CNRS - INPT - UPS - UT1 - UTM

The computation of word meaning

Distributional similarity and its applications

Tim Van de Cruys
IRIT Toulouse

28 January 2013

Distributional similarity

- Semantic similarity is based on the DISTRIBUTIONAL HYPOTHESIS [Harris 1954]

  - Take a word and its contexts:
    - freshly baked *etteugab*
    - crunchy *etteugab*
    - stale *etteugab*
    - buttered *etteugab*

- By looking at a word's context, one can infer its meaning

Distributional similarity

- Semantic similarity is based on the DISTRIBUTIONAL HYPOTHESIS [Harris 1954]

  - Take a word and its contexts:
    - freshly baked *etteugab*
    - crunchy *etteugab*
    - stale *etteugab*
    - buttered *etteugab*

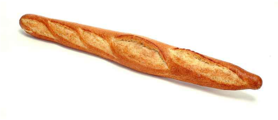
    $\Rightarrow$ **food**

- By looking at a word's context, one can infer its meaning

Distributional similarity

- Semantic similarity is based on the DISTRIBUTIONAL HYPOTHESIS [Harris 1954]

  - Take a word and its contexts:
    - freshly baked *etteugab*
    - crunchy *etteugab*
    - stale *etteugab*
    - buttered *etteugab*



- By looking at a word's context, one can infer its meaning

Matrix

- captures co-occurrence frequencies of two entities

	red	tasty	fast	second-hand
raspberry	2	1	0	0
strawberry	2	2	0	0
car	1	0	1	2
truck	1	0	1	1

Matrix

- captures co-occurrence frequencies of two entities

	red	tasty	fast	second-hand
raspberry	7	9	0	0
strawberry	12	6	0	0
car	7	0	8	4
truck	2	0	3	4

Matrix

- captures co-occurrence frequencies of two entities

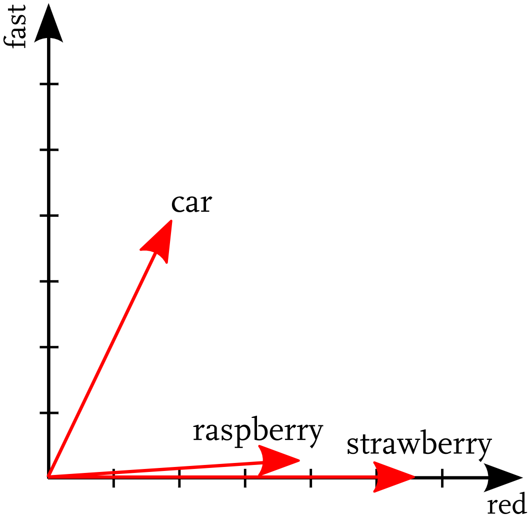
	red	tasty	fast	second-hand
raspberry	56	98	0	0
strawberry	44	34	0	0
car	23	0	31	39
truck	4	0	18	29

Matrix

- captures co-occurrence frequencies of two entities

	red	tasty	fast	second-hand
raspberry	728	592	1	0
strawberry	1035	437	0	2
car	392	0	487	370
truck	104	0	393	293

Vector space model



Term-document matrix

	doc1	doc2	doc3	doc4
term1				
term2				
term3				
term4				

Word-context matrix

	context1	context2	context3	context4
word1				
word2				
word3				
word4				

- Different notions of context
    - window around word
    - dependency-based features (extracted from parse trees)

He drove his second-hand **car** a couple of miles down the road .

Word-context matrix

	context1	context2	context3	context4
word1				
word2				
word3				
word4				

- Different notions of context
  - window around word (2 words)
  - dependency-based features (extracted from parse trees)

He drove [his second-hand **car** a couple] of miles down the road .

Word-context matrix

	context1	context2	context3	context4
word1				
word2				
word3				
word4				

- Different notions of context
  - window around word (sentence)
  - dependency-based features (extracted from parse trees)

[He drove his second-hand **car** a couple of miles down the road .]

Word-context matrix

	context1	context2	context3	context4
word1				
word2				
word3				
word4				

- Different notions of context
    - window around word
    - dependency-based features (extracted from parse trees)

obj

mod

He drove his second-hand **car** a couple of miles down the road .

Different kinds of semantic similarity

- **'tight', synonym-like similarity**: (near-)synonymous or (co-)hyponymous
- **loosely related, topical similarity**: more loose relationships, such as association and meronymy

Different kinds of semantic similarity

- **'tight', synonym-like similarity**: (near-)synonymous or (co-)hyponymous
- **loosely related, topical similarity**: more loose relationships, such as association and meronymy

Example

- **doctor**: *nurse*, *GP*, *physician*, *practitioner*, *midwife*, *dentist*, *surgeon*
- **doctor**: *medication*, *disease*, *surgery*, *hospital*, *patient*, *clinic*, *nurse*, *treatment*, *illness*

Relation context – similarity

- Different context leads to different kind of similarity
- Syntax, small window $\leftrightarrow$ large window, documents
- The former models induce **tight, synonymous similarity**
- The latter models induce **topical relatedness**

Step 1: linguistic preprocessing

French actor Gérard Depardieu wants to be a Belgian.

