0		6
2	0	0	5		6		0
3	0	1	1		0		2
4							
5	0	0	0		0		7
…							
n	2	0	6		0		5

Represents each co-occurring word in the 1^{st} order vector as a vector itself

instance	**abandon**	**ability**	**able**	**…**	**ant**	**…**	**zone**
1	0	0	1		0		6

2st order co-occurrence vectors

	abandon	ability	able	…	ant	…	zone
abandon	0	0	1		0		6
ability	0	0	5		6		0
able	0	1	1		0		2
…							
ant	0	0	0		0		7
…							
zone	2	0	6		0		5

able	0	1	1	…	0	…	2

zone	2	0	6	…	0	…	5

instance	abandon	ability	able	…	ant	…	zone
1	0	0	1		0		6

2st order co-occurrence vectors

	abandon	ability	able	…	ant	…	zone
abandon	0	0	1		0		6
ability	0	0	5		6		0
able	0	1	1		0		2
…							
ant	0	0	0		0		7
…							
zone	2	0	6		0		5

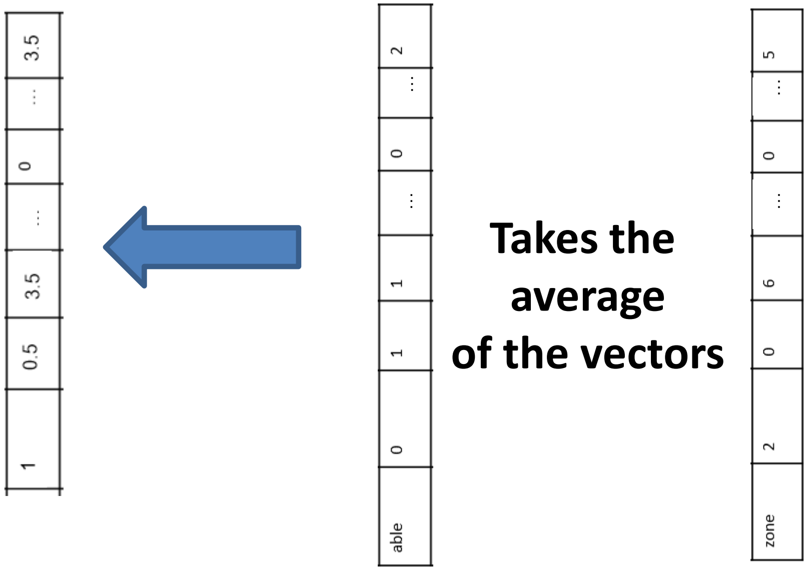
able	0	1	1	…	0	…	2
zone	2	0	6	…	0	…	5

Takes the average of the vectors

instance	abandon	ability	able	…	ant	…	zone
1	0	0	1		0		6

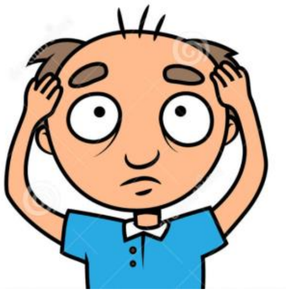
2st order co-occurrence vectors

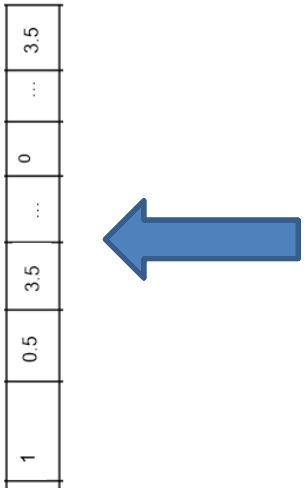
	abandon	abilit y	able	…	ant	…	zone
abandon	0	0	1		0		6
ability	0	0	5		6		0
able	0	1	1		0		2
…							
ant	0	0	0		0		7
…							
zone	2	0	6		0		5

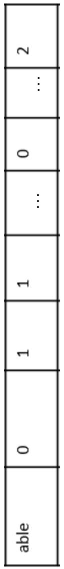


instance	abandon	ability	able	…	ant	…	zone
1	0	0	1		0		6

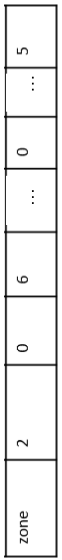
2st order co-occurrence vectors







Takes the average of the vectors



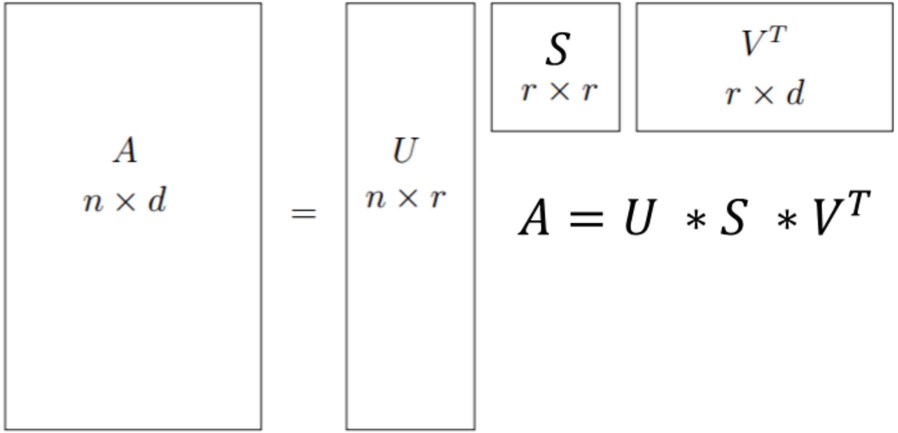
instance	abandon	ability	able	…	ant	…	zone
1	0	0	1		0		6

Singular Value Decomposition

technique to decompose a matrix into a product of three simpler matrices

Singular Value Decomposition

technique to decompose a matrix into a product of three simpler matrices



$$A = U * S * V^T$$

typically decomposition can be achieved
without any loss of information

except

SVD in NLP reduces matrices from tens of thousands down to a few hundred

so

the hope is the information lost is noise causing the similarity between the words to be more apparent

TABLE 2.1
Titles for Topics on Music and Baking

Label	*Titles*
M1	*Rock* and *Roll Music* in the 1960's
M2	Different *Drum Rolls*, a Demonstration of Techniques
M3	*Drum* and Bass *Composition*
M4	A Perspective of *Rock Music* in the 90's
M5	*Music* and *Composition* of Popular Bands
B1	How to Make *Bread* and *Rolls*, a *Demonstration*
B2	*Ingredients* for Crescent *Rolls*
B3	A *Recipe* for Sourdough *Bread*
B4	A Quick *Recipe* for Pizza *Dough* using Organic *Ingredients*

Note. Keywords are in italics.

TABLE 2.2
The 10 x 9 Type-by-Document Matrix With Type Frequencies Corresponding to the Titles in Table 2.1

Types	*Documents*								
	M1	M2	M3	M4	M5	B1	B2	B3	B4
Bread	0	0	0	0	0	1	0	1	0
Composition	0	0	1	0	1	0	0	0	0
Demonstration	0	1	0	0	0	1	0	0	0
Dough	0	0	0	0	0	0	0	1	1
Drum	0	1	1	0	0	0	0	0	0
Ingredients	0	0	0	0	0	0	1	0	1
Music	1	0	0	1	1	0	0	0	0
Recipe	0	0	0	0	0	0	0	1	1
Rock	1	0	0	1	0	0	0	0	0
Roll	1	1	0	0	0	1	1	0	0

TABLE 2.3
The 10 × 9 Weighted Type-by-Document Matrix Corresponding to the Titles in Table 2.1

Types	*Documents*								
	M1	M2	M3	M4	M5	B1	B2	B3	B4
Bread	0	0	0	0	0	.474	0	.474	0
Composition	0	0	.474	0	.474	0	0	0	0
Demonstration	0	.474	0	0	0	.474	0	0	0
Dough	0	0	0	0	0	0	0	.474	.474
Drum	0	.474	.474	0	0	0	0	0	0
Ingredients	0	0	0	0	0	0	.474	0	.474
Music	.347	0	0	.347	.347	0	0	0	0
Recipe	0	0	0	0	0	0	0	.474	.474
Rock	.474	0	0	.474	0	0	0	0	0
Roll	.256	.256	0	0	0	.256	.256	0	0

TABLE 2.3
The 10 × 9 Weighted Type-by-Document Matrix Corresponding to the Titles in Table 2.1

Types	*Documents*								
	M1	M2	M3	M4	M5	B1	B2	B3	B4
Bread	0	0	0	0	0	.474	0	.474	0
Composition	0	0	.474	0	.474	0	0	0	0
Demonstration	0	.474	0	0	0	.474	0	0	0
Dough	0	0	0	0	0	0	0	.474	.474
Drum	0	.474	.474	0	0	0	0	0	0
Ingredients	0	0	0	0	0	0	.474	0	.474
Music	.347	0	0	.347	.347	0	0	0	0
Recipe	0	0	0	0	0	0	0	.474	.474
Rock	.474	0	0	.474	0	0	0	0	0
Roll	.256	.256	0	0	0	.256	.256	0	0

$$\underset{n \times d}{A} = \underset{n \times r}{U}\ \underset{r \times r}{D}\ \underset{r \times d}{V^T}$$

$$A = U * S * V^T$$

U – Word Vector matrix

D – Singular value matrix

V - Document Vector matrix

$A = U * S * V^T$

Matrix of the "singular values" of each dimension, showing what fraction of total variance is captured by each dimension

Matrix Σ-Singular Values

1.10	0	0	0	0	0	0	0	0
0	.96	0	0	0	0	0	0	0
0	0	.86	0	0	0	0	0	0
0	0	0	.76	0	0	0	0	0
0	0	0	0	.66	0	0	0	0
0	0	0	0	0	.47	0	0	0
0	0	0	0	0	0	.27	0	0
0	0	0	0	0	0	0	.17	0
0	0	0	0	0	0	0	0	.07
0	0	0	0	0	0	0	0	0

$A = U * S * V^T$

Text x Dimension matrix

Matrix V-Document Vectors

M1	.07	–.38	.53	.27	.08	.12	.20	.50	.42
M2	.17	–.54	–.41	.00	.28	.43	–.34	.22	–.28
M3	.06	–.40	–.11	–.67	–.12	.12	.49	–.23	.23
M4	.03	–.29	.55	.19	–.05	.22	–.04	–.62	–.37
M5	.03	–.29	.27	–.40	–.27	–.55	–.48	.21	–.17
B1	.31	–.36	–.36	.46	–.15	–.45	.00	–.32	.31
B2	.19	–.04	.06	–.02	.65	–.45	.41	.07	–.40
B3	.66	.17	.00	.06	–.51	.12	.27	.25	–.35
B4	.63	.27	.18	–.24	.35	.10	–.35	–.20	.37

$A = U * S * V^T$

Text x Dimension matrix

Matrix V-Document Vectors

M1	.07	−.38	.53	.27	.08	.12	.20	.50	.42
M2	.17	−.54	−.41	.00	.28	.43	−.34	.22	−.28
M3	.06	−.40	−.11	−.67	−.12	.12	.49	−.23	.23
M4	.03	−.29	.55	.19	−.05	.22	−.04	−.62	−.37
M5	.03	−.29	.27	−.40	−.27	−.55	−.48	.21	−.17
B1	.31	−.36	−.36	.46	−.15	−.45	.00	−.32	.31
B2	.19	−.04	.06	−.02	.65	−.45	.41	.07	−.40
B3	.66	.17	.00	.06	−.51	.12	.27	.25	−.35
B4	.63	.27	.18	−.24	.35	.10	−.35	−.20	.37

Word x Dimension matrix
the “semantic space” or “LSA space”

$A = U * S * V^T$

The SVD of the Weighted Type-by-Document Matrix Represented in Table 2.3

Matrix U-Type Vectors

Bread	.42	–.09	–.20	.33	–.48	–.33	.46	–.21	–.28
Composition	.04	–.34	.09	–.67	–.28	–.43	.02	–.06	.40
Demonstration	.21	–.44	–.42	.29	.09	–.02	–.60	–.29	.21
Dough	.55	.22	.10	–.11	–.12	.23	–.15	.15	.11
Drum	.10	–.46	–.29	–.41	.11	.55	.26	–.02	–.37
Ingredients	.35	.12	.13	–.17	.72	–.35	.10	–.37	–.17
Music	.04	–.35	.54	.03	–.12	–.16	–.41	.18	–.58
Recipe	.55	.22	.10	–.11	–.12	.23	–.15	.15	.11
Rock	.05	–.33	.60	.29	.02	.33	.28	–.35	.37
Roll	.17	–.35	–.05	.24	.33	–.19	.25	.73	.22

