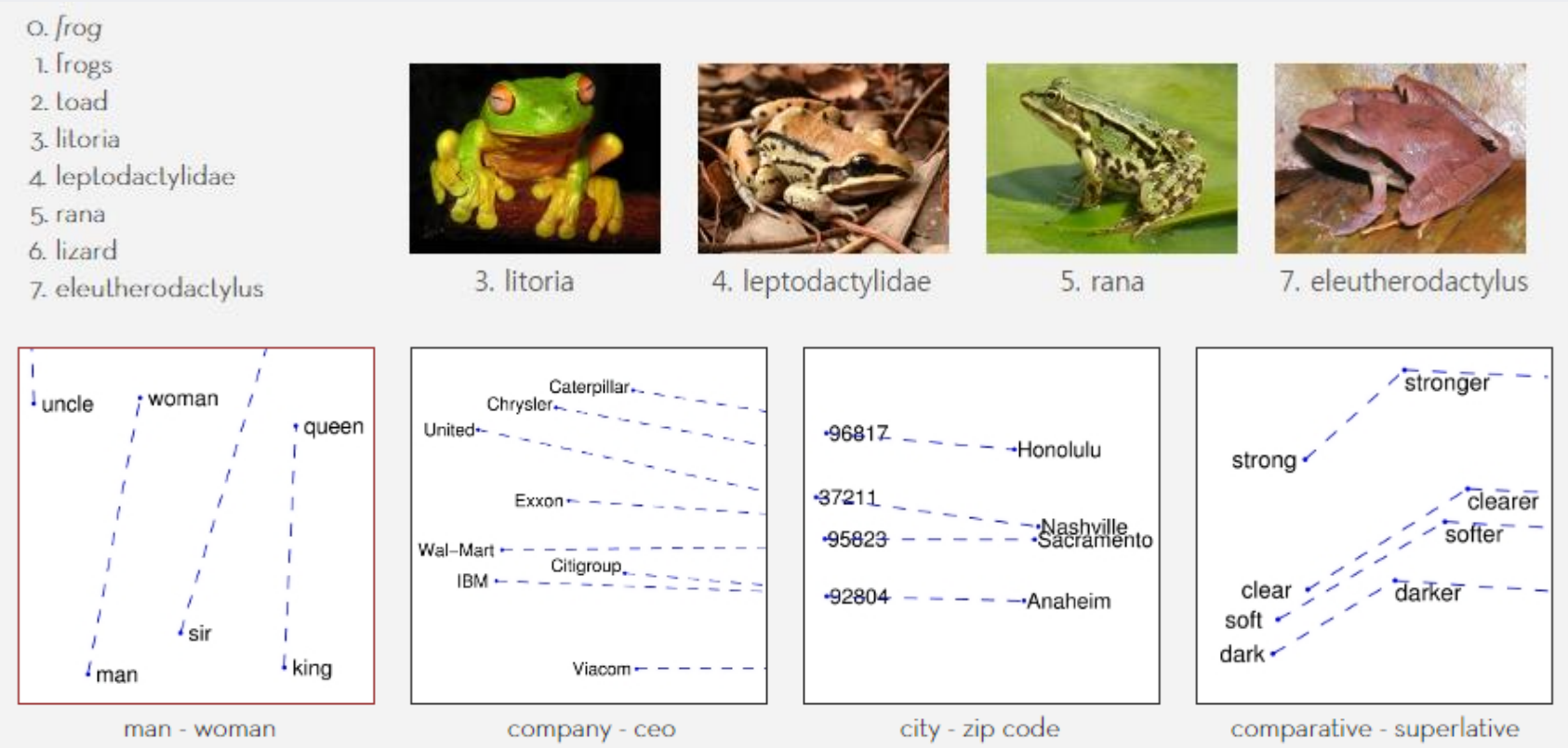


Lecture 4: Document Representation

Pilsung Kang

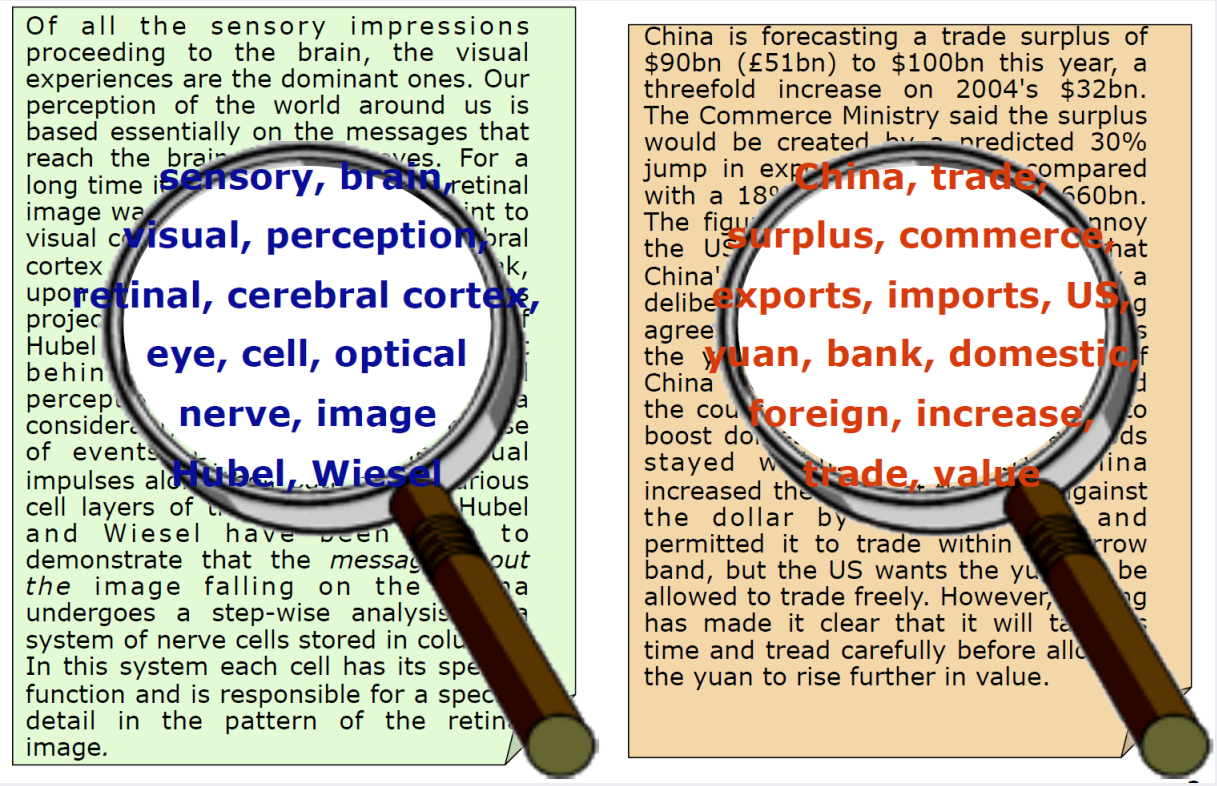
School of Industrial Management Engineering

Korea University

AGENDA

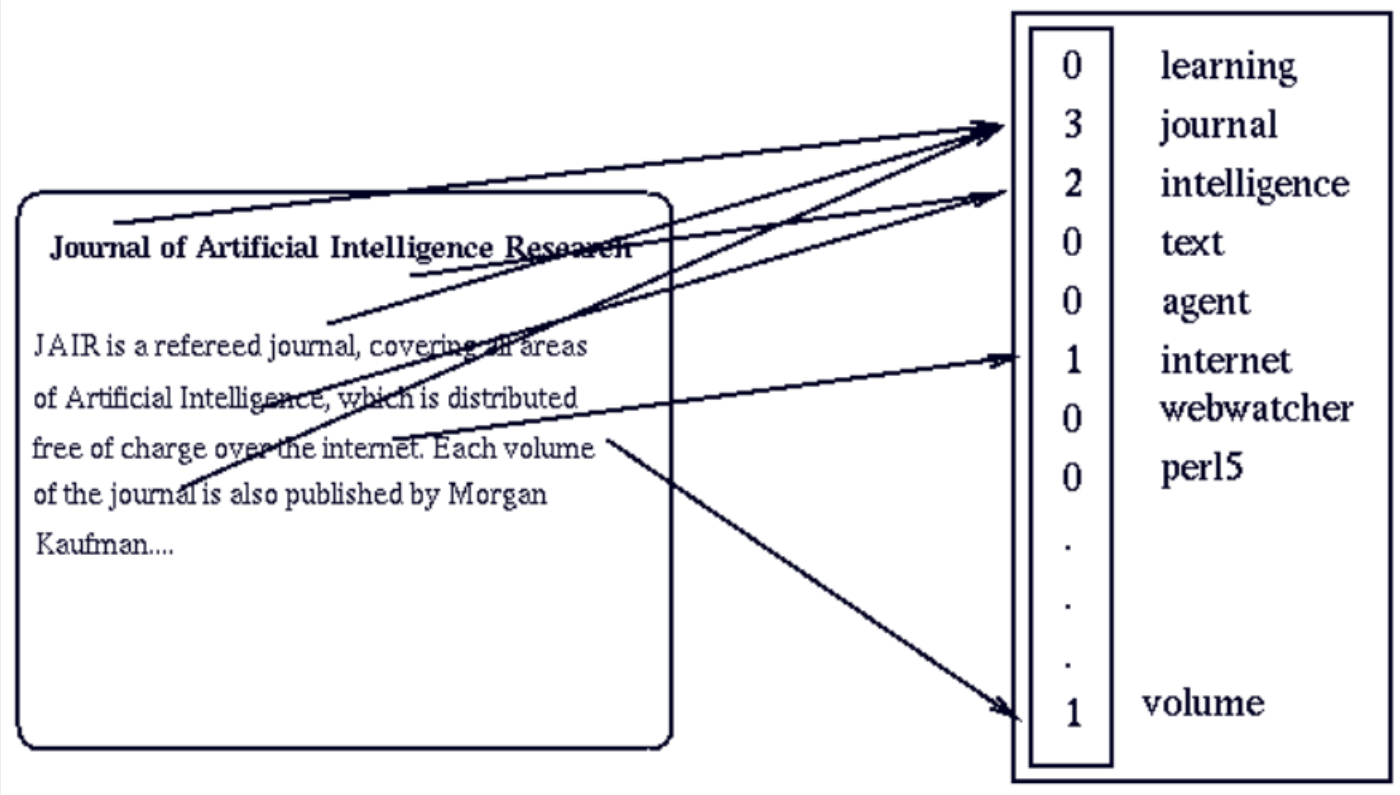
Bag of Words: Motivation

- Document Representation
  - ✓ How to represent a document in a structured way?
  - ✓ How to convert a unstructured text into a matrix form to apply those machine learning algorithms based on a vector space?



Bag of Words: Idea

- Bag-of-words
  - ✓ Simplifying representation method for documents where a text is represented in a vector of an unordered collection of words



Bag of Words: Idea

- Bag-of-words: Term-Document Matrix
  - ✓ Simplifying representation method for documents where a text is represented in a vector of an unordered collection of words

S1: Jon likes to watch movies. Mary likes too.

S2: John also likes to watch football game.

Binary representation

Word	S1	S2
John	1	1
Likes	1	1
To	1	1
Watch	1	1
Movies	1	0
Also	0	1
Football	0	1
Games	0	1
Mary	1	0
too	1	0

Frequency representation

Word	S1	S2
John	1	1
Likes	2	1
To	1	1
Watch	1	1
Movies	1	0
Also	0	1
Football	0	1
Games	0	1
Mary	1	0
too	1	0

Bag of Words: Idea

- Bag of words Representation in a Vector Space
  - ✓ The contents can be inferred from the frequency of words
  - ✓ Vector representation does not consider the ordering of words in a document
    - Visual words = independent features
    - John is quicker than Mary = Mary is quicker than John in BOW representation
  - ✓ We cannot reconstruct the original text based on the term-document matrix

Original text: John loves Mary

↓

BOW: (..., John, ..., loves, ..., Mary, ...)

(..., 1, ..., 1, ..., 1, ...)

↓

Reconstruction: John loves Mary?

Mary loves John?

Text Preprocessing

- Remove unnecessary information
  - ✓ They vs. they: different words in many systems
    - lower case is commonly used
  - ✓ Punctuation
    - Punctuations do not contain significant information → Remove them!
  - ✓ Numbers
    - Numbers are not critical in some domains but critical in other domains
    - Removing numbers should be carefully determined based on the domain for which a collection of text is about to be analyzed

Stop Words

- What are stop words?
  - ✓ Words that do not carry any information
    - Mainly functional role
    - Usually remove them to help the machine learning algorithms to perform better
  - ✓ Natural language dependent
    - English: a, about, above, across, after, again, against, all, also, etc.
    - 한국어: …습니다, …로서(써), …를 등

[Original text]

Information Systems Asia Web - provides research, IS-related commercial materials, interaction, and even research sponsorship by interested corporations with a focus on Asia Pacific region.

[After removing stop words]

Information Systems Asia Web provides research IS-related commercial materials interaction research sponsorship interested corporations focus Asia Pacific region

Stop Words

- Example 1: SMART stop words list
  - ✓ SMART: **S**ystem for the **M**echanical **A**nalysis and **R**etrieval of **T**ext
    - A total of 571 stop words

[1] "a"	"a's"	"able"	"about"	"above"	"according"	"accordingly"	"across"	"actually"	"after"	"afterwards"
[12] "again"	"against"	"ain't"	"all"	"allow"	"allows"	"almost"	"alone"	"along"	"already"	"also"
[23] "although"	"always"	"am"	"among"	"amongst"	"an"	"and"	"another"	"any"	"anybody"	"anyhow"
[34] "anyone"	"anything"	"anyway"	"anyways"	"anywhere"	"apart"	"appear"	"appreciate"	"appropriate"	"are"	"aren't"
[45] "around"	"as"	"aside"	"ask"	"asking"	"associated"	"at"	"available"	"away"	"awfully"	"b"
[56] "be"	"became"	"because"	"become"	"becomes"	"becoming"	"been"	"before"	"beforehand"	"behind"	"being"
[67] "believe"	"below"	"beside"	"besides"	"best"	"better"	"between"	"beyond"	"both"	"brief"	"but"
[78] "by"	"c"	"c'mon"	"c's"	"came"	"can"	"can't"	"cannot"	"cant"	"cause"	"causes"
[89] "certain"	"certainly"	"changes"	"clearly"	"co"	"com"	"come"	"comes"	"concerning"	"consequently"	"consider"
[100] "considering"	"contain"	"containing"	"contains"	"corresponding"	"could"	"couldn't"	"course"	"currently"	"d"	"definitely"
[111] "described"	"despite"	"did"	"didn't"	"different"	"do"	"does"	"doesn't"	"doing"	"don't"	"done"
[122] "down"	"downwards"	"during"	"e"	"each"	"edu"	"eg"	"eight"	"either"	"else"	"elsewhere"
[133] "enough"	"entirely"	"especially"	"et"	"etc"	"even"	"ever"	"every"	"everybody"	"everyone"	"everything"
[144] "everywhere"	"ex"	"exactly"	"example"	"except"	"f"	"far"	"few"	"fifth"	"first"	"five"
[155] "followed"	"following"	"follows"	"for"	"former"	"formerly"	"forth"	"four"	"from"	"further"	"furthermore"
[166] "g"	"get"	"gets"	"getting"	"given"	"gives"	"go"	"goes"	"going"	"gone"	"got"
[177] "gotten"	"greetings"	"h"	"had"	"hadn't"	"happens"	"hardly"	"has"	"hasn't"	"have"	"haven't"
[188] "having"	"he"	"he's"	"hello"	"help"	"hence"	"her"	"here"	"here's"	"hereafter"	"hereby"
[199] "herein"	"hereupon"	"hers"	"herself"	"hi"	"him"	"himself"	"his"	"hither"	"hopefully"	"how"
[210] "howbeit"	"however"	"i"	"i'd"	"i'll"	"i'm"	"i've"	"ie"	"if"	"ignored"	"immediate"
[221] "in"	"inasmuch"	"inc"	"indeed"	"indicate"	"indicated"	"indicates"	"inner"	"insofar"	"instead"	"into"
[232] "inward"	"is"	"isn't"	"it"	"it'd"	"it'll"	"it's"	"its"	"itself"	"j"	"just"
[243] "k"	"keep"	"keeps"	"kept"	"know"	"knows"	"known"	"l"	"last"	"lately"	"later"
[254] "latter"	"latterly"	"least"	"less"	"lest"	"let"	"let's"	"like"	"liked"	"likely"	"little"
[265] "look"	"looking"	"looks"	"ltd"	"m"	"mainly"	"many"	"may"	"maybe"	"me"	"mean"
[276] "meanwhile"	"merely"	"might"	"more"	"moreover"	"most"	"mostly"	"much"	"must"	"my"	"myself"
[287] "n"	"name"	"namely"	"nd"	"near"	"nearly"	"necessary"	"need"	"needs"	"neither"	"never"
[298] "nevertheless"	"new"	"next"	"nine"	"no"	"nobody"	"non"	"none"	"noone"	"nor"	"normally"
[309] "not"	"nothing"	"novel"	"now"	"nowhere"	"o"	"obviously"	"of"	"off"	"often"	"oh"
[320] "ok"	"okay"	"old"	"on"	"once"	"one"	"ones"	"only"	"onto"	"or"	"other"
[331] "others"	"otherwise"	"ought"	"our"	"ours"	"ourselves"	"out"	"outside"	"over"	"overall"	"own"
[342] "p"	"particular"	"particularly"	"per"	"perhaps"	"placed"	"please"	"plus"	"possible"	"presumably"	"probably"
[353] "provides"	"q"	"que"	"quite"	"qv"	"r"	"rather"	"rd"	"re"	"really"	"reasonably"
[364] "regarding"	"regardless"	"regards"	"relatively"	"respectively"	"right"	"s"	"said"	"same"	"saw"	"say"
[375] "saying"	"says"	"second"	"secondly"	"see"	"seeing"	"seem"	"seemed"	"seeming"	"seems"	"seen"
[386] "self"	"selves"	"sensible"	"sent"	"serious"	"seriously"	"seven"	"several"	"shall"	"she"	"should"
[397] "shouldn't"	"since"	"six"	"so"	"some"	"somebody"	"somehow"	"someone"	"something"	"sometime"	"sometimes"
[408] "somewhat"	"somewhere"	"soon"	"sorry"	"specified"	"specify"	"specifying"	"still"	"sub"	"such"	"sup"
[419] "sure"	"t"	"t's"	"take"	"taken"	"tell"	"tends"	"th"	"than"	"thank"	"thanks"
[430] "thanx"	"that"	"that's"	"thats"	"the"	"their"	"theirs"	"them"	"themselves"	"then"	"thence"
[441] "there"	"there's"	"thereafter"	"thereby"	"therefore"	"therein"	"theres"	"thereupon"	"these"	"they"	"they'd"
[452] "they'll"	"they're"	"they've"	"think"	"third"	"this"	"thorough"	"thoroughly"	"those"	"though"	"three"
[463] "through"	"throughout"	"thru"	"thus"	"to"	"together"	"too"	"took"	"toward"	"towards"	"tried"
[474] "tries"	"truly"	"try"	"trying"	"twice"	"two"	"u"	"un"	"under"	"unfortunately"	"unless"
[485] "unlikely"	"until"	"unto"	"up"	"upon"	"us"	"use"	"used"	"useful"	"uses"	"using"
[496] "usually"	"uucp"	"v"	"value"	"various"	"very"	"via"	"viz"	"vs"	"w"	"want"
[507] "wants"	"was"	"wasn't"	"way"	"we"	"we'd"	"we'll"	"we're"	"we've"	"welcome"	"well"
[518] "went"	"were"	"weren't"	"what"	"what's"	"whatever"	"when"	"whence"	"whenever"	"where"	"where's"
[529] "whereafter"	"whereas"	"whereby"	"wherein"	"whereupon"	"wherever"	"whether"	"which"	"while"	"whither"	"who"
[540] "who's"	"whoever"	"whole"	"whom"	"whose"	"why"	"will"	"willing"	"wish"	"with"	"within"
[551] "without"	"won't"	"wonder"	"would"	"would"	"wouldn't"	"x"	"y"	"yes"	"yet"	"you"
[562] "you'd"	"you'll"	"you're"	"you've"	"your"	"yours"	"yourself"	"yourselves"	"z"	"zero"	