1. tokenization
   - `French actor Gérard_Depardieu wants to be a Belgian .`
2. normalization (lemmatization)
   - `French actor Gérard_Depardieu want to be a Belgian .`
3. annotation (part of speech tagging, parsing)
   - `French/JJ actor/NN Gérard_Depardieu/NP want/VB to/TO be/VB a/DT Belgian/NN ./PT`

Step 2: mathematical processing

1. construction of frequency matrix
2. weight elements of matrix
3. optional: smooth matrix (dimensionality reduction)
4. calculate similarity

Construction of frequency matrix

	red	tasty	fast	second-hand	big
raspberry	728	592	0	0	823
strawberry	1035	633	0	0	890
car	392	0	487	370	920
truck	104	0	393	293	846
banana	0	489	0	0	500

Weighting elements of matrix

- give more weight to surprising co-occurrences and less weight to expected co-occurrences
- (positive) pointwise mutual information
- quantifies mismatch between joint probability and probability of individual entities assuming independence (i.e. co-occurrence probability by chance)
- $pmi(x, y) = \log(\frac{p(x,y)}{p(x)p(y)})$

Weighting elements of matrix

- $pmi(x, y) = \log(\frac{p(x,y)}{p(x)p(y)})$

	red	tasty	fast	second-hand	big
raspberry	728	592	0	0	823
strawberry	1035	633	0	0	890
car	392	0	487	370	920
truck	104	0	393	293	846
banana	0	489	0	0	500

Weighting elements of matrix

- $pmi(x, y) = \log(\frac{p(x,y)}{p(x)p(y)})$

	red	tasty	fast	second-hand	big
raspberry	.08	.06	0	0	.09
strawberry	.11	.07	0	0	.09
car	.04	0	.05	.04	.10
truck	.01	0	.04	.03	.09
banana	0	.05	0	0	.05

Weighting elements of matrix

- $pmi(x, y) = \log(\frac{p(x,y)}{p(x)p(y)})$

	red	tasty	fast	second-hand	big	$p(X)$
raspberry	.08	.06	0	0	.09	.23
strawberry	.11	.07	0	0	.09	.27
car	.04	0	.05	.04	.10	.23
truck	.01	0	.04	.03	.09	.17
banana	0	.05	0	0	.05	.10
$p(Y)$	.24	.18	.09	.07	.42	

Weighting elements of matrix

- $pmi(x, y) = \log(\frac{p(x,y)}{p(x)p(y)})$

	red	tasty	fast	second-hand	big
raspberry	.36	.42	0	0	−.09
strawberry	.53	.31	0	0	−.19
car	−.27	0	.88	.89	.01
truck	−1.31	0	.95	.94	.21
banana	0	1.00	0	0	.19

Weighting elements of matrix

- $pmi(x, y) = \log(\frac{p(x,y)}{p(x)p(y)})$

	red	tasty	fast	second-hand	big
raspberry	.36	.42	0	0	0
strawberry	.53	.31	0	0	0
car	0	0	.88	.89	.01
truck	0	0	.95	.94	.21
banana	0	1.00	0	0	.19

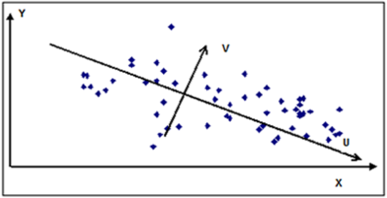
Smoothing the matrix

Two reasons for performing dimensionality reduction:

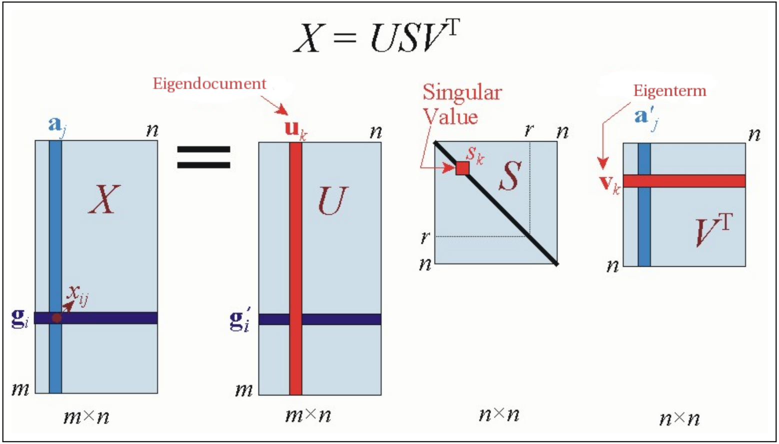
- Intractable computations
  - When number of elements and number of features is too large, similarity computations may become intractable
  - reduction of the number of features makes computation tractable again
- Generalization capacity
  - the dimensionality reduction is able to describe the data better, or is able to capture intrinsic semantic features
  - dimensionality reduction is able to improve the results (counter data sparseness and noise)

Singular value decomposition (SVD)

