Vector space models for words and documents

Statistical Natural Language Processing
ELEC-E5550
Jan 31, 2023

Tiina Lindh-Knuutila D.Sc. (Tech)
Lingsoft Language Services & Aalto University
tiina.lindh-knuutila -at- lingsoft dot fi
Material also by Mari-Sanna Paukkeri
(mari-sanna.paukkeri -at- utopiaanalytics dot com)

Today’s Agenda

- Short introduction to Lingsoft
- Vector space models
  - word-document matrices
  - word vectors
  - stemming, weighting, dimensionality reduction
  - similarity measures
  - Count models vs. predictive models
  - Word2vec
  - Other models
- Information retrieval (Briefly)

- Course book: Speech and Language Processing. Daniel Jurafsky & James H. Martin. Draft of December 2021. Chapter 6 - Vector semantics and embeddings https://web.stanford.edu/~jurafsky/slp3/

Lingsoft

A Full Service Language Management Company

Lingsoft®
LANGUAGE
MANAGEMENT

Lingsoft in a Nutshell

Experienced innovation partner for the analysis, production and management of spoken and written language.

Founded in **1986**

Offices in **Turku, Helsinki, Stockholm**

Revenue ~**12 million €** (2021)

On top 100 largest language service providers in the world

About **150 employees**

More than **2000 partners**

Nearly **1 000 end customers**

Regular **collaborations with national and international organisations in research projects**

Language Service Solutions for: Accessibility, Discoverability, Productivity, Interoperability

Speech and Language Technology Assisted Service Design

Lingsoft

Core Services

Translation & Localisation

Subtitling

Medical Transcription

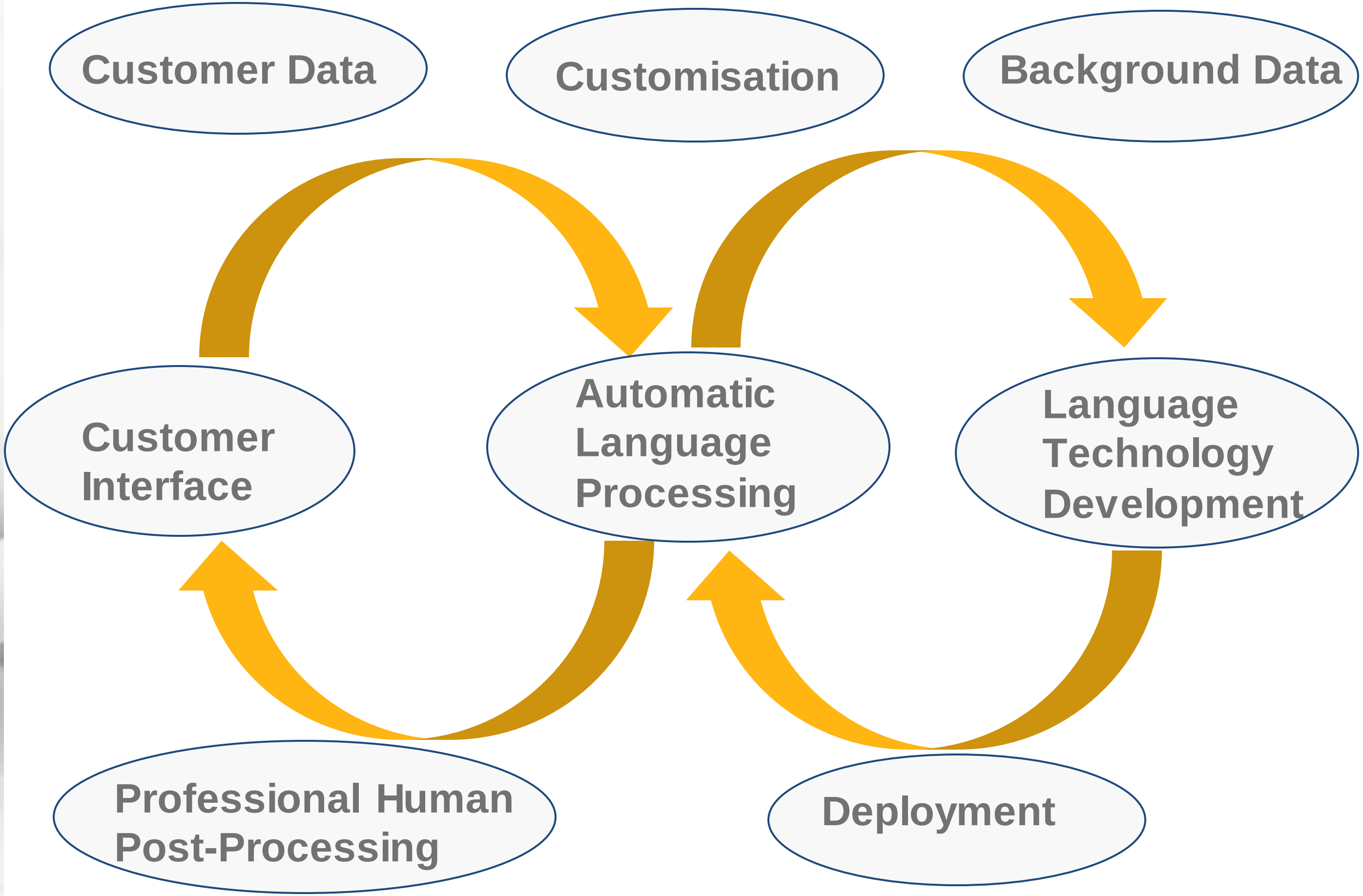
Transcription

Proofing

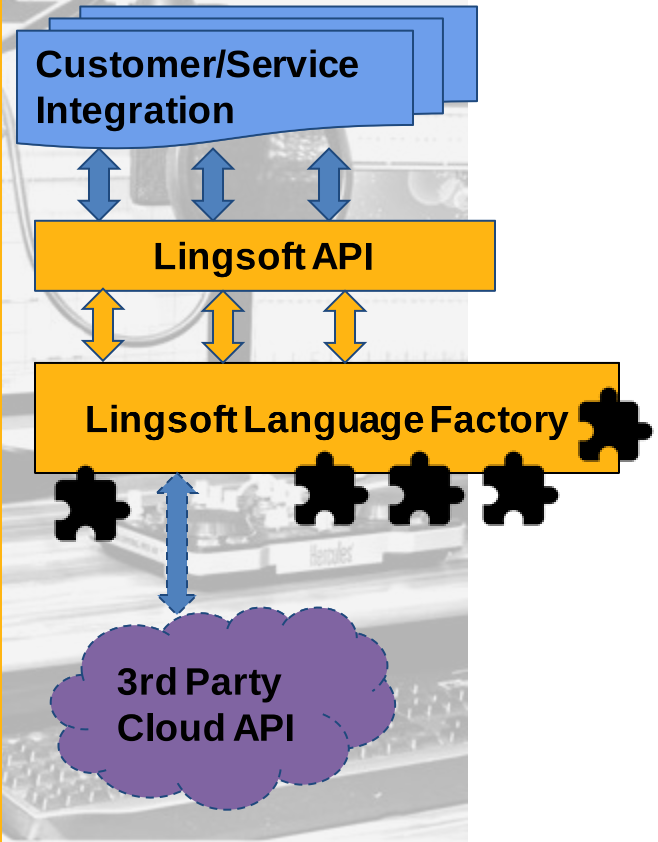
Terminology

Metadata

Lingsoft Process Design Model


Customer Data
Customisation
Background Data
Customer Interface
Automatic Language Processing
Language Technology Development
Professional Human Post-Processing
Deployment

Lingsoft’s Core Technologies



- Current main functionalities:
  - Automatic **speech recognition** and **subtitling**
  - Neural **machine translation**
  - **Spelling** and/or **grammar checking**
  - Text analysis (**lemmatization**, named entity recognition, **terminology recognition**, …)
- Core languages: **Finnish** and **Swedish** (and translation to/from those languages)
- Easy extension to new languages and functionality
- Easy integration of open source and 3rd party components
- **Easy integration to customer solutions via industry standard APIs**

Lingsoft®

Career opportunities at Lingsoft

- Language services
  - Freelance translator
  - Freelance transcriber
  - Translation coordinator
- Language solutions
  - Project management
  - Solution design
  - Development (software/linguistic)
  - Support

Contact: Jobs@lingsoft.fi

Vector space models

You shall know the word by the company it keeps

- Language is symbolic in nature
  - Surface form is in an arbitrary relation with the meaning of the word
  - Hat vs. cat
    - One substitution: Levenshtein distance of 1
    - Does not measure the semantic similarity of the words
- Distributional semantics
  - **Linguistic items** which appear in **similar contexts** in large samples of language data tend to **have similar meanings**

Firth, John R. 1957. A synopsis of linguistic theory 1930–1955. In Studies in linguistic analysis, 1–32. Oxford: Blackwell.