Word x Dimension matrix
the “semantic space” or “LSA space”

$$A = U * S * V^T$$

The SVD of the Weighted Type-by-Document Matrix Represented in Table 2.3

Matrix U-Type Vectors

Bread	.42	–.09	–.20	.33	–.48	–.33	.46	–.21	–.28
Composition	.04	–.34	.09	–.67	–.28	–.43	.02	–.06	.40
Demonstration	.21	–.44	–.42	.29	.09	–.02	–.60	–.29	.21
Dough	.55	.22	.10	–.11	–.12	.23	–.15	.15	.11
Drum	.10	–.46	–.29	–.41	.11	.55	.26	–.02	–.37
Ingredients	.35	.12	.13	–.17	.72	–.35	.10	–.37	–.17
Music	.04	–.35	.54	.03	–.12	–.16	–.41	.18	–.58
Recipe	.55	.22	.10	–.11	–.12	.23	–.15	.15	.11
Rock	.05	–.33	.60	.29	.02	.33	.28	–.35	.37
Roll	.17	–.35	–.05	.24	.33	–.19	.25	.73	.22

$$A = U * S * V^T$$

Word x Dimension matrix
the "semantic space" or "LSA space"

The SVD of the Weighted Type-by-Document Matrix Represented in Table 2.3

Matrix U-Type Vectors

Bread	.42	–.09	–.20	.33	–.48	–.33	.46	–.21	–.28
Composition	.04	–.34	.09	–.67	–.28	–.43	.02	–.06	.40
Demonstration	.21	–.44	–.42	.29	.09	–.02	–.60	–.29	.21
Dough	.55	.22	.10	–.11	–.12	.23	–.15	.15	.11
Drum	.10	–.46	–.29	–.41	.11	.55	.26	–.02	–.37
Ingredients	.35	.12	.13	–.17	.72	–.35	.10	–.37	–.17
Music	.04	–.35	.54	.03	–.12	–.16	–.41	.18	–.58
Recipe	.55	.22	.10	–.11	–.12	.23	–.15	.15	.11
Rock	.05	–.33	.60	.29	.02	.33	.28	–.35	.37
Roll	.17	–.35	–.05	.24	.33	–.19	.25	.73	.22

Query "Recipe for White Bread"

The SVD of the Weighted Type-by-Document Matrix Represented in Table 2.3

Matrix U-Type Vectors

Bread	.42	–.09	–.20	.33	–.48	–.33	.46	–.21	–.28
Composition	.04	–.34	.09	–.67	–.28	–.43	.02	–.06	.40
Demonstration	.21	–.44	–.42	.29	.09	–.02	–.60	–.29	.21
Dough	.55	.22	.10	–.11	–.12	.23	–.15	.15	.11
Drum	.10	–.46	–.29	–.41	.11	.55	.26	–.02	–.37
Ingredients	.35	.12	.13	–.17	.72	–.35	.10	–.37	–.17
Music	.04	–.35	.54	.03	–.12	–.16	–.41	.18	–.58
Recipe	.55	.22	.10	–.11	–.12	.23	–.15	.15	.11
Rock	.05	–.33	.60	.29	.02	.33	.28	–.35	.37
Roll	.17	–.35	–.05	.24	.33	–.19	.25	.73	.22

Query "Recipe for White Bread"

AVERAGE(Vector(Bread), Vector(Recipe))

Bread	.42	–.09	–.20	.33	–.48	–.33	.46	–.21	–.28
Recipe	.55	.22	.10	–.11	–.12	.23	.–.15	.15	.11

You can then project new instances into the semantic space

Matrix V-Document Vectors

M1	.07	–.38	.53	.27	.08	.12	.20	.50	.42
M2	.17	–.54	–.41	.00	.28	.43	–.34	.22	–.28
M3	.06	–.40	–.11	–.67	–.12	.12	.49	–.23	.23
M4	.03	–.29	.55	.19	–.05	.22	–.04	–.62	–.37
M5	.03	–.29	.27	–.40	–.27	–.55	–.48	.21	–.17
B1	.31	–.36	–.36	.46	–.15	–.45	.00	–.32	.31
B2	.19	–.04	.06	–.02	.65	–.45	.41	.07	–.40
B3	.66	.17	.00	.06	–.51	.12	.27	.25	–.35
B4	.63	.27	.18	–.24	.35	.10	–.35	–.20	.37

You can then project new instances into the semantic space

Results for the Query "Recipe for White Bread" Using a Cosine Threshold of .80

Document	*Cosine*
B2: Ingredients for Crescent Rolls	.99800
B3: A Recipe for Sourdough Bread	.90322
B1: How to make Bread and Rolls, a Demonstration	.84171
B4: A Quick Recipe for Pizza Dough using Organic Ingredients	.83396

A query for "Rock and Roll Drum Technique" would return different documents as similar.

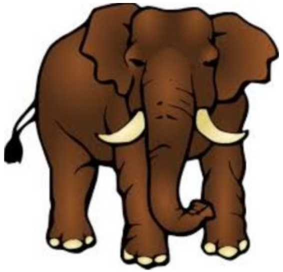
- Each word’s “meaning” is captured by its loading on all the dimensions
- Similarity of meaning between two words is determined by correlating their loadings on the factors.
- Words don’t have to have appeared in the same text to have similar meaning.
- Can also examine similarity between longer texts

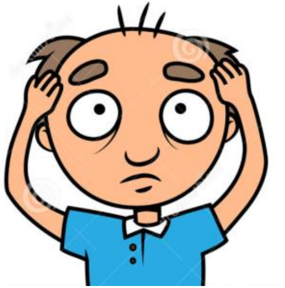
Some considerations in using LSA

- Meaning of words depends on the training corpus used.
  - E.g., “sugar” will be highly similar to “nutrition” and “diabetes” if trained on medical texts, and will be similar to “dough” and “cake” if trained on cookbooks

- Word order is not taken into account
  - E.g., according to LSA, “Donald mashed the potatoes” and “The potatoes mashed Donald” are identical.

SVD

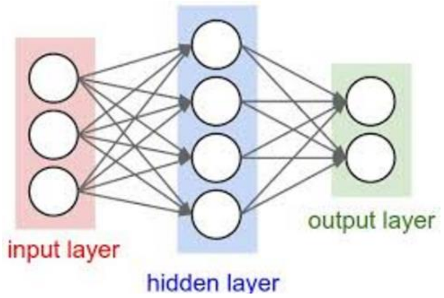




Word Embeddings

Basic idea:

– Train a Neural Network to tell us for any given some context what is the probability of a word



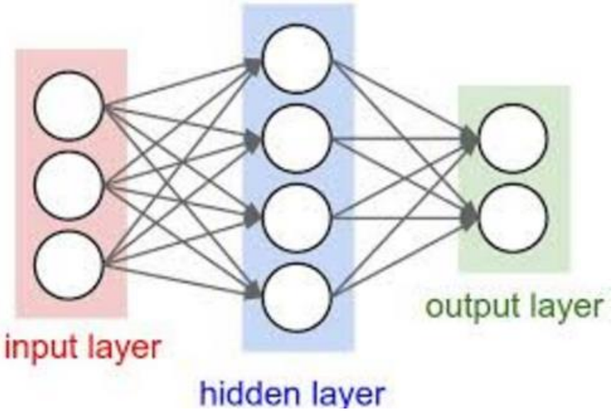
From a NY Times story...

- Stocks ...
- Stocks plunged this ....
- Stocks plunged this morning, despite a cut in interest rates
- Stocks plunged this morning, despite a cut in interest rates by the Federal Reserve, as Wall ...
- Stocks plunged this morning, despite a cut in interest rates by the Federal Reserve, as Wall Street began

...

word2vec

Basic idea: Train a Neural Network to tell us for any given some context what is the probability of a word

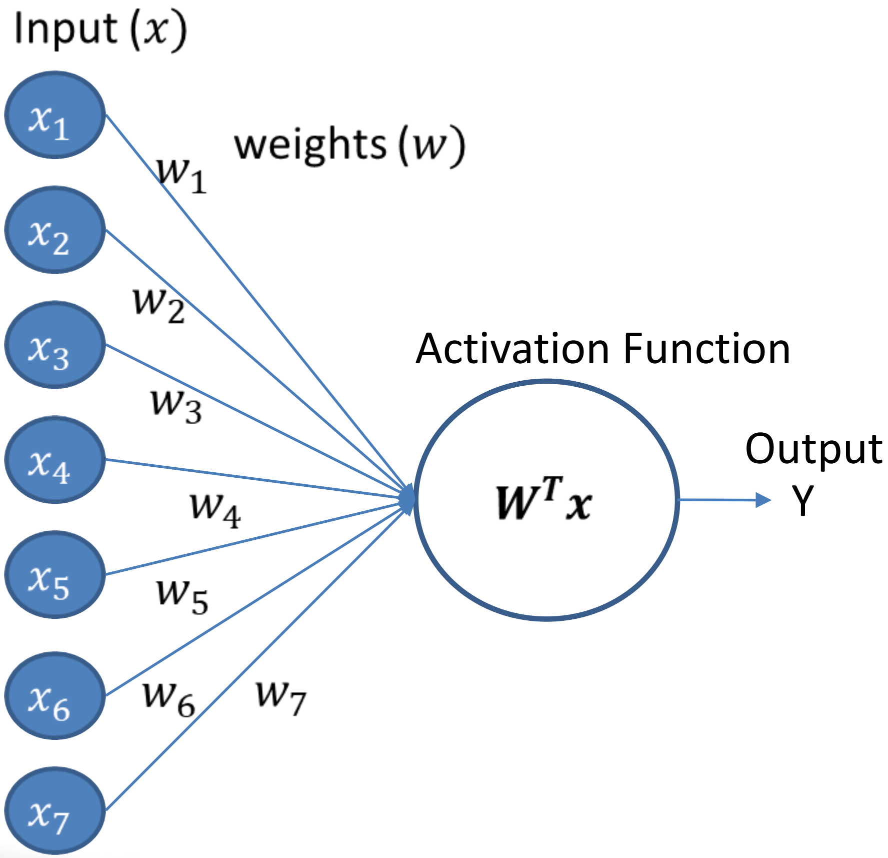


…

Neural Networks

TRAINING DATA

Positive	0	1	1	0	1	1	0
Positive	0	1	1	0	1	0	0
Negative	1	0	0	0	1	1	0
Negative	1	1	1	0	0	0	1

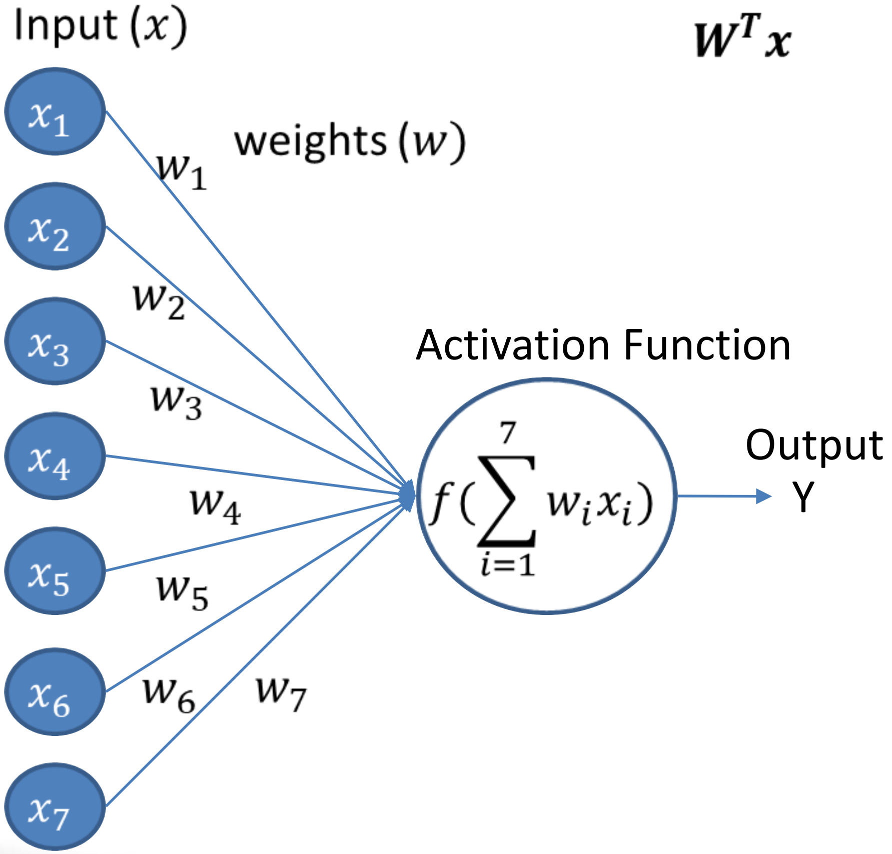


Neural Networks

TRAINING DATA

Positive	0	1	1	0	1	1	0
Positive	0	1	1	0	1	0	0
Negative	1	0	0	0	1	1	0
Negative	1	1	1	0	0	0	1