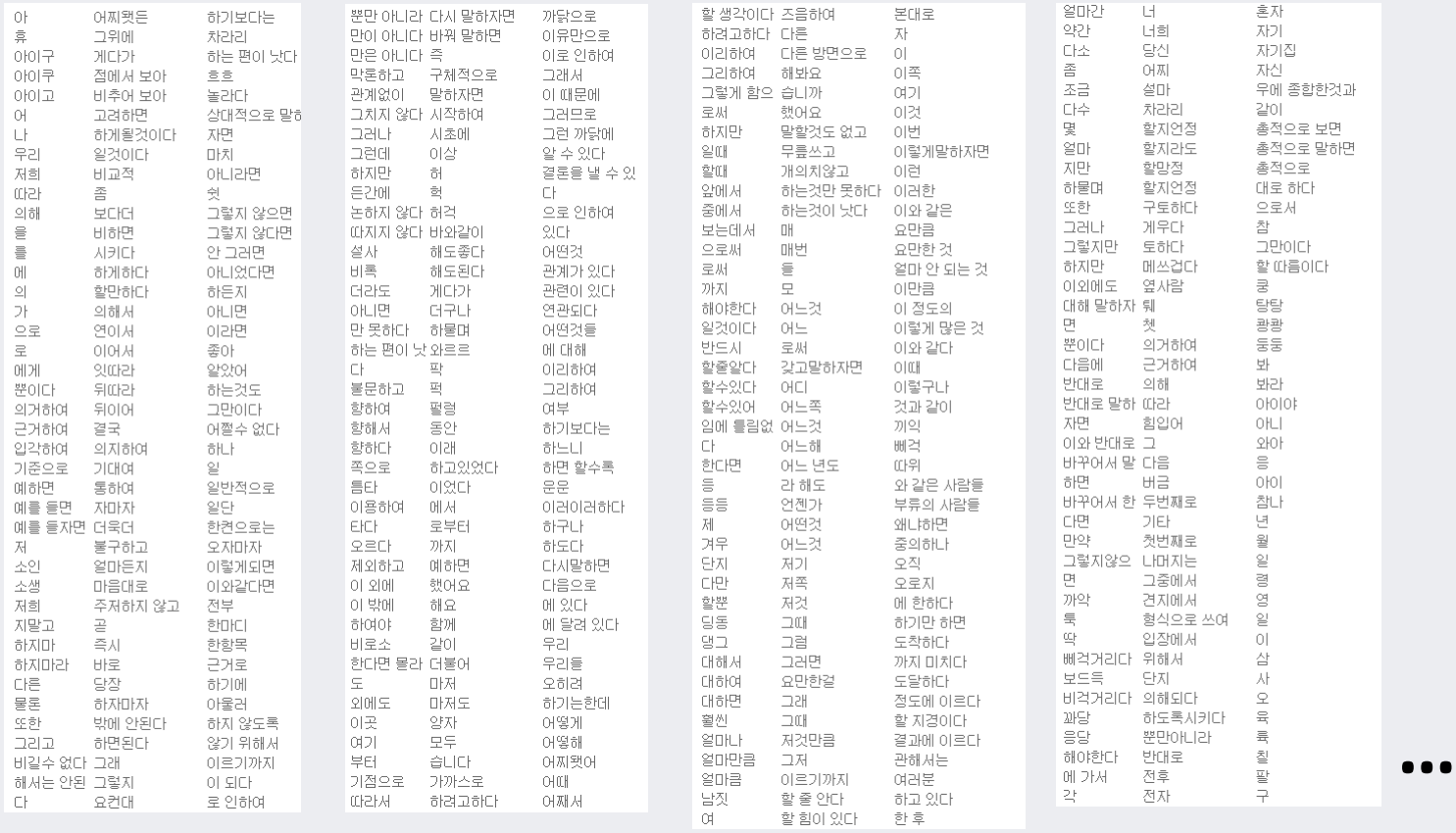
Stop Words

- Example 2: MySQL Stop words list
  - ✓ http://dev.mysql.com/doc/refman/5.1/en/fulltext-stopwords.html
    - A total of 543 stop words

a's	able	about	above	according	her	here	here's	hereafter	hereby	serious	seriously	seven	several	shall
accordingly	across	actually	after	afterwards	herein	hereupon	hers	herself	hi	she	should	shouldn't	since	six
again	against	ain't	all	allow	him	himself	his	hither	hopefully	so	some	somebody	somehow	someone
allows	almost	alone	along	already	how	howbeit	however	i'd	i'll	something	sometime	sometimes	somewhat	somewhere
also	although	always	am	among	i'm	i've	ie	if	ignored	soon	sorry	specified	specify	specifying
amongst	an	and	another	any	immediate	in	inasmuch	inc	indeed	still	sub	such	sup	sure
anybody	anyhow	anyone	anything	anyway	indicate	indicated	indicates	inner	insofar	t's	take	taken	tell	tends
anyways	anywhere	apart	appear	appreciate	instead	into	inward	is	isn't	th	than	thank	thanks	thanx
appropriate	are	aren't	around	as	it	it'd	it'll	it's	its	that	that's	thats	the	their
aside	ask	asking	associated	at	itself	just	keep	keeps	kept	theirs	them	themselves	then	thence
available	away	awfully	be	became	know	known	knows	last	lately	there	there's	thereafter	thereby	therefore
because	become	becomes	becoming	been	later	latter	latterly	least	less	therein	theres	thereupon	these	they
before	beforehand	behind	being	believe	lest	let	let's	like	liked	they'd	they'll	they're	they've	think
below	beside	besides	best	better	likely	little	look	looking	looks	third	this	thorough	thoroughly	those
between	beyond	both	brief	but	ltd	mainly	many	may	maybe	though	three	through	throughout	thru
by	c'mon	c's	came	can	me	mean	meanwhile	merely	might	thus	to	together	too	took
can't	cannot	cant	cause	causes	more	moreover	most	mostly	much	toward	towards	tried	tries	truly
certain	certainly	changes	clearly	co	must	my	myself	name	namely	try	trying	twice	two	un
com	come	comes	concerning	consequently	nd	near	nearly	necessary	need	under	unfortunately	unless	unlikely	until
consider	considering	contain	containing	contains	needs	neither	never	nevertheless	new	unto	up	upon	us	use
corresponding	could	couldn't	course	currently	next	nine	no	nobody	non	used	useful	uses	using	usually
definitely	described	despite	did	didn't	none	noone	nor	normally	not	value	various	very	via	viz
different	do	does	doesn't	doing	nothing	novel	now	nowhere	obviously	vs	want	wants	was	wasn't
don't	done	down	downwards	during	of	off	often	oh	ok	way	we	we'd	we'll	we're
each	edu	eg	eight	either	okay	old	on	once	one	we've	welcome	well	went	were
else	elsewhere	enough	entirely	especially	ones	only	onto	or	other	weren't	what	what's	whatever	when
et	etc	even	ever	every	others	otherwise	ought	our	ours	whence	whenever	where	where's	whereafter
everybody	everyone	everything	everywhere	ex	ourselves	out	outside	over	overall	whereas	whereby	wherein	whereupon	wherever
exactly	example	except	far	few	own	particular	particularly	per	perhaps	whether	which	while	whither	who
fifth	first	five	followed	following	placed	please	plus	possible	presumably	who's	whoever	whole	whom	whose
follows	for	former	formerly	forth	probably	provides	que	quite	qv	why	will	willing	wish	with
four	from	further	furthermore	get	rather	rd	re	really	reasonably	within	without	won't	wonder	would
gets	getting	given	gives	go	regarding	regardless	regards	relatively	respectively	wouldn't	yes	yet	you	you'd
goes	going	gone	got	gotten	right	said	same	saw	say	you'll	you're	you've	your	yours
greetings	had	hadn't	happens	hardly	saying	says	second	secondly	see	yourself	yourselves	zero		
has	hasn't	have	haven't	having	seeing	seem	seemed	seeming	seems					
he	he's	hello	help	hence	seen	self	selves	sensible	sent					

Stop Words

- Example 3: Stop words list in Korean
  - ✓ http://www.ranks.nl/stopwords/korean
    - A total of 677 stop words



AGENDA

Word Weighting: Term-Frequency (TF)

Nayak & Raghavan (2014)

- Term frequency $tf_{t,d}$
  - ✓ The number of times that the term **t** occurs in the document **d**

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	157	73	0	0	0	0
Brutus	4	157	0	1	0	0
Caesar	232	227	0	2	1	1
Calpurnia	0	10	0	0	0	0
Cleopatra	57	0	0	0	0	0
mercy	2	0	3	5	5	1
worser	2	0	1	1	1	0

Word Weighting: Term-Frequency (TF)

이재천, 김수경, 홍성연 (2015)

- Term frequency $tf_{t,d}$
  - ✓ The more frequently occurs, the more important it is

<산공 강의 상위 25%>

<산공 강의 하위 25%>

Word Weighting: Document Frequency (DF)

- Document frequency df_t
  - ✓ The number of documents in which the term **t** appears.
- Issues on DF
  - ✓ Rare terms are more informative than frequent terms across the document collection
    - is, can, the, of, …
  - ✓ Consider a term in the query that is rare in the collection (e.g., Pneumonoultramicroscopicsilicovolcanoconiosis (longest word in English, ))
  - ✓ A document containing this term is very likely to be relevant to the query
  - ✓ We should give a high weight for rare terms than common terms

Word Weighting: Inverse Document Frequency (IDF)

- Inverse document frequency idf_t
  - ✓ $idf_t = \log_{10}(N/df_t)$
  - ✓ We use $\log(N/df_t)$ instead of N/df_t to "dampen" the effect of idf
- IDF example with N = 1 million

term	df_t	idf_t
calpurnia	1	6
animal	100	5
sunday	1,000	4
fly	10,000	3
under	100,000	2
the	1,000,000	1

Word Weighting: TF-IDF

- TF-IDF
  - ✓ TF-IDF weight of a term is the product of its tf weight and its idf weight

$$TF - IDF(w) = tf(w) \times \log\left(\frac{N}{df(w)}\right)$$

More important if the term occur more frequently in a document

More important if the term occur less frequently in the other document

- ✓ Best known weighting scheme in information retrieval
- ✓ Increases with the number of occurrences within a document
- ✓ Increases with the rarity of the term in the collection

Word Weighting: TF-IDF

Nayak & Raghavan (2014)

- Example revisited
  - ✓ Each document is now represented by a real-valued vector of tf-idf weights in $R^{|V|}$

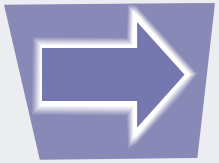
	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	5.25	3.18	0	0	0	0
Brutus	1.21	6.1	0	1	0	0
Caesar	8.59	2.54	0	1.51	0.25	0.35
Calpurnia	0	1.54	0	0	0	0
Cleopatra	2.85	0	0	0	0	0
mercy	1.51	0	1.9	0.12	5.25	0.88
worser	1.37	0	0.11	4.15	0.25	1.95

  - ✓ So, we have a |V|-dimensional vector space
    - Terms are axes of the space
    - Documents are points or vectors in this space
    - Very high dimensional: need to reduce the number of features!
    - Sparseness: most entries are zero

Word Weighting: TF-IDF

- TF-IDF Example
  - ✓ Q1: Which term is the most important for the document 1?
  - ✓ Q2: Which term is the least important for the document 1?

	Doc1	Doc2	Doc3
Term1	5	0	0
Term2	1	0	0
Term3	5	5	5
Term4	3	3	3
Term5	3	0	1



Doc1	TF	DF	IDF	TF-IDF
Term1	5	1	Log3	5log3
Term2	1	1	Log3	1log3
Term3	5	3	Log1	0
Term4	3	3	Log1	0
Term5	3	2	Log(3/2)	3log(3/2)

Word weighting: Term 1 > Term 5 > Term 2 > Term 3 = Term 4

TF Variants

Roelleke (2013)

- TF Variants

Definition 2.1 TF Variants: $\mathrm{TF}(t,d)$. $\mathrm{TF}(t,d)$ is a quantification of the within-document term frequency, tf_d. The main variants are:

$$\mathrm{tf}_d := \mathrm{TF}_{\mathrm{total}}(t,d) := \mathrm{lf}_{\mathrm{total}}(t,d) := n_L(t,d) \tag{2.1}$$

$$\mathrm{TF}_{\mathrm{sum}}(t,d) := \mathrm{lf}_{\mathrm{sum}}(t,d) := \frac{n_L(t,d)}{N_L(d)} \left(= \frac{\mathrm{tf}_d}{\mathrm{dl}}\right) \tag{2.2}$$

$$\mathrm{TF}_{\mathrm{max}}(t,d) := \mathrm{lf}_{\mathrm{max}}(t,d) := \frac{n_L(t,d)}{n_L(t_{\mathrm{max}},d)} \tag{2.3}$$

$$\mathrm{TF}_{\mathrm{log}}(t,d) := \mathrm{lf}_{\mathrm{log}}(t,d) := \log(1+n_L(t,d)) \left(= \log(1+\mathrm{tf}_d)\right) \tag{2.4}$$

$$\mathrm{TF}_{\mathrm{frac},K}(t,d) := \mathrm{lf}_{\mathrm{frac},K}(t,d) := \frac{n_L(t,d)}{n_L(t,d)+K_d} \left(= \frac{\mathrm{tf}_d}{\mathrm{tf}_d+K_d}\right) \tag{2.5}$$

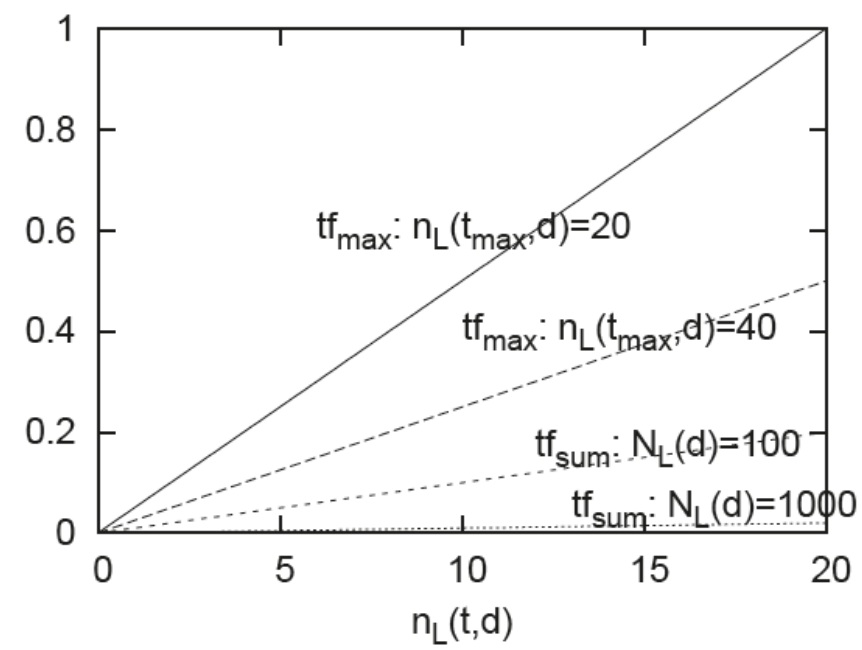
$$\mathrm{TF}_{\mathrm{BM25},k_1,b}(t,d) := \mathrm{lf}_{\mathrm{BM25},k_1,b}(t,d) := \frac{n_L(t,d)}{n_L(t,d)+k_1\cdot(b\cdot \mathrm{pivdl}(d,c)+(1-b))} \tag{2.6}$$

- K_d: (document length)/(average document length)

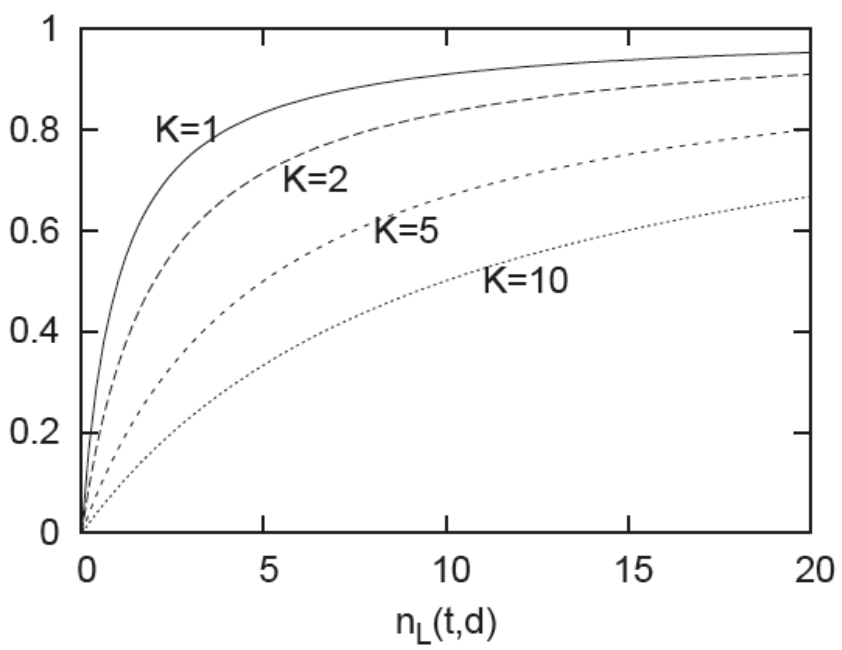
TF Variants

Roelleke (2013)

- TF Variants



(a) TF_{sum} and TF_{max}



(b) TF_{frac}

DF & IDF Variants

Roelleke (2013)

- DF & IDF Variants

Definition 2.3 DF Variants. $\mathrm{DF}(t,c)$ is a quantification of the document frequency, $\mathrm{df}(t,c)$. The main variants are:

$$\mathrm{df}(t,c) := \mathrm{df}_{\mathrm{total}}(t,c) := n_D(t,c) \tag{2.18}$$

$$\mathrm{df}_{\mathrm{sum}}(t,c) := \frac{n_D(t,c)}{N_D(c)} \left(= \frac{\mathrm{df}(t,c)}{N_D(c)}\right) \tag{2.19}$$

$$\mathrm{df}_{\mathrm{sum,smooth}}(t,c) := \frac{n_D(t,c)+0.5}{N_D(c)+1} \tag{2.20}$$

$$\mathrm{df}_{\mathrm{BIR}}(t,c) := \frac{n_D(t,c)}{N_D(c)-n_D(t,c)} \tag{2.21}$$

$$\mathrm{df}_{\mathrm{BIR,smooth}}(t,c) := \frac{n_D(t,c)+0.5}{N_D(c)-n_D(t,c)+0.5} \tag{2.22}$$

Definition 2.4 IDF Variants. $\mathrm{IDF}(t,c)$ is the negative logarithm of a DF quantification. The main variants are:

$$\mathrm{idf}_{\mathrm{total}}(t,c) := -\log \mathrm{df}_{\mathrm{total}}(t,c) \tag{2.23}$$

$$\mathrm{idf}(t,c) := \mathrm{idf}_{\mathrm{sum}}(t,c) := -\log \mathrm{df}_{\mathrm{sum}}(t,c) \tag{2.24}$$

$$\mathrm{idf}_{\mathrm{sum,smooth}}(t,c) := -\log \mathrm{df}_{\mathrm{sum,smooth}}(t,c) \tag{2.25}$$

$$\mathrm{idf}_{\mathrm{BIR}}(t,c) := -\log \mathrm{df}_{\mathrm{BIR}}(t,c) \tag{2.26}$$

$$\mathrm{idf}_{\mathrm{BIR,smooth}}(t,c) := -\log \mathrm{df}_{\mathrm{BIR,smooth}}(t,c) \tag{2.27}$$

TF-IDF Variants Summary

Roelleke (2013)

- The most commonly used TF-IDF in general

Term frequency		Document frequency		Normalization	
n (natural)	$\mathrm{tf}_{t,d}$	n (no)	1	n (none)	1
l (logarithm)	$1 + \log(\mathrm{tf}_{t,d})$	t (idf)	$\log \frac{N}{\mathrm{df}_t}$	c (cosine)	$\frac{1}{\sqrt{w_1^2+w_2^2+...+w_M^2}}$
a (augmented)	$0.5 + \frac{0.5 \times \mathrm{tf}_{t,d}}{\max_t(\mathrm{tf}_{t,d})}$	p (prob idf)	$\max\{0, \log \frac{N-\mathrm{df}_t}{\mathrm{df}_t}\}$	u (pivoted unique)	$1/u$
b (boolean)	$\begin{cases} 1 & \text{if } \mathrm{tf}_{t,d} > 0 \\ 0 & \text{otherwise} \end{cases}$			b (byte size)	$1/CharLength^{\alpha}$, $\alpha < 1$
L (log ave)	$\frac{1+\log(\mathrm{tf}_{t,d})}{1+\log(\mathrm{ave}_{t \in d}(\mathrm{tf}_{t,d}))}$				