George Miller and Walter Charles. 1991. Contextual correlates of semantic similarity. Language and Cognitive Processes, 6(1):1–28.

Word similarity and relatedness

- Words tend not to have many ‘true’ synonyms but
    - Words have many words that are ‘similar’
    - Cat and dog vs. cat and democracy
    - How do we know? - Ask people

- Relatedness: Words are somehow related to each other
    - association
    - part-whole relation
    - belonging to same thematic domain

- How to use?
    - Similarity and association databases such as SimLEX-999
    - Used to evaluate the quality of the vector space models

Count vs. Predict

- Count-based methods
  - compute the word co-occurrence statistics with its neighbor words in a large text corpus - **sparse representation**
  - followed by a mapping (through weighting and dimensionality reduction) to dense vectors
- Predictive models
  - try to predict a word from its neighbors by **directly** learning a **dense representation**

Vector space models (VSM)

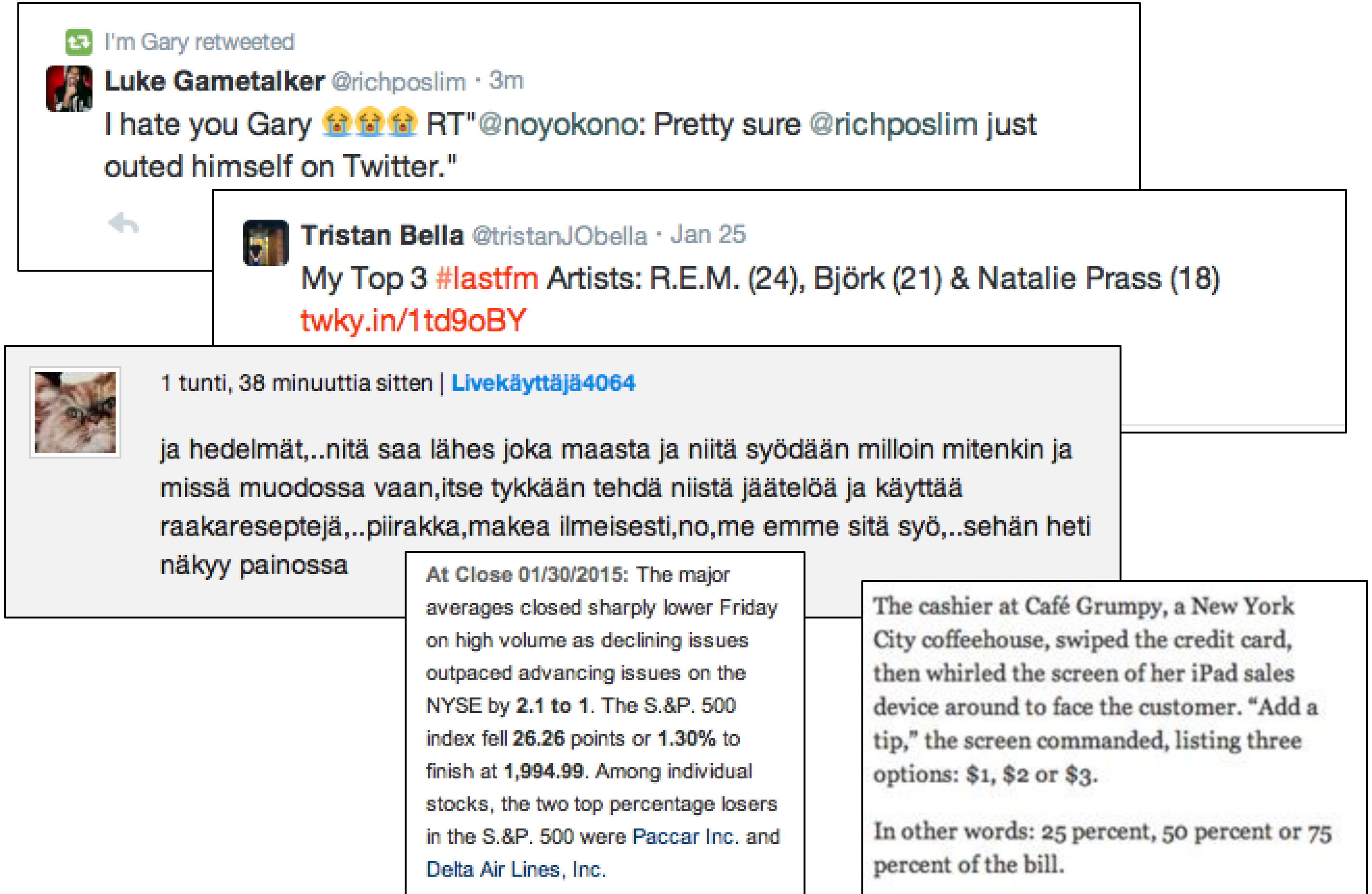
- The use of a high-dimensional space of documents (or words)
- Closeness in the vector space resembles closeness in the semantics or structure of the documents (depending on the features extracted).
- Makes the use of data mining possible
- Applications:
  - Document clustering/classification/…
    - Finding similar documents
    - Finding similar words
  - Word disambiguation
  - Information retrieval
    - Term discrimination: ranking keywords in the order of usefulness

Vector space models (VSM)

Steps to build a vector space model

1. Preprocessing
2. Defining word-document or word-word matrix
    a. choosing features
3. Dimensionality reduction
    a. choosing features
    b. removing noise
    c. easing computation
4. Weighting and normalization
    a. emphasizing the features
5. Similarity / distance measures
    a. comparing the vectors

Preprocessing



I'm Gary retweeted

Luke Gametalker @richposlim · 3m

I hate you Gary 😭😭😭 RT"@noyokono: Pretty sure @richposlim just outed himself on Twitter."