$\boldsymbol{W^T x}$



Neural Networks

TRAINING DATA

Positive	0	1	1	0	1	1	0
Positive	0	1	1	0	1	0	0
Negative	1	0	0	0	1	1	0
Negative	1	1	1	0	0	0	1

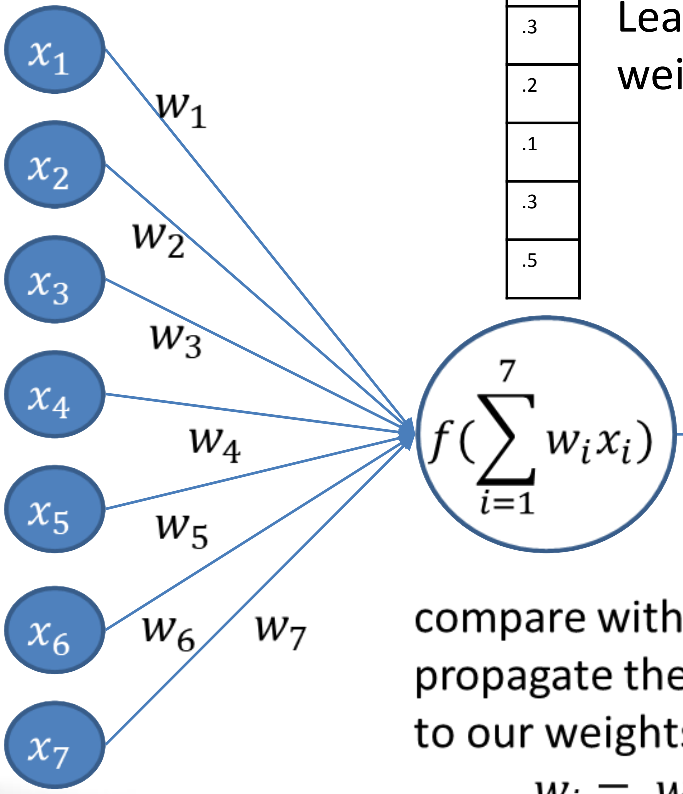
0	1	1	0	1	1	0

Positive

$\boldsymbol{w}$

.5
.4
.3
.2
.1
.3
.5

Learning these weights



Output
Y

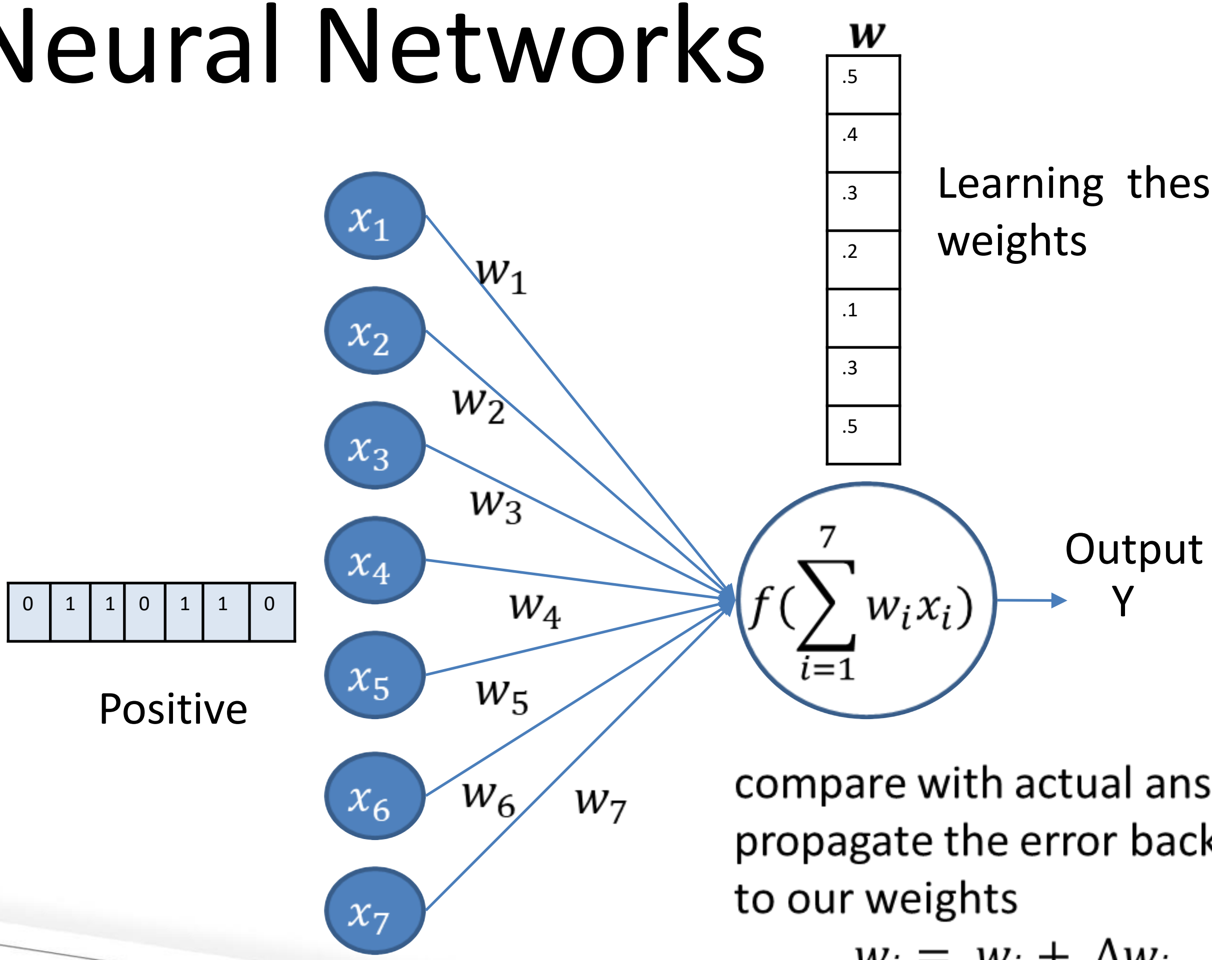
compare with actual answer
propagate the error back
to our weights

$$w_i = w_i + \Delta w_i$$

Neural Networks

TRAINING DATA

Positive	0	1	1	0	1	1	0
Positive	0	1	1	0	1	0	0
Negative	1	0	0	0	1	1	0
Negative	1	1	1	0	0	0	1

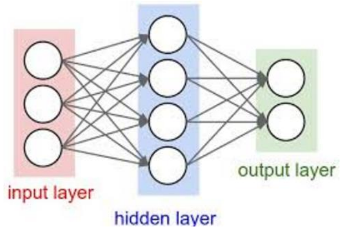


Learning these weights

compare with actual answer
propagate the error back
to our weights

$$w_i = w_i + \Delta w_i$$

word2vec



Basic idea: Train a Neural Network to tell us for any given some context what is the probability of a word

2 models:

- Skip-gram
  - trying to predict: P(context|target)
- Continuous bag of words (CBOW)
  - trying to predict: P(target|context)

…

Skip Gram: P(context|target)

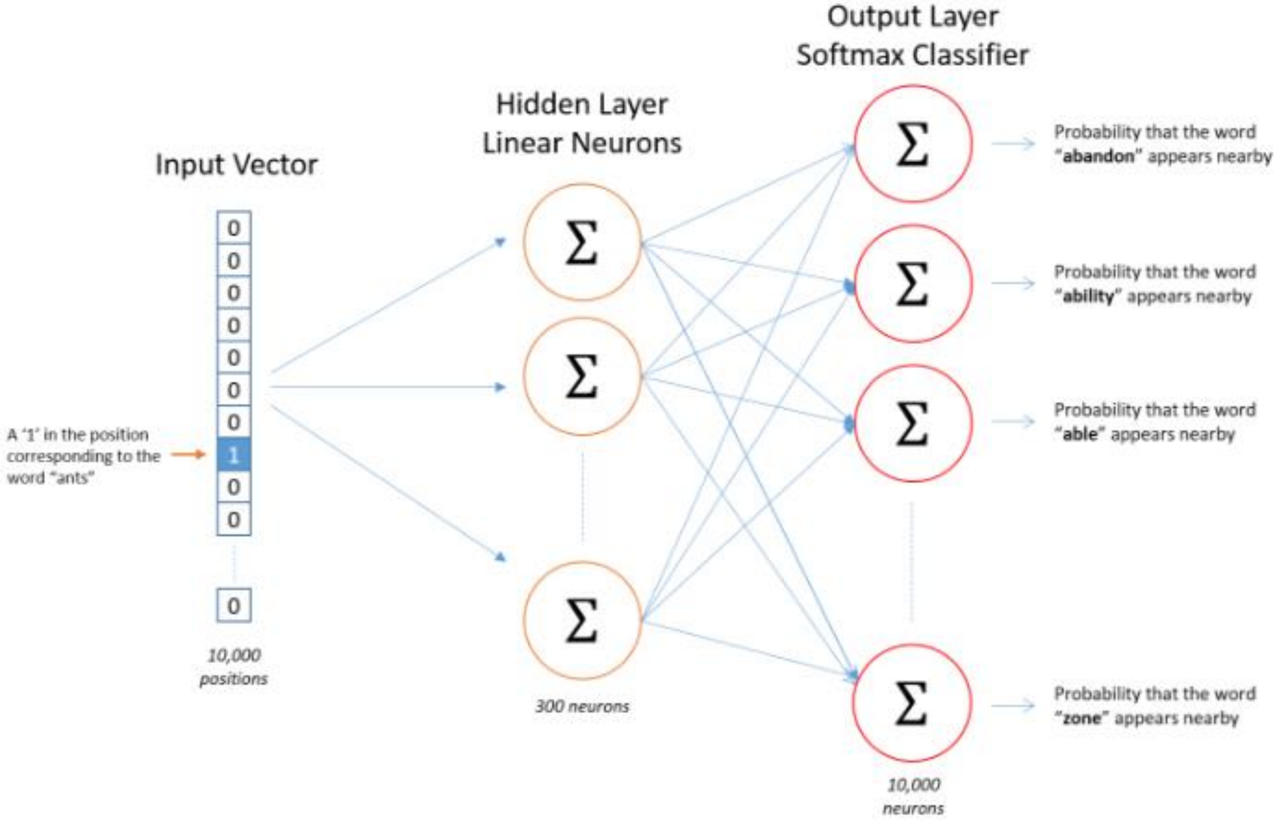
what is the probability of a word given its context

the quick **brown** fox jumped over the lazy dog

(target, context)

(brown, dog) (brown, quick) (brown, fox) (brown, jumped)

(brown, the) (brown, lazy) (brown, over) (brown, the)


Output Layer
Softmax Classifier
Hidden Layer
Linear Neurons
Input Vector
A '1' in the position corresponding to the word "ants"
10,000 positions
300 neurons
10,000 neurons
Probability that the word "**abandon**" appears nearby
Probability that the word "**ability**" appears nearby
Probability that the word "**able**" appears nearby
Probability that the word "**zone**" appears nearby

Skip Gram: P(context|target)

	abandon	ability	able	…	ant	…	zone
abandon	0	0	1		0		6
ability	0	0	5		6		0
able	0	0	0		0		0
…							
brown	0	0	0		0		7
…							
zone	0	0	0		0		0

VxV bigram table

$$P(\text{quick}\,|\,\text{brown}) = \frac{Freq(quick, brown)}{Freq(brown)}$$

Training a Neural Network to maximize this probability over all the words in our vocabulary

(brown, fox) (brown, the)

