WORD SIMILARITY

David Kauchak
CS159 Fall 2014

Admin

Assignment 4

Quiz #2 Thursday

- Same rules as quiz #1
  - First 30 minutes of class
  - Open book and notes

Assignment 5 out on Thursday

Quiz #2

Topics

- Linguistics 101
- Parsing
  - Grammars, CFGs, PCFGs
  - Top-down vs. bottom-up
  - CKY algorithm
  - Grammar learning
  - Evaluation
  - Improved models
- Text similarity
  - Will also be covered on Quiz #3, though

Text Similarity

A common question in NLP is how similar are texts

score: sim(,) = ?

rank: ?

Bag of words representation

For now, let's ignore word order:

Obama said banana repeatedly last week on tv, "banana, banana, banana"

(4, 1, 1, 0, 0, 1, 0, 0, …)

banana obama said california across tv wrong capital

"Bag of words representation": multi-dimensional vector, one dimension per word in our vocabulary

Frequency of word occurrence

Vector based word

A

a_1: When 1
a_2: the 2
a_3: defendant 1
a_4: and 1
a_5: courthouse 0
…

B

b_1: When 1
b_2: the 2
b_3: defendant 1
b_4: and 0
b_5: courthouse 1
…

Multi-dimensional vectors, one dimension per word in our vocabulary

How do we calculate the similarity based on these vectors?

Normalized distance measures

Cosine

$$sim_{\cos}(A,B) = A \cdot B = \sum_{i=1}^{n} a'_i b'_i = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}\sqrt{\sum_{i=1}^{n} b_i^2}}$$

L2

$$dist_{L2}(A,B) = \sqrt{\sum_{i=1}^{n} (a'_i - b'_i)^2}$$

L1

$$dist_{L1}(A,B) = \sum_{i=1}^{n} |a'_i - b'_i|$$

a' and b' are length normalized versions of the vectors

Our problems

So far…

- word order
- length
- synonym
- spelling mistakes
- word importance
- word frequency

Word importance

Include a weight for each word/feature

A

a_1: When	1	w_1
a_2: the	2	w_2
a_3: defendant	1	w_3
a_4: and	1	w_4
a_5: courthouse	0	w_5
…		…

B

b_1: When	1	w_1
b_2: the	2	w_2
b_3: defendant	1	w_3
b_4: and	0	w_4
b_5: courthouse	1	w_5
…		…

Distance + weights

We can incorporate the weights into the distances

Think of it as either (*both work out the same*):

- preprocessing the vectors by multiplying each dimension by the weight
- incorporating it directly into the similarity measure

$$sim_{\cos}(A,B) = A \cdot B = \sum_{i=1}^{n} a_i' b_i' = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}\sqrt{\sum_{i=1}^{n} b_i^2}}$$

$$= \frac{\sum_{i=1}^{n} w_i a_i w_i b_i}{\sqrt{\sum_{i=1}^{n} (w_i a_i)^2}\sqrt{\sum_{i=1}^{n} (w_i b_i)^2}}$$ with weights

Document vs. overall frequency

The overall frequency of a word is the number of occurrences in a dataset, counting multiple occurrences

Example:

Word	Overall frequency	Document frequency
insurance	10440	3997
try	10422	8760

Which word is a more informative (and should get a higher weight)?

Document frequency

Word	Collection frequency	Document frequency
insurance	10440	3997
try	10422	8760

Document frequency is often related to word importance, but we want an actual weight. Problems?

$$sim_{\cos}(A,B) = A \cdot B = \frac{\sum_{i=1}^{n} w_i a_i w_i b_i}{\sqrt{\sum_{i=1}^{n} (w_i a_i)^2}\sqrt{\sum_{i=1}^{n} (w_i b_i)^2}}$$

From document frequency to weight

Word	Collection frequency	Document frequency
insurance	10440	3997
try	10422	8760

weight and document frequency are **inversely** related
- higher document frequency should have lower weight and vice versa

document frequency is unbounded

document frequency will change depending on the size of the data set (i.e. the number of documents)

Inverse document frequency

$$\text{idf}_w = \log \frac{N}{\text{df}_w}$$

N: # of documents in dataset

df_w: document frequency of w

IDF is inversely correlated with DF
- higher DF results in lower IDF

N incorporates a dataset dependent normalizer

log dampens the overall weight

IDF example, suppose *N*=1 million

word	df_w	idf_w
calpurnia	1	
animal	100	
sunday	1,000	
fly	10,000	
under	100,000	
the	1,000,000	