Tristan Bella @tristanJObella · Jan 25

My Top 3 #lastfm Artists: R.E.M. (24), Björk (21) & Natalie Prass (18) twky.in/1td9oBY

1 tunti, 38 minuuttia sitten | **Livekäyttäjä4064**

ja hedelmät,..nitä saa lähes joka maasta ja niitä syödään milloin mitenkin ja missä muodossa vaan,itse tykkään tehdä niistä jäätelöä ja käyttää raakareseptejä,..piirakka,makea ilmeisesti,no,me emme sitä syö,..sehän heti näkyy painossa

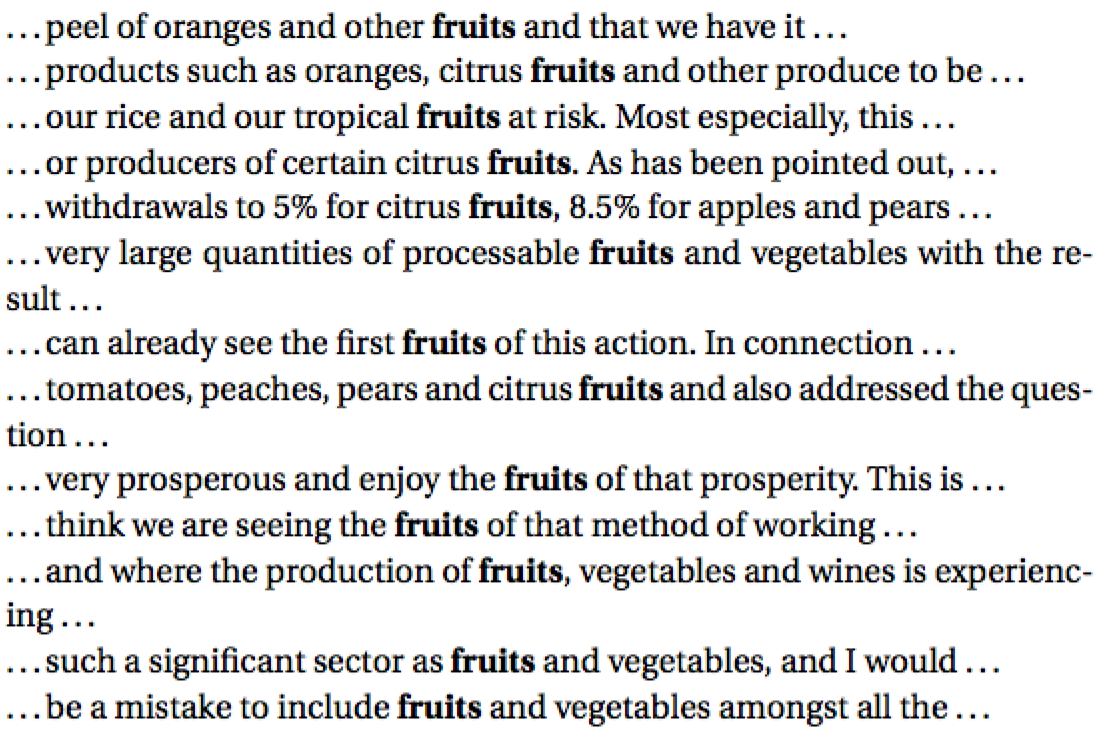
At Close 01/30/2015: The major averages closed sharply lower Friday on high volume as declining issues outpaced advancing issues on the NYSE by **2.1 to 1**. The S.&P. 500 index fell **26.26** points or **1.30%** to finish at **1,994.99**. Among individual stocks, the two top percentage losers in the S.&P. 500 were Paccar Inc. and Delta Air Lines, Inc.

The cashier at Café Grumpy, a New York City coffeehouse, swiped the credit card, then whirled the screen of her iPad sales device around to face the customer. “Add a tip,” the screen commanded, listing three options: $1, $2 or $3.

In other words: 25 percent, 50 percent or 75 percent of the bill.

Word-word matrix

- Example from Europarl corpus (Koehn, 2005):



…peel of oranges and other **fruits** and that we have it …

…products such as oranges, citrus **fruits** and other produce to be …

…our rice and our tropical **fruits** at risk. Most especially, this …

…or producers of certain citrus **fruits**. As has been pointed out, …

…withdrawals to 5% for citrus **fruits**, 8.5% for apples and pears …

…very large quantities of processable **fruits** and vegetables with the result …

…can already see the first **fruits** of this action. In connection …

…tomatoes, peaches, pears and citrus **fruits** and also addressed the question …

…very prosperous and enjoy the **fruits** of that prosperity. This is …

…think we are seeing the **fruits** of that method of working …

…and where the production of **fruits**, vegetables and wines is experiencing …

…such a significant sector as **fruits** and vegetables, and I would …

…be a mistake to include **fruits** and vegetables amongst all the …

Word-word matrix

- Choosing features for *word meaning*
- **First-order co-occurences**: collected for *target word* ("fruits") by counting the frequencies of *context words*

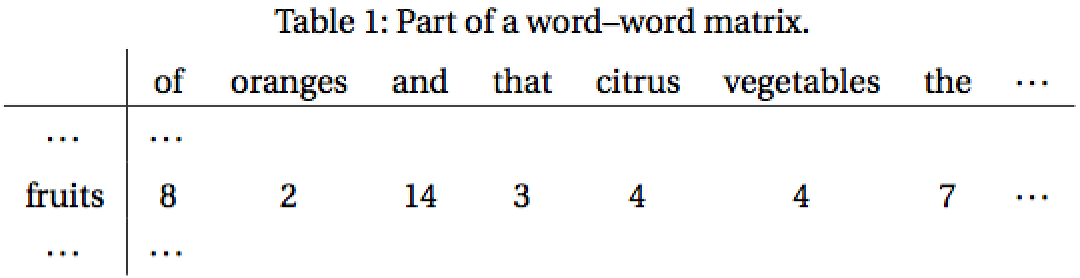


Table 1: Part of a word–word matrix.

	of	oranges	and	that	citrus	vegetables	the	⋯
⋯	⋯							
fruits	8	2	14	3	4	4	7	⋯
⋯	⋯							

- **Second-order co-occurrences***:* words that co-occur with the same target words
- *e.g. "trees"* which also co-occurs with both *"oranges"* and *"citrus"* -> second-order similarity between *"fruits"* and *"trees"*

Word-document matrix

- Choosing features for *document contents*
- A document may be
  - text document
  - e-mail message
  - tweet
  - paragraph of a text
  - sentence of a text
  - phrase

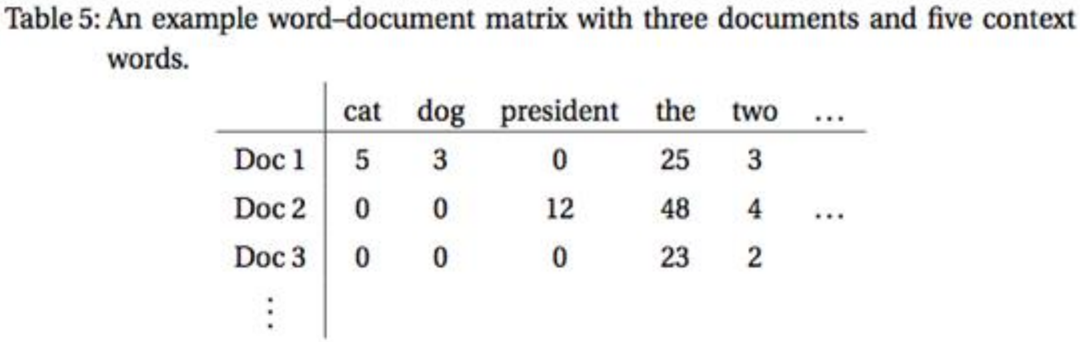


Table 5: An example word–document matrix with three documents and five context words.

	cat	dog	president	the	two	…
Doc 1	5	3	0	25	3	
Doc 2	0	0	12	48	4	…
Doc 3	0	0	0	23	2	
⋮						

Word-word/document matrix

- The values to word-word or word-document matrix can be collected in many ways – **Context**
- Sliding window
    - n words before and after the target
  - Using word order
        - word order taken into account
  - Using syntactic information
  - Bag-of-words
      - word order not taken into account
  - N-grams
      - unigrams, bigrams, trigrams, n-grams

Word-document matrix

Word-word matrix

Dimensionality reduction

- Choosing features, removing noise, easing computation
- Feature selection
  - Choose the best features (= representation words) for your task, remove the rest
  - Can be mapped back to the original features
- Feature extraction: reparametrization
  - Calculate a new, usually lower-dimensional feature space
  - New features are (complex) combinations of the old features
  - Mapping back to the original features (representation words) might be difficult