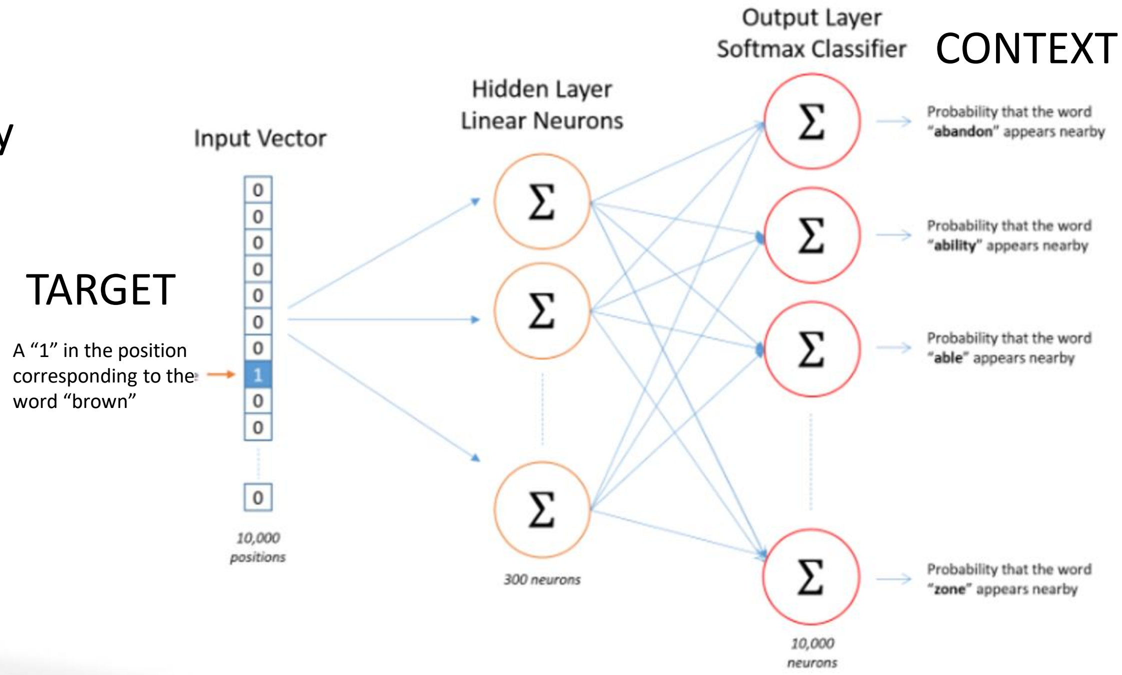
(brown, quick) (brown, jumped)

(brown, the) (brown, lazy) (brown, over) (brown, dog)

Input vectors = *"one-hot" vector*

This a vector of size |V|
where 1 corresponds
to the word of that vector

So we actually have |V| input vectors
-- one for each word in our vocabulary

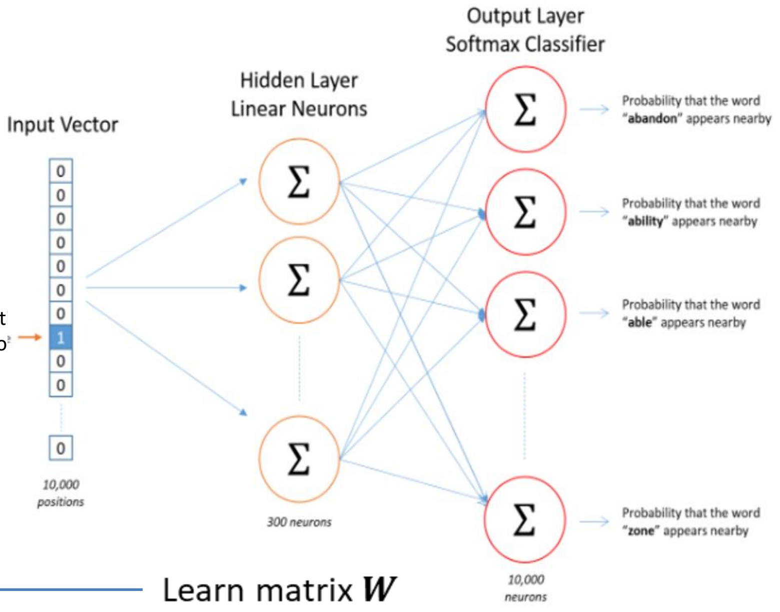


Input vectors = *"one-hot" vector*

This a vector of size |V|
where 1 corresponds
to the word of that vector

So we actually have |V| input vectors
-- one for each word in our vocabulary

A "1" in the posit corresponding to word "brown"



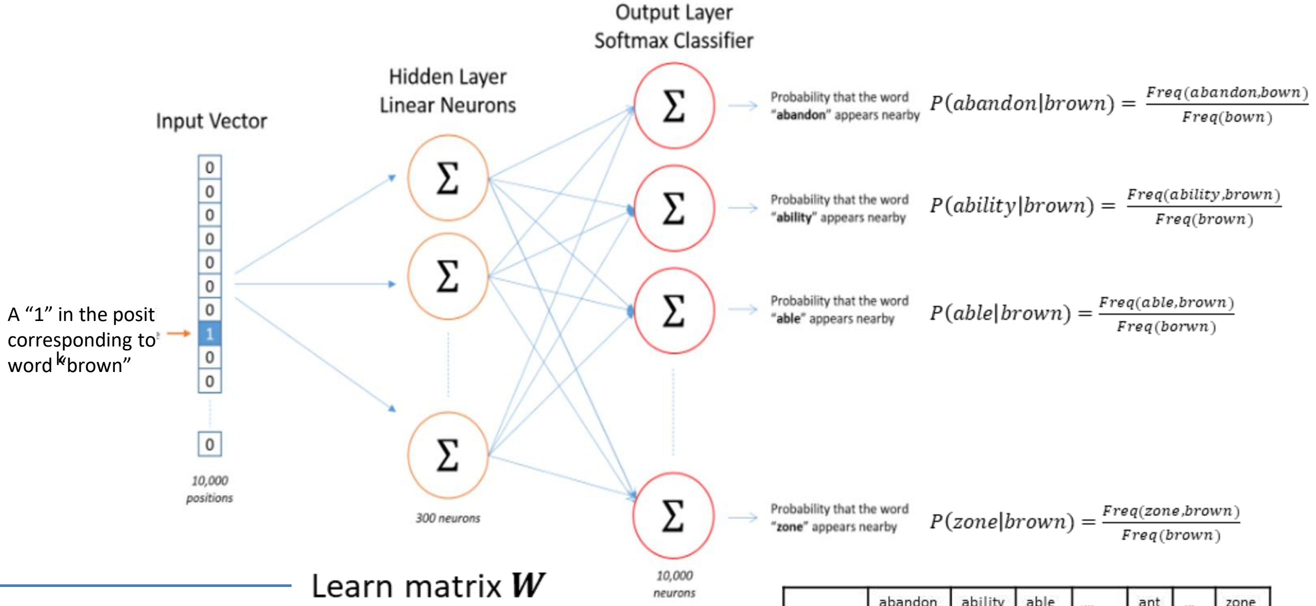
W	w_1	w_2	......	w_{300}
abandon	.003	.0004		.00005
ability	.002	.0001		.0002
able	.00001	.0006		.0007
......	......	......	......	......
brown	.00005	.0005		.004
......	......	......	......	......
zone	.00004	.0002		.0006

Learn matrix $\boldsymbol{W}$

Input vectors = *"one-hot" vector*

This a vector of size |V|
where 1 corresponds
to the word of that vector

So we actually have |V| input vectors
-- one for each word in our vocabulary



Learn matrix $\boldsymbol{W}$

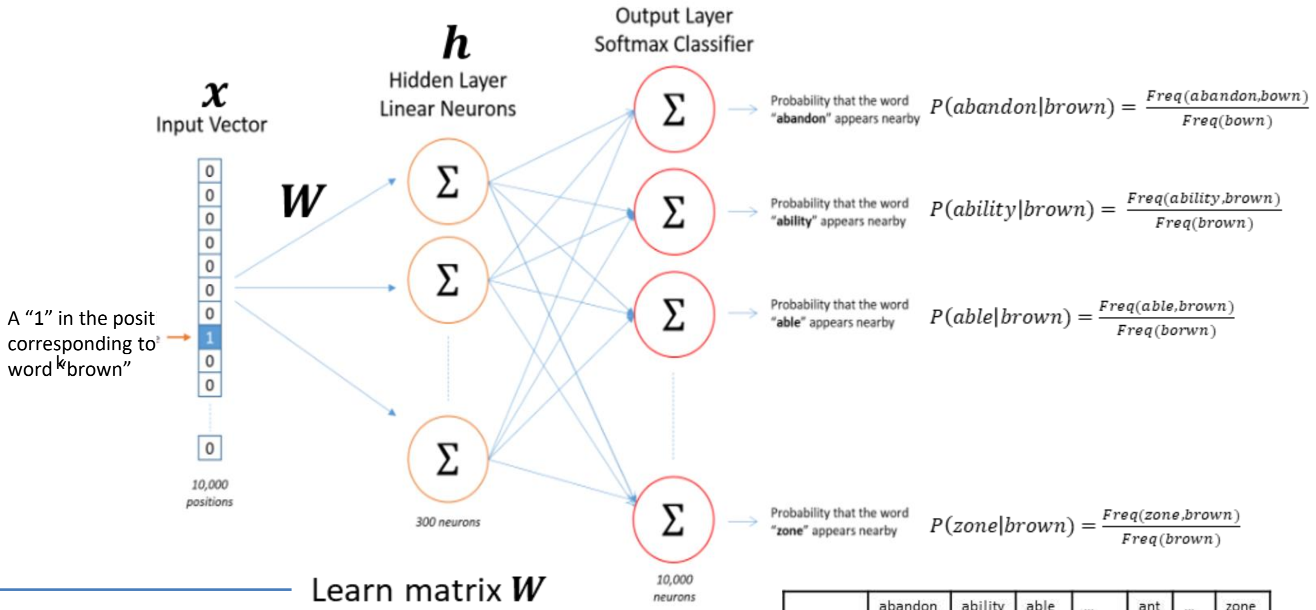
W	w_1	w_2	……	w_{300}
abandon	.003	.0004		.00005
ability	.002	.0001		.0002
able	.00001	.0006		.0007
……	……	……	……	……
brown	.00005	.0005		.004
……	……	……	……	……
zone	.00004	.0002		.0006

	abandon	ability	able	...	ant	...	zone
abandon	0	0	1		0		6
ability	0	0	5		6		0
able	0	0	0		0		0
...							
brown	0	0	0		0		7
...							
zone	0	0	0		0		0

Input vectors = "*one-hot*" *vector*

This a vector of size |V|
where 1 corresponds
to the word of that vector

So we actually have |V| input vectors
-- one for each word in our vocabulary



W	w_1	w_2	……	w_{300}
abandon	.003	.0004		.00005
ability	.002	.0001		.0002
able	.00001	.0006		.0007
……	……	……	……	……
brown	.00005	.0005		.004
……	……	……	……	……
zone	.00004	.0002		.0006

Learn matrix $\boldsymbol{W}$

$$\boldsymbol{h} = \boldsymbol{W}^T\boldsymbol{x}$$

	abandon	ability	able	...	ant	...	zone
abandon	0	0	1		0		6
ability	0	0	5		6		0
able	0	0	0		0		0
...							
brown	0	0	0		0		7
...							
zone	0	0	0		0		0

$$h = W^T x$$

since x is a one-hot vector

W

w	w_1	w_2	……	w_{300}
abandon	.003	.0004		.00005
ability	.002	.0001		.0002
able	.00001	.0006		.0007
……	……	……	……	……
ant	.00005	.0005		.004
……	……	……	……	……
zone	.00004	.0002		.0006

W^T

	abandon	ability	able	…	brown	…	zone
w_1	.003	.002	.00001		.00005		.00004
w_2	.0004	.0001	.0006		.0005		.0002
……							
w_{300}	.00005	.0002	.0007		.004		.0006