What are the IDFs assuming log base 10?

$$\text{idf}_w = \log \frac{N}{\text{df}_w}$$

IDF example, suppose *N*=1 million

word	df_w	idf_w
calpurnia	1	6
animal	100	4
sunday	1,000	3
fly	10,000	2
under	100,000	1
the	1,000,000	0

There is one idf value/weight for each word

$$\text{idf}_w = \log \frac{N}{\text{df}_w}$$

IDF example, suppose *N*=1 million

word	df_w	idf_w
calpurnia	1	
animal	100	
sunday	1,000	
fly	10,000	
under	100,000	
the	1,000,000	

What if we didn't use the log to dampen the weighting?

$$\mathrm{idf}_w = \log \frac{N}{\mathrm{df}_w}$$

IDF example, suppose *N*=1 million

word	df_w	idf_w
calpurnia	1	1,000,000
animal	100	10,000
sunday	1,000	1,000
fly	10,000	100
under	100,000	10
the	1,000,000	1

Tends to overweight rare words!

TF-IDF

One of the most common weighting schemes

TF = term frequency

IDF = inverse document frequency

$$a'_i = \underbrace{a_i}_{\text{TF}} \times \underbrace{\log N / \mathrm{df}_i}_{\text{IDF (word importance weight)}}$$

We can then use this with any of our similarity measures!

Stoplists: extreme weighting

Some words like 'a' and 'the' will occur in almost every document

- IDF will be 0 for any word that occurs in all documents
- For words that occur in almost all of the documents, they will be nearly 0

A ***stoplist*** is a list of words that should **not** be considered (in this case, similarity calculations)

- Sometimes this is the *n* most frequent words
- Often, it's a list of a few hundred words manually created

Stoplist

I	all-over	around	beneath	due	go
a	almost	as	beside	durin	goddamn
aboard	along	aside	besides	during	goody
about	alongside	astride	between	each	gosh
above	altho	at	bewteen	eh	half
across	although	atop	beyond	either	have
after	amid	avec	bi	en	he
afterwards	amidst	away	both	every	hell
against	among	back	but	ever	her
agin	amongst	be	by	everyone	herself
ago	an	because	ca.	everything	hey
agreed-upon	and	before	de	except	him
ah	another	beforehand	des	far	himself
alas	any	behind	despite	fer	his
albeit	anyone	behynde	do	for	ho
all	anything	below	down	from	how

If most of these end up with low weights anyway, why use a stoplist?

Stoplists

Two main benefits

- More fine grained control: some words may not be frequent, but may not have any content value (alas, teh, gosh)
- Often does contain many frequent words, which can drastically reduce our storage and computation

Any downsides to using a stoplist?

- For some applications, some stop words may be important

Our problems

Which of these have we addressed?

- word order
- length
- synonym
- spelling mistakes
- word importance
- word frequency

A model of word similarity!

Word overlap problems

A: When the defendant and his *lawyer* walked into the *court*, some of the victim supporters *turned* their backs to him.

B: When the defendant walked into the *courthouse* with his *attorney*, the crowd *truned* their backs on him.

Word similarity

How similar are two words?

score: $\text{sim}(w_1, w_2) = ?$

rank: w ? w_1 w_2 w_3

list: w_1 and w_2 are synonyms

applications?

Word similarity applications

- General text similarity
- Thesaurus generation
- Automatic evaluation
- Text-to-text
  - paraphrasing
  - summarization
  - machine translation
- information retrieval (search)

Word similarity

How similar are two words?

score: $\text{sim}(w_1, w_2) = ?$

rank: w ? w_1 w_2 w_3

list: w_1 and w_2 are synonyms

ideas? useful resources?

Word similarity

Four categories of approaches (maybe more)

- Character-based
  - turned vs. truned
  - cognates (night, nacht, nicht, natt, nat, noc, noch)
- Semantic web-based (e.g. WordNet)
- Dictionary-based
- Distributional similarity-based
  - similar words occur in similar contexts

Character-based similarity

sim(*turned*, *truned*) = ?

How might we do this using only the words (i.e. no outside resources?

Edit distance (Levenshtein distance)

The edit distance between w_1 and w_2 is the minimum number of operations to transform w_1 into w_2

Operations:

- insertion
- deletion
- substitution

EDIT(turned, truned) = ?
EDIT(computer, commuter) = ?
EDIT(banana, apple) = ?
EDIT(wombat, worcester) = ?