Effects of TF-IDF Variants

- Comparative Study (Paltoglou and Thelwall, 2010)
  - ✓ Task 1: Classification of 2,000 movie reviews: positive vs. negative
  - ✓ Task 2: Multi-Domain Sentiment Data set (MDSD)
    - Four different product types: books, electronics, DVDs, and kitchen appliances
    - 1,000 positive & 1,000 negative for each type, 8,000 in total

Term Frequency

Notation	Term frequency
n (natural)	tf
l (logarithm)	$1 + log(tf)$
a (augmented)	$0.5 + \frac{0.5 \cdot tf}{max_t(tf)}$
b (boolean)	$\begin{cases} 1, & tf > 0 \\ 0, & otherwise \end{cases}$
L (log ave)	$\frac{1+log(tf)}{1+log(avg\_dl)}$
o (BM25)	$\frac{(k_1+1) \cdot tf}{k_1\left((1-b)+b \cdot \frac{dl}{avg\_dl}\right)+tf}$

Inverse Document Frequency

Notation	Inverse Document Frequency
n (no)	1
t (idf)	$log\frac{N}{df}$
p (prob idf)	$log\frac{N-df}{df}$
k (BM25 idf)	$log\frac{N-df+0.5}{df+0.5}$
$\Delta(t)$ (Delta idf)	$log\frac{N_1 \cdot df_2}{N_2 \cdot df_1}$
$\Delta(t')$ (Delta smoothed idf)	$log\frac{N_1 \cdot df_2+0.5}{N_2 \cdot df_1+0.5}$
$\Delta(p)$ (Delta prob idf)	$log\frac{(N_1-df_1) \cdot df_2}{df_1 \cdot (N_2-df_2)}$
$\Delta(p')$ (Delta smoothed prob idf)	$log\frac{(N_1-df_1) \cdot df_2+0.5}{(N_2-df_2) \cdot df_1+0.5}$
$\Delta(k)$ (Delta BM25 idf)	$log\frac{(N_1-df_1+0.5) \cdot df_2+0.5}{(N_2-df_2+0.5) \cdot df_1+0.5}$

Normalization

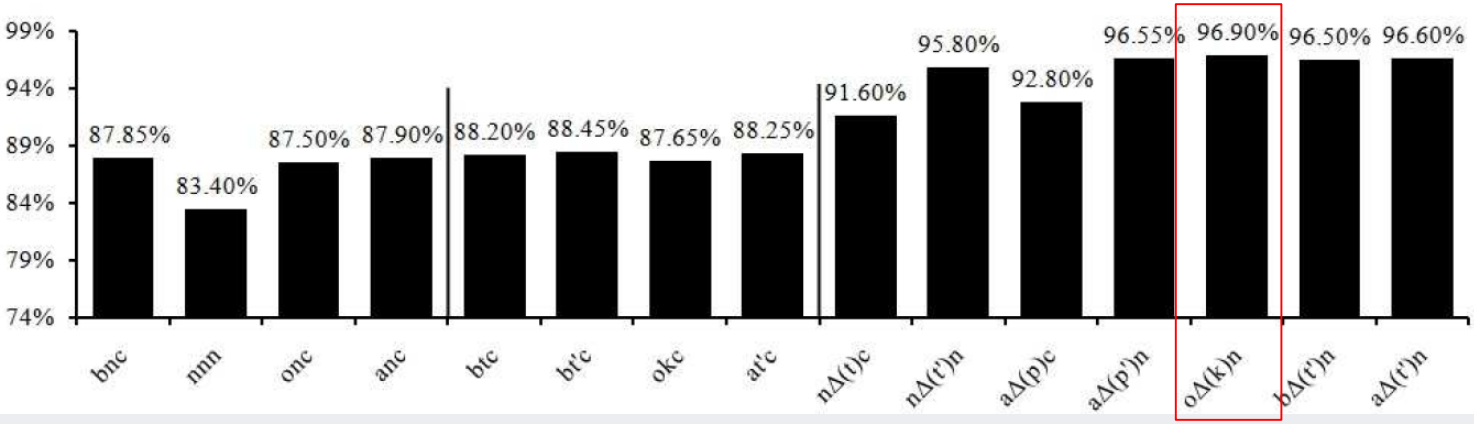
Notation	Normalization
n (none)	1
c (cosine)	$\frac{1}{\sqrt{w_1^2+w_2^2+...+w_n^2}}$

Effects of TF-IDF Variants

Paltoglou and Thelwall (2010)

- Experimental Result 1: Movie Reviews
  - ✓ Base classifier: support vector machine (SVM)

Data set	#Documents	#Terms	#Unique Terms	Average #Terms per Document
Movie Reviews	2,000	1,336,883	39,399	668
Multi-Domain Sentiment Dataset (MDSD)	8,000	1,741,085	455,943	217
BLOGS06	17,898	51,252,850	367,899	2,832



o (BM25)	$\frac{(k_1+1)\cdot tf}{k_1\left((1-b)+b\cdot\frac{dl}{avg\_dl}\right)+tf}$

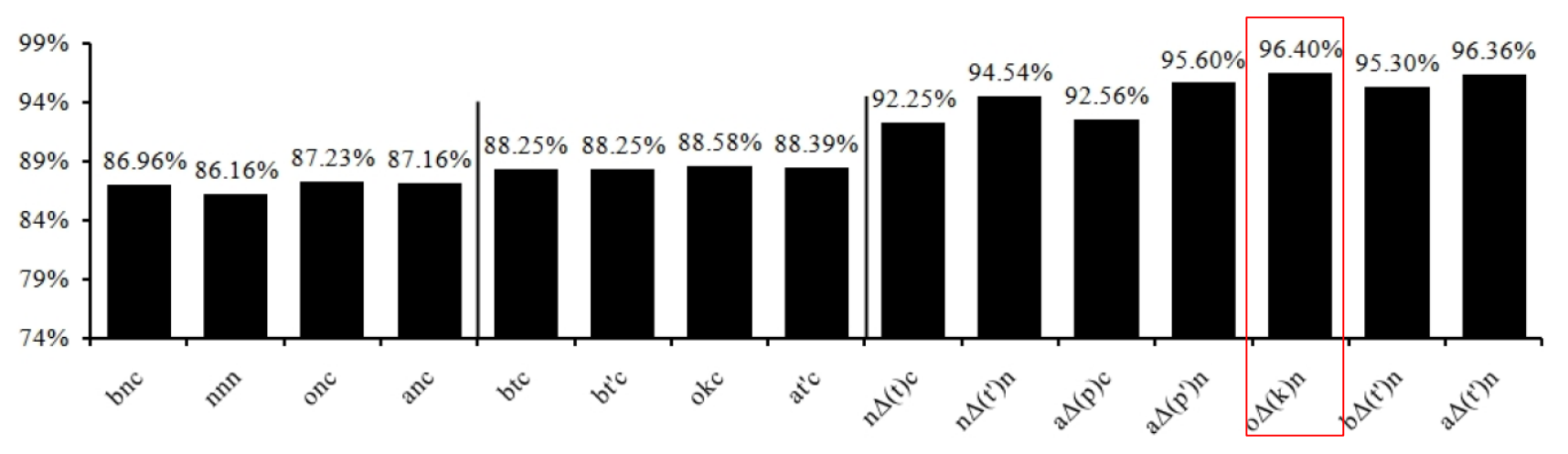
$\Delta(k)$ (Delta BM25 idf)	$log\frac{(N_1-df_1+0.5)\cdot df_2+0.5}{(N_2-df_2+0.5)\cdot df_1+0.5}$

n (none)	1

Effects of TF-IDF Variants

Paltoglou and Thelwall (2010)

- Experimental Result 2: MDSD
  - ✓ Base classifier: support vector machine (SVM)



o (BM25)	$\frac{(k_1+1)\cdot tf}{k_1\left((1-b)+b\cdot\frac{dl}{avg\_dl}\right)+tf}$

$\Delta(k)$ (Delta BM25 idf)	$log\frac{(N_1-df_1+0.5)\cdot df_2+0.5}{(N_2-df_2+0.5)\cdot df_1+0.5}$

n (none)	1

AGENDA

01 Bag of Words

02 Word Weighting

03 N-Grams

04 Distributed Representations

05 R Exercise

N-Grams

- N-Gram models in NLP
  - ✓ Use the previous N-1 words in a sequence to predict the next word

$$P(w_n|w_{n-1}, w_{n-2},\dots, w_1) = \frac{P(w_n,w_{n-1},w_{n-2},\dots, w_1)}{P(w_{n-1},w_{n-2},\dots, w_1)}$$

  - ✓ Q) One of the hottest topics in artificial intelligence is deep _____
    - blue vs. frying vs. learning ?
- N-Gram in Text Mining
  - ✓ Some phrases are very useful in text clustering/categorization!
    - Six sigma, supply chain management, big data, etc.
  - ✓ Term-frequency for n-grams can be utilized.
  - ✓ Domain-dependent.

N-Grams

- Bigram example
  - ✓ Total counts in a corpus

	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0

N-Grams

Furnkranz (1998)

- Empirical evaluation
  - ✓ Data sets
    - 20 newsgroup data set: 20,000 articles (1,000 for each category)
    - 21578 REUTERS newswire articles: 21,578 articles with 90 categories
  - ✓ Classification algorithm: RIPPER
- Results for 20 newsgroup dataset

Pruning	n-grams	Error rate	CPU secs.	No. Features
set-of-words		47.07 ± 0.92	n.a.	71,731
DF: 3 TF: 5	1	46.18 ± 0.94	12686.12	36,534
	2	45.28 ± 0.51	15288.32	113,716
	3	45.05 ± 1.22	15253.27	155,184
	4	45.18 ± 1.17	14951.17	189,933
DF: 5 TF: 10	1	45.51 ± 0.83	12948.31	22,573
	2	45.34 ± 0.68	13280.73	44,893
	3	46.11 ± 0.73	12995.66	53,238
	4	46.11 ± 0.72	13063.68	59,455
DF: 10 TF: 20	1	45.88 ± 0.89	10627.10	13,805
	2	45.53 ± 0.86	13080.32	20,295
	3	45.58 ± 0.87	11640.18	22,214
	4	45.74 ± 0.62	11505.92	23,565
DF: 25 TF: 50	1	48.23 ± 0.69	10676.43	n.a.
	2	48.97 ± 1.15	8870.05	n.a.
	3	48.69 ± 1.04	10141.25	n.a.
	4	48.36 ± 1.01	10436.58	n.a.
	5	48.36 ± 1.01	10462.65	n.a.
DF: 50 TF: 100	1	51.54 ± 0.60	8547.43	n.a.
	2	49.71 ± 0.53	8164.27	n.a.
	3	51.21 ± 1.26	8079.59	n.a.
	4	51.21 ± 1.26	8078.55	n.a.
	5	51.21 ± 1.26	8147.75	n.a.
DF: 75 TF: 150	1	52.59 ± 0.71	6609.05	n.a.
	2	52.83 ± 0.25	6532.80	n.a.
	3	52.36 ± 0.48	6128.49	n.a.
	4	52.36 ± 0.48	6128.49	n.a.
	5	52.36 ± 0.48	6119.27	n.a.

N-Grams

Furnkranz (1998)

- Results for 21578 REUTERS
  - ✓ Classification accuracy is the highest with bigram features

Pruning	n-grams	Recall	Precision	F1	Accuracy	No. Features
set-of-words		76.71	83.42	79.92	99.5140	n.a.
	1	77.22	83.55	80.26	99.5211	9,673
DF: 3	2	80.34	82.03	81.18	99.5302	28,045
TF: 5	3	77.56	82.74	80.07	99.5130	38,646
	4	78.18	82.31	80.19	99.5130	45,876
	1	77.19	83.65	80.29	99.5221	6,332
DF: 5	2	80.05	82.06	81.04	99.5278	13,598
TF: 10	3	77.96	82.29	80.07	99.5106	17,708
	4	78.21	82.13	80.12	99.5106	20,468
	1	76.92	83.99	80.30	99.5241	4,068
DF: 10	2	79.06	82.04	80.52	99.5177	7,067
TF: 20	3	77.32	82.67	79.91	99.5096	8,759
	4	76.98	82.91	79.84	99.5096	9,907

AGENDA

01 Bag of Words

02 Word Weighting

03 N-Grams

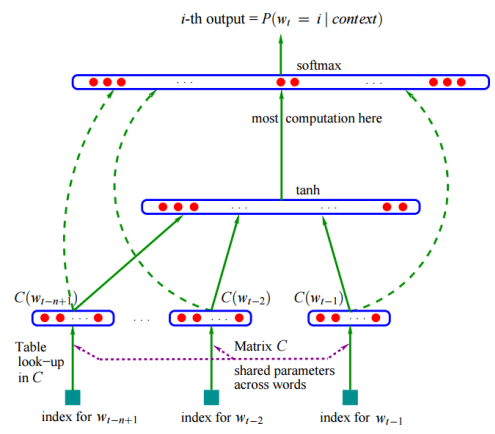
04 Distributed Representations

05 R Exercise

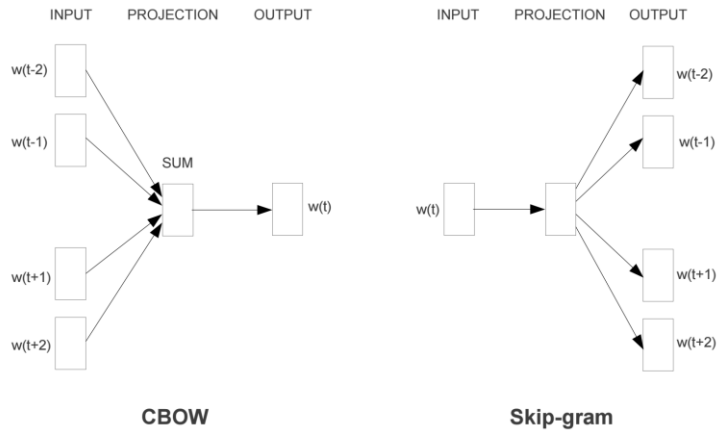
Distributed Representation: Word Embedding

- Word Embedding
  - ✓ The purpose of word embedding is to map the words in a language into a vector space so that semantically similar words are located close to each other.
  - ✓ Hypothetically, the number of token in English is estimated about 13M, there exist a d (< 13M) dimensional optimal space that can embed the meaning of all words.

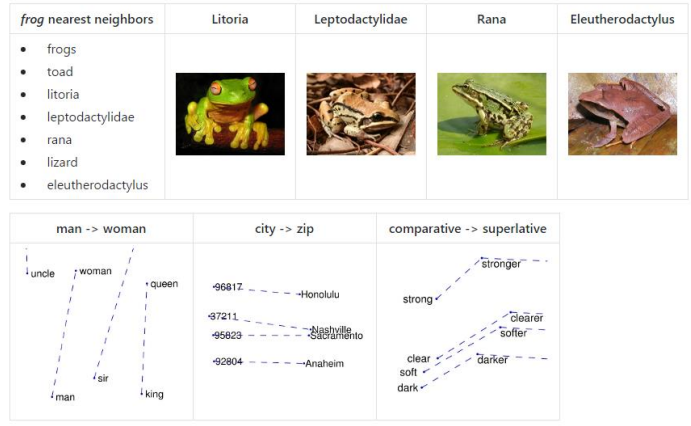
NNLM



Word2Vec



GloVe



Distributed Representation: Word Embedding

- Word vectors: one-hot vector
  - ✓ The most simple & intuitive representation

$$w^{aardvark} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, w^{a} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, w^{at} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \ldots w^{zebra} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}$$

  - ✓ Can make a vector representation, but similarities between words cannot be preserved.

motel [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0] AND
hotel [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0] = 0

$$(w^{hotel})^{\top} w^{motel} = (w^{hotel})^{\top} w^{cat} = 0$$

Distributed Representation: Word Embedding

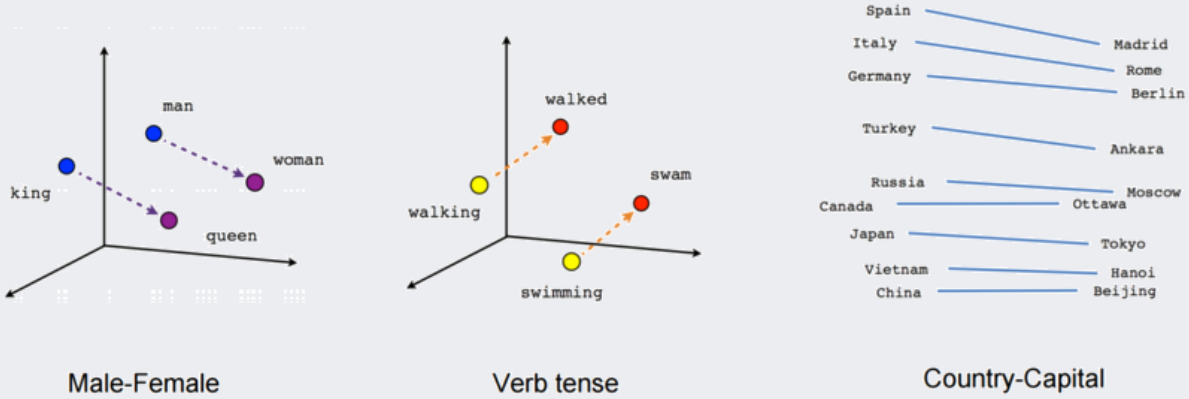
- Word vectors: distributed representation
  - ✓ A parameterized function mapping words in some language to a certain dimensional vectors

$$W : \text{words} \rightarrow \mathbb{R}^n$$

$$W(\text{“cat”}) = (0.2,\ \text{-}0.4,\ 0.7,\ \ldots)$$

$$W(\text{“mat”}) = (0.0,\ 0.6,\ \text{-}0.1,\ \ldots)$$

- Interesting feature of word embedding
  - ✓ Semantic relationship between words can be preserved



Male-Female | Verb tense | Country-Capital

Neural Network Language Model (NNLM)

Bengio et al. (2003)

- Purpose
  - ✓ Fighting the curse of dimensionality with distributed representations

✓ Associate with each word in the vocabulary a **distributed word feature vector** (a real valued vector in R^m),

✓ Express the joint **probability function** of word sequences in terms of the feature vectors of these words in the sequence,

✓ Learn simultaneously the **word feature vectors** and the parameters of that **probability function**

Neural Network Language Model (NNLM)

- Why it works?
  - ✓ If we knew that `dog` and `cat` played similar roles (semantically and synthetically), and similarity for (`the`, `a`), (`bedroom`, `room`), (`is`, `was`), (`running`, `walking`), we could naturally generalize from

```
The cat is walking in the bedroom
```

to

```
A dog was running in a room
The cat is running is a room
A dog is walking in a bedroom
The dog was waling in the room
...
```

Neural Network Language Model (NNLM)

Kim et al. (2016)

- Comparison with Count-based Language Models
  - ✓ Count-based Language Models

By the chain rule, any distribution can be factorized as

$$p(w_1, \ldots, w_T) = \prod_{t=1}^{T} p(w_t | w_1, \ldots, w_{t-1})$$

Count-based n-gram language models make a Markov assumption:

$$p(w_t | w_1, \ldots, w_t) \approx p(w_t | w_{t-n}, \ldots, w_{t-1})$$

Need smoothing to deal with rare n-grams.

Neural Network Language Model (NNLM)

Kim et al. (2016)

- Comparison with Count-based Language Models

✓NNLM

- Represent words as dense vectors in $\mathbb{R}^n$ (word embeddings).

$\mathbf{w}_t \in \mathbb{R}^{|\mathcal{V}|}$: One-hot representation of word $\in \mathcal{V}$ at time t
$\Rightarrow \mathbf{x}_t = \mathbf{X}\mathbf{w}_t$: Word embedding ($\mathbf{X} \in \mathbb{R}^{n \times |\mathcal{V}|}, n < |\mathcal{V}|$)

- Train a neural net that composes history to predict next word.