300 * V

x

x

ant
0
0
0
…
1
…
0

V * 1

=

h

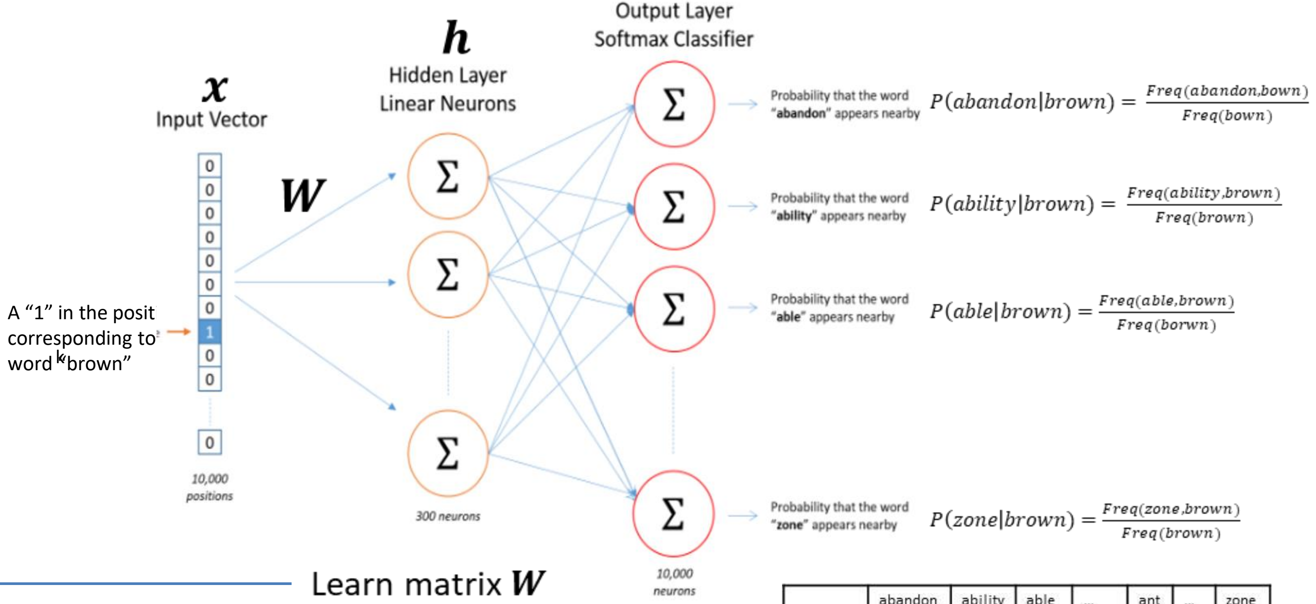
brown
.00005
.0005
.004

Basically this is a look up table

Input vectors = *"one-hot" vector*

This a vector of size |V|
where 1 corresponds
to the word of that vector

So we actually have |V| input vectors
-- one for each word in our vocabulary



W	w_1	w_2	……	w_{300}
abandon	.003	.0004		.00005
ability	.002	.0001		.0002
able	.00001	.0006		.0007
……	……	……	……	……
brown	.00005	.0005		.004
……	……	……	……	……
zone	.00004	.0002		.0006

Learn matrix W

$$h = W^T x =$$

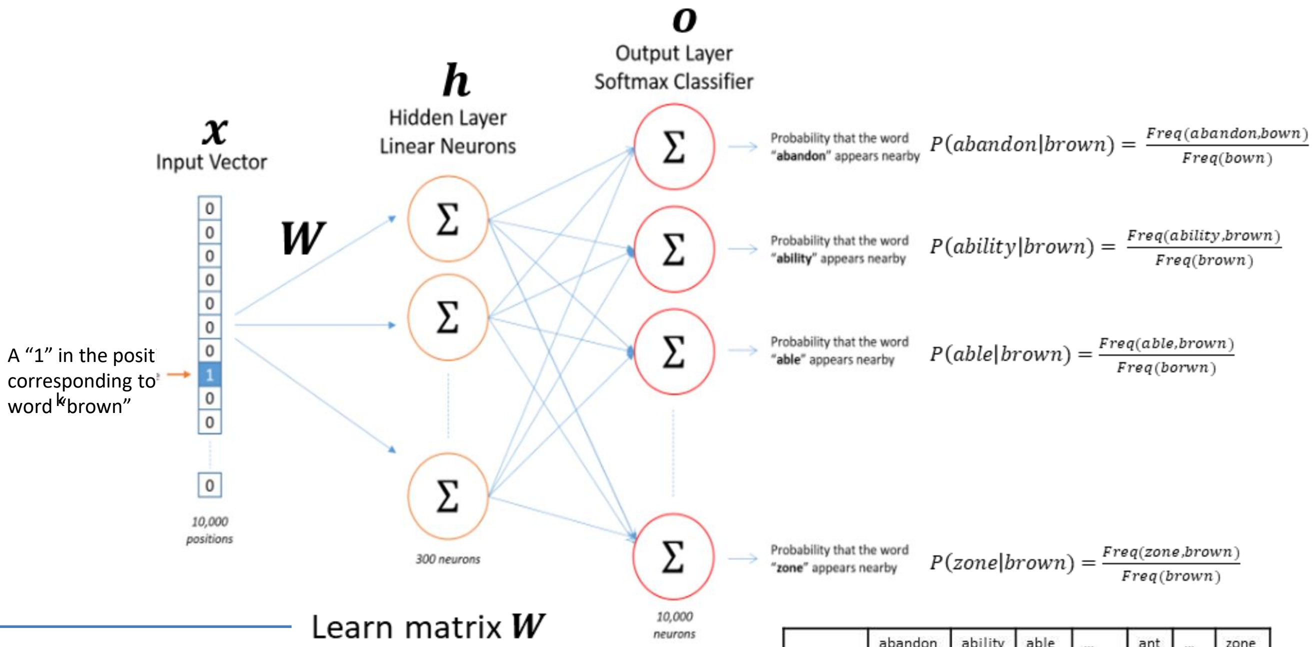
brown
.00005
.0005
.004

	abandon	ability	able	...	ant	...	zone
abandon	0	0	1		0		6
ability	0	0	5		6		0
able	0	0	0		0		0
...							
brown	0	0	0		0		7
...							
zone	0	0	0		0		0

Input vectors = *"one-hot" vector*

This a vector of size |V|
where 1 corresponds
to the word of that vector

So we actually have |V| input vectors
-- one for each word in our vocabulary



Learn matrix W

W	w_1	w_2	......	w_{300}
abandon	.003	.0004		.00005
ability	.002	.0001		.0002
able	.00001	.0006		.0007
......	......	......	......	......
brown	.00005	.0005		.004
......	......	......	......	......
zone	.00004	.0002		.0006

$$h = W^T x =$$

brown
.00005
.0005
.004

$$o = \frac{e^h}{\sum e^h}$$

	abandon	ability	able	...	ant	...	zone
abandon	0	0	1		0		6
ability	0	0	5		6		0
able	0	0	0		0		0
...							
brown	0	0	0		0		7
...							
zone	0	0	0		0		0

$$h = W^T x$$

since x is a one-hot vector

W

W	w_1	w_2	......	w_{300}
abandon	.003	.0004		.00005
ability	.002	.0001		.0002
able	.00001	.0006		.0007
......	......	......	......	......
ant	.00005	.0005		.004
......	......	......	......	......
zone	.00004	.0002		.0006

W^T

	abandon	ability	able	...	brown	...	zone
w_1	.003	.002	.00001		.00005		.00004
w_2	.0004	.0001	.0006		.0005		.0002
......							
w_{300}	.00005	.0002	.0007		.004		.0006

X

x

ant
0
0
0
...
1
...
0

=

h

brown
.00005
.0005
.004

$$\frac{e^h}{\sum e^h} =$$

o

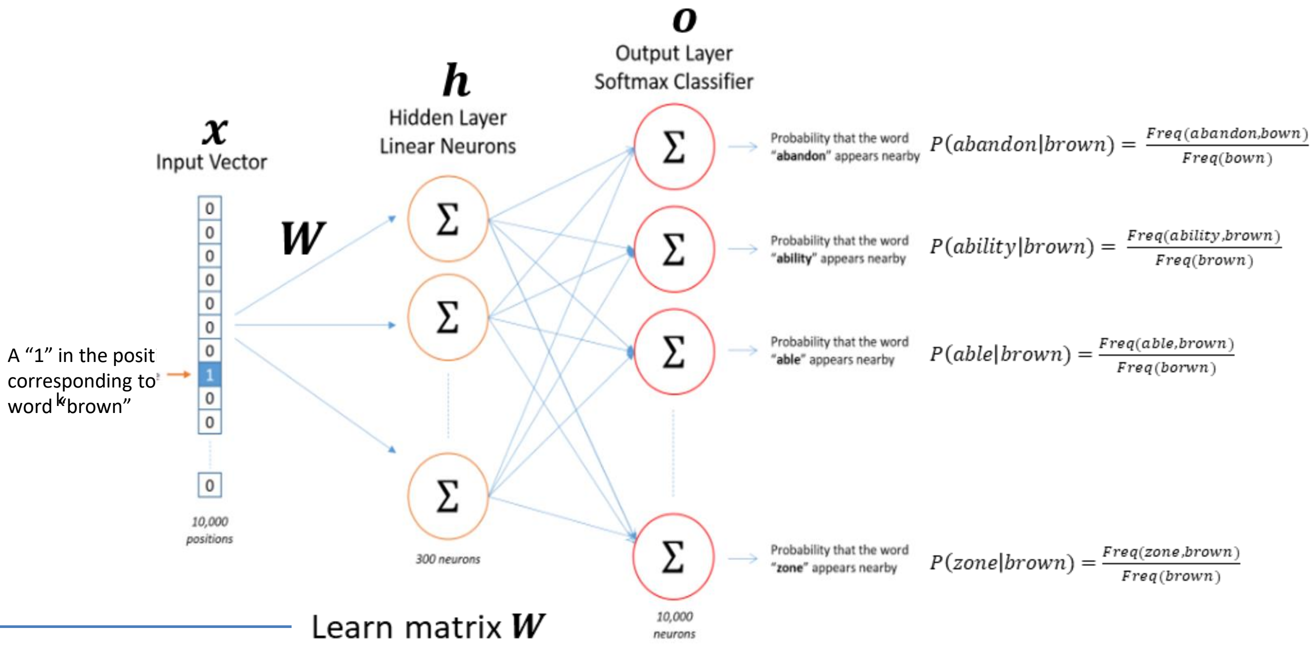
.003	$P(abandon \mid brown) = \frac{Freq(abandon,bown)}{Freq(bown)}$
....	
.002	$P(ability \mid brown) = \frac{Freq(ability,brown)}{Freq(brown)}$
......	
.00001	$P(able \mid brown) = \frac{Freq(able,brown)}{Freq(borwn)}$
.....	
......	
.00004	$P(zone \mid brown) = \frac{Freq(zone,brown)}{Freq(brown)}$

Input vectors = *"one-hot" vector*

This a vector of size |V|
where 1 corresponds
to the word of that vector

So we actually have |V| input vectors
-- one for each word in our vocabulary

W	w_1	w_2	......	w_{300}
abandon	.003	.0004		.00005
ability	.002	.0001		.0002
able	.00001	.0006		.0007
......	......	......	......	......
brown	.00005	.0005		.004
......	......	......	......	......
zone	.00004	.0002		.0006