Edit distance

EDIT(turned, truned) = 2

- delete u
- insert u

EDIT(computer, commuter) = 1

- replace p with m

EDIT(banana, apple) = 5

- delete b
- replace n with p
- replace a with p
- replace n with l
- replace a with e

EDIT(wombat, worcester) = 6

Better edit distance

Are all operations equally likely?

- No

Improvement, give different weights to different operations

- replacing a for e is more likely than z for y

Ideas for weightings?

- Learn from actual data (known typos, known similar words)
- Intuitions: phonetics
- Intuitions: keyboard configuration

Vector character-based word similarity

sim(*turned*, *truned*) = ?

Any way to leverage our vector-based similarity approaches from last time?

Vector character-based word similarity

sim(*turned*, *truned*) = ?

Generate a feature vector based on the characters
(or could also use the set based measures at the character level)

a: 0	a: 0
b: 0	b: 0
c: 0	c: 0
d: 1	d: 1
e: 1	e: 1
f: 0	f: 0
g: 0	g: 0
…	…

problems?

Vector character-based word similarity

sim(*restful*, *fluster*) = ?

Character level loses a lot of information

a: 0	a: 0
b: 0	b: 0
c: 0	c: 0
d: 1	d: 1
e: 1	e: 1
f: 0	f: 0
g: 0	g: 0
…	…

ideas?

Vector character-based word similarity

sim(*restful*, *fluster*) = ?

Use character bigrams or even trigrams

aa: 0	aa: 0
ab: 0	ab: 0
ac: 0	ac: 0
…	…
es: 1	er: 1
…	…
fu: 1	fl: 1
…	…
re: 1	lu: 1
…	…

Word similarity

Four general categories

- Character-based
  - turned vs. truned
  - cognates (night, nacht, nicht, natt, nat, noc, noch)
- Semantic web-based (e.g. WordNet)
- Dictionary-based
- Distributional similarity-based
  - similar words occur in similar contexts

WordNet

Lexical database for English

- 155,287 words
- 206,941 word senses
- 117,659 synsets (synonym sets)
- ~400K relations between senses
- Parts of speech: nouns, verbs, adjectives, adverbs

Word graph, with word senses as nodes and edges as relationships

Psycholinguistics

- WN attempts to model human lexical memory
- Design based on psychological testing

Created by researchers at Princeton

- http://wordnet.princeton.edu/

Lots of programmatic interfaces

WordNet relations

- synonym
- antonym
- hypernyms
- hyponyms
- holonym
- meronym
- troponym
- entailment
- (and a few others)

WordNet relations

synonym – X and Y have similar meaning

antonym – X and Y have opposite meanings

hypernyms – subclass

- beagle is a hypernym of dog

hyponyms – superclass

- dog is a hyponym of beagle

holonym – contains part

- car is a holonym of wheel

meronym – part of

- wheel is a meronym of car

WordNet relations

troponym – for verbs, a more specific way of doing an action

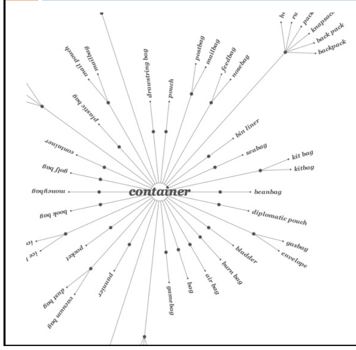
- run is a troponym of move
- dice is a troponym of cut

entailment – for verbs, one activity leads to the next

- sleep is entailed by snore

(and a few others)

WordNet



Graph, where nodes are words and edges are relationships

There is some hierarchical information, for example with hyp-er/o-nomy

WordNet: dog

Noun

- S: (n) **dog**, domestic dog, Canis familiaris (a member of the genus Canis (probably descended from the common wolf) that has been domesticated by man since prehistoric times; occurs in many breeds) "*the dog barked all night*"
- S: (n) frump, **dog** (a dull unattractive unpleasant girl or woman) "*she got a reputation as a frump*"; "*she's a real dog*"
- S: (n) **dog** (informal term for a man) "*you lucky dog*"
- S: (n) cad, bounder, blackguard, **dog**, hound, heel (someone who is morally reprehensible) "*you dirty dog*"
- S: (n) frank, frankfurter, hotdog, hot dog, **dog**, wiener, wienerwurst, weenie (a smooth-textured sausage of minced beef or pork usually smoked; often served on a bread roll)
- S: (n) pawl, detent, click, **dog** (a hinged catch that fits into a notch of a ratchet to move a wheel forward or prevent it from moving backward)
- S: (n) andiron, firedog, **dog**, dog-iron (metal supports for logs in a fireplace) "*the andirons were too hot to touch*"