- popular technique for dimensionality reduction
- SVD decomposes matrix **X** into product of three matrices $\mathbf{USV}^T$
- **U** contains left-singular vectors, **V** contains right-singular vectors, **S** is diagonal matrix which contains singular values
- mathematical technique able to find new dimensions that explain most variance



SVD: graphical representation



Example

$M =$

$$U \begin{bmatrix} -2.0513e-03 & -4.2637e-01 & -3.9208e-01 & 1.7899e-01 & -7.9526e-01 \\ -1.6467e-03 & -3.8275e-01 & -7.3766e-01 & -1.5740e-01 & 5.3346e-01 \\ -6.7952e-01 & 1.9344e-02 & -3.7455e-02 & 7.1240e-01 & 1.7018e-01 \\ -7.3352e-01 & -7.2086e-04 & 2.7325e-02 & -6.6006e-01 & -1.5975e-01 \\ -1.3561e-02 & -8.1935e-01 & 5.4771e-01 & -2.2125e-03 & 1.6879e-01 \end{bmatrix}$$

$$S \begin{bmatrix} 1.8443492 & 0 & 0 & 0 & 0 \\ 0 & 1.1924855 & 0 & 0 & 0 \\ 0 & 0 & 0.5633409 & 0 & 0 \\ 0 & 0 & 0 & 0.1350923 & 0 \\ 0 & 0 & 0 & 0 & 0.0021991 \end{bmatrix}$$

$$V \begin{bmatrix} -8.7025e-04 & -2.9779e-01 & -9.4323e-01 & -1.4679e-01 & 9.2507e-03 \\ -8.1633e-03 & -9.4569e-01 & 2.7049e-01 & 1.7971e-01 & -1.1854e-02 \\ -7.0479e-01 & 1.3778e-02 & -1.2635e-02 & 1.2733e-02 & -7.0906e-01 \\ -7.0371e-01 & 1.3920e-02 & -1.3702e-02 & 1.0821e-01 & 7.0193e-01 \\ -8.9450e-02 & -1.2886e-01 & 1.9184e-01 & -9.6658e-01 & 6.5632e-02 \end{bmatrix}^T$$

Example

$M =$

$$U \begin{bmatrix} -2.0513e-03 & -4.2637e-01 & -3.9208e-01 & 1.7899e-01 & -7.9526e-01 \\ -1.6467e-03 & -3.8275e-01 & -7.3766e-01 & -1.5740e-01 & 5.3346e-01 \\ -6.7952e-01 & 1.9344e-02 & -3.7455e-02 & 7.1240e-01 & 1.7018e-01 \\ -7.3352e-01 & -7.2086e-04 & 2.7325e-02 & -6.6006e-01 & -1.5975e-01 \\ -1.3561e-02 & -8.1935e-01 & 5.4771e-01 & -2.2125e-03 & 1.6879e-01 \end{bmatrix}$$

$$S \begin{bmatrix} 1.8443492 & 0 & 0 & 0 & 0 \\ 0 & 1.1924855 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$V \begin{bmatrix} -8.7025e-04 & -2.9779e-01 & -9.4323e-01 & -1.4679e-01 & 9.2507e-03 \\ -8.1633e-03 & -9.4569e-01 & 2.7049e-01 & 1.7971e-01 & -1.1854e-02 \\ -7.0479e-01 & 1.3778e-02 & -1.2635e-02 & 1.2733e-02 & -7.0906e-01 \\ -7.0371e-01 & 1.3920e-02 & -1.3702e-02 & 1.0821e-01 & 7.0193e-01 \\ -8.9450e-02 & -1.2886e-01 & 1.9184e-01 & -9.6658e-01 & 6.5632e-02 \end{bmatrix}^T$$

Example

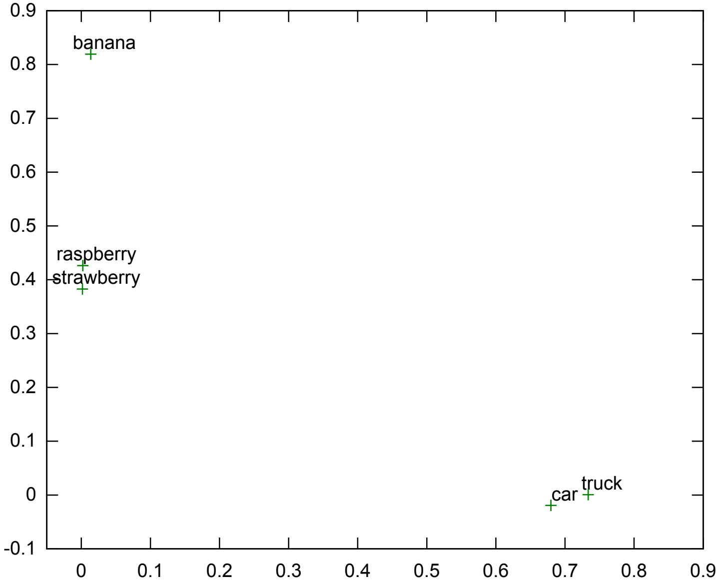
$M =$

$$U \begin{bmatrix} -2.0513e-03 & -4.2637e-01 \\ -1.6467e-03 & -3.8275e-01 \\ -6.7952e-01 & 1.9344e-02 \\ -7.3352e-01 & -7.2086e-04 \\ -1.3561e-02 & -8.1935e-01 \end{bmatrix}$$