Dimensionality reduction

- Feature selection
  - excluding very frequent and/or very rare words
  - excluding stop words ('of', 'the', 'and', 'or, ...)
    - Words which are filtered out prior to processing of natural language texts, in particular, before storing the documents in the inverted index. A stop word list contains typically words such as “a”, “about”, “above”, “across”, “after”, “afterwards”, “again”, etc. The list reduces the size of the index but can also prevent from querying some special phrases like “it magazine”, “The Who”, “Take That”.
  - remove punctuation, non-alphabetic characters
  - keyphrase extraction

Dimensionality reduction

Feature extraction: reparametrization

- Stemming and lemmatizing
- Singular value decomposition (SVD), Latent semantic indexing/analysis (LSI/LSA)
- Principal component analysis (PCA)
- Independent component analysis (ICA)
- Random projection (random indexing)

Dimensionality reduction -> Feature extraction

-> Stemming and lemmatizing

- **Lemmatizing**: Finding the base form of an inflected word (requires a dictionary)
  - laughs -> laugh, matrices -> matrix,
  - Helsin**gille** -> Helsinki, saunoihin -> sauna
- **Stemming** is an approximation for morpological analysis (a set of rules is enough). The stem of each word is used instead of the inflected form.
- Examples:

Stem	Word forms
laugh-	**laughing, laugh, laughs, laughed**
galler-	**gallery, galleries**
yö-	yöllinen, yötön, yöllä
öi-	öisin, öinen
saun-	saunan, saunaan, saunoihin, saunasta, saunoistamme

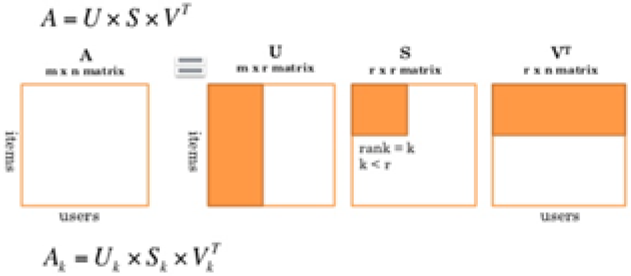
Stemming is a simplifying solution and does not suit well for languages like Finnish in all NLP applications. For one basic word form there may be several stems for search (e.g. "yö-" and "öi-" in the table refer to the same base form "yö" (night))

VSM: (3) Dimensionality reduction -> Feature extraction -> **Singular value decomposition (SVD)**

- Latent Semantic Indexing (LSI) finds a low-rank approximation to the original term-document matrix using Singular Value Decomposition (SVD).
- *W is a document-word matrix*, the elements of which contain a value of a function based on the number of a word in a document
  - E.g., using normalized entropy of words in the whole corpus
  - Often tf-idf weighting is in use.
- A singular value decomposition of rank R is calculated:

  $$\text{SVD(W):} \qquad (W) = USV^T$$

  in which *S* is a *diagonal matrix* which contains the singular values in its diagonal, *U* and *V* are used to *project the words and documents* into the latent space (*T*: matrix transpose).
- SVD calculates an optimal R-dimensional approximation for W.
- A typical value of R ranges between 100 and 200.



http://xieyan87.com/2015/06/stochastic-gradient-descent-sgd-singular-value-decomposition-svd-algorithms-notes/

VSM: (3) Dimensionality reduction -> Feature extraction

-> Random projection

- A *random matrix* is used to project data vectors into a lower dimensional space while the distances between data points are approximately preserved.
  - n_i - original document vector for document i
  - *R* - random matrix, the columns are normally distributed unit vectors. Dimensionality is *rdim* × *ddim*, in which *ddim* is the original dimension and *rdim* the new one, *rdim* << *ddim*
  - x_i - new, randomly projected document vector for document *i*, with vector dimension *rdim*.

$$x_i = Rn_i$$

- It is essential that the unit vectors of the projection matrix are as **orthogonal** as possible (i.e. correlations between them are small). In *R*, the vectors are not fully orthogonal but if *rdim* is sufficiently large, and the vectors are taken randomly from an even distribution on a hyperball, the correlation between any two vectors tends to be small.
- The values used for *rdim* typically range between 100 and 1000.

Weighting and normalization

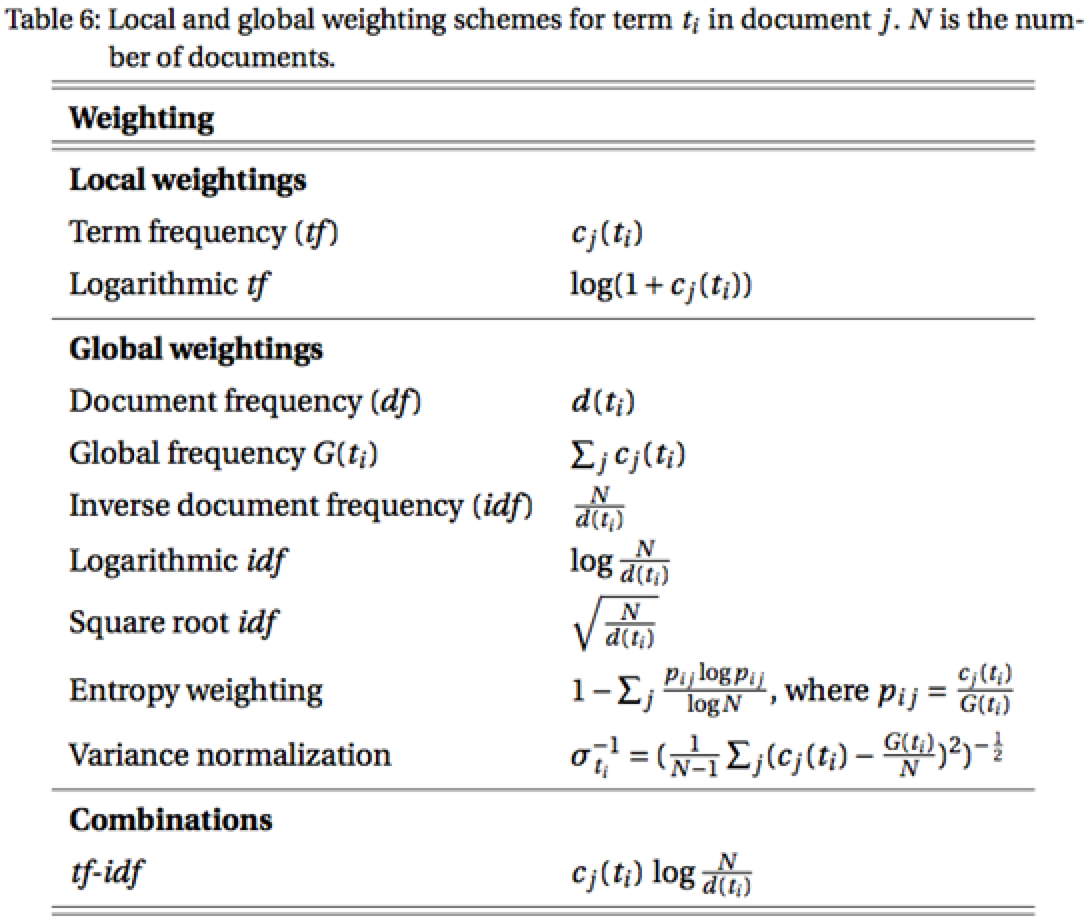


Table 6: Local and global weighting schemes for term t_i in document j. N is the number of documents.