Learn matrix W

$$h = W^T x$$

$$o = \frac{e^h}{\sum e^h}$$

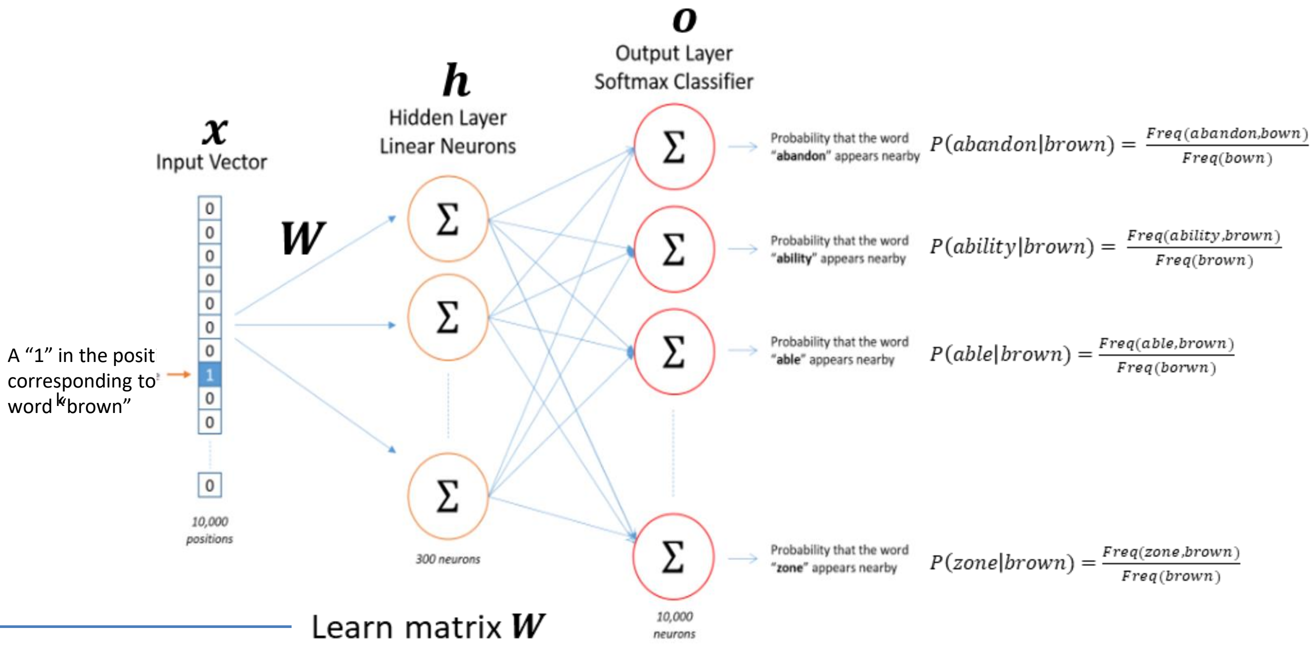
compare with actual answer
propagate the error
back to our weights

$$w_i = w_i + \Delta w_i$$

Input vectors = “*one-hot” vector*

This a vector of size |V|
where 1 corresponds
to the word of that vector

So we actually have |V| input vectors
-- one for each word in our vocabulary



W	w_1	w_2	......	w_{300}
abandon	.003	.0004		.00005
ability	.002	.0001		.0002
able	.00001	.0006		.0007
......	......	......	......	......
brown	.00005	.0005		.004
......	......	......	......	......
zone	.00004	.0002		.0006

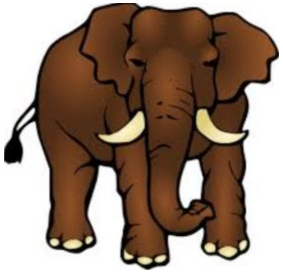
Learn matrix ***W***

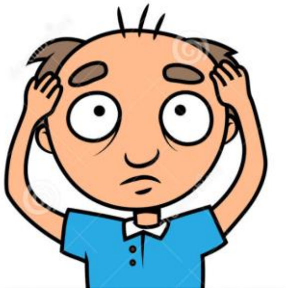
$$h = W^T x$$

$$o = \frac{e^h}{\sum e^h}$$

Do this for each input vector in our vocabulary continually updating our weights is intractable so we use a *sampling*

Word embeddings





CBOW: P(word|context)

what is the probability of a word given its context

the quick **brown** fox jumped over the lazy dog

(target, context)

(brown, the)
(brown, dog)
(brown, quick)
(brown, fox)
(brown, jumped)
(brown, the)
(brown, lazy)
(brown, over)

CBOW: P(word|context)

	abandon	ability	able	…	ant	…	zone
abandon	0	0	1		0		6
ability	0	0	5		6		0
able	0	0	0		0		0
…							
brown	0	0	0		0		7
…							
zone	0	0	0		0		0

VxV bigram table

$$P(\text{brown}\,|\,\text{quick}) = \frac{Freq(brown,quick)}{Freq(quick)}$$

Training a Neural Network to maximize this probability over all the words in our vocabulary

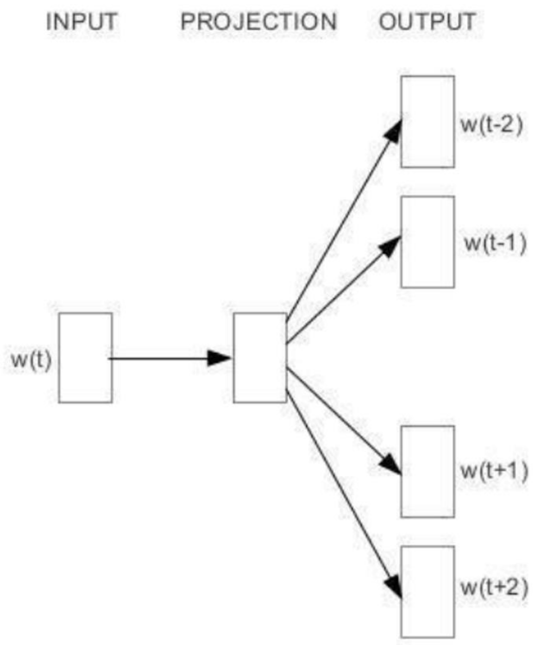
(brown, fox) (brown, the)

(brown, quick) (brown, jumped)

(brown, the) (brown, lazy) (brown, over) (brown, dog)

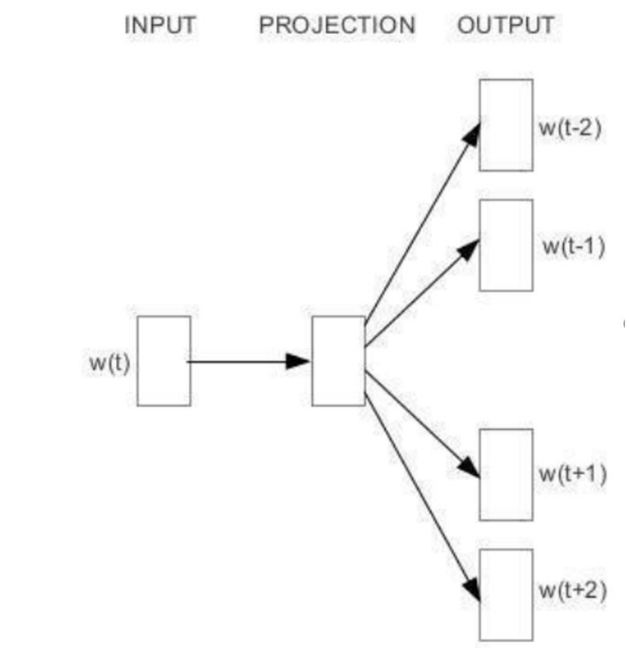
SkipGram

P(context|word)



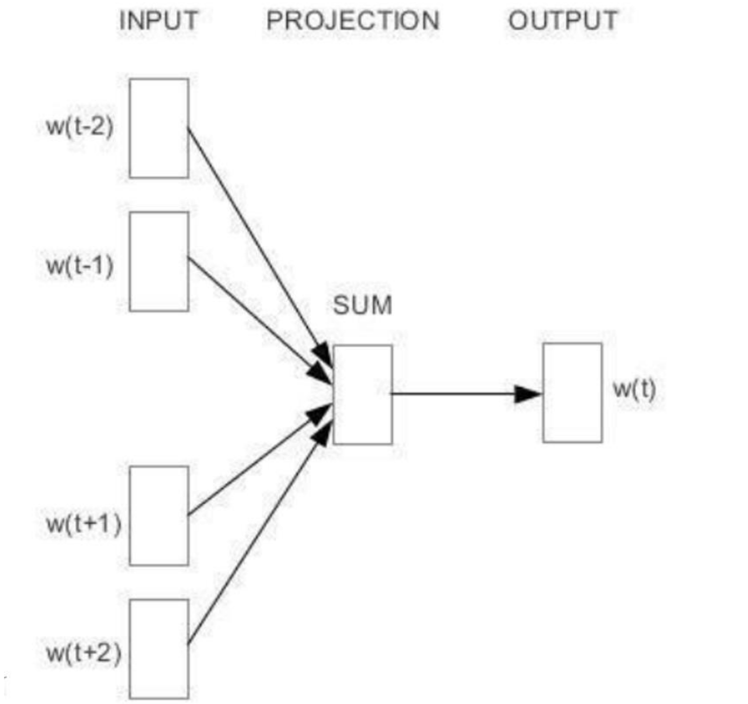
SkipGram

P(context|word)



CBOW

P(word|context)

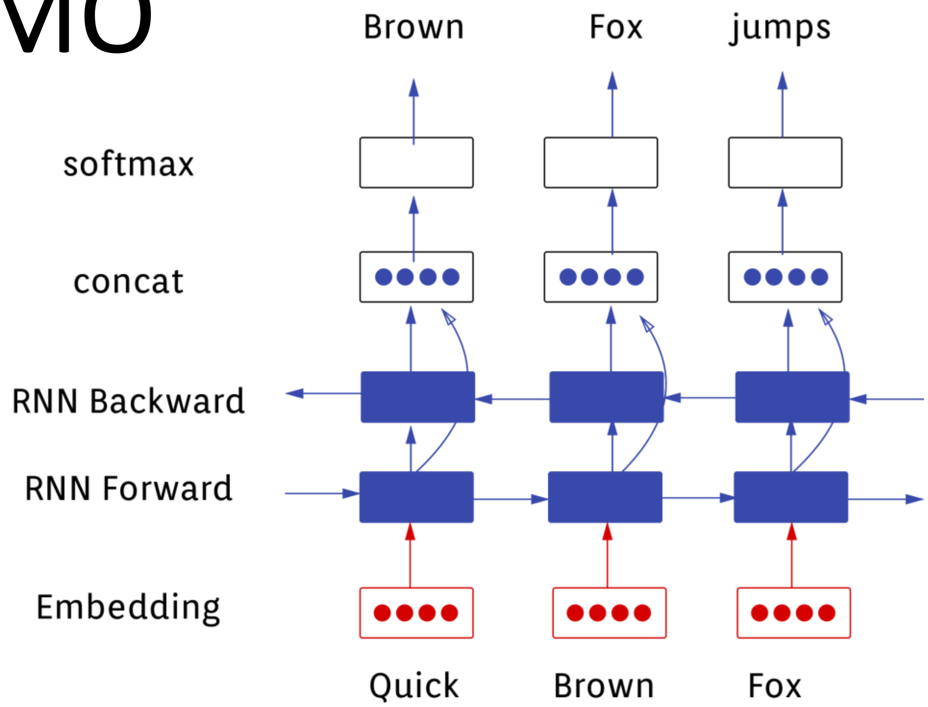


Sesame Street

- Recent advances
  - eLMO
  - BERT
  - ERNIE
  - UML-FiT
  - XLNEt

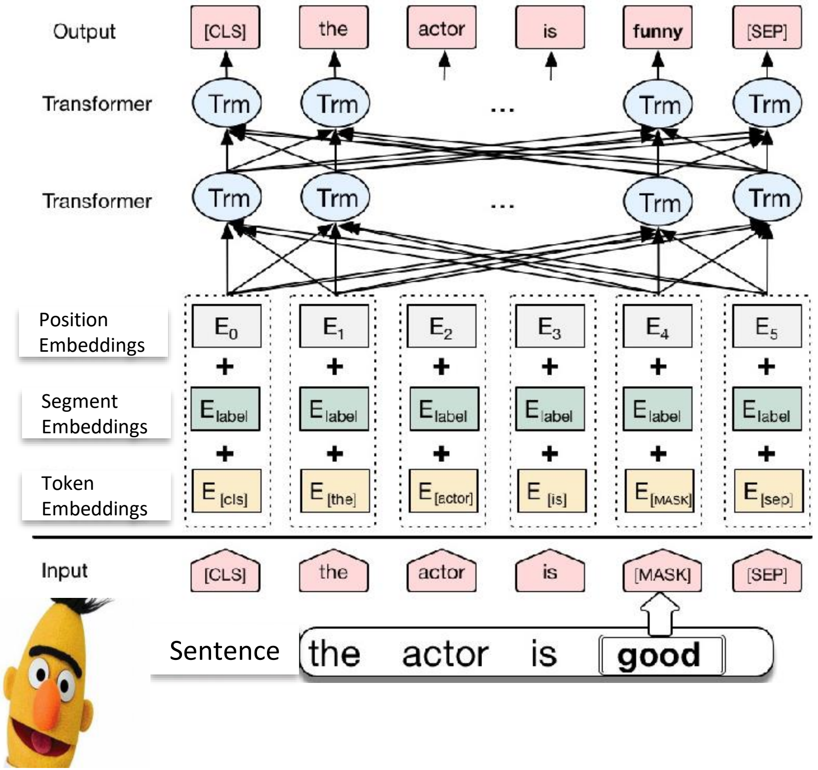
eLMO

- Learning model: LSTM
- Input: character embeddings
- Output: Token level prediction