$$S \begin{bmatrix} 1.8443492 & 0 \\ 0 & 1.1924855 \end{bmatrix}$$

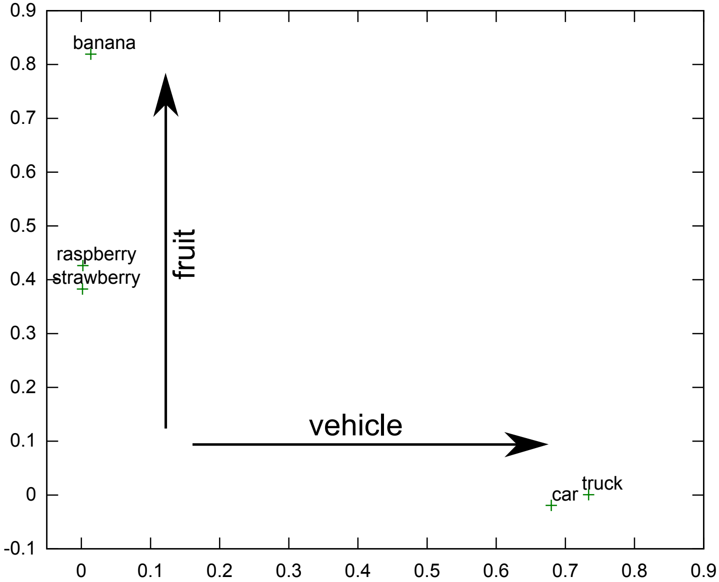
$$V \begin{bmatrix} -8.7025e-04 & -2.9779e-01 \\ -8.1633e-03 & -9.4569e-01 \\ -7.0479e-01 & 1.3778e-02 \\ -7.0371e-01 & 1.3920e-02 \\ -8.9450e-02 & -1.2886e-01 \end{bmatrix}^T$$

Example

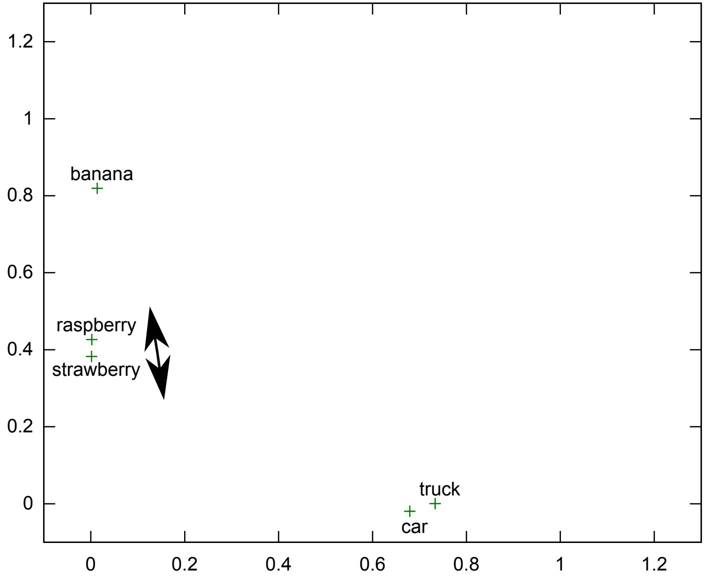


0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
banana
raspberry
strawberry
car
truck

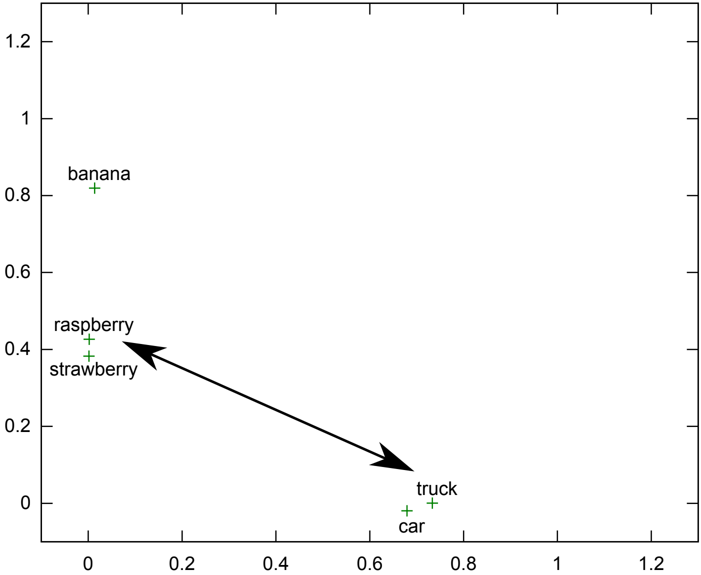
Example


0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
banana
raspberry
strawberry
fruit
vehicle
car
truck