Weighting	
Local weightings	
Term frequency (*tf*)	$c_j(t_i)$
Logarithmic *tf*	$\log(1 + c_j(t_i))$
Global weightings	
Document frequency (*df*)	$d(t_i)$
Global frequency $G(t_i)$	$\sum_j c_j(t_i)$
Inverse document frequency (*idf*)	$\frac{N}{d(t_i)}$
Logarithmic *idf*	$\log \frac{N}{d(t_i)}$
Square root *idf*	$\sqrt{\frac{N}{d(t_i)}}$
Entropy weighting	$1 - \sum_j \frac{p_{ij} \log p_{ij}}{\log N}$, where $p_{ij} = \frac{c_j(t_i)}{G(t_i)}$
Variance normalization	$\sigma_{t_i}^{-1} = (\frac{1}{N-1} \sum_j (c_j(t_i) - \frac{G(t_i)}{N})^2)^{-\frac{1}{2}}$
Combinations	
tf-idf	$c_j(t_i) \log \frac{N}{d(t_i)}$

(Positive) pointwise mutual information

- Alternative to tf.idf weighting
- What is the probability of word *w* occuring in context *c, P(w,c)* compared to the probability of word *w P(w)* and context *c P(c)* if they occur independently of each other?
- Pointwise mutual information:

$$PMI(w,c) = log_2 \frac{P(w,c)}{P(w)P(c)}$$

- Negative values are unreliable unless corpora is enormous
  - Positive pointwise mutual information by setting all negative values to zero

Weighting and normalization

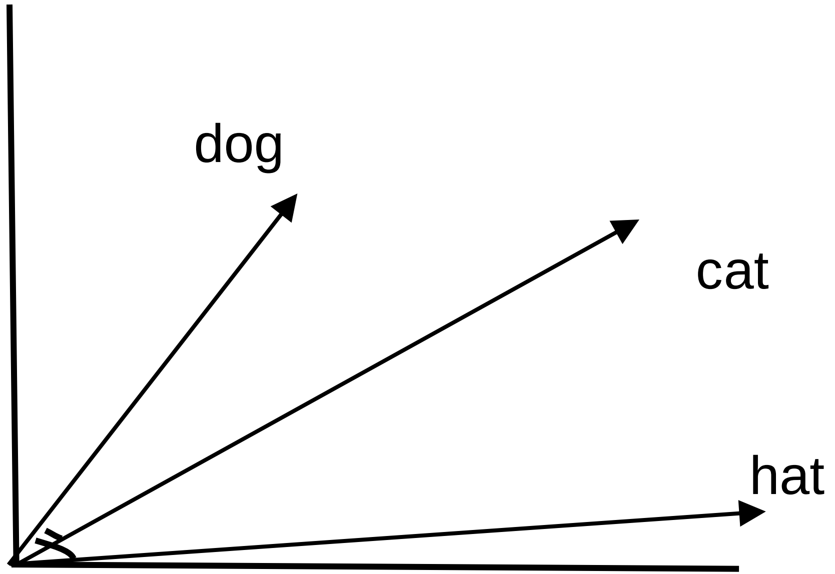
- Length Normalization
  - Vector length affects e.g. Euclidean distance calculation
  - L1 Norm: Divide every item of a vector with the Manhattan distance i.e. City Block distance
  - L2 Norm: Divide every item of a vector with the Euclidean length of the vector

$$\|\boldsymbol{x}\| := \sqrt{x_1^2 + \cdots + x_n^2}.$$

  - Not required for cosine distance

Word vectors

- Words represented by their context as vectors
- Proximity in vector space indicates semantic or functional similarity



Vector similarity such as cosine

$$1 - \frac{\mathbf{x}_i^T \mathbf{x}_j}{\|\mathbf{x}_i\|_2 \, \|\mathbf{x}_j\|_2}$$

Similarity / distance measures

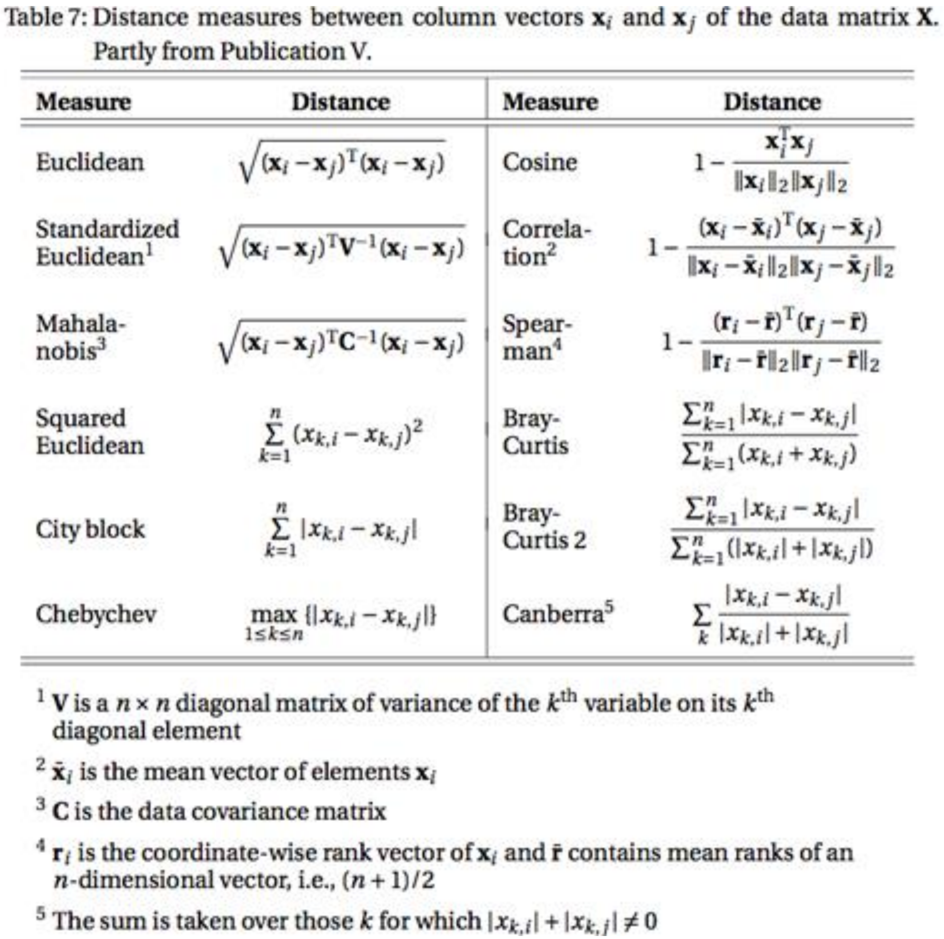


Table 7: Distance measures between column vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ of the data matrix $\mathbf{X}$. Partly from Publication V.

Measure	Distance	Measure	Distance
Euclidean	$\sqrt{(\mathbf{x}_i-\mathbf{x}_j)^{\mathrm{T}}(\mathbf{x}_i-\mathbf{x}_j)}$	Cosine	$1-\frac{\mathbf{x}_i^{\mathrm{T}}\mathbf{x}_j}{\Vert\mathbf{x}_i\Vert_2\Vert\mathbf{x}_j\Vert_2}$
Standardized Euclidean[1]	$\sqrt{(\mathbf{x}_i-\mathbf{x}_j)^{\mathrm{T}}\mathbf{V}^{-1}(\mathbf{x}_i-\mathbf{x}_j)}$	Correlation[2]	$1-\frac{(\mathbf{x}_i-\bar{\mathbf{x}}_i)^{\mathrm{T}}(\mathbf{x}_j-\bar{\mathbf{x}}_j)}{\Vert\mathbf{x}_i-\bar{\mathbf{x}}_i\Vert_2\Vert\mathbf{x}_j-\bar{\mathbf{x}}_j\Vert_2}$
Mahalanobis[3]	$\sqrt{(\mathbf{x}_i-\mathbf{x}_j)^{\mathrm{T}}\mathbf{C}^{-1}(\mathbf{x}_i-\mathbf{x}_j)}$	Spearman[4]	$1-\frac{(\mathbf{r}_i-\bar{\mathbf{r}})^{\mathrm{T}}(\mathbf{r}_j-\bar{\mathbf{r}})}{\Vert\mathbf{r}_i-\bar{\mathbf{r}}\Vert_2\Vert\mathbf{r}_j-\bar{\mathbf{r}}\Vert_2}$
Squared Euclidean	$\sum_{k=1}^{n}(x_{k,i}-x_{k,j})^2$	Bray-Curtis	$\frac{\sum_{k=1}^{n}\vert x_{k,i}-x_{k,j}\vert}{\sum_{k=1}^{n}(x_{k,i}+x_{k,j})}$
City block	$\sum_{k=1}^{n}\vert x_{k,i}-x_{k,j}\vert$	Bray-Curtis 2	$\frac{\sum_{k=1}^{n}\vert x_{k,i}-x_{k,j}\vert}{\sum_{k=1}^{n}(\vert x_{k,i}\vert+\vert x_{k,j}\vert)}$
Chebychev	$\max_{1\leq k\leq n}\{\vert x_{k,i}-x_{k,j}\vert\}$	Canberra[5]	$\sum_k\frac{\vert x_{k,i}-x_{k,j}\vert}{\vert x_{k,i}\vert+\vert x_{k,j}\vert}$