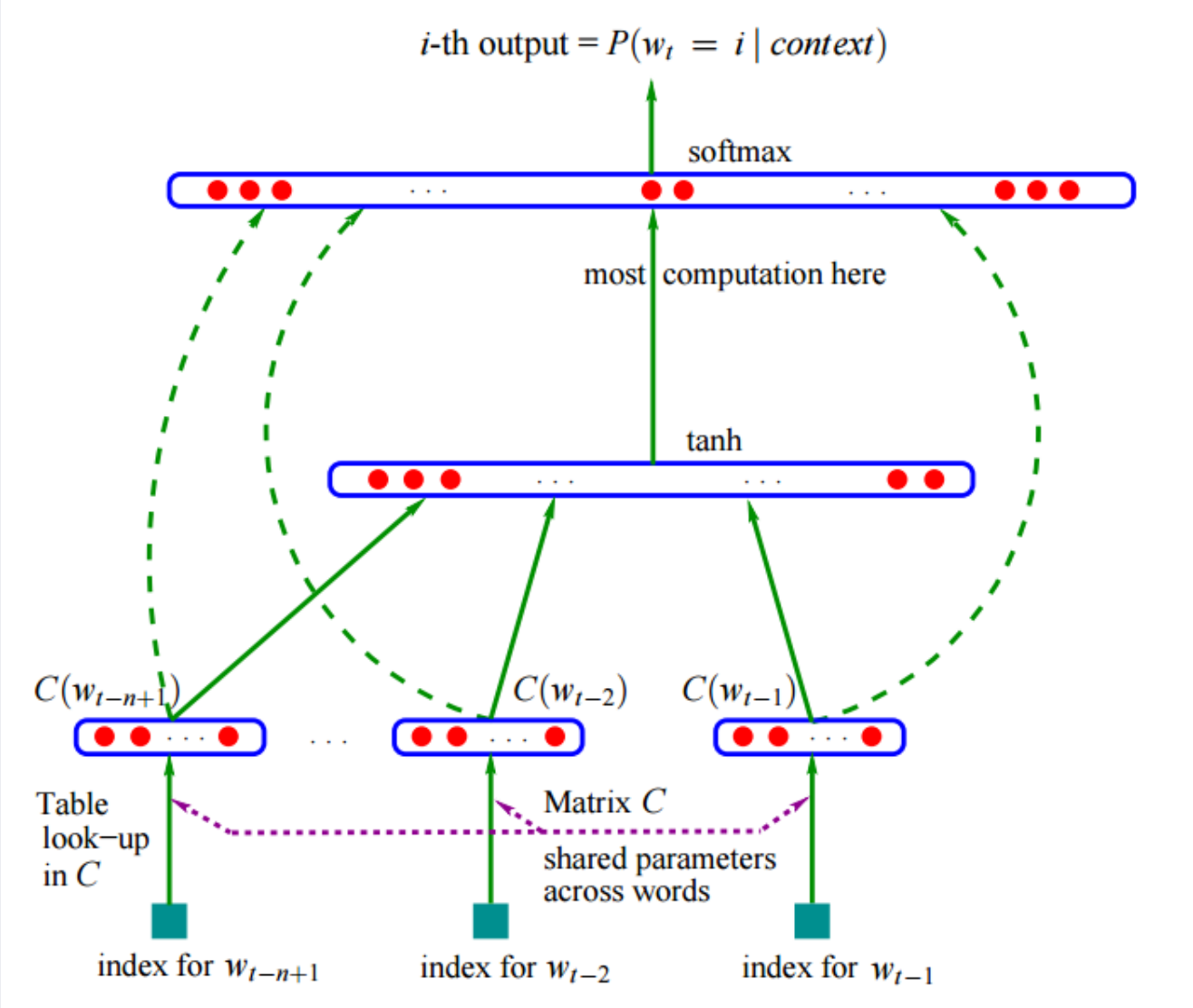
$$p(w_t = j | w_1, \ldots, w_{t-1}) = \frac{\exp(\mathbf{p}^j \cdot g(\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}) + q^j)}{\sum_{j' \in \mathcal{V}} \exp(\mathbf{p}^{j'} \cdot g(\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}) + q^{j'})}$$
$$= \text{softmax}(\mathbf{P} g(\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}) + \mathbf{q})$$

$\mathbf{p}^j \in \mathbb{R}^m, q^j \in \mathbb{R}$: Output word embedding/bias for word $j \in \mathcal{V}$
g : Composition function

Neural Network Language Model (NNLM)

Bengio et al. (2003)

- Learning NNLM



Neural Network Language Model (NNLM)

Bengio et al. (2003)

- Learning NNLM
  - ✓ The objective is to learn a good model $f(w_t, \cdots w_{t-n+1}) = \hat{P}(w_t|w_1^{t-1})$, in the sense that it gives high out-of-sample likelihood
  - ✓ Two constraints
    - For any choice of $w_1^{t-1}, \sum_{i=1}^{|V|} f(i, w_{t-1} \cdots w_{t-n+1}) = 1$
    - f > 0

Neural Network Language Model (NNLM)

Bengio et al. (2003)

- Learning NNLM

  ✓ Decompose the function $f(w_t, \cdots w_{t-n+1}) = \hat{P}(w_t|w_1^{t-1})$ in two parts:

  - A mapping *C*, a.k.a the lookup table, from any element *i* of *V* to a real vector $C(i) \in R^m$, it represents the distributed feature vectors associated with each word in the vocabulary. *C* is represented by a $|V| \times m$ matrix of free parameters

  - The probability function over words, expressed with *C*: a function g maps an input sequence of feature vectors for words in context, $(C(w_{t-n+1}), \cdots C(w_{t-1}))$, to a conditional probability distribution over words in V for the next word w_t. The output of *g* is a vector whose *i*-th element estimates the probability $\hat{P}(w_t = i|w_1^{t-1})$

Neural Network Language Model (NNLM)

Bengio et al. (2003)

- Learning NNLM

$$f(i, w_{t-1} \cdots w_{t-n+1}) = g(i, C(w_{t-1}), \cdots, C(w_{t-n+1}))$$

✓ The function f is a composition of these two mappings (*C* and *g*), with *C* being shared across all the words in the context.

- The parameters of the mapping C are simply the feature vectors themselves, represented by a $|V| \times m$ matrix C whose row i is the feature vector C(i) for word i
- The function g may be implemented by a feed-forward or recurrent neural network or another parameterized function, with parameters ω.

✓ Training is done by maximizing the penalized log0likelihood of the training corpus

$$L = \frac{1}{T} \sum_{t} \log f(i, w_{t-1} \cdots w_{t-n+1}; \theta) + R(\theta)$$

Neural Network Language Model (NNLM)

Bengio et al. (2003)

- Learning NNLM
  - ✓ The neural network has one hidden layer beyond the word features mapping, and optionally, direct connections from the word features to the output.
  - ✓ Computation of the output layer

$$\hat{P}(w_t|w_{t-1},\cdots,w_{t-n+1}) = \frac{\exp(y_{w_t})}{\sum_i \exp(y_i)}$$

$$y = b + Wx + U \cdot tanh(d + Hx)$$

  - W is optionally zero (no direct connections from the input to the output)
  - x is the word features layer activation vector $x = (C(w_{t-1}),\cdots,C(w_{t-n+1}))$
  - h is the number of hidden units

Neural Network Language Model (NNLM)

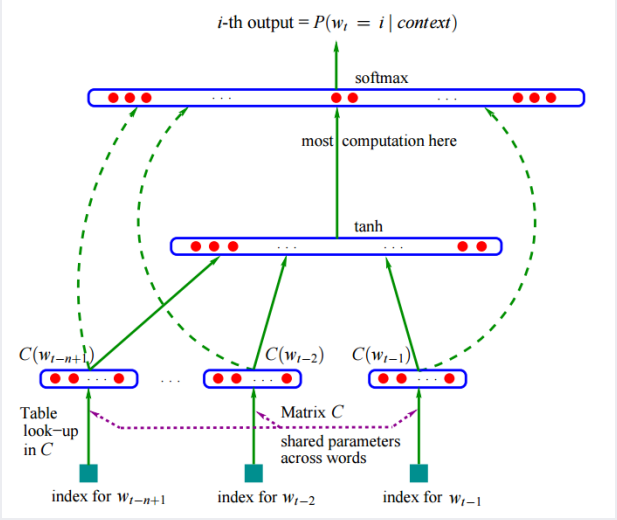
Bengio et al. (2003)

- Learning NNLM

  ✓ The free parameters of the model

$$y = b + Wx + U \cdot tanh(d + Hx)$$

  - the output bias b (|V| elements)
  - the hidden layer biases d (with h elements)
  - the hidden-to-output weights U (a |V|*h matrix)
  - the word features to output weights W (a |V|*(n-1)m matrix)
  - the hidden layer weight H (a h*(n-1)*m matrix)



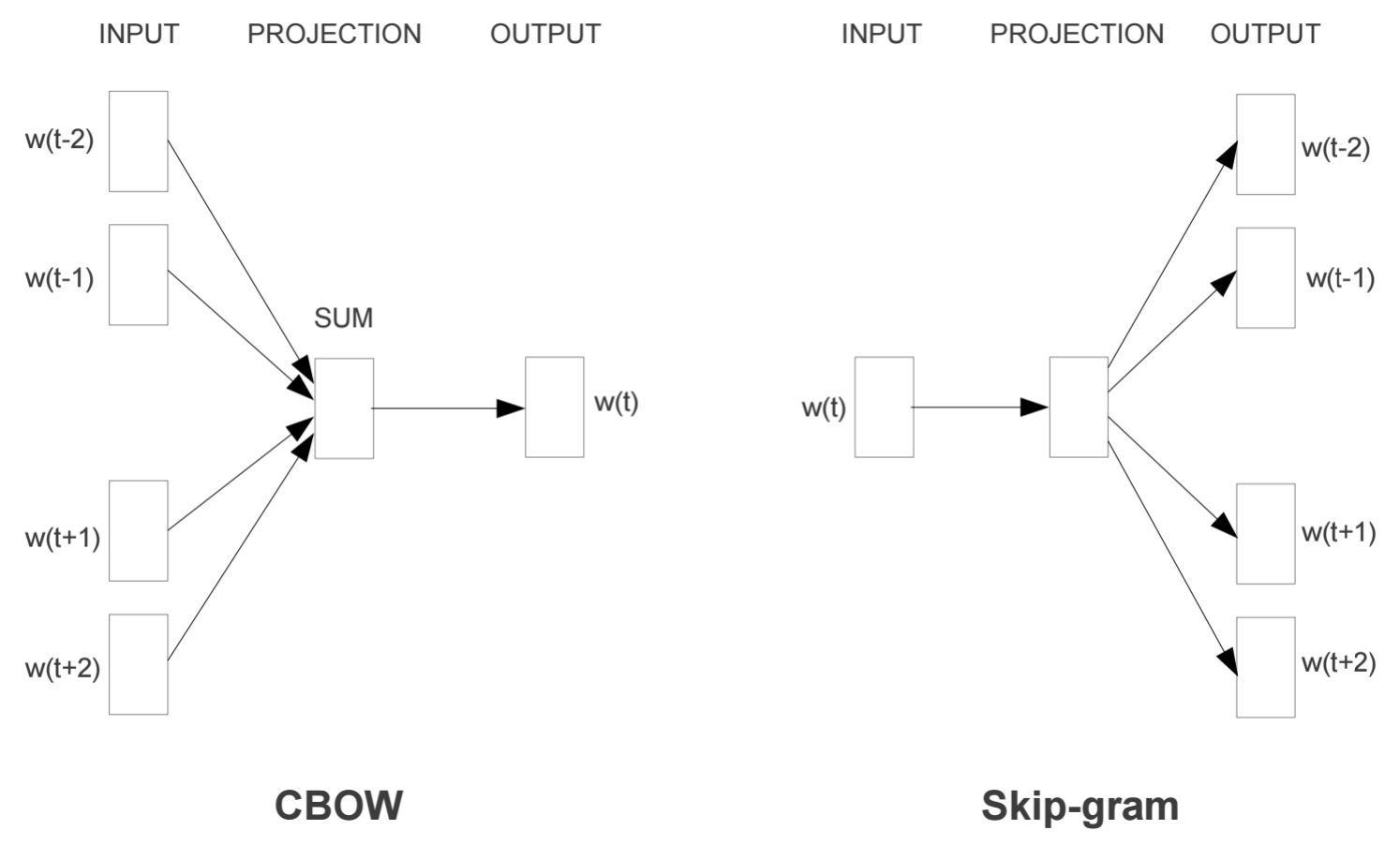
  ✓ Stochastic gradient ascent

$$\theta \leftarrow \theta + \varepsilon \frac{\partial \log \hat{P}(w_t | w_{t-1}, \cdots, w_{t-n+1})}{\partial \theta}$$

Word2Vec

Mikolov et al. (2013)

- Two Architectures
  - ✓ Continuous bag-of-words (CBOW) vs. Skip-gram



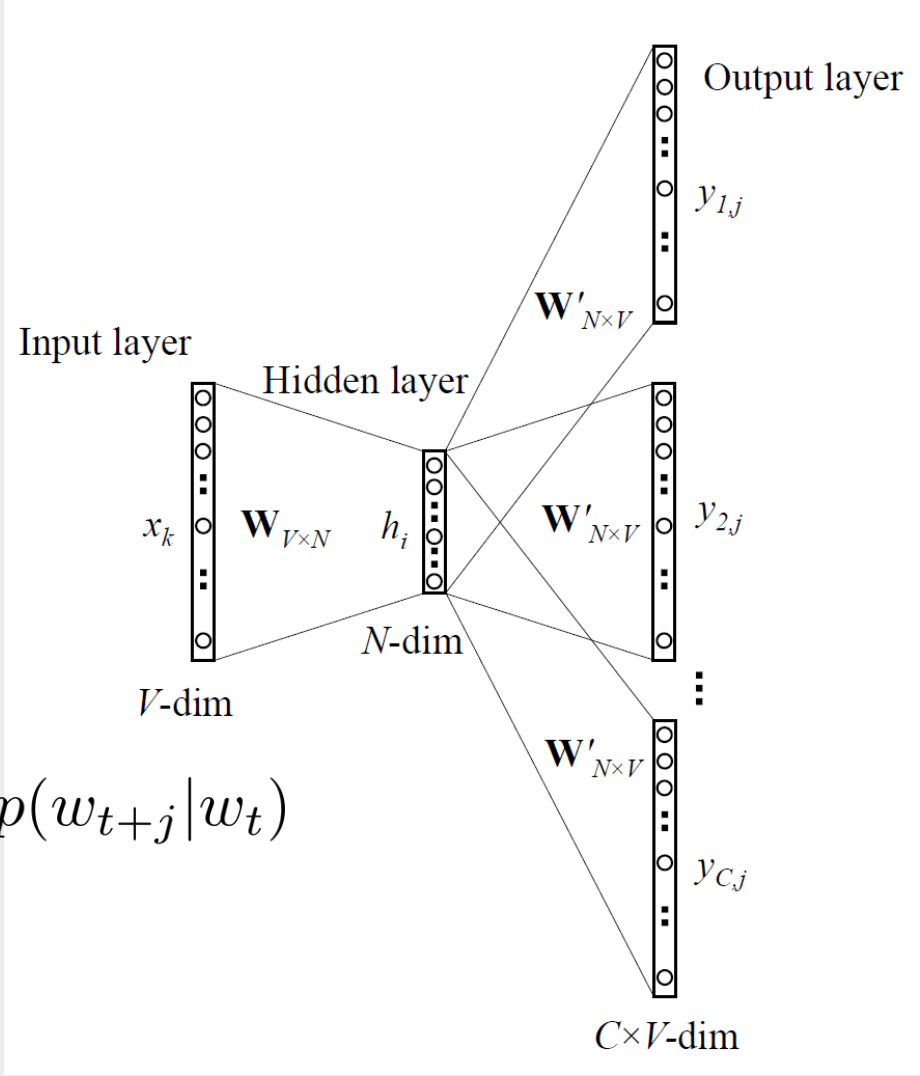
Word2Vec

Rong (2014)

- Learning representations: Skip-gram approach
  - ✓ Predict surrounding words in a window of length m of every word
- Objective function
  - ✓ Maximize the log probability of any context word given the current center word

$$J(\theta) = \frac{1}{T}\sum_{t=1}^{T} \sum_{-m \le j \le m, j \ne 0} \log p(w_{t+j}|w_t)$$

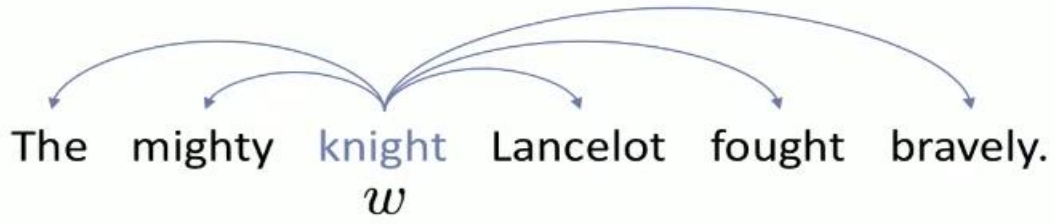
  - ✓ where θ represents all variables we optimize



Word2Vec

Bojanowski (2016)

- Skip-gram model



knight → The
knight → mighty
knight → Lancelot
knight → fought
knight → bravely.

- Model probability of a context word given a word

feature for word w: x_w
classifier for word c: v_c

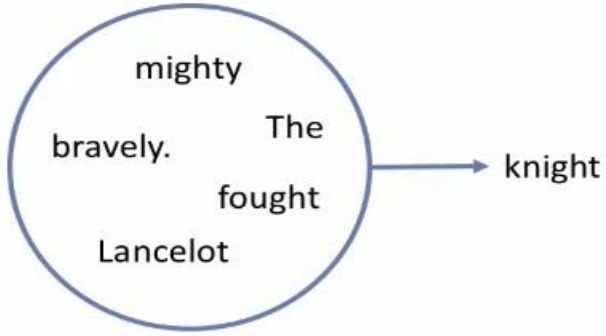
$$p(c|w) = \frac{e^{x_w^\top v_c}}{\sum_{k=1}^{K} e^{x_w^\top v_k}}$$

- Word vectors $x_w \in \mathbb{R}^d$

Word2Vec

Bojanowski (2016)

- CBOW model



- Model probability of a word given a context

feature for context $\mathcal{C}$: $h_{\mathcal{C}}$

classifier for word w: v_w

$$p(w|\mathcal{C}) = \frac{e^{h_{\mathcal{C}}^\top v_w}}{\sum_{k=1}^{K} e^{h_{\mathcal{C}}^\top v_k}}$$

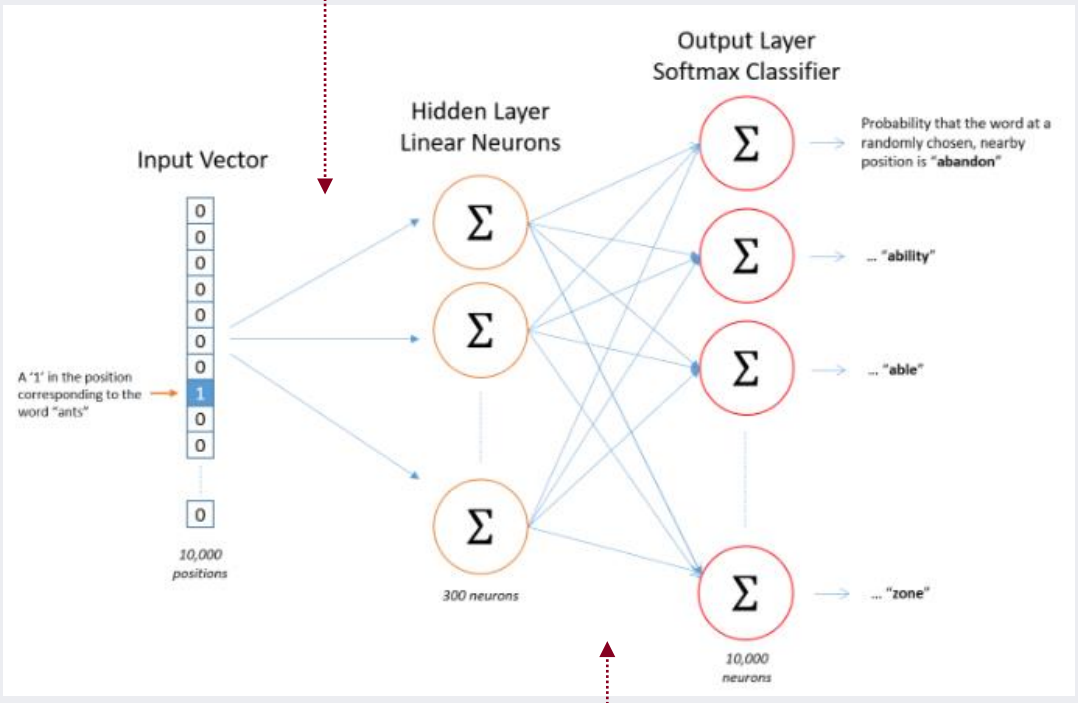
- Continuous Bag Of Words

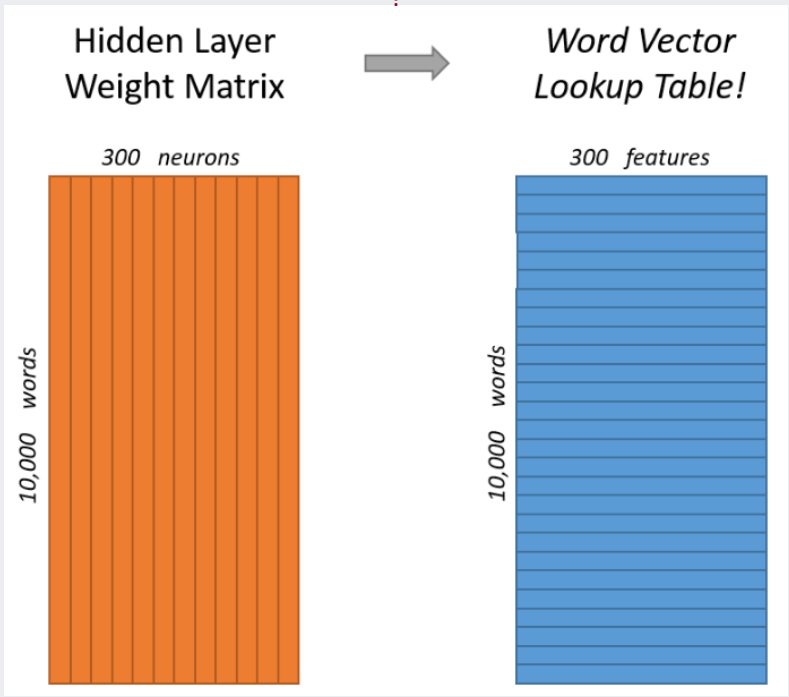
$$h_{\mathcal{C}} = \sum_{c \in \mathcal{C}} x_c$$