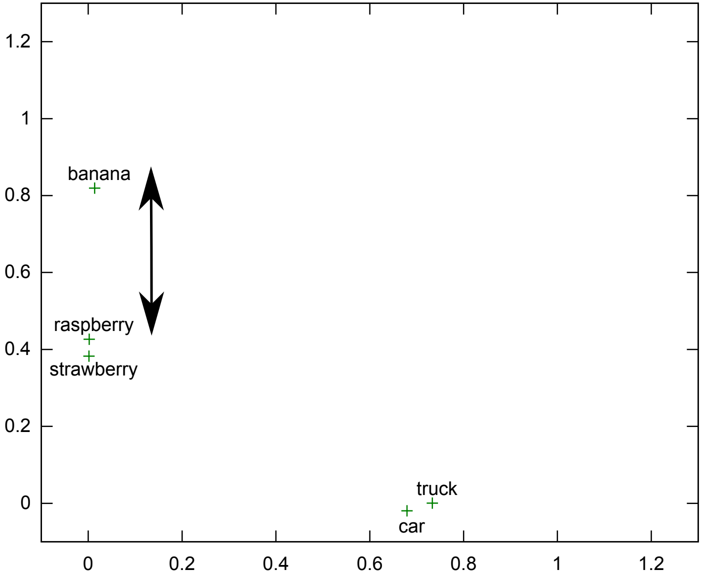
Measure similarity



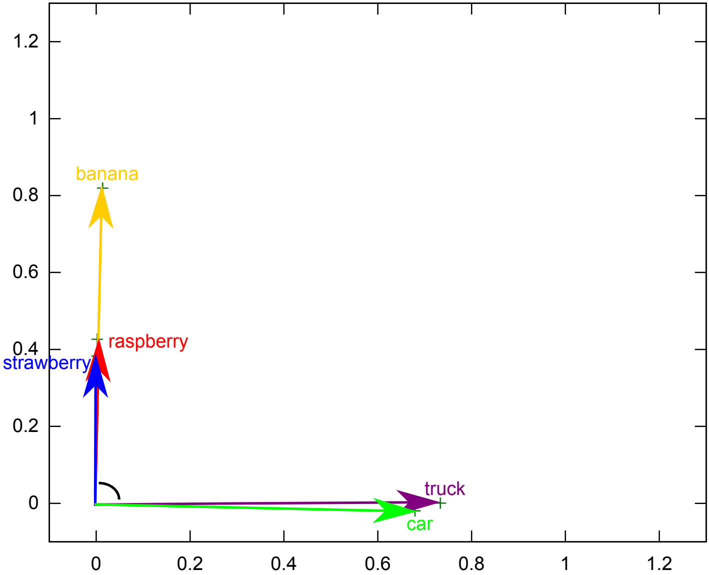
Measure similarity



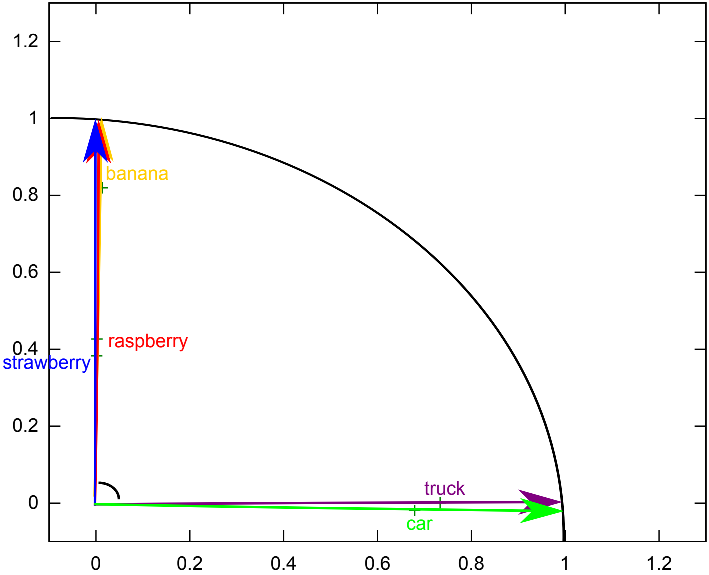
Measure similarity



Measure similarity



Measure similarity



Cosine

Formulae

- $cos(\overrightarrow{x}, \overrightarrow{y}) = \frac{\overrightarrow{x} \cdot \overrightarrow{y}}{|\overrightarrow{x}||\overrightarrow{y}|} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2 \sum_{i=1}^{n} y_i^2}}$ (standard)
- $cos(\overrightarrow{x}, \overrightarrow{y}) = \overrightarrow{x} \cdot \overrightarrow{y} = \sum_{i=1}^{n} x_i y_i$ (normalized)

- examples:
  - $cos(raspberry, strawberry)$ = (.005 x .004) + (.999 x .999) = .998
  - $cos(raspberry, car)$ = (.005 x .999) + (.999 x –.003) = .002