[1] $\mathbf{V}$ is a $n\times n$ diagonal matrix of variance of the k^{th} variable on its k^{th} diagonal element

[2] $\bar{\mathbf{x}}_i$ is the mean vector of elements $\mathbf{x}_i$

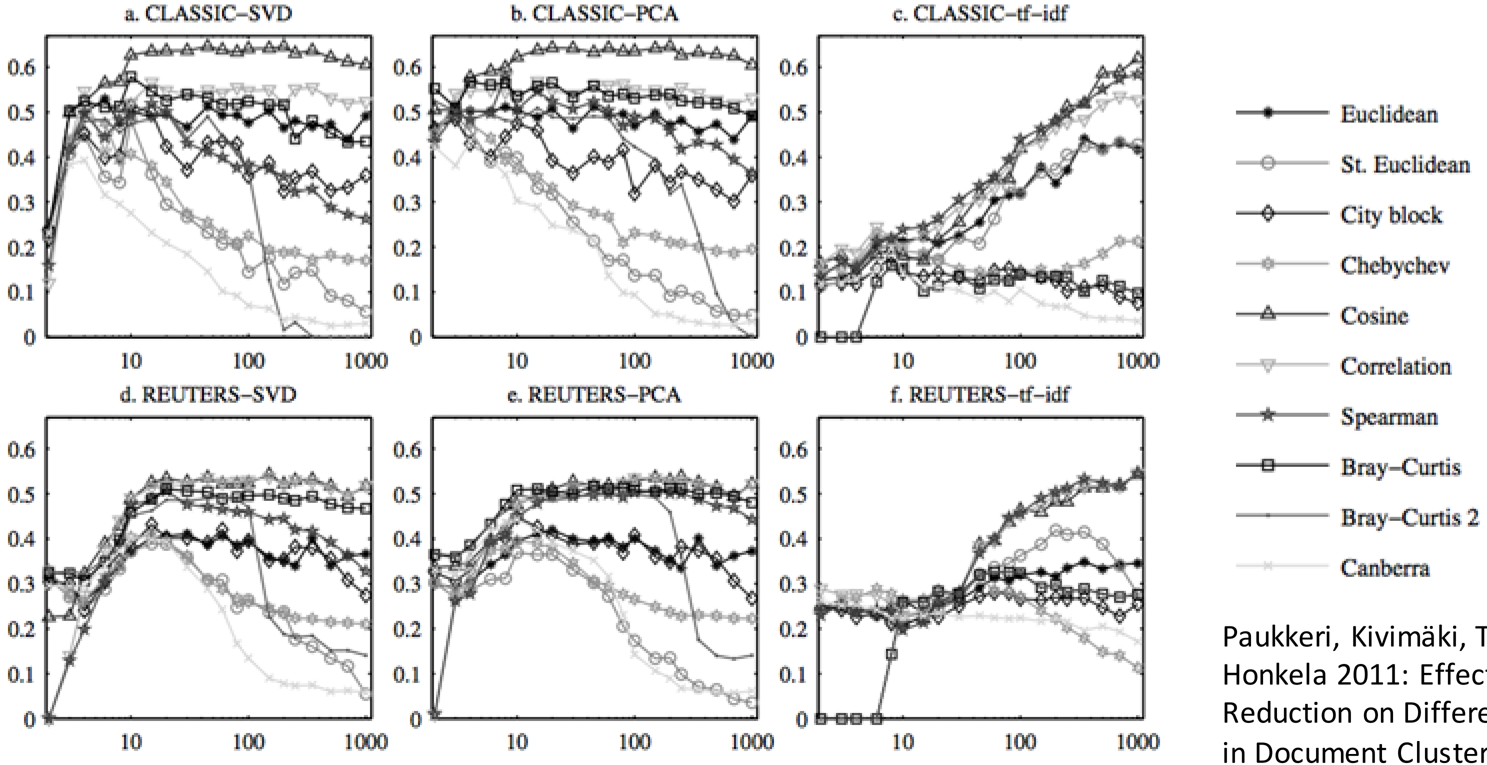
[3] $\mathbf{C}$ is the data covariance matrix

[4] $\mathbf{r}_i$ is the coordinate-wise rank vector of $\mathbf{x}_i$ and $\bar{\mathbf{r}}$ contains mean ranks of an n-dimensional vector, i.e., $(n+1)/2$

[5] The sum is taken over those k for which $|x_{k,i}|+|x_{k,j}|\neq 0$

Comparison of similarity / distance measures

- Application: document classification accuracy (on y axis) when using dimensionality reduction to 2-1000 dimensions (on x axis) on two data sets CLASSIC and REUTERS



Paukkeri, Kivimäki, Tirunagari, Oja, Honkela 2011: Effect of Dimensionality Reduction on Different Distance Measures in Document Clustering. LNCS 7064.

Group/pair discussion

- What are the benefits of the distributional semantics?
- What kind of problems there might be?

- What kind of applications can you come up with?

Distributional semantics - benefits and problems

- What are the benefits of distributional semantics?
    - Easy to make calculations if you have enough data
    - Many application areas
- What kind of problems might arise?
    - Polysemy may be an issue if the different senses are not taken into account separately, the representation may be a mix of all senses or might just represent one sense and not necessarily the most relevant in general usage
    - The symbol grounding problem: we are still stuck at the level of words. Knowing X and Y are similar does not tell us anything about what X and Y are unless we know what something similar to them is.

Applications?

- Document clustering / classification
  - Finding similar documents
  - Finding similar words
- Word disambiguation
- Information retrieval
- Term discrimination
- Sentiment analysis
- Named entity recognition
- Even brain research: See for example Kivisaari et al. 2019.
  - https://www.nature.com/articles/s41467-019-08848-0

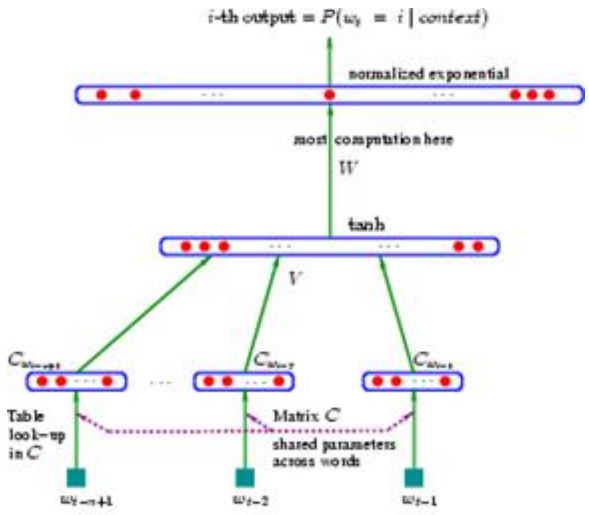
Predict(ive) models

- Based on neural language modeling
- Language model: Predict the next word from a sequence of words
  - $P(w_n | w_1, w_2, w_3, \dots w_{n-1})$