BERT

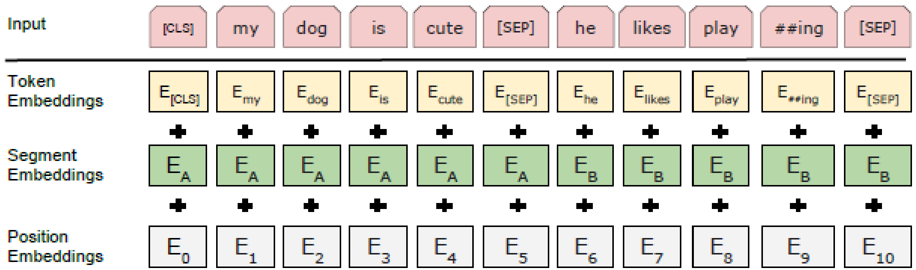
- Learning model: Transformer
- Input:
  - Token Embeddings
  - Subword Embeddings
  - Position Embeddings
- Output: masked token and
  sentence prediction
- Feature space: sub words

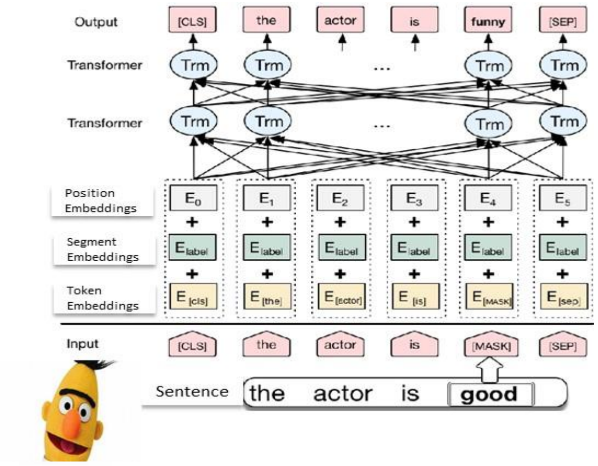


BERT

- Learning model: Transformer
- Input:
  - Token Embeddings
  - Segment Embeddings
  - Position Embeddings

- Output: masked token and sentence prediction
- Feature space: sub words





What vector representations could we use?

What vector representations could we use?

- Feature-based vectors
  - Consists of numeric or nominal values that encode linguistic information

- Example features:
  - Lexical Information
    - Bag-of-words -- words surrounding the target word
    - N-grams - Extension of bag-of-words (which is just unigrams)
    - Collocation - the information about the words located to the left or right of the target word
  - Syntactic Information
    - Part of speech of the target word
    - Part of speech of the previous word
  - Semantic information
    - Concept of the surrounding words

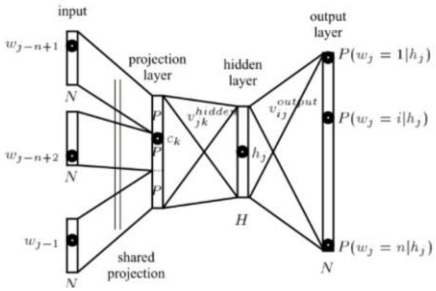
What vector representations could we use?

- Feature-based vectors
  - Consists of numeric or nominal values that encode linguistic information

- Example features:
  - Lexical Information
    - Bag-of-words -- words surrounding the target word
    - N-grams - Extension of bag-of-words (which is just unigrams)
    - Collocation - the information about the words located to the left or right of the target word
  - Syntactic Information
    - Part of speech of the target word
    - Part of speech of the previous word
  - Semantic information
    - Concept of the surrounding words

Feature-less based word embeddings

- word2vec
- glove
- BERT
- ELMO



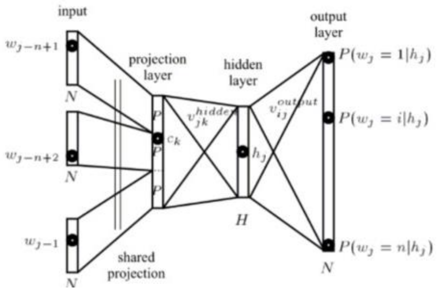
What vector representations could we use?

- Feature-based vectors
  - Consists of numeric or nominal values that encode linguistic information

- Example features:
  - Lexical Information
    - Bag-of-words -- words surrounding the target word
    - N-grams - Extension of bag-of-words (which is just unigrams)
    - Collocation - the information about the words located to the left or right of the target word
  - Syntactic Information
    - Part of speech of the target word
    - Part of speech of the previous word
  - Semantic information
    - Concept of the surrounding words

Feature-less based word embeddings

- word2vec
- glove
- BERT
- ELMO



What do we need to be aware of when using pre-trained word embeddings?

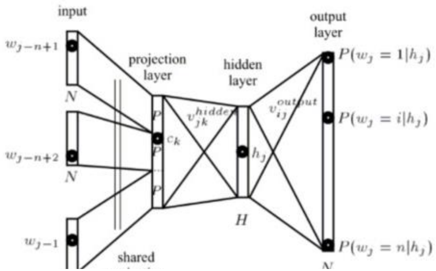
What vector representations could we use?

- Feature-based vectors
  - Consists of numeric or nominal values that encode linguistic information

- Example features:
  - Lexical Information
    - Bag-of-words -- words surrounding the target word
    - N-grams - Extension of bag-of-words (which is just unigrams)
    - Collocation - the information about the words located to the left or right of the target word
  - Syntactic Information
    - Part of speech of the target word
    - Part of speech of the previous word
  - Semantic information
    - Concept of the surrounding words

Feature-less based word embeddings

- word2vec
- glove
- BERT
- ELMO
- ERNIE



Questions

BUDANITSKY & HIRST

DESCRIPTION OF THE STUDY:

EVALUATE SEMANTIC SIMILARITY MEASURES ON TWO TASKS

INTRINSIC EVALUTION: WORD SIMILARITY
EXTRINSIC EVALUATION: MALAPROPISMS

malapropisms

Dictionary

malapropisms

mal·a·prop

/ˈmaləˌpräp/

noun

plural noun: **malapropisms**

the mistaken use of a word in place of a similar-sounding one, often with unintentionally amusing effect, as in, for example, “dance a *flamingo*” (instead of *flamenco*).

synonyms: wrong word, solecism, misuse, misapplication, infelicity, Freudian slip, blunder; More

Translations, word origin, and more definitions

Method

Each possible CORRECTION of a **MALPROPISM** is assigned a score
[sum similarity between it and its surrounding terms]