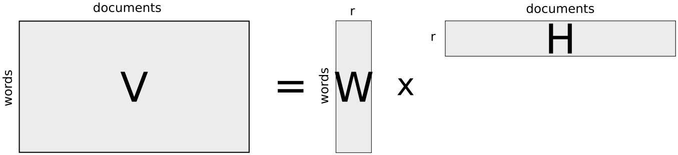
Word clusters

- Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday
- blackberry, blackcurrant, blueberry, raspberry, redcurrant, strawberry
- anthropologist, biologist, economist, linguist, mathematician, psychologist, physicist, sociologist, statistician
- drought, earthquake, famine, flood, flooding, storm, tsunami

Non-negative matrix factorization

- Given a non-negative matrix V, find non-negative matrix factors W and H such that:
$$\mathbf{V}_{n\times m} \approx \mathbf{W}_{n\times r}\mathbf{H}_{r\times m} \tag{1}$$
- Choosing $r \ll n, m$ reduces data
- Constraint on factorization: all values in three matrices need to be *non-negative values* (≥ 0)
- Constraint brings about a *parts-based* representation: only additive, no subtractive relations are allowed

Graphical Representation



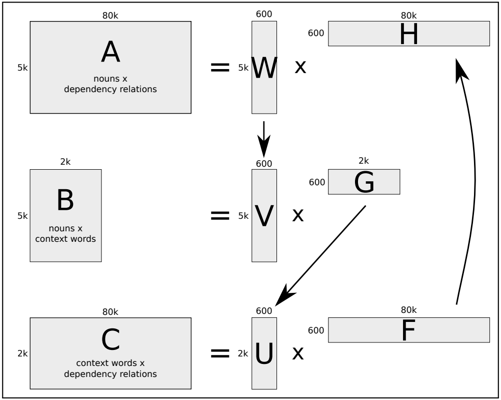
Latent factors (*k=300*)

dim 60	**dim 88**	**dim 89**	**dim 120**
rail	journal	filename	bathroom
bus	book	null	lounge
ferry	preface	integer	bedroom
train	anthology	string	kitchen
freight	author	parameter	WC
commuter	monograph	String	ensuite
tram	article	char	fireplace
airport	magazine	boolean	room
Heathrow	publisher	default	patio
Gatwick	pamphlet	int	dining

Application: word meaning in context

- Can we combine 'topical' similarity and tight, synonym-like similarity to disambiguate meaning of word in a particular context?
- Goal: classification of nouns according to both window-based context (with large window) and syntactic context
- $\Rightarrow$ Construct three matrices capturing co-occurrence frequencies for each mode
  - nouns cross-classified by dependency relations
  - nouns cross-classified by (bag of words) context words
  - dependency relations cross-classified by context words
- $\Rightarrow$ Apply NMF to matrices, but interleave the process
- Result of former factorization is used to initialize factorization of the next one

Graphical representation



Implementational details

- method applied to English
- ukwac corpus (1.5 billion words crawled from the web)
- parsed using MaltParser
- 5k nouns, 80k dependency relations, 2k context words
- nmf model: K=600, 100 iterations

Example dimension 44

nouns	context words	dependency relations
building/NN	building/NN	dobj-1#redevelop/VB
factory/NN	construction/NN	conj_and/cc#warehouse/NN
center/NN	build/VB	prep_of/in-1#redevelopment/NN
refurbishment/NN	station/NN	prep_of/in-1#refurbishment/NN
warehouse/NN	store/NN	conj_and/cc#dock/NN
store/NN	open/VB	prep_by/in-1#open/VB
station/NN	center/NN	nn#refurbishment/NN
construction/NN	industrial/JJ	prep_of/in-1#ft/NN
complex/NN	Street/NNP	amod#multi-storey/JJ
headquarters/NN	close/VB	prep_of/in-1#opening/NN

Example dimension 89

words	context words	dependency relations
virus/NN	security/NN	amod#malicious/JJ
software/NN	Microsoft/NNP	nn-1#vulnerability/NN
security/NN	Internet/NNP	conj_and/cc#worm/NN
firewall/NN	Windows/NNP	nn-1#worm/NN
spam/NN	computer/NN	nn-1#flaw/NN
Security/NNP	network/NN	nn#antivirus/NN
vulnerability/NN	attack/NN	nn#IM/NNP
system/NN	software/NN	prep_of/in#worm/NN
Microsoft/NNP	protect/VB	nn#Trojan/NNP
computer/NN	protection/NN	conj_and/cc#virus/NN

Example dimension 319

words	context words	dependency relations
virus/NN	brain/NN	dobj-1#infect/VB
disease/NN	animal/NN	nsubjpass-1#infect/VB
bacterium/NN	disease/NN	rcmod#infect/VB
infection/NN	human/JJ	nsubj-1#infect/VB
human/NN	blood/NN	prep_with/in-1#infect/VB
rat/NN	cell/NN	conj_and/cc#rat/NN
cell/NN	cancer/NN	prep_of/in#virus/NN
animal/NN	skin/NN	amod#infected/JJ
mouse/NN	scientist/NN	prep_of/in#flu/NN
cancer/NN	drug/NN	nn#monkey/NN