Verb

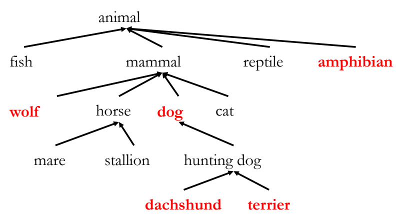
- S: (v) chase, chase after, trail, tail, tag, give chase, **dog**, go after, track (go after with the intent to catch) "*The policeman chased the mugger down the alley*"; "*the dog chased the rabbit*"

WordNet: dog

- S: (n) **dog, domestic dog, Canis familiaris (a member of the genus Canis (probably descended from the common wolf) that has been domesticated by man since prehistoric times; occurs in many breeds) "*the dog barked all night*"**
  - *direct hyponym* / *full hyponym*
  - *part meronym*
  - *member holonym*
  - *direct hypernym* / *inherited hypernym* / *sister term*

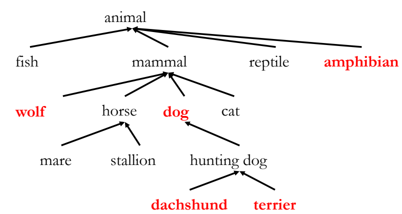
- ***direct hyponym*** / *full hyponym*
  - S: (n) puppy (a young dog)
  - S: (n) pooch, doggie, doggy, barker, bow-wow (informal terms for dogs)
  - S: (n) cur, mongrel, mutt (an inferior dog or one of mixed breed)
  - S: (n) lapdog (a dog small and tame enough to be held in the lap)
  - S: (n) toy dog, toy (any of several breeds of very small dogs kept purely as pets)
  - S: (n) hunting dog (a dog used in hunting game)
  - S: (n) working dog (any of several breeds of usually large powerful dogs bred to work as draft animals and guard and guide dogs)
  - S: (n) dalmatian, coach dog, carriage dog (a large breed having a smooth white coat with black or brown spots; originated in Dalmatia)
  - S: (n) basenji (small smooth-haired breed of African origin having a tightly curled tail and the inability to bark)
  - S: (n) pug, pug-dog (small compact smooth-coated breed of Asiatic origin having a tightly curled tail and broad flat wrinkled muzzle)

WordNet-like Hierarchy



To utilize WordNet, we often want to think about some graph-based measure.

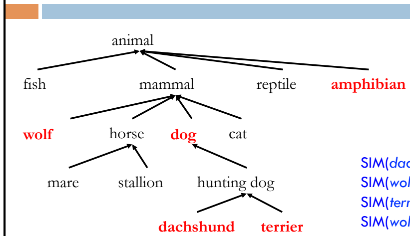
WordNet-like Hierarchy



Rank the following based on similarity:

SIM(*wolf*, *dog*)
SIM(*wolf*, *amphibian*)
SIM(*terrier*, *wolf*)
SIM(*dachshund*, *terrier*)

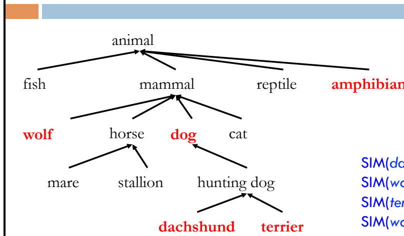
WordNet-like Hierarchy



SIM(*dachshund*, *terrier*)
SIM(*wolf*, *dog*)
SIM(*terrier*, *wolf*)
SIM(*wolf*, *amphibian*)

What information/heuristics did you use to rank these?

WordNet-like Hierarchy



SIM(*dachshund*, *terrier*)
SIM(*wolf*, *dog*)
SIM(*terrier*, *wolf*)
SIM(*wolf*, *amphibian*)

- path length is important (but not the only thing)
- words that share the same ancestor are related
- words lower down in the hierarchy are finer grained and therefore closer

WordNet similarity measures

path length doesn't work very well

Some ideas:
- path length scaled by the depth (Leacock and Chodorow, 1998)

With a little cheating:
- Measure the "information content" of a word using a corpus: how specific is a word?
  - words higher up tend to have less information content
  - more frequent words (and ancestors of more frequent words) tend to have less information content

WordNet similarity measures

Utilizing information content:
- information content of the lowest common parent (Resnik, 1995)
- information content of the words minus information content of the lowest common parent (Jiang and Conrath, 1997)
- information content of the lowest common parent divided by the information content of the words (Lin, 1998)