Assign MALPROPISM the correction with highest score

He played base at the rock concert

He played base at the rock concert

bass

base

He played base at the rock concert

bass

base

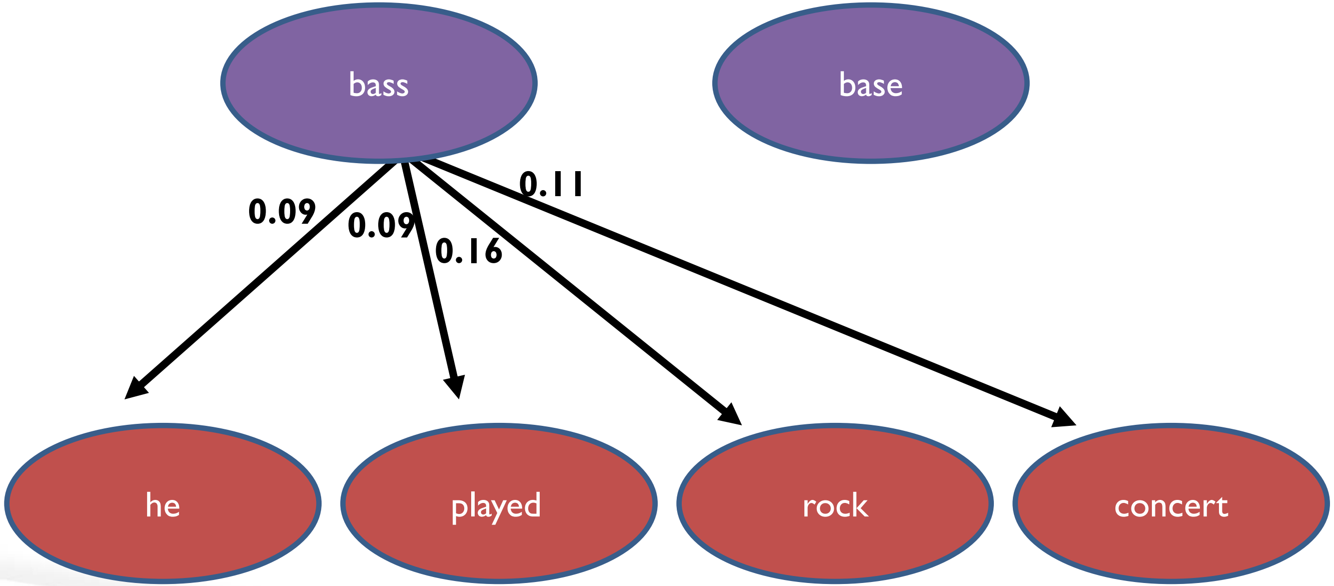
he

played

rock

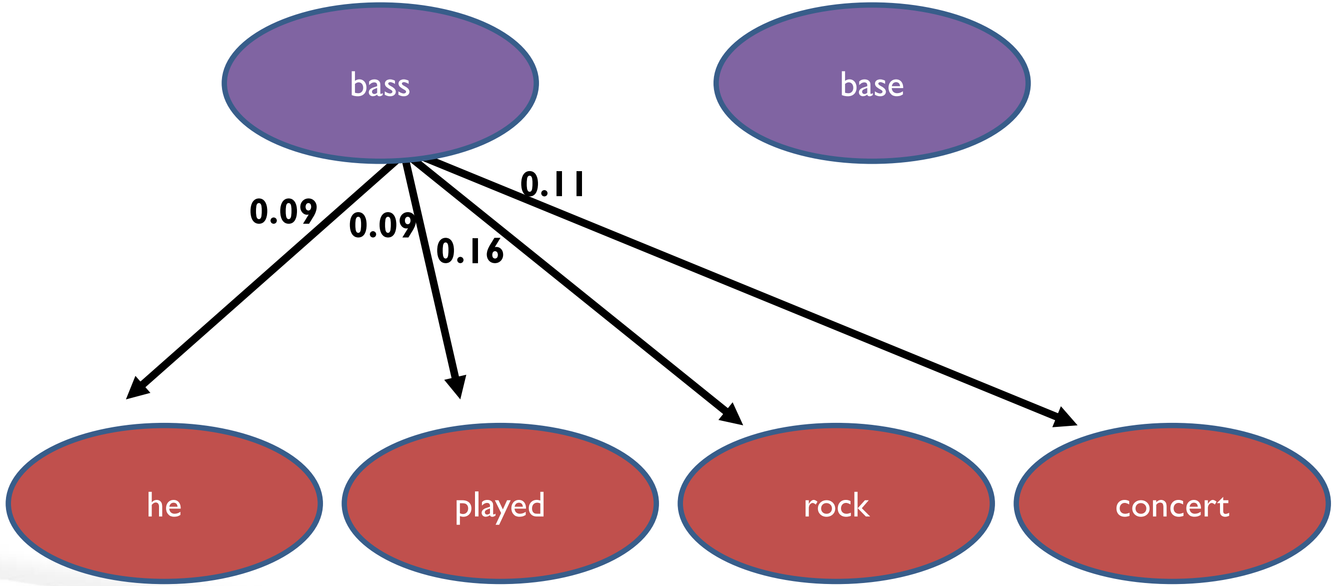
concert

He played base at the rock concert



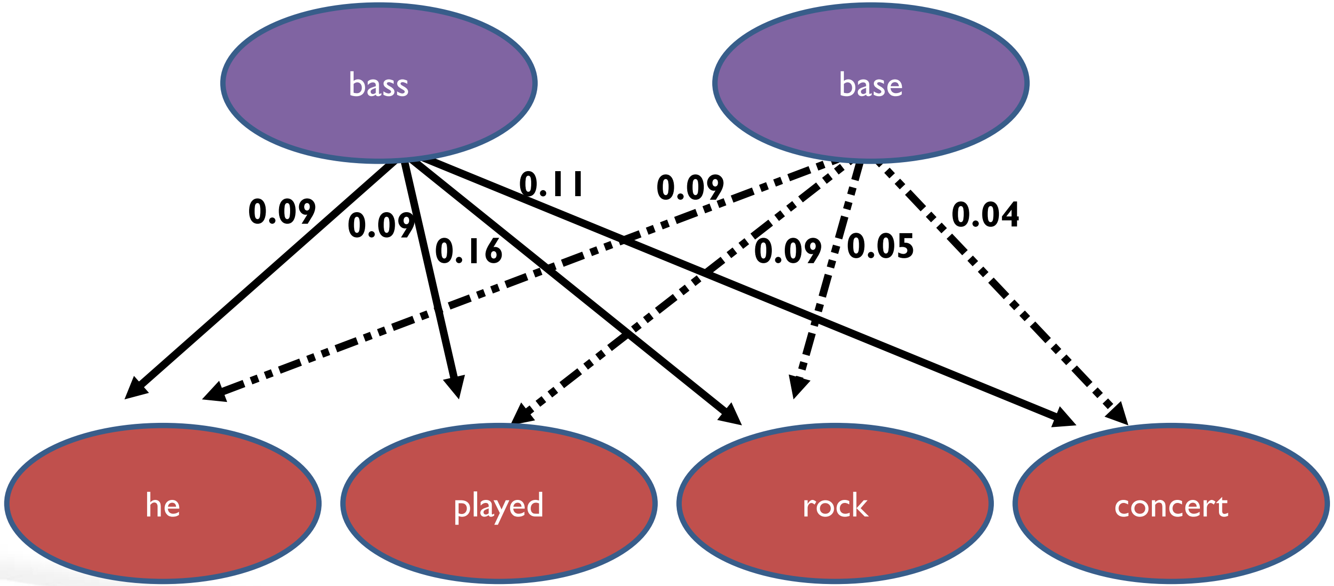
He played base at the rock concert

bass
Score = 0.09 + 0.09 + 0.16 + 0.11 = 0.45



He played base at the rock concert

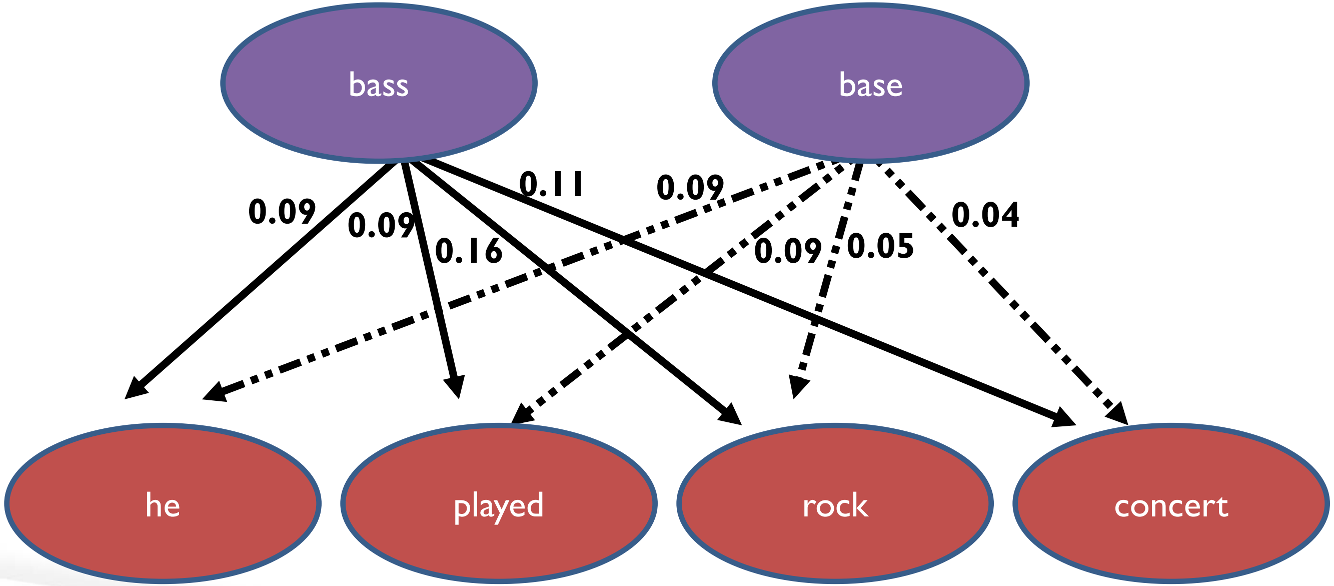
bass
Score = 0.09 + 0.09 + 0.16 + 0.11 = 0.45



He played base at the rock concert

<u>bass</u>
Score = 0.09 + 0.09 + 0.16 + 0.11 = 0.45

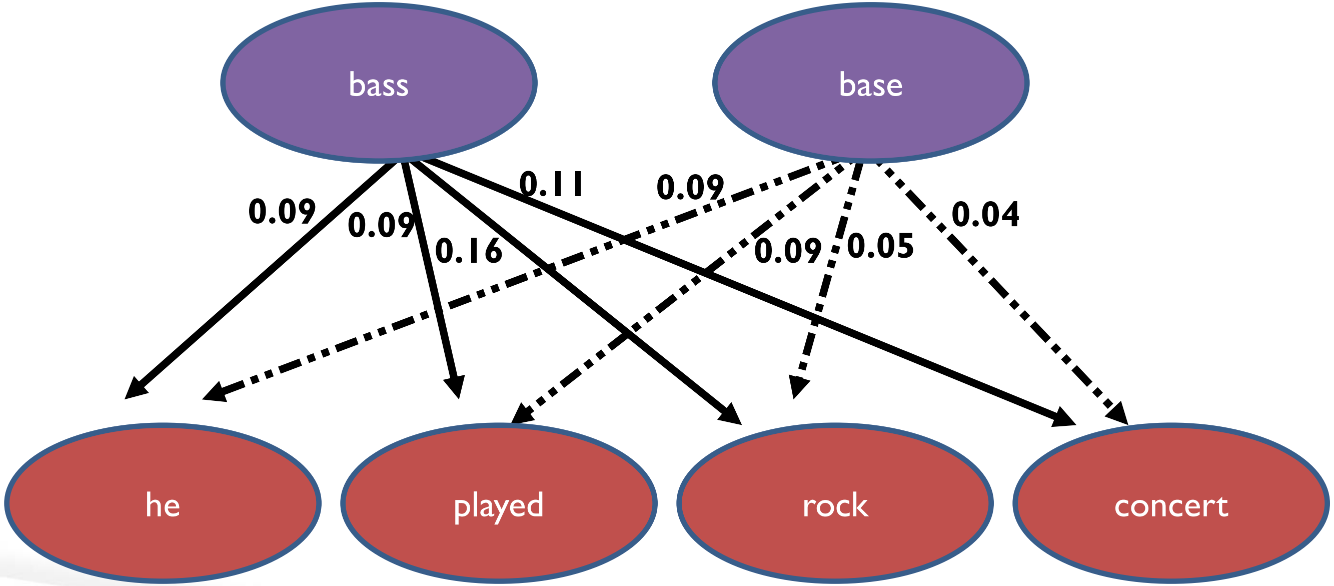
<u>base</u>
Score = 0.09 + 0.09 + 0.05 + 0.05 = 0.27



He played base at the rock concert

bass
Score = 0.09 + 0.09 + 0.16 + 0.11 = 0.45

base
Score = 0.09 + 0.09 + 0.05 + 0.05 = 0.27



malapropism dataset

- wall street journal corpus
  - removed: proper nouns & stop-list words (non content words)
- replaced every 200 word with a spelling variant
- spelling varients
  - come from wordnet nouns with at least one spelling variation

- resulting dataset
  - 107,233 words
  - 1,408 of which were malapropisms

malapropism results

- evaluated
  - Measures
  - Scope
    - the number of surrounding words

- DISCUSSION?

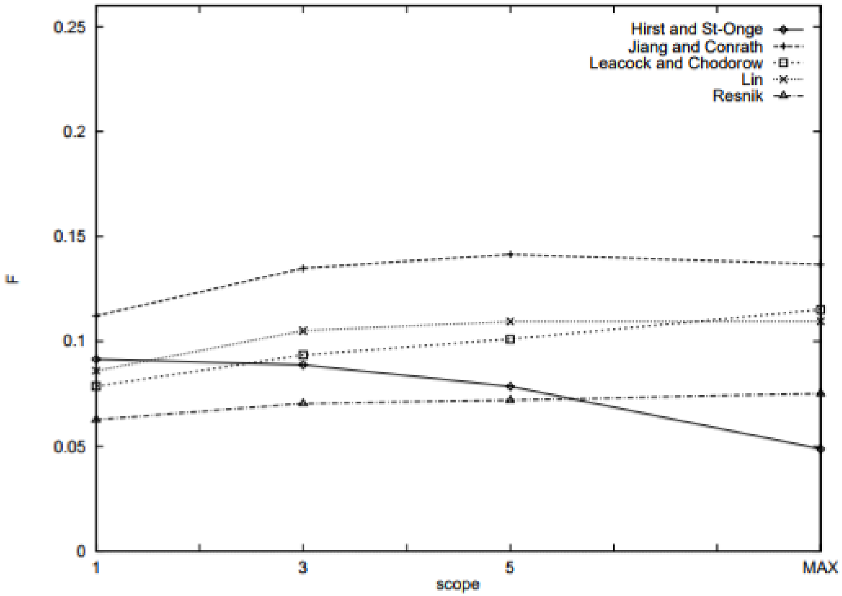


Figure 1: Suspicion F-measure (F_S), by measure and scope.

WORDNET

DOWNLOAD LINK: https://wordnet.princeton.edu/wordnet/download/current-version/#nix

WordNet 3.0 for UNIX-like systems (including: Linux, Mac OS X, Solaris)

Before you download: The WordNet 3.0 README file contains additional information about the release. You can read about the changes from version 2.1.

Source code and binaries:

Download tar-gzipped: WordNet-3.0.tar.gz

Download tar-bzip2'ed: WordNet-3.0.tar.bz2

Download just database files: WNdb-3.0.tar.gz

INSTALLATION GUIDE: http://people.vcu.edu/~henryst/WordNet%20Installation%20Procedure.pdf

wordnet::similarity

DOWNLOAD LINK: http://wn-similarity.sourceforge.net/

WordNet::Similarity

This is a Perl module that implements a variety of semantic similarity and relatedness measures based on information found in the lexical database WordNet. In particular, it supports the measures of Resnik, Lin, Jiang-Conrath, Leacock-Chodorow, Hirst-St.Onge, Wu-Palmer, Banerjee-Pedersen, and Patwardhan-Pedersen.

We have a mailing list designed to support users of WordNet::Similarity.

Want to report a bug or request a feature? Do that here!

Try the Web Interface here. (version 2.07)

Download the Current Version (v2.07, released October 4, 2015) from CPAN or Sourceforge

INSTALLATION GUIDE: http://people.vcu.edu/~henryst/WordNet%20Installation%20Procedure.pdf

WORDNET::Similarity INTERFACE

WordNet::Similarity

Read an overview of WordNet::Similarity.

You may enter any two words in one of three formats:

1. `word`
2. `word#part_of_speech` (where part_of_speech is one of n, v, a, or r)
3. `word#part_of_speech#sense` (where sense is a positive integer)

If words are entered in format 1 or 2, then the relatedness of all valid forms of the words will be computed (e.g., if 'dogs' is entered, then 'dog' will be used to compute relatedness). More instructions.

Word 1:		◉ Use all senses	○ Pick a sense by gloss	○ Pick a sense by synset
Word 2:		◉ Use all senses	○ Pick a sense by gloss	○ Pick a sense by synset
Measure:	Path Length ▾	About the measures		

☑ Use root node?

Compute Clear

Show version info

Created by Ted Pedersen and Jason Michelizzi
E-mail: tpederse (at) d (dot) umn (dot) edu

http://maraca.d.umn.edu/cgi-bin/similarity/similarity.cgi

wordnet::Similarity precomputed pairs

- **Pre-computed Pairwise Similarity Values for Nouns and Verbs**

  We are pre-computing all pairwise similarity values for all senses in WordNet, slowly but surely. This began in June 2010 - by March 2011 we had completed all verb pairs for all similarity measures, and in August 2011 we completed all noun pairs for the path measure. We continue to work on the other measures.
  - WordNet Similarity Pairs

- **Information Content Computed on Various Corpora**

  We have pre-computed information content files from the British National Corpus (World Edition), the Penn Treebank (version 2), the Brown Corpus, the complete works of Shakespeare, and SemCor (with and without sense tags). These were created using the *Freq.pl programs found in WordNet::Similarity. These information content files should be used with WordNet::Similarity for the given version of WordNet.
  - WordNet-InfoContent-3.0 - README
  - WordNet-InfoContent-2.1
  - WordNet-InfoContent-2.0
  - WordNet-InfoContent-1.7.1

LINK: http://wn-similarity.sourceforge.net/