Calculating word meaning in context

- NMF can be interpreted probabilistically
- $p(\mathbf{z}|C) = \frac{\sum_{c_i \in C} p(\mathbf{z}|c_i)}{|C|}$ – the probability distribution over latent factors given the context ('semantic fingerprint')
- $p(\mathbf{d}|C) = p(\mathbf{z}|C)p(\mathbf{d}|\mathbf{z})$ – probability distribution over dependency features given the context
- $p(\mathbf{d}|w_i, C) = p(\mathbf{d}|w_i) \cdot p(\mathbf{d}|C)$ – weight each dependency feature of the original noun vector according to its prominence given the context
- Using the distribution over latent senses, it is possible to calculate the precise meaning of a word in context

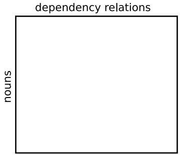
Example

1. Jack is listening to a **record**.
   - $p(\textbf{topic}|listen_{pc(to)}) \rightarrow p(\textbf{feature}|record_N, listen_{pc(to)})$
   - **record**$_N$: *album*, *song*, *recording*, *track*, *cd*

2. Jill updated the **record**.
   - $p(\textbf{topic}|update_{obj}) \rightarrow p(\textbf{feature}|record_N, update_{obj})$
   - **record**$_N$: *file*, *data*, *document*, *database*, *list*

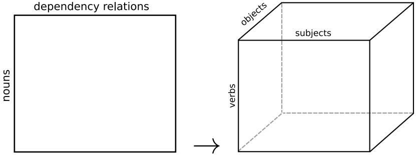
Multi-way co-occurrences

- all methods use two way co-occurrence frequencies $\longrightarrow$ matrix
- suitable for two-way problems
  - words $\times$ documents
  - nouns $\times$ dependency relations
- not suitable for *n*-way problems
  - words $\times$ documents $\times$ authors
  - verbs $\times$ subjects $\times$ direct objects



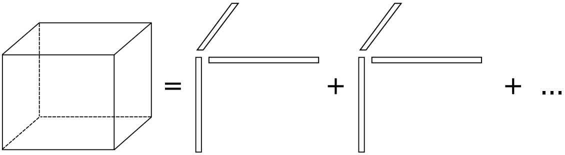
Multi-way co-occurrences

- all methods use two way co-occurrence frequencies $\longrightarrow$ matrix
- suitable for two-way problems
    - words $\times$ documents
    - nouns $\times$ dependency relations
- not suitable for *n*-way problems $\longrightarrow$ tensor
    - words $\times$ documents $\times$ authors
    - verbs $\times$ subjects $\times$ direct objects

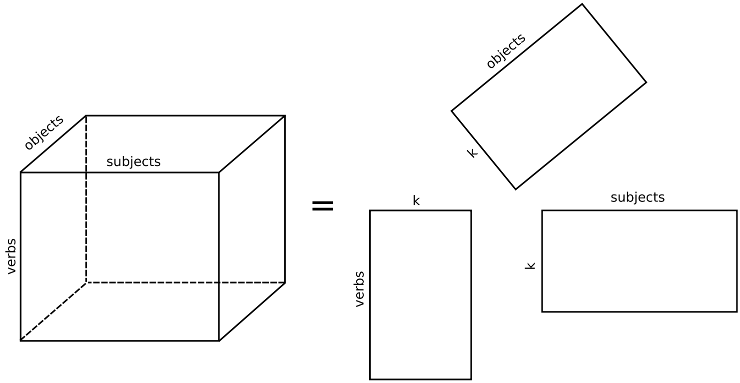


Non-negative tensor factorization

- Idea similar to non-negative matrix factorization
- Calculations are different
- $\min_{x_i \in \mathbb{R}^{D1}_{\geq 0}, y_i \in \mathbb{R}^{D2}_{\geq 0}, z_i \in \mathbb{R}^{D3}_{\geq 0}} \parallel T - \sum_{i=1}^{k} x_i \circ y_i \circ z_i \parallel_F^2$



Graphical representation



Application: selectional preference induction

- selectional preference models model preference of verb for particular arguments
- Standard selectional preference models: two-way co-occurrences
- Keeping track of single relationships
- But: two-way selectional preference models are not sufficiently rich
- Compare:
  - *The skyscraper is playing coffee.*
  - *The turntable is playing the piano.*

Application: selectional preference induction

- *The skyscraper is playing coffee.*
  - $(play, su, scyscraper) \downarrow$
  - $(play, obj, coffee) \downarrow$
- *The turntable is playing the piano.*
  - $(play, su, turntable) \uparrow$
  - $(play, obj, piano) \uparrow$
  - $(play, su, turntable, obj, piano) \downarrow$

Methodology

- Three-way extraction of selectional preferences
- Approach applied to Dutch, using TWENTE NIEUWS CORPUS (500M words of newspaper texts)
- parsed with Dutch dependency parser ALPINO
- three-way co-occurrence of verbs with subjects and direct objects extracted
- adapted with extension of pointwise mutual information
- Resulting tensor 1K verbs $\times$ 10K subjects $\times$ 10K direct objects
- reduction to k dimensions (k=50,100,300)