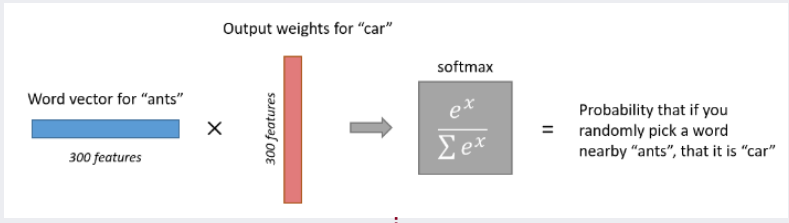
Word2Vec

McCormick

- Another architecture explanation







Word2Vec

- For simplicity, we use the following notation instead of $p(w_{t+j}|w_t)$

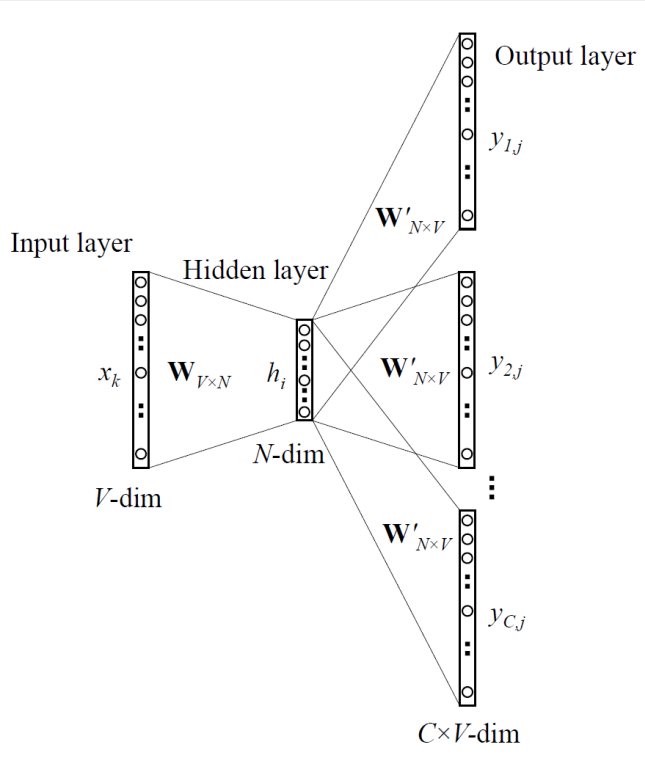
$$p(o|c) = \frac{exp(u_o^T v_c)}{\sum_{w=1}^{W} exp(u_w^T v_c)}$$

✓ where o is the outside (output) word id, c is the center word id, u and v are "outside" and "center" vectors of o and c

✓ Every word has two vectors!

- v is a row of matrix W
- u is a column of matrix W'

✓ Use W' = W^T in practice for efficient computation



Word2Vec

- Learning parameters with Gradient Ascent

$$J(\theta) = \frac{1}{T}\sum_{t=1}^{T}\sum_{-m\leq j\leq m, j\neq 0} \log p(w_{t+j}|w_t)$$

$$p(o|c) = \frac{exp(u_o^T v_c)}{\sum_{w=1}^{W} exp(u_w^T v_c)}$$

✓ Compute the gradient

$$\frac{\partial}{\partial v_c}\log p(o|c) = \frac{\partial}{\partial v_c}\log\frac{exp(u_o^T v_c)}{\sum_{w=1}^{W} exp(u_w^T v_c)}$$

$$= \frac{\partial}{\partial v_c}u_o^T v_c - \frac{\partial}{\partial v_c}\log\sum_{w=1}^{W} exp(u_w^T v_c)$$

A B

Word2Vec

- Learning parameters with Gradient Ascent
  - ✓ For chunk A

$$\frac{\partial}{\partial v_c} u_o^T v_c = u_o$$

  - ✓ For chunk B

$$-\frac{\partial}{\partial v_c} \log \sum_{w=1}^{W} exp(u_w^T v_c)$$

$$= -\frac{1}{\sum_{w=1}^{W} exp(u_w^T v_c)} \cdot \left(\sum_{w=1}^{W} exp(u_w^T v_c) \cdot u_w \right)$$

$$= -\sum_{w=1}^{W} \frac{exp(u_w^T v_c)}{\sum_{w=1}^{W} exp(u_w^T v_c)} \cdot u_w = \sum_{w=1}^{W} P(w|c) \cdot u_w$$

Word2Vec

- Learning parameters with Gradient Ascent

$$\frac{\partial}{\partial v_c} \log p(o|c) = u_o - \sum_{w=1}^{W} P(w|c) \cdot u_w$$

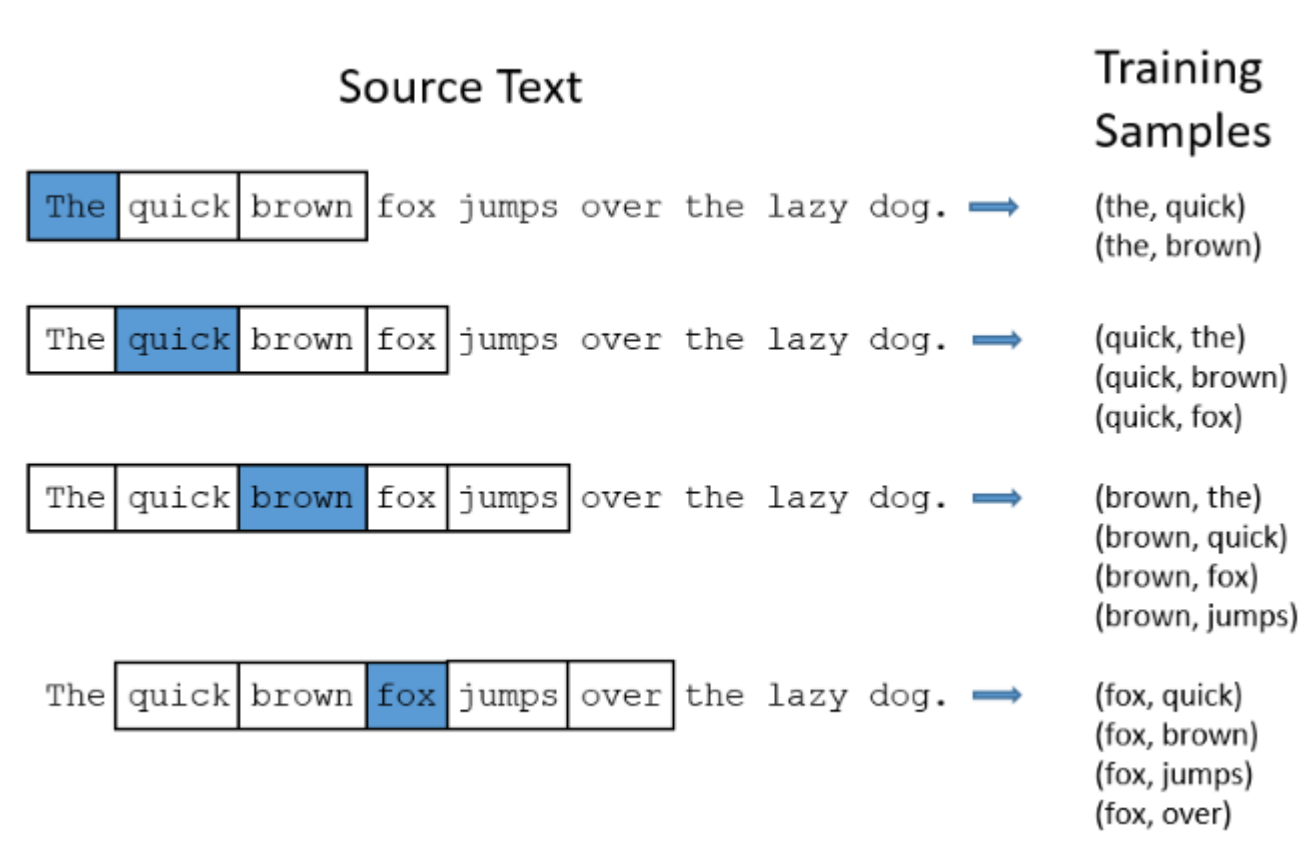
- Update the weight vector

$$v_c(t+1) = v_c(t) + \alpha\left(u_o - \sum_{w=1}^{W} P(w|c) \cdot u_w\right)$$

Word2Vec

McCormick

- Learning strategy
  - ✓ Do not use all nearby words, but one per each training



Word2Vec

- The number of weights to be trained: 2 x V x N (Huge network!)
  - ✓ Word pairs and phrases
    - Treating common word pairs or phrases as single "word"
  - ✓ Subsampling frequent words
    - To decrease the number of training examples

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}}$$

← Threshold (10^{-5})

← Term frequency in the corpus

Word2Vec

- The number of weights to be trained: 2 x V x N (Huge network!)
  - ✓ Negative sampling
    - Instead of updating the weights associated with all output words, update the weight of a few (5-20) words

$$J(\theta) = \frac{1}{T}\sum_{t=1}^{T}\sum_{-m\leq j\leq m, j\neq 0} \log p(w_{t+j}|w_t) = \frac{1}{T}\sum_{t=1}^{T} J_t(\theta)$$

$$J_t(\theta) = \log\sigma(u_o^T v_c) + \sum_{i=1}^{k} E_{i\sim P(w)}\left[\log\sigma(-u_i^T v_c)\right]$$

$$P(w_i) = \frac{f(w_i)^{3/4}}{\sum_{j=0}^{n}\left(f(w_i)^{3/4}\right)}$$

wevi: word embedding visual inspector

Everything you need to know about this tool - Source code

Control Panel

Config:

{"hidden_size":5,"random_state":1,"learning_rate":0.2}

Training data (context|target):

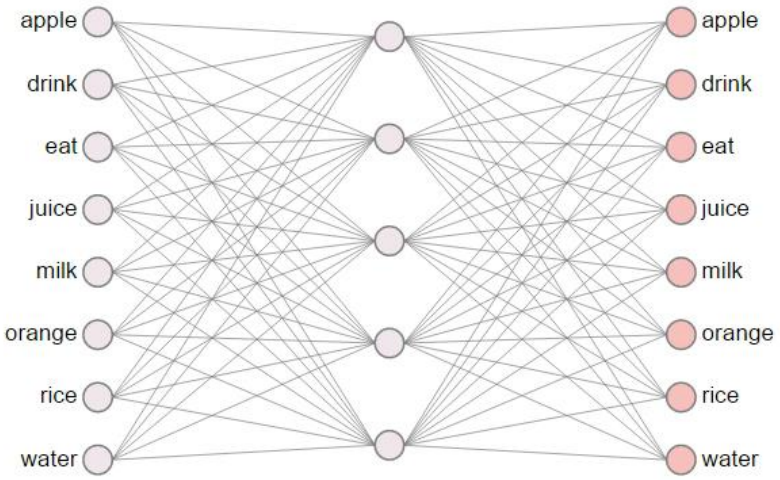
eat|apple,eat|orange,eat|rice,drink|juice,drink|milk,drink|water,orange|juice,apple|juice,rice|milk,milk|drink,water|drink,juice|drink

Presets: Fruit and juice

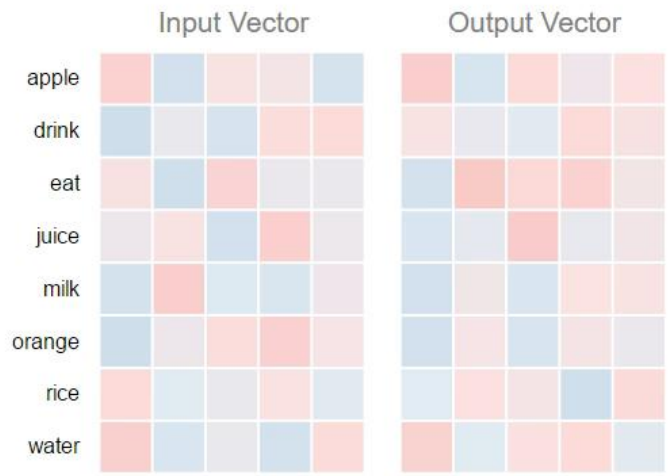
Update and Restart | Update Learning Rate

Next | 20 | 100 | 500 | PCA

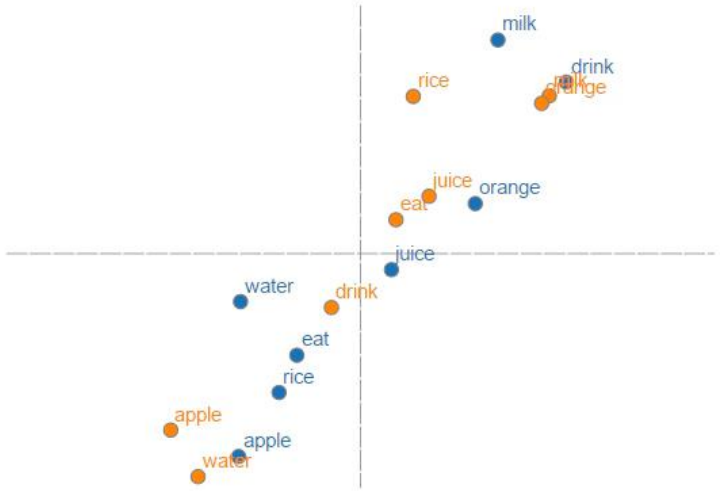
Neurons



Weight Matrices



Vectors



GloVe

Pennington et al. (2014)

- Limitations of Word2Vec
  - ✓ The network spends so much time to train some overwhelmingly used words
    - Ex: to learn a distribution for P(w|the)

Theatre or theater is a collaborative form of fine art that uses live performers to present the experience of a real or imagined event before a live audience in a specific place. The performers may communicate this experience to the audience through combinations of gesture, speech, song, music, and dance. Elements of art and stagecraft are used to enhance the physicality, presence and immediacy of the experience. The specific place of the performance is also named by the word "theatre" as derived from the Ancient Greek (thatron, "a place for viewing"), itself from (theomai, "to see", "to watch", "to observe"). Modern Western theatre comes from large measure from ancient Greek drama, from which it borrows technical terminology, classification into genres, and many of its themes, stock characters, and plot elements. Theatre artist Patrice Pavis denes theatricality, theatrical language, stage writing, and the specificity of theatre as synonymous expressions that differentiate theatre from the other performing arts, literature, and the arts in general. Theatre today, broadly denfied, includes performances of plays and musicals, ballets, operas and various other forms.

GloVe

Pennington et al. (2014)

- GloVe
  - ✓ Based on matrix factorization method
  - ✓ http://nlp.stanford.edu/projects/glove/
  - ✓ Notations

- $X \in \mathbb{R}^{V \times V}$ word co-occurrence matrix
- X_{ij} frequency of word i co-occurring with word j
- $X_i = \sum_k^V X_{ik}$ total number of occurrences of word i in corpus
- $P_{ij} = P(j|i) = \frac{X_{ij}}{X_i}$ a.k.a. probability of word j occurring within the context of word i
- $w \in \mathbb{R}^d$ a word embedding of dimension d
- $\tilde{w} \in \mathbb{R}^d$ a context word embedding of dimension d

GloVe

Pennington et al. (2014)

- Motivation

Prob. 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}
$\frac{P(k\|ice)}{P(k\|steam)}$	8.9	8.5×10^{-2}	1.36	0.96

- ✓ For words *k* related ice but not steam (solid), the ratio P_{ik}/P_{jk} is large
- ✓ For words *k* related steam ice but not ice (gas) the ratio P_{ik}/P_{jk} is small
- ✓ For words *k* that are either related to both ice and steam, or to neither, the ratio should be close to 1

GloVe

Pennington et al. (2014)

- Derivation

$$F(w_i, w_j, \tilde{w}_k) = \frac{P_{ik}}{P_{jk}}$$

F should encode information in the ratio $\frac{P_{ik}}{P_{jk}}$

$$F(w_i - w_j, \tilde{w}_k) = \frac{P_{ik}}{P_{jk}}$$

The argument of *F* should be scalar

$$F\left((w_i - w_j)^T \tilde{w}_k\right) = \frac{P_{ik}}{P_{jk}}$$

F should be unchanged by exchanging $w \to \tilde{w}$ and $X \to X^T$

$$F\left((w_i - w_j)^T \tilde{w}_k\right) = \frac{F\left(w_i^T \tilde{w}_k\right)}{F\left(w_j^T \tilde{w}_k\right)}$$

$$\Rightarrow F(w_i^T \tilde{w}_k) = P_{ik}$$

GloVe

Pennington et al. (2014)

- Derivation

$$F\left(w_i^T \tilde{w}_k - w_j^T \tilde{w}_k\right) = \frac{F\left(w_i^T \tilde{w}_k\right)}{F\left(w_j^T \tilde{w}_k\right)}$$

$$\exp\left(w_i^T \tilde{w}_k - w_j^T \tilde{w}_k\right) = \frac{\exp\left(w_i^T \tilde{w}_k\right)}{\exp\left(w_j^T \tilde{w}_k\right)}$$

$$w_i^T \tilde{w}_k = \log P_{ik} = \log X_{ik} - \log X_i$$

$$w_i^T \tilde{w}_k = \log P_{ik} = \log X_{ik} - \log X_i$$

$$w_i^T \tilde{w}_k = \log X_{ik} - b_i - \tilde{b}_k$$

$$w_i^T \tilde{w}_k + b_i + \tilde{b}_k = \log X_{ik}$$

GloVe

Pennington et al. (2014)

- Objective Function
  - ✓ A least squared objective function

$$J = \sum_{i,j=1}^{V} \left(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij} \right)^2$$

$$\Rightarrow J = \sum_{i,j=1}^{V} f\left(X_{ij}\right) \left(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij} \right)^2$$

where f has the following desiderata:

1. $f(0) = 0$
2. $f(x)$ should be non-decreasing so that rare co-occurrences are not overweighted.
3. $f(x)$ should be relatively small for large values of x, so that frequent co-occurrences are not overweighted.