- Language modeling is used for example in speech recognition, machine translation, part-of-speech-tagging, handwriting recognition
- Represent the ‘meaning�either of the word with a vector for which the features are learned from data
- Predictive vector space models are usually called ‘embeddings’

Neural language modeling (briefly)

- Replace word with continuous-valued vector representation
- Parameters are the values in the vector, shared across words
- The NN learns to map a sequence of features to a prediction of interest
- Generalizes to unseen examples



http://www.scholarpedia.org/article/Neural_net_language_models,

Word2Vec

- Computationally efficient algorithm family for learning dense word embeddings
  - dimensionality of vector usually [100...1000]
- Readily available in good Python and C packages
- Two architectures: skip-gram and continuous bag of words (CBOW)
- Computational example:
  - Skipgram model
  - Finnish Internet corpus
  - 2B words in little over 10 hours
  - with 4 cores

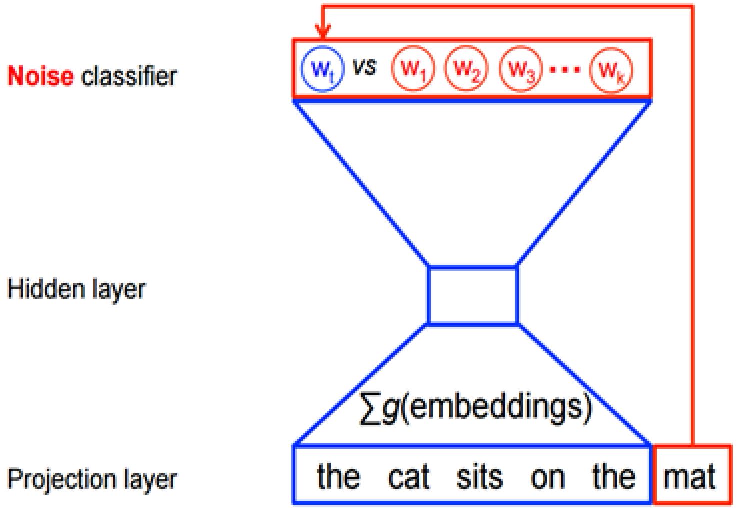
Word2vec demo

- http://epsilon-it.utu.fi/wv_demo/
- Thanks to Turku NLP group, especially Prof. Filip Ginter
- Examples in Finnish and English
    - Nearest neighbors and similarity
    - Analogy
    - Try it out yourself!
    - Let me know about online demos in other languages

- Compare Google News English results with Danish State Library demo on Project Gutenberg corpus
    - http://labs.statsbiblioteket.dk/dsc/

Word2vec architecture(s)

- Continuous Bag-Of-Words (CBOW): Predict target word given context
- Skip-gram: Predict each context word given the target word
- Negative sampling: Learn to distinguish correct target from noise words
- Computationally less intensive: calculate logistic regression instead full probabilistic model



CBOW architecture

https://www.tensorflow.org/tutorials/word2vec/

Algorithms

Maximize the log-likelihood for skipgram:

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

Hierarchical softmax

- Softmax to calculate the log-likelihood computationally infeasible
- Hierarchical approximation
- Calculate only $\log_2$(W) instead of W nodes in NN
- Word2vec uses binary Huffman tree

Negative sampling

- Minimize the log-likelihood of the negative instances
- Distinguish target word w_t from k draws from the noise distribution P_n(w) using logistic regression
- k negative samples for each data sample
- Small data sets: k = [5..20]
- Large data sets: k = [2..5]

Subsampling

- Another way to speed up the process
- Imbalance between the most frequent and the rarest words
  - ‘a’, ‘an’, ‘the’ vs. ‘broccoli’
- Do not use all of the instances in the training corpus
- Discard a word with probability

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

t = threshold, f(x), f(x) =the frequency of the word

Context window

- Context window defines which context words are taken into account when building a model
- Rule of thumb
    - Large contexts: Topic level similarities
    - Small context: words that behave similarly
- Using a **sliding window**
- Often window does not cross sentence border
- Check documentation of your tool to verify the context size is what you intend!

This is an example sentence with context window of 2

This is an example sentence

is an example sentence with

an example sentence with context

example sentence with context window

sentence with context window of

with context window of 2

context window of 2

window of 2

Performance

- architecture: skip-gram (slower, better for infrequent words) vs CBOW (faster)
- the training algorithm: hierarchical softmax (better for infrequent words) vs negative sampling (better for frequent words, better with low dimensional vectors)
- sub-sampling of frequent words: can improve both accuracy and speed for large data sets (useful values are in range 1e-3 to 1e-5)
- dimensionality of the word vectors: usually more is better, but not always (usually range of 100-1000 is used)
- context (window) size: for skip-gram usually around 10, for CBOW around 5

Ref: https://code.google.com/archive/p/word2vec/

Beyond single words

- Extended to phrases that occur often together but rarely separate
  - ‘Toronto Maple Leaves’
  - ‘New York Times’
  - Add into lexicon as individual items
- Analogy tasks (work at least in English)
  - ‘King’ - ‘man’ + ‘woman’ = ‘queen’
- Word2vec works surprisingly well with element wise addition
  - ‘French + actress’ = ‘Juliette Binoche’
  - ‘Vietnam + capital’ = ‘Hanoi’
- Models for phrases, sentences etc: Simple averaging of vectors in a sentence, doc2vec, Word Movers Distance etc..

Other word-level models

- GloVe:
  - Not only local context
  - takes **global** word-word co-occurrence statistics into account

- FastTEXT:
  - Based on skipgram model
  - **subword level**
    - for example n = 3
    - “where”: “wh”, “her”, “ere”, “re”
    - n=3...6, but different segmentations are possible
  - Word vector is the sum of character ngram vectors
  - Extends to unseen word forms

Contextualized representations

- Word2vec and other such models are context-independent at word level:
  - one single representation for an single word
  - does not capture polysemy well

- Enter contextualized representations
  - ULMFit, GPT and GPT2, ELMo and BERT (among others)
  - Based on transfer learning
  - Details will be covered at a later lecture

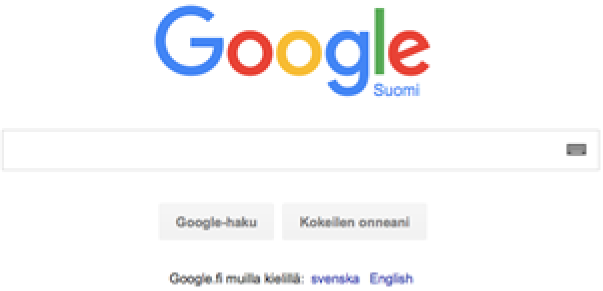
BERT

- Bidirectional Encoder Representations from Transformers
- Huge models - you basically can’t train your own base model
    - Finnish model: 3.3 B tokens trained on Puhti supercomputer at CSC on 8 Nvidia V100 GPUs for 12 days per model
    - Finnish Sentence BERT demo: http://epsilon-it.utu.fi/sbert400m
- Base models available for fine tuning!
    - Google: English, Chinese (simplified) and multilingual BERT
    - Turku NLP: FinBERT in Finnish
    - CamemBERT in French
    - Deepset.ai: German BERT
    - Russian and Slavic BERT
- Tools:
    - For example: Huggingface Transformers
        - many model architectures available in addition to BERT: GPT, GPT2, Transformer-XL, XL-Net,XLM, RoBERTa, DistilBERT, CTRL, ALBERT, T5, XLM-RoBERTa, MMBT, …

Information retrieval (IR)

- A traditional research area, currently part of NLP research
- Information retrieval from a large document collection
- Produce an **index**ed version (e.g. vector space) of the collection
- User provides a **query term**/phrase/document
- Query is compared to the index and the best matching results are given

- Example: Google search engine



IR: More terminology

- **Index term**: A term (character string) that is part of an index. Index terms are typically full words but can also be, for instance, numerical codes or word segments such as stems.
- **(Inverted) index**: The purpose of using an index is to optimize speed and performance in finding relevant documents for a search query. Without an index, the search engine would scan every document in the corpus, which would require considerable time and computing power.
- **Relevance**: How well the retrieved document(s) meet(s) the information need of the user.
- **Relevance feedback**: Taking the results that are initially returned from a given query and to use information about whether or not those results are relevant to perform a new query. The feedback can be explicit or implicit.