Example dimension: police action

subjects	su_s	**verbs**	v_s	**objects**	obj_s
politie 'police'	.99	*houd_aan* 'arrest'	.64	*verdachte* 'suspect'	.16
agent 'policeman'	.07	*arresteer* 'arrest'	.63	*man* 'man'	.16
autoriteit 'authority'	.05	*pak_op* 'run in'	.41	*betoger* 'demonstrator'	.14
Justitie 'Justice'	.05	*schiet_dood* 'shoot'	.08	*relschopper* 'rioter'	.13
recherche 'detective force'	.04	*verdenk* 'suspect'	.07	*raddraaier* 'instigator'	.13
marechaussee 'military police'	.04	*tref_aan* 'find'	.06	*overvaller* 'raider'	.13
justitie 'justice'	.04	*achterhaal* 'overtake'	.05	*Roemeen* 'Romanian'	.13
arrestatieteam 'special squad'	.03	*verwijder* 'remove'	.05	*actievoerder* 'campaigner'	.13
leger 'army'	.03	*zoek* 'search'	.04	*hooligan* 'hooligan'	.13
douane 'customs'	.02	*spoor_op* 'track'	.03	*Algerijn* 'Algerian'	.13

Example dimension: legislation

subjects	su_s	verbs	v_s	objects	obj_s
meerderheid 'majority'	.33	*steun* 'support'	.83	*motie* 'motion'	.63
VVD	.28	*dien_in* 'submit'	.44	*voorstel* 'proposal'	.53
D66	.25	*neem_aan* 'pass'	.23	*plan* 'plan'	.28
Kamermeerderheid	.25	*wijs_af* 'reject'	.17	*wetsvoorstel* 'bill'	.19
fractie 'party'	.24	*verwerp* 'reject'	.14	*hem* 'him'	.18
PvdA	.23	*vind* 'think'	.08	*kabinet* 'cabinet'	.16
CDA	.23	*aanvaard* 'accepts'	.05	*minister* 'minister'	.16
Tweede Kamer	.21	*behandel* 'treat'	.05	*beleid* 'policy'	.13
partij 'party'	.20	*doe* 'do'	.04	*kandidatuur* 'candidature'	.11
Kamer 'Chamber'	.20	*keur_goed* 'pass'	.03	*amendement* 'amendment'	.09

Example dimension: exhibition

subjects	su_s	**verbs**	v_s	**objects**	obj_s
tentoonstelling 'exhibition'	.50	*toon* 'display'	.72	*schilderij* 'painting'	.47
expositie 'exposition'	.49	*omvat* 'cover'	.63	*werk* 'work'	.46
galerie 'gallery'	.36	*bevat* 'contain'	.18	*tekening* 'drawing'	.36
collectie 'collection'	.29	*presenteer* 'present'	.17	*foto* 'picture'	.33
museum 'museum'	.27	*laat* 'let'	.07	*sculptuur* 'sculpture'	.25
oeuvre 'oeuvre'	.22	*koop* 'buy'	.07	*aquarel* 'aquarelle'	.20
Kunsthal	.19	*bezit* 'own'	.06	*object* 'object'	.19
kunstenaar 'artist'	.15	*zie* 'see'	.05	*beeld* 'statue'	.12
dat 'that'	.12	*koop_aan* 'acquire'	.05	*overzicht* 'overview'	.12
hij 'he'	.10	*in huis heb* 'own'	.04	*portret* 'portrait'	.11

Three-way selectional preferences

- Selectional preference value according to model
  - $x_{svo} = \sum_{i=1}^{k} s_{si} v_{vi} o_{oi}$
- Tensor factorization model largely outperforms other selectional preference models on standard evaluation task

Conclusion

- meaning of words is closely connected to statistics of word usage
- vector space models provide an adequate framework for the modeling of these statistics
- Dimensionality reduction models provide ‘latent semantics’
  - ability to generalize over co-occurrence data
  - useful in applications (computation of word meaning in context)
- tensor (factorization) models provide adequate means to model multi-way language phenomena

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *The Journal of Machine Learning Research*, 3:993–1022.

Zellig S. Harris. 1954. Distributional structure. *Word*, 10(23):146–162.

Thomas Landauer and Susan Dumais. 1997. A solution to Plato's problem: The Latent Semantic Analysis theory of the acquisition, induction, and representation of knowledge. *Psychology Review*, 104:211–240.

Daniel D. Lee and H. Sebastian Seung. 2000. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems 13*, pages 556–562.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (COLING-ACL98), Volume 2*, pages 768–774, Montreal, Quebec, Canada.

Tim Van de Cruys. A non-negative tensor factorization model for selectional preference induction. 2010. *Journal of Natural Language Engineering*, 16(4):417–437.

Tim Van de Cruys and Marianna Apidianaki. 2011. Latent semantic word sense induction and disambiguation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1476–1485, Portland, Oregon, USA.

Tim Van de Cruys, Thierry Poibeau, and Anna Korhonen. 2011. Latent vector weighting for word meaning in context. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1012–1022, Edinburgh, Scotland, UK.