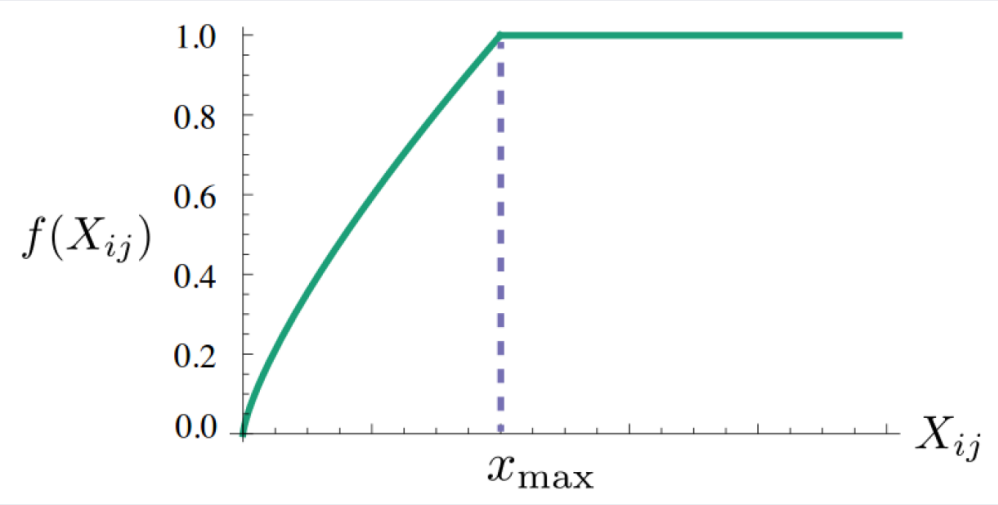
GloVe

Pennington et al. (2014)

- Objective Function
  - ✓ A least squared objective function

$$J = \sum_{i,j=1}^{V} f\left(X_{ij}\right)\left(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij}\right)^2$$

$$\text{where } f(x) = \begin{cases} \left(\frac{x}{x_{\max}}\right)^{\alpha} & \text{if } x < x_{\max} \\ 1 & \text{otherwise} \end{cases}$$



GloVe

Pennington et al. (2014)

- Results

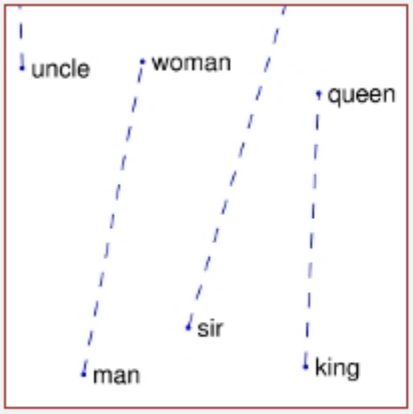
0. *frog*
1. frogs
2. toad
3. litoria
4. leptodactylidae
5. rana
6. lizard
7. eleutherodactylus


3. litoria

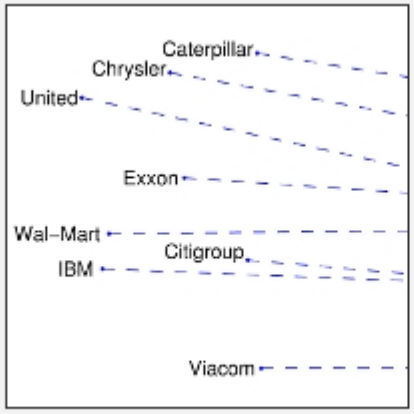

4. leptodactylidae


5. rana

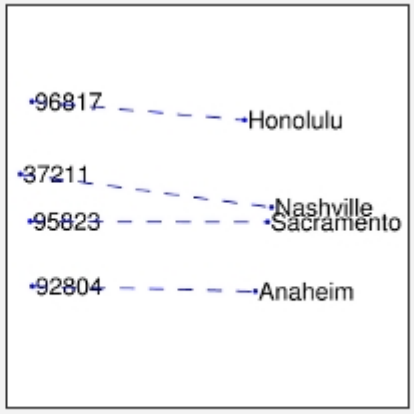

7. eleutherodactylus



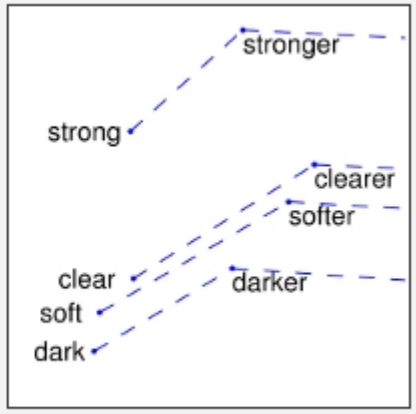
man - woman



company - ceo



city - zip code



comparative - superlative

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

FastText

Bojanowski et al. (2016)

- Limitations of NNLM, Word2Vec, and GloVe
  - ✓ Ignores the morphology or words by assigning a distinct vector to each word
  - ✓ Difficult to apply to morphologically rich languages with large vocabularies and many rare words (Turkish or Finnish)
- Goal
  - ✓ Learn representations for character n-grams
  - ✓ Represent words as the sum of n-gram vectors

FastText

Bojanowski et al. (2016)

- Revisit Negative Sampling in Word2Vec

$$J_t(\theta) = \log\sigma(u_o^T v_c) + \sum_{i=1}^{k} E_{i\sim P(w)} \left[\log\sigma(-u_i^T v_c)\right]$$

  - ✓ Score is just a dot product between the two embeddings

- Subword model

  - ✓ Define the set of n-grams appearing in w: $\mathcal{G}_w \subset \{1, ..., G\}$

$$score(w, c) = \sum_{g \in \mathcal{G}_w} \mathbf{z}_g^T \mathbf{v}_c$$

  - ✓ Represent a word by the sum of the vector representations of its n-grams

FastText

Bojanowski et al. (2016)

- Subword model
  - ✓ n-gram representation
    - Include the word w in the set of its n-grams
    - Keep all the n-grams of size 3, 4, 5, and 6
    - Different vectors are assigned to a word and a n-gram sharing the same sequence of characters

Word2Vec

parameter	...	...	...	...	...

FastText

p	...	...	...	...	...
...	...	...	...	...	...
r	...	...	...	...	...
pa	...	...	...	...	...
...	...	...	...	...	...
er	...	...	...	...	...
par	...	...	...	...	...
...	...	...	...	...	...
ameter	...	...	...	...	...
Avg.	...	...	...	...	...

Word Embedding Examples

- Word Embedding examples: English
  - ✓ Word lists that are close to a given word after embedding

FRANCE	JESUS	XBOX	REDDISH	SCRATCHED	MEGABITS
AUSTRIA	GOD	AMIGA	GREENISH	NAILED	OCTETS
BELGIUM	SATI	PLAYSTATION	BLUISH	SMASHED	MB/S
GERMANY	CHRIST	MSX	PINKISH	PUNCHED	BIT/S
ITALY	SATAN	IPOD	PURPLISH	POPPED	BAUD
GREECE	KALI	SEGA	BROWNISH	CRIMPED	CARATS
SWEDEN	INDRA	PSNUMBER	GREYISH	SCRAPED	KBIT/S
NORWAY	VISHNU	HD	GRAYISH	SCREWED	MEGAHERTZ
EUROPE	ANANDA	DREAMCAST	WHITISH	SECTIONED	MEGAPIXELS
HUNGARY	PARVATI	GEFORCE	SILVERY	SLASHED	GBIT/S
SWITZERLAND	GRACE	CAPCOM	YELLOWISH	RIPPED	AMPERES

Collobert et al. (2011)

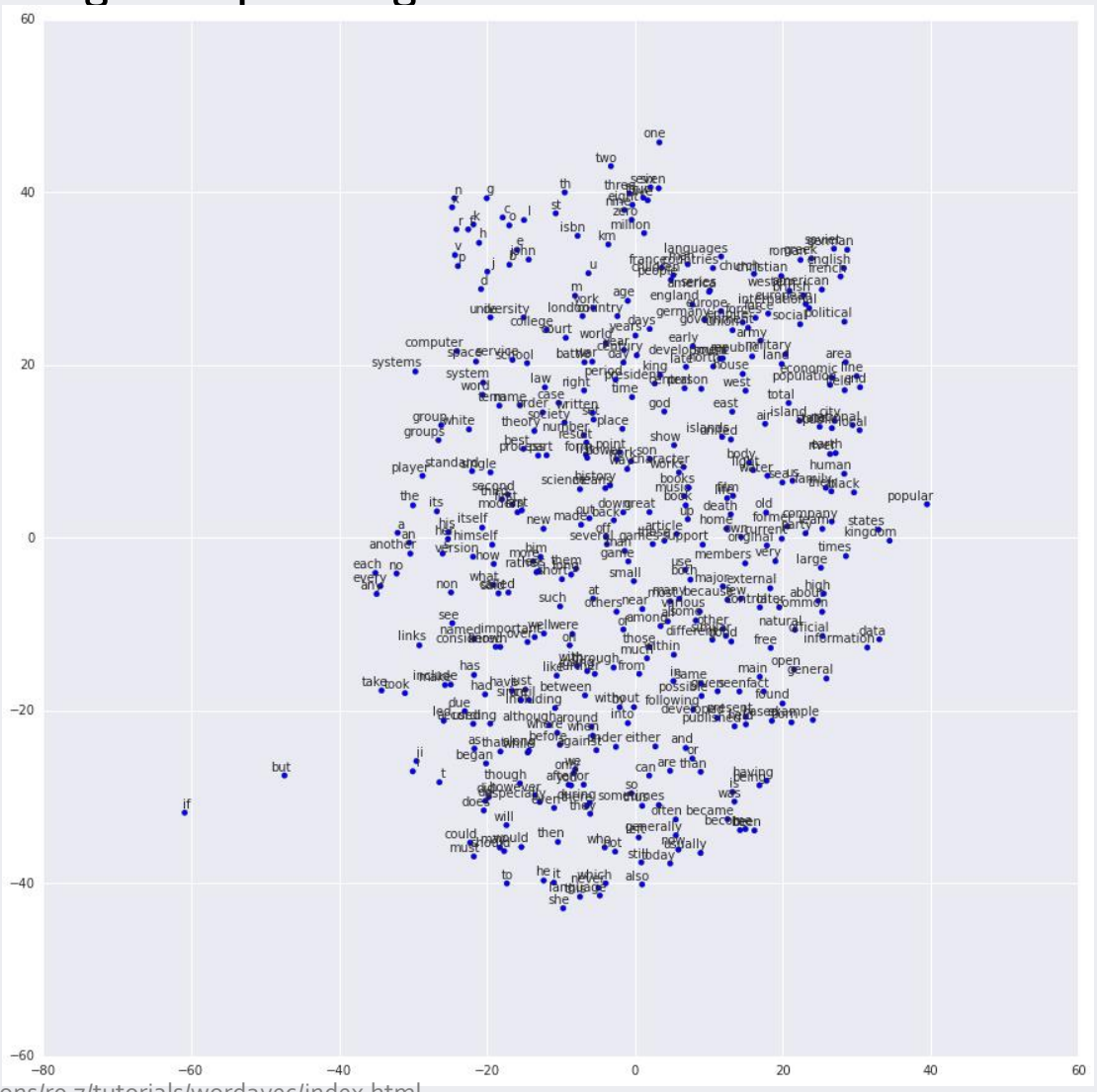
  - ✓ Relationship pairs in a word embedding

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

Mikolov et al. (2013)

Word Embedding Examples

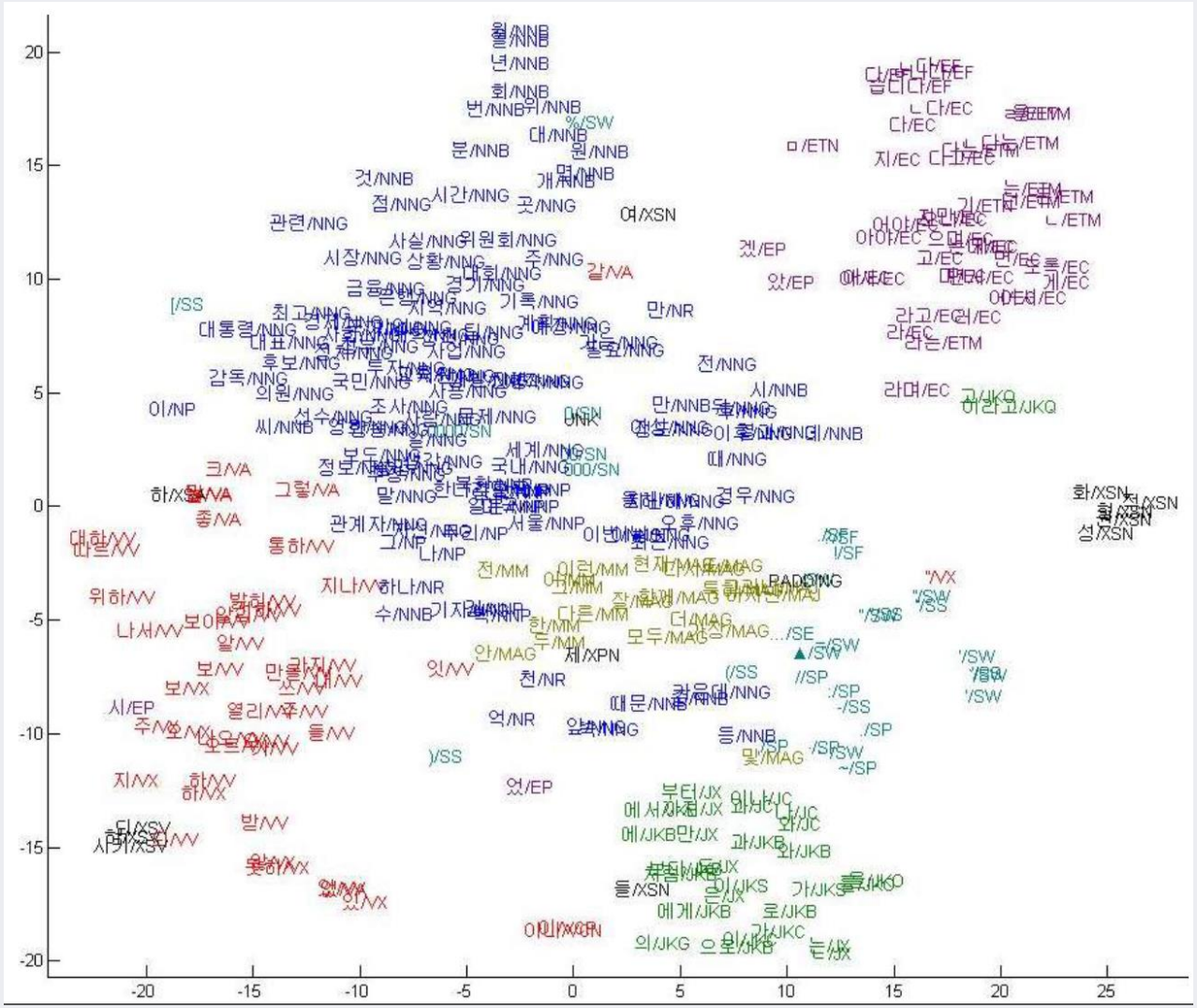
- Word Embedding examples: English



https://www.tensorflow.org/versions/ro.7/tutorials/word2vec/index.html

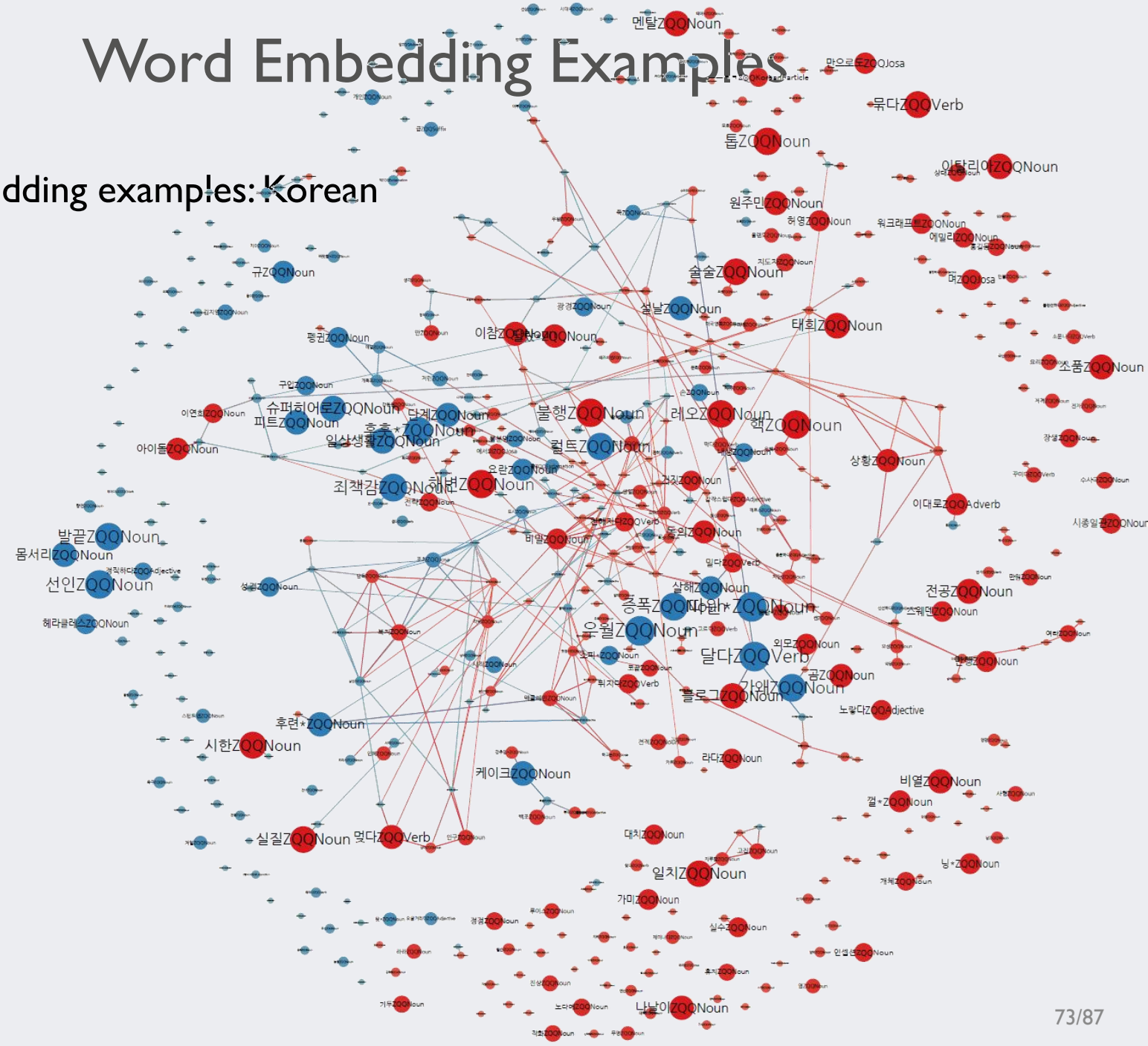
Word Embedding Examples

- Word Embedding examples: Korean



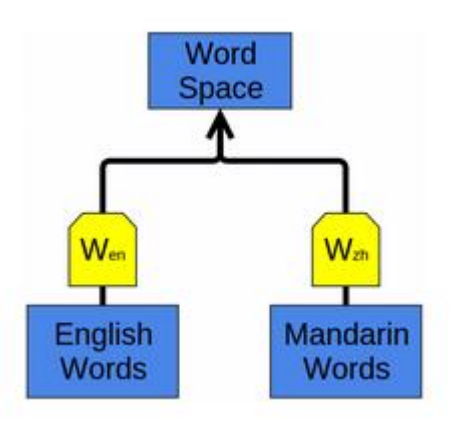
Word Embedding Examples

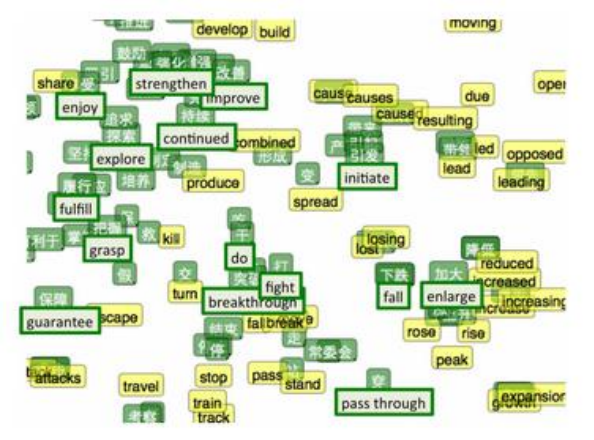
- Word Embedding examples: Korean



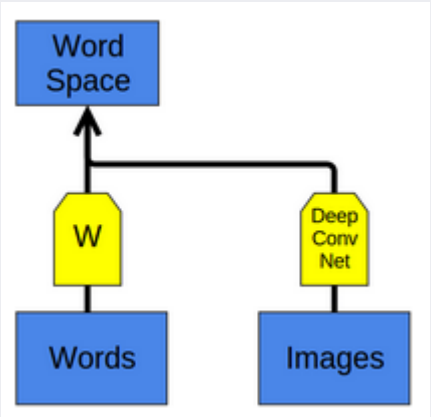
Word Embedding Examples

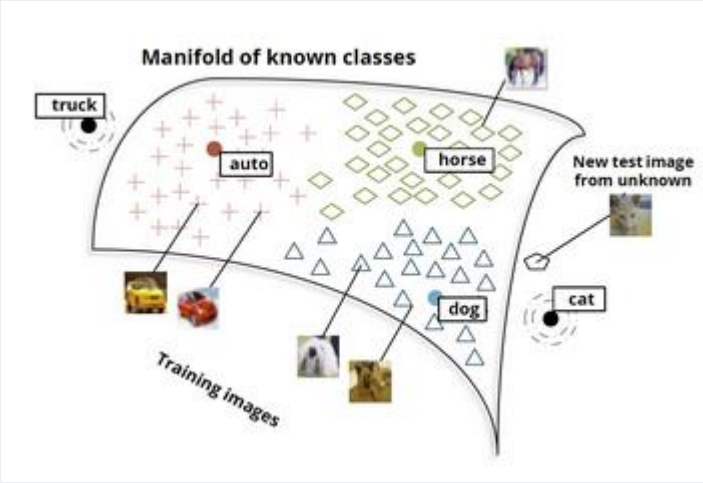
- Word Embedding with two different languages