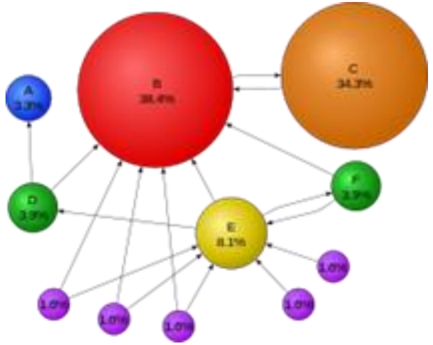
- **Information extraction**: A type of information retrieval for which the goal is to automatically extract structured information, i.e. categorized and contextually and semantically well-defined data from a certain domain, from unstructured machine-readable documents.

IR Traditionally: Exact match retrieval

- No NLP processing of the query nor index
- Often Boolean queries (AND, OR, NOT) can be used
    - e.g. Q = (mouse OR mice) AND (dog OR dogs OR puppy OR puppies) AND NOT (cat OR cats)
- Works well for small document sets and if the user is experienced with IR
- **Problems** especially with large and heterogeneous collections:
    - **Order:** The results are not ordered by any meaningful criteria.
    - **Size:** The result may be an empty set or there may be a very large set of results.
    - **Relevance:** It is difficult to formulate a query in such a manner that one would receive relevant documents but as small number of non-relevant ones as possible.
    - One cannot know what kind of relevant documents there are that do not quite match the search criteria.

IR: Ranking of query results

- Most IR systems compute a numeric score on how well each object in the database matches the query
  - Distance in the vector space
  - Content and structure of the document collection can be used
  - Number of hits in a document
  - Number of hits in title, first paragraph, elsewhere
  - Other meta information in the documents or external knowledge
- The retrieved objects are ranked according to this numeric score and the top ranking objects are then shown to the user.
- For instance, *Google's PageRank* is a link analysis algorithm that assigns a numerical weighting to each element of a hyperlinked set of documents. The purpose is to measure its relative importance within the set.

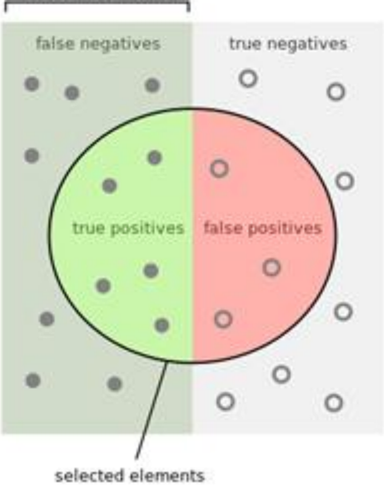


https://fi.wikipedia.org/wiki/PageRank

IR: Indexing & VSM

- The documents in the document collection are processed in the similar way as in the vector space modelling
  - Preprocessing
    - removing punctuation
    - removing capitalization
    - stemming / lemmatizing
  - Defining word-document matrix
  - Weighting and normalizing
    - Tf.idf
    - …
- The queries are then mapped to the same vector space
- The *relevance* is assessed in terms of (partial) similarity between query and document.
- The vector space model is one of the most used models for ad-hoc retrieval

IR: Evaluation



http://en.wikipedia.org/wiki/Precision_and_recall

- N = number of documents retrieved (true positive + false positive)
- REL = number of relevant documents in the whole collection (true positive + false negative)
- rel = number of relevant documents in the retrieved set of documents (true positive)

- **Precision P**: the number of relevant documents retrieved by a search divided by the total number of documents retrieved by that search,

  P = rel/N = true positive / (true positive + false positive)
- **Recall R**: the number of relevant documents retrieved by a search divided by the total number of relevant documents

  R = rel/REL = true positive / (true positive + false negative)
- An inverse relationship typically exists between P and R. It is not possible to increase one without the cost of reducing the other. One can usually increase R by retrieving a larger number of documents, also increasing number of irrelevant documents and thus decreasing P.

IR: Evaluation

- **F-measure**
  - Precision and Recall scores can be combined into a single measure, such as the F-measure, which is the weighted harmonic mean of P and R:

$$F = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{R}}$$

- **Accuracy**
  - Not a good measure if the number of relevant documents is small, which is the case usually in IR

    (true positive + true negative)/(true positive + true negative + false positive + false negative)
- Method comparison
  - Different IR methods are usually compared using precision (P) and recall (R) measures or the F-measure over a number of queries (e.g. 50), and the obtained averages are studied.
  - A statistical test (e.g. Student’s t-test) can be used to ensure the statistical significance of the observed differences.

IR: Various data types

- In addition to text documents (in any language) there are also other types of data to be retrieved, such as
  - Pictures (image retrieval)
  - Videos (video/multimedia retrieval)
  - Audio (speech retrieval, music retrieval)
  - Data/Document classifications, tags, categories (e.g. hashtags), graphs, …
  - Cross-language information retrieval

- How can these types of documents be retrieved using previously seen information retrieving methods?

Recommended reading for this lecture

- Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." arXiv preprint arXiv:1301.3781 (2013). https://arxiv.org/pdf/1301.3781.pdf
- Mikolov, Thomas, et al. Distributed representations of words and phrases and their compositionality Advances in neural information processing systems. 2013. https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf
- Baroni, Marco, Georgiana Dinu, and Germán Kruszewski. "Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors." ACL (1). 2014. http://anthology.aclweb.org/P/P14/P14-1023.pdf

References

Distributional semantics

Firth, John R. 1957. A synopsis of linguistic theory 1930–1955. In Studies in linguistic analysis, 1–32. Oxford:Blackwell.

George Miller and Walter Charles. 1991. Contextual correlates of semantic similarity. Language and Cognitive Processes, 6(1):1–28.

Vector space models and neural language models

Baroni, Marco, Georgiana Dinu, and Germán Kruszewski. "Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors." ACL (1). 2014.

Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." arXiv preprint arXiv:1301.3781 (2013). https://arxiv.org/pdf/1301.3781.pdf

Mikolov, Thomas, et al. Distributed representations of words and phrases and their compositionality Advances in neural information processing systems. 2013. https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf

Yoshua Bengio (2008) Neural net language models. Scholarpedia, 3(1):3881.

Tensorflow Tutorials:

Word2vec https://www.tensorflow.org/tutorials/word2vec/

Other models

Bengio et al: http://www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf

Collobert et al: https://ronan.collobert.com/pub/matos/2008_nlp_icml.pdf, and https://arxiv.org/abs/1103.0398

Huang et al. , GloVe: http://nlp.stanford.edu/pubs/HuangACL12.pdf

Turian et al: http://www.aclweb.org/anthology/P10-1040

References

(Manning, Schütze, 1999): Foundations of Statistical Natural Language Processing. The MIT Press.

(Koehn, 2005) Europarl: A parallel corpus for statistical machine translation. MT Summit.

(Paukkeri, 2012) Language- and domain-independent text mining. Doctoral dissertation, Aalto University.

Figures and tables:

(Paukkeri, 2012) Language- and domain-independent text mining. Doctoral dissertation, Aalto University.

Online tutorials and demos

- http://www.scholarpedia.org/article/Neural_net_language_models
- https://www.tensorflow.org/tutorials/word2vec/
- https://rare-technologies.com/word2vec-tutorial/
- https://code.google.com/archive/p/word2vec/
- http://mccormickml.com/2017/01/11/word2vec-tutorial-part-2-negative-sampling/
- http://epsilon-it.utu.fi/wv_demo/