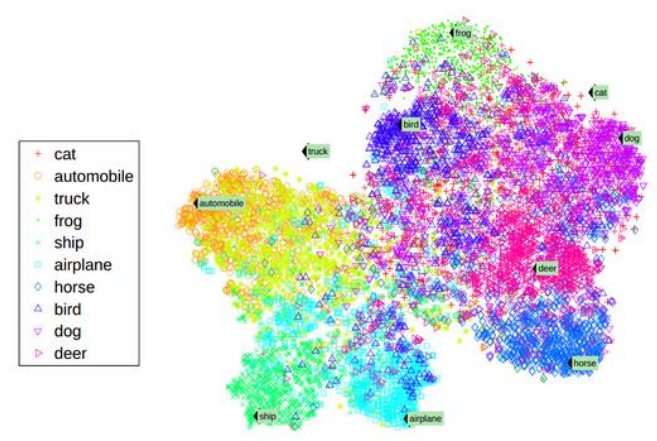




- Word Embedding with Images







AGENDA

01 Bag of Words

02 Word Weighting

03 N-Grams

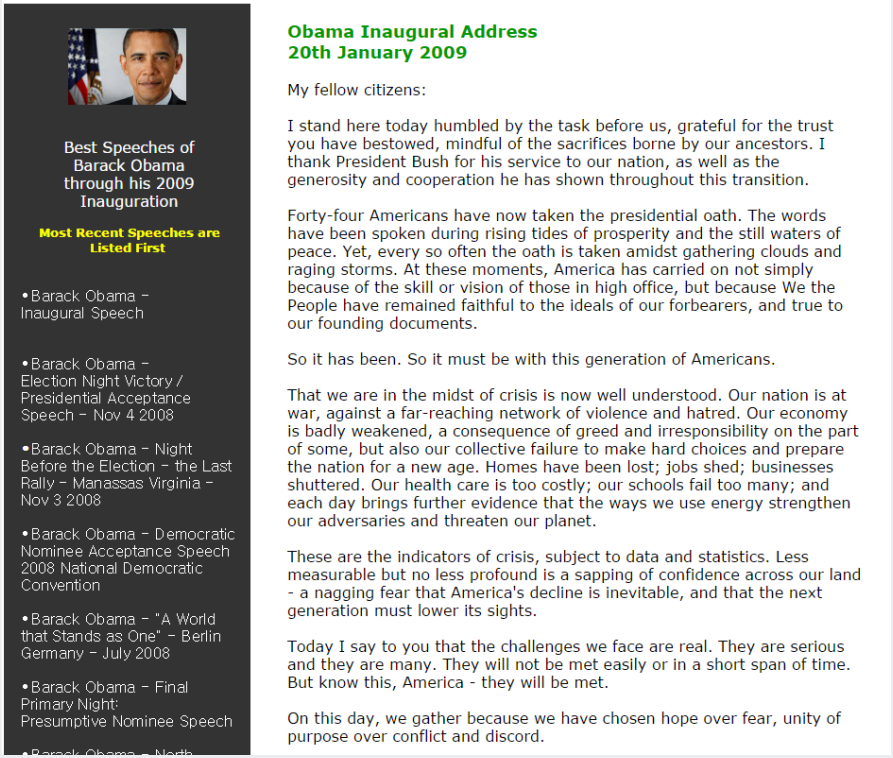
04 Distributed Representations

05 R Exercise

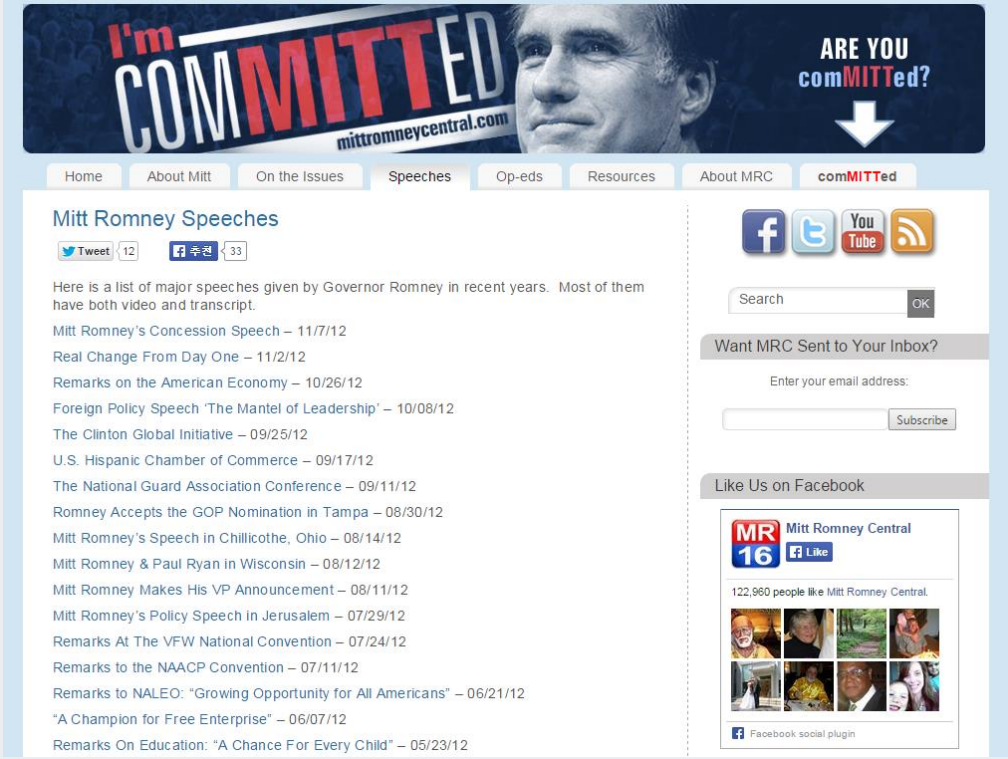
R Exercise

- 10 speeches delivered by Mr. Barack Obama & Mitt Romney

http://www.obamaspeeches.com/



http://mittromneycentral.com/speeches/



R Exercise

- 10 speeches delivered by Mr. Barack Obama & Mitt Romney

	Speaker	Speech
1	Obama	<U+FEFF>I stand here today humbled by the task before us, grateful for the trust you have bestowed, mindful of the sacrifices borne by our ancestors. I thank President Bush for his service to
2	Obama	<U+FEFF>If there is anyone out there who still doubts that America is a place where all things are possible; who still wonders if the dream of our founders is alive in our time; who still que
3	Obama	<U+FEFF>What a scene. What a crowd. Thank you for Virginia. (Crowd chants "yes we can.") Let me start by noting, Virginia that this is our last rally. This is the last rally of a campaign tha
4	Obama	<U+FEFF>Well, we are in the middle of a very close race right now in Texas, and we may not even know the final results until morning. We do know that Senator Clinton has won Rhode Island, and
5	Obama	<U+FEFF>To Chairman Dean and my great friend Dick Durbin; and to all my fellow citizens of this great nation; With profound gratitude and great humility, I accept your nomination for the pres
6	Obama	<U+FEFF>Tonight, after fifty-four hard-fought contests, our primary season has finally come to an end. Sixteen months have passed since we first stood together on the steps of the Old State C
7	Obama	<U+FEFF>You know, some were saying that North Carolina would be a game-changer in this election. But today, what North Carolina decided is that the only game that needs changing is the one in
8	Obama	<U+FEFF>I want to start by congratulating Senator Clinton on her victory tonight, and I want to thank the hundreds of thousands of Pennsylvanians who stood with our campaign today. There were
9	Obama	<U+FEFF>Good afternoon. I know I kept a lot of you guys busy this weekend with the comments I made last week. Some of you might even be a little bitter about that. As I said yesterday, I regr
10	Obama	<U+FEFF>"We the people, in order to form a more perfect union." Two hundred and twenty one years ago, in a hall that still stands across the street, a group of men gathered and, with these si
11	Rommey	<U+FEFF>Thank you all. It��s great to be back in Iowa. And don��t think that this is the last time you are going to see Paul Ryan and me, because you Iowans may well be the ones who decide
12	Rommey	<U+FEFF>Thank you. It��s good to be in the home state of the next Vice President of the United States. Next to Ann Romney, Paul Ryan is the best choice I ever made. Tonight, we enter the fi We are so very grateful to you and to people across the country, for all that you have given of yourselves to this campaign. This is not just about Paul and me<U+2013>it is about America, an
13	Rommey	<U+FEFF>I particularly appreciate the introduction from my good friend and tireless campaign companion, Gov. Bob McDonnell. He is showing what conservative leadership can do to build a strong This is the struggle that is now shaking the entire Middle East to its foundation. It is the struggle of millions and millions of people<U+2014>men and women, young and old, Muslims, Christi It is time to change course in the Middle East. That course should be organized around these bedrock principles: America must have confidence in our cause, clarity in our purpose and resolve I know many Americans are asking a different question: ��Why us?�� I know many Americans are asking whether our country today<U+2014>with our ailing economy, and our massive debt, and aft
14	Rommey	<U+FEFF>Thank you, Mr. President. I appreciate the kind words and your invitation here today. If there��s one thing we��ve learned this election season, it��s that a few words from Bill I just want to work. Work. That must be at the heart of our effort to help people build economies that can create jobs for people, young and old alike. Work builds self-esteem. It transforms
15	Rommey	<U+FEFF>Thank you, Hector, for that warm introduction. And congratulations to Cristina. I��m pleased to be your guest, and to speak as we begin National Hispanic Heritage Month. I��m also I love hearing stories like that. I am convinced that the Republican Party is the rightful home of Hispanic Americans. But my speech today isn��t about my political party. It��s about the
16	Rommey	<U+FEFF>Major General Vavala, thank you for your generous introduction. And thank you for your years of service as Chairman of the Board <U+2013> and for your decades of service to our nation Two weeks ago, I saw the Guard in action in Louisiana after it was hit by Hurricane Isaac. For many of the people of the Gulf <U+2013> who had just finished repairing their homes and getting
17	Rommey	<U+FEFF>Thank you for that kind introduction, Mayor Barkat, and thank you all for that warm welcome. It��s a pleasure and a privilege to be in Israel again. To step foot into Israel is to s So it is today, as Israel faces enemies who deny past crimes against the Jewish people and seek to commit new ones. When Iran��s leaders deny the Holocaust or speak of wiping this nation o We have seen the horrors of history. We will not stand by. We will not watch them play out again. It would be foolish not to take Iran��s leaders at their word. They are, after all, the pr
18	Rommey	<U+FEFF>Thank you. Commander Richard DeNoyer, I appreciate the introduction, and I��m proud to see a combat veteran from Massachusetts serving as National Commander of the VFW. Ladies Auxil This conduct is contemptible. It betrays our national interest. It compromises our men and women in the field. And it demands a full and prompt investigation by a special counsel, with expla Exactly who in the White House betrayed these secrets? Did a superior authorize it? These are things that Americans are entitled to know <U+2013> and they are entitled to know right now. If Why is flexibility with Russian leaders more important than transparency to the American people? President Obama had a moment of candor, however, just the other day. He said that the actions
19	Rommey	<U+FEFF>Thank you all very much. It��s good to be back in Iowa. So many friends here hold a special place in my heart. I��ve come here today to talk to you about an issue that affects the Imagine if the federal government was the sole legal supplier of cell phones. First, they��d still be under review, with hearings in Congress. When finally approved, the contract to make t We see the same bureaucracy and overhead in our anti-poverty programs. Last year, the federal government spent more than $600 billion on more than 100 different programs that aim to help the
20	Rommey	<U+FEFF>For the graduates, this moment marks a clear ending and a clear beginning. The task set before you four years ago is now completed in full. To the class of 2012: Well done, and congra Maybe the most confident step Jerry ever took was to open the doors of this school 41 years ago. He believed that Liberty might become one of the most respected Christian universities anywhe In this life, the commitments that come closest to forever are those of family. My Dad, George Romney, was a CEO, a governor, and a member of the President��s Cabinet. My wife Ann asked hi Ann and I feel the same way about our family. I have never once regretted missing a business opportunity so that I could be with my children and grandchildren. Among the things in life that Promotions often mark the high points in a career, and I hope I haven��t seen my last. But sometimes the high points come in unexpected ways. I was asked to help rescue the 2002 Olympics i The Olympics were not a logical choice, but it was one of the best and most fulfilling choices of my life. Opportunities for you to serve in meaningful ways may come at inconvenient times, b

R Exercise

- Preprocessing
  - ✓ Lower case transformation, remove punctuation/numbers/stopwords
  - ✓ Stemming

```
install.packages("RWeka", dependencies = TRUE)
install.packages("tm", dependencies = TRUE)

library(RWeka)
library(tm)

load("SpeechData.RData")

# Construct corpuses
# VectorSource specifies that the source is character vectors.
myCorpus <- Corpus(VectorSource(SpeechData$Speech))

# Preprocessing
# 1: to lower case
PreCorpus <- tm_map(myCorpus, content_transformer(tolower))

# 2: remove puntuations
PreCorpus <- tm_map(PreCorpus, content_transformer(removePunctuation))

# 3. remove numbers
PreCorpus <- tm_map(PreCorpus, content_transformer(removeNumbers))

# 4. remove stopwords (SMART stopwords list + obama + romney)
myStopwords <- c(stopwords("SMART"), "obama", "romney")

PreCorpus <- tm_map(PreCorpus, removeWords, myStopwords)

# 5. Stemming
stemCorpus <- tm_map(PreCorpus, stemDocument)
```

R Exercise

- Preprocessing
  - ✓ Lower case transformation, remove punctuation/numbers/stopwords
  - ✓ Stemming

```
> myCorpus[[3]]
<<PlainTextDocument (metadata: 7)>>
<U+FEFF>What a scene. What a crowd. Thank you for Virginia. (Crowd chants "yes we can.") Let me start by noting, Virginia that this is our last rally. This is
the last rally of a campaign that began nearly 2 years ago. We've gone to every corner of this country, from here in Northern Virginia to the rocky coasts of M
aine, to the open plains of Texas, to the open skies of Montana. I just want to say that whatever happens tomorrow, I have been deeply humbled by this journey.
You have welcomed Michelle and me and the girls into your homes. You have shared your stories of struggle, you have spoken of your dreams, along the way, talki
ng with all of you about your own lives. You have enriched my life, you have moved me again and again. You have inspired me. Sometimes when I have been down yo
u have lifted me up. You filled me with new hope for our future and you have reminded me about what makes America so special. In the places I have gone and the
people I have met, I have been struck again and again by the fundamental decency and generosity and dignity of men and women who work hard without complaint, t
o meet their responsibilities every day. I come away with an unyielding belief that if we only had a government as responsible as all of you, as compassionate
as the American people, that there is no obstacle that we can't overcome. There is no destiny that we cannot fulfill. Virginia, I have just one word for you, j
ust one word. Tomorrow. Tomorrow. After decades of broken politics in Washington, 8 years of failed policies from George Bush, twenty-one months of campaigning
, we are less then one day away from bringing about change in America. Tomorrow you can turn the page on policies that put greed and irresponsibility before ha
rd work and sacrifice. Tomorrow you can choose policies that invest in our middle class, create new jobs and grow this economy so that everybody has a chance t
o succeed. Not just the CEO but the secretary and the janitor; not just the factory owner but the men and women who work the factory floor. Tomorrow you can pu
```

```
> stemCorpus[[3]]
<<PlainTextDocument (metadata: 7)>>
   scene    crowd    virginia crowd chant      start  note virginia      ralli     ralli   campaign  began   year ago weve    corner   countri    northern virgi
nia   rocki coast  main   open plain  texa   open sky  montana        tomorrow   deepli humbl   journey   welcom michell     girl   home   share  stori  stru
ggl   spoken   dream    talk        live   enrich  life   move       inspir          lift    fill    hope   futur    remind    make america  special   place
peopl   met    struck      fundament decenc  generos  digniti  men  women  work hard  complaint  meet  respons  day      unyield belief       govern  respons
compassion   american peopl     obstacl    overcom    destini    fulfil virginia     word     word tomorrow tomorrow  decad  broken polit  washington  year  fa
il polici  georg bush twentyon month  campaign      day   bring  chang  america tomorrow   turn  page  polici  put greed  irrespons  hard work  sacrific tomorr
ow    choos polici  invest   middl class creat  job  grow  economi      chanc  succeed    ceo   secretari   janitor   factori owner   men  women  work  factori
floor tomorrow   put  end   polit   divid  nation   win  elect  put reason  reason  citi  town republican  democrat  ask   fear   time     hope tomorrow   defi
n moment  histori   give  countri  chang     start   virginia  start   manassa    chang begin  campaign   perfect   time    back  ive     wouldnt       though
t     bit   ill        campaign weve   lot   proud       tone    set    argu issu   engag  person attack weve  fierc  defend   weve   make      remind  suppor
t        black white hispan nativ american asian democrat  republican young   rich  poor gay  straight disabl   disabl      contribut    communic     year
afford   polit game   tactic  pit      make  afraid      afford  anymor   time    oppon  claim    real  fake part  virginia anymor     real  fake part  america
citi  town    proamerica        nation    proud    patriot    salut  flag  men  women  serv   battlefield    walk  life  polit parti   fought    bled   die
proud flag    serv red america  blue america   serv  unit state  america      campaign     call    serv  unit state  america   campaign     privileg  wit     am
```

R Exercise

- Construct a Term-Document Matrix

```
31  # Term-Document Matrix: Without Preprocessing
32  myTDM <- TermDocumentMatrix(myCorpus, control = list(minWordLength = 1))
33
34  # Term-Document Matrix: With Preprocessing
35  stemTDM <- TermDocumentMatrix(stemCorpus, control = list(minWordLength = 1))
```

✓ Without preprocessing

- 7,873 terms in 20 documents

```
> myTDM
<<TermDocumentMatrix (terms: 7873, documents: 20)>>
Non-/sparse entries: 18734/138726
Sparsity           : 88%
Maximal term length: 27
Weighting          : term frequency (tf)
```

✓ With preprocessing

- 3,491 terms in 20 documents

```
> stemTDM
<<TermDocumentMatrix (terms: 3491, documents: 20)>>
Non-/sparse entries: 11045/58775
Sparsity           : 84%
Maximal term length: 19
Weighting          : term frequency (tf)
```

Term reduction ratio:

4382/7873 = 55.66%

R Exercise

- Construct a Term-Document Matrix
  - ✓ Check the term-frequency

```
> as.matrix(stemTDM)[1:30,1:20]
                Docs
Terms            1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
  abandon        1 0 0 0 1 0 0 0 0  0  0  0  1  0  0  0  0  1  0  0
  abbottabad     0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  1  0  0  0  0
  abid           0 0 0 0 0 0 0 0 0  0  0  0  2  0  1  0  0  0  0  0
  abil           1 0 0 0 0 0 1 0 0  0  0  0  2  1  0  0  1  2  0  1
  abort          0 0 0 0 1 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0
  aboutit        0 0 0 0 0 0 0 0 0  0  0  1  0  0  0  0  0  0  0  0
  aboutmemori    0 0 0 0 0 0 0 0 0  1  0  0  0  0  0  0  0  0  0  0
  abroad         0 0 0 0 1 0 0 0 0  0  0  0  0  0  0  0  1  1  0  0
  abrupt         0 0 0 0 0 0 0 0 0  0  0  0  1  0  0  0  0  0  0  0
  absenc         0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  1  0  0  0
  absent         0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  1
  absolut        0 0 0 0 0 1 0 0 0  1  0  0  0  0  1  0  0  1  0  1
  absorb         0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  1  0
  abund          0 0 0 0 0 0 0 0 0  0  0  0  0  1  0  0  0  0  0  0
  abus           0 0 0 0 0 0 0 0 0  0  1  0  0  0  0  0  0  0  0  0
  accept         1 0 0 0 1 0 0 0 0  1  0  0  0  0  1  0  0  0  0  1
  access         0 0 0 0 1 0 0 0 0  1  1  0  0  2  0  0  0  1  0  0
  accompani      0 0 0 0 1 0 0 0 0  0  0  0  1  0  0  0  0  0  0  0
  accomplish     0 0 0 0 0 1 1 0 0  0  0  2  1  0  0  0  0  1  0  1
  account        1 0 0 0 2 0 0 0 0  1  1  0  1  1  0  0  0  3  1  0
  accus          0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  1  0
  achiev         1 4 0 0 1 0 1 0 0  3  2  5  1  1  3  0  4  1  0  3
  acknowledg     0 0 0 0 0 0 0 0 1  1  0  0  0  0  0  0  0  0  0  0
  acquir         0 0 0 0 0 0 0 0 0  0  0  0  1  0  0  0  0  0  0  0
  acrosstheboard 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  1  0  0
  act            1 0 2 0 0 0 0 1 0  0  1  5  2  0  0  3  0  0  1  0
  action         1 0 0 0 0 0 0 0 0  3  0  0  0  0  1  2  0  2  0  1
  actionsnot     0 0 0 0 0 0 0 0 0  0  0  0  1  0  0  0  0  0  0  0
  activeduti     0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  1  0  0  0  0
  activelyto     0 0 0 0 0 0 0 0 0  0  0  0  1  0  0  0  0  0  0  0
```

R Exercise

- TF Variants
  - ✓ Natural, Logarithm, Augmented

```
40  # TF-IDF Variants
41  # TF 1: Natural
42  natural_tf <- as.matrix(stemTDM)
43
44  # TF 2: Logarithm
45  log_tf <- log(1+natural_tf)
46
47  # TF 3: augmented
48  max_tf <- apply(natural_tf, 2, max)
49  augmented_tf <- 0.5+0.5*t(t(natural_tf)/max_tf)
```

```
> natural_tf[1:20,1:5]
              Docs
Terms          1 2 3 4 5
  abandon      1 0 0 0 1
  abbottabad   0 0 0 0 0
  abid         0 0 0 0 0
  abil         1 0 0 0 0
  abort        0 0 0 0 1
  aboutit      0 0 0 0 0
  aboutmemori  0 0 0 0 0
  abroad       0 0 0 0 1
  abrupt       0 0 0 0 0
  absenc       0 0 0 0 0
  absent       0 0 0 0 0
  absolut      0 0 0 0 0
  absorb       0 0 0 0 0
  abund        0 0 0 0 0
  abus         0 0 0 0 0
  accept       1 0 0 0 1
  access       0 0 0 0 1
  accompani    0 0 0 0 1
  accomplish   0 0 0 0 0
  account      1 0 0 0 2
```

```
> log_tf[1:20,1:5]
              Docs
Terms                1 2 3 4         5
  abandon      0.6931472 0 0 0 0.6931472
  abbottabad   0.0000000 0 0 0 0.0000000
  abid         0.0000000 0 0 0 0.0000000
  abil         0.6931472 0 0 0 0.0000000
  abort        0.0000000 0 0 0 0.6931472
  aboutit      0.0000000 0 0 0 0.0000000
  aboutmemori  0.0000000 0 0 0 0.0000000
  abroad       0.0000000 0 0 0 0.6931472
  abrupt       0.0000000 0 0 0 0.0000000
  absenc       0.0000000 0 0 0 0.0000000
  absent       0.0000000 0 0 0 0.0000000
  absolut      0.0000000 0 0 0 0.0000000
  absorb       0.0000000 0 0 0 0.0000000
  abund        0.0000000 0 0 0 0.0000000
  abus         0.0000000 0 0 0 0.0000000
  accept       0.6931472 0 0 0 0.6931472
  access       0.0000000 0 0 0 0.6931472
  accompani    0.0000000 0 0 0 0.6931472
  accomplish   0.0000000 0 0 0 0.0000000
  account      0.6931472 0 0 0 1.0986123
```

```
> augmented_tf[1:20,1:5]
              Docs
Terms                1   2   3   4         5
  abandon      0.5333333 0.5 0.5 0.5 0.515625
  abbottabad   0.5000000 0.5 0.5 0.5 0.500000
  abid         0.5000000 0.5 0.5 0.5 0.500000
  abil         0.5333333 0.5 0.5 0.5 0.500000
  abort        0.5000000 0.5 0.5 0.5 0.515625
  aboutit      0.5000000 0.5 0.5 0.5 0.500000
  aboutmemori  0.5000000 0.5 0.5 0.5 0.500000
  abroad       0.5000000 0.5 0.5 0.5 0.515625
  abrupt       0.5000000 0.5 0.5 0.5 0.500000
  absenc       0.5000000 0.5 0.5 0.5 0.500000
  absent       0.5000000 0.5 0.5 0.5 0.500000
  absolut      0.5000000 0.5 0.5 0.5 0.500000
  absorb       0.5000000 0.5 0.5 0.5 0.500000
  abund        0.5000000 0.5 0.5 0.5 0.500000
  abus         0.5000000 0.5 0.5 0.5 0.500000
  accept       0.5333333 0.5 0.5 0.5 0.515625
  access       0.5000000 0.5 0.5 0.5 0.515625
  accompani    0.5000000 0.5 0.5 0.5 0.515625
  accomplish   0.5000000 0.5 0.5 0.5 0.500000
  account      0.5333333 0.5 0.5 0.5 0.531250
```

R Exercise

- IDF Vector

```
> idf_vec[1:20]
    abandon  abbottabad        abid        abil       abort     aboutit aboutmemori      abroad      abrupt      absenc      absent      absolut      absorb
  1.6094379   2.9957323   2.3025851   1.0498221   2.9957323   2.9957323   2.9957323   1.8971200   2.9957323   2.9957323   2.9957323   1.3862944   2.9957323
      abund        abus      accept      access   accompani  accomplish     account
  2.9957323   2.9957323   1.3862944   1.3862944   2.3025851   1.2039728   0.9162907
```

- TF-IDF Variants
  - ✓ Natural TF * IDF vs. Log TF * IDF (cosine normalized)

```
> TFIDF_1[1:20,1:10]
             Docs
Terms               1 2 3 4         5         6         7 8 9        10
  abandon   1.6094379 0 0 0 1.609438 0.000000 0.000000 0 0 0.0000000
  abbottabad 0.0000000 0 0 0 0.000000 0.000000 0.000000 0 0 0.0000000
  abid      0.0000000 0 0 0 0.000000 0.000000 0.000000 0 0 0.0000000
  abil      1.0498221 0 0 0 0.000000 0.000000 1.049822 0 0 0.0000000
  abort     0.0000000 0 0 0 2.995732 0.000000 0.000000 0 0 0.0000000
  aboutit   0.0000000 0 0 0 0.000000 0.000000 0.000000 0 0 0.0000000
  aboutmemori 0.0000000 0 0 0 0.000000 0.000000 0.000000 0 0 2.9957323
  abroad    0.0000000 0 0 0 1.897120 0.000000 0.000000 0 0 0.0000000
  abrupt    0.0000000 0 0 0 0.000000 0.000000 0.000000 0 0 0.0000000
  absenc    0.0000000 0 0 0 0.000000 0.000000 0.000000 0 0 0.0000000
  absent    0.0000000 0 0 0 0.000000 0.000000 0.000000 0 0 0.0000000
  absolut   0.0000000 0 0 0 0.000000 1.386294 0.000000 0 0 1.3862944
  absorb    0.0000000 0 0 0 0.000000 0.000000 0.000000 0 0 0.0000000
  abund     0.0000000 0 0 0 0.000000 0.000000 0.000000 0 0 0.0000000
  abus      0.0000000 0 0 0 0.000000 0.000000 0.000000 0 0 0.0000000
  accept    1.3862944 0 0 0 1.386294 0.000000 0.000000 0 0 1.3862944
  access    0.0000000 0 0 0 1.386294 0.000000 0.000000 0 0 1.3862944
  accompani 0.0000000 0 0 0 2.302585 0.000000 0.000000 0 0 0.0000000
  accomplish 0.0000000 0 0 0 0.000000 1.203973 1.203973 0 0 0.0000000
  account   0.9162907 0 0 0 1.832581 0.000000 0.000000 0 0 0.9162907
```

```
> TFIDF_2[1:20,1:10]
             Docs
                    1 2 3 4         5         6         7 8 9          10
  abandon   0.03323211 0 0 0 0.02963354 0.00000000 0.00000000 0 0 0.00000000
  abbottabad 0.00000000 0 0 0 0.00000000 0.00000000 0.00000000 0 0 0.00000000
  abid      0.00000000 0 0 0 0.00000000 0.00000000 0.00000000 0 0 0.00000000
  abil      0.02167701 0 0 0 0.00000000 0.00000000 0.03004081 0 0 0.00000000
  abort     0.00000000 0 0 0 0.05515848 0.00000000 0.00000000 0 0 0.00000000
  aboutit   0.00000000 0 0 0 0.00000000 0.00000000 0.00000000 0 0 0.00000000
  aboutmemori 0.00000000 0 0 0 0.00000000 0.00000000 0.00000000 0 0 0.03873698
  abroad    0.00000000 0 0 0 0.03493045 0.00000000 0.00000000 0 0 0.00000000
  abrupt    0.00000000 0 0 0 0.00000000 0.00000000 0.00000000 0 0 0.00000000
  absenc    0.00000000 0 0 0 0.00000000 0.00000000 0.00000000 0 0 0.00000000
  absent    0.00000000 0 0 0 0.00000000 0.00000000 0.00000000 0 0 0.00000000
  absolut   0.00000000 0 0 0 0.00000000 0.03851873 0.00000000 0 0 0.01792579
  absorb    0.00000000 0 0 0 0.00000000 0.00000000 0.00000000 0 0 0.00000000
  abund     0.00000000 0 0 0 0.00000000 0.00000000 0.00000000 0 0 0.00000000
  abus      0.00000000 0 0 0 0.00000000 0.00000000 0.00000000 0 0 0.00000000
  accept    0.02862458 0 0 0 0.02552494 0.00000000 0.00000000 0 0 0.01792579
  access    0.00000000 0 0 0 0.02552494 0.00000000 0.00000000 0 0 0.01792579
  accompani 0.00000000 0 0 0 0.04239601 0.00000000 0.00000000 0 0 0.00000000
  accomplish 0.00000000 0 0 0 0.00000000 0.03345285 0.03445186 0 0 0.00000000
  account   0.01891982 0 0 0 0.02674001 0.00000000 0.00000000 0 0 0.01184830
```

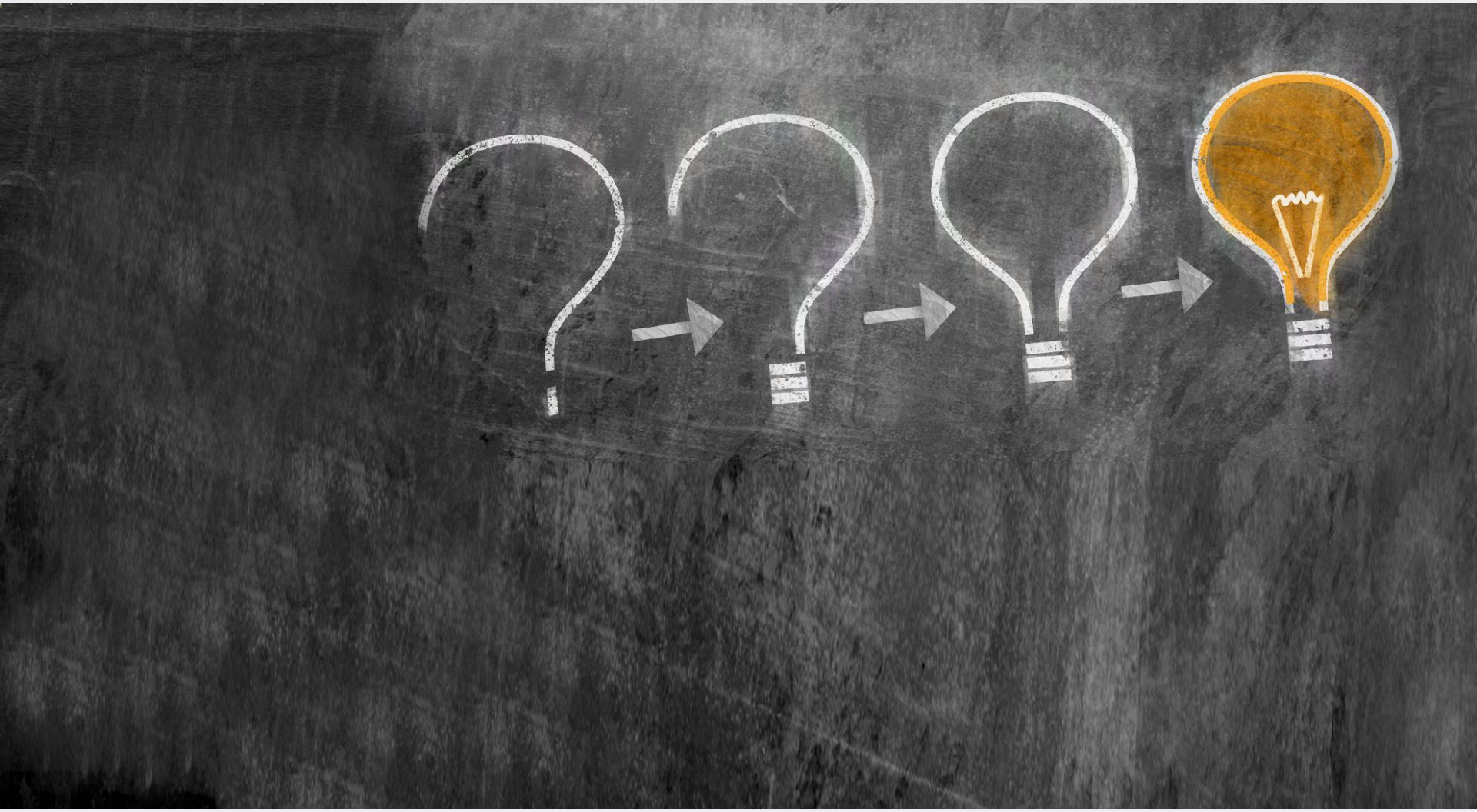
R Exercise

- N-Gram TF matrix (N=2,3)

```
69  # Construct N-gram matrix
70  tmpTokenizer <- function(x) NGramTokenizer(x, Weka_control(min = 2, max = 3))
71  NgramTDM <- TermDocumentMatrix(stemCorpus, control = list(tokenize = tmpTokenizer))
72  inspect(NgramTDM[30000:30100,1:20])
```

```
> inspect(NgramTDM[30000:30100,1:20])
<<TermDocumentMatrix (terms: 101, documents: 20)>>
Non-/sparse entries: 111/1909
Sparsity           : 95%
Maximal term length: 26
Weighting          : term frequency (tf)

                        Docs
Terms                    1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
  shift divers visa      0 0 0 0 0 0 0 0 0  0  0  0  0  0  1  0  0  0  0  0
  shift full             0 0 0 1 0 1 1 0 0  0  0  0  0  0  0  0  0  0  0  0
  shift full day         0 0 0 1 0 1 1 0 0  0  0  0  0  0  0  0  0  0  0  0
  shift mom              0 0 0 0 1 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0
  shift mom rais         0 0 0 0 1 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0
  shift servic           0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  1  0
  shift servic program   0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  1  0
  shini object           0 0 0 0 0 0 0 0 0  0  1  0  0  0  0  0  0  0  0  0
  shini object elect     0 0 0 0 0 0 0 0 0  0  1  0  0  0  0  0  0  0  0  0
  ship china             0 0 0 0 1 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0
  ship china choke       0 0 0 0 1 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0
  ship job               0 0 0 0 1 0 1 0 0  0  0  0  0  0  0  0  0  0  0  0
  ship job oversea       0 0 0 0 1 0 1 0 0  0  0  0  0  0  0  0  0  0  0  0
  ship oversea           0 0 0 0 0 0 0 0 0  2  0  0  0  0  0  0  0  0  0  0
  ship oversea pension   0 0 0 0 0 0 0 0 0  1  0  0  0  0  0  0  0  0  0  0
  ship oversea profit    0 0 0 0 0 0 0 0 0  1  0  0  0  0  0  0  0  0  0  0
  ship unit              0 0 0 0 0 0 0 0 0  0  0  0  0  0  1  0  0  0  0  0
  ship unit state        0 0 0 0 0 0 0 0 0  0  0  0  0  0  1  0  0  0  0  0
  ship year              0 0 0 0 0 0 0 0 0  0  0  0  1  0  0  0  0  0  0  0
  ship year includ       0 0 0 0 0 0 0 0 0  0  0  0  1  0  0  0  0  0  0  0
  shock flame            0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  1  0  0  0  0
  shock flame smoke      0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  1  0  0  0  0
  shock ignor            0 0 0 0 0 0 0 0 0  1  0  0  0  0  0  0  0  0  0  0
  shock ignor struggl    0 0 0 0 0 0 0 0 0  1  0  0  0  0  0  0  0  0  0  0
```



References

Research Papers

- Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A Neural Probabilistic Language Model. Journal of Machine Learning Research, 3, 1137-1155.
- Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2016). Enriching word vectors with subword information. arXiv preprint arXiv:1607.04606.
- Collobert et al. (2011). Natural Language Processing (Almost) from Scratch. Journal of Machine Learning Research 12: 2493-2537.
- Furnkranz, J. (1998). A Study using N-gram Features for Text Categorization. Austrian Research Institute for Artificial Intelligence Technical Report OEFAI-TR-98-30 Schottengasse.
- 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).
- Paltoglou, G. and Thelwall, M. (2010). A Study of Information Retrieval Weighting Schemes for Sentiment Analysis. In Proceddings of the 48th Annual Meeting of the Association for Computational Linguistics: 1386-1395.
- Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global Vectors for Word Representation. In EMNLP (Vol. 14, pp. 1532-1543).
- Roelleke, T. (2013). Information Retrieval Models: Foundations and Relationships. Morgan & Claypool Publishers.
- Rong, X. (2014). word2vec parameter learning explained. arXiv preprint arXiv:1411.2738.

References

Other materials

- Bojanowski, P. (2016). fastest: A library for efficient text classification and word representation. https://www.youtube.com/watch?v=CHcExDsDeHU
- CS224d-L2: Lecture note on "Simple Word Vector representations: word2vec, GloVe", http://cs224d.stanford.edu/lectures/CS224d-Lecture2.pdf
- CS224d-L3: Lecture note on "Advanced word vector representations: language models, softmax, single layer networks", http://cs224d.stanford.edu/lectures/CS224d-Lecture3.pdf
- Images in the title page: http://nlp.stanford.edu/projects/glove/
- Kauchak, D. (2009). TF-IDF. http://www.cs.pomona.edu/~dkauchak/classes/f09/cs160-f09/lectures/lecture5-tfidf.pdf
- Kim, Y., Hernite, Y., Sontag, D., and Rush, A. (2016). Character-Aware Neural Language Models, AAAI 2016 presentation slides, http://www.people.fas.harvard.edu/~yoonkim/data/aaai_2016_slides.pdf
- McCormick, C. Word2Vec Tutorial Part I & 2, http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/
- Nayak, P. and Raghavan, P. (2014). Lecture 6: Scoring, Term Weighting and the Vector Space Model. http://web.stanford.edu/class/cs276/handouts/lecture6-tfidf-handout-1-per.pdf
- Word2Vec online example: http://bit.ly/wevi-online
- 이재천, 김수경, 홍성연. (2015). Data Mining을 이용한 고려대 강의평가 분석: Klue 사이트 강의평가를 기준으로. 2015 캡스톤디자인 II Term Project.