Advanced Text mining topics

MICHALIS VAZIRGIANNIS

February 2016

Outline.

Topic modelling

Advanced keyword extraction and summarization

- Gow visualization & summarization - gowis – ACL 2016
- Heuristics and Submodularity based keyword extraction – EMNLP 2016 + EACL 2017
- keyword extraction for summarization - off line – IEEE-ICASSP(submitted)
- Abstractive summary (fillipova) + openpaas

Advanced GoW topics

- Collection level graph weights - tw-icw (under-submission)
- Shortest-Path Graph Kernels for Document Similarity -
- Graph based regularization for text classification – EMNLP2016

Topic Modeling

- Flux of information: Wikipedia articles, blogs, Flickr images, astronomical survey data, social networking activity
- Need algorithms to organize, search, and understand this information.

Topic modeling

- aims at discovering the theme(s) of documents
- is a method for analyzing large quantities of unlabeled data.

Topic is a probability distribution over a collection of words

Topic model

- statistical relationship between a group of observed and latent (unknown) random variables
- specifies a probabilistic procedure to generate the topics—a generative model.
- provides a “thematic summary” of a collection of documents.
- answers the question themes documents discuss – i.e. collection of news articles could discuss e.g. political, sports, and business related themes.

Probabilistic LSA

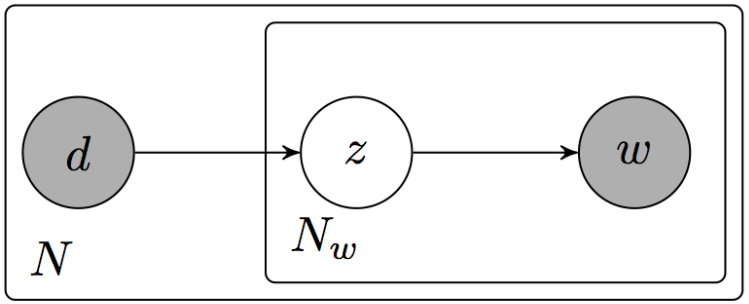
- Probabilistic Latent Semantic Analysis (pLSA) is topic model method
- main goal: model co- occurrence information under a probabilistic framework to discover the underlying semantic structure of the data.

- Developed Th. Hofmann,1999 = initially used for text-based applications (indexing, retrieval, clustering);
- spread in other fields: such as computer vision or audio processing.

- Goal of pLSA: use co-occurrence matrix to extract the “topics” and explain the documents as a mixture of them.

PLSA

Documents: d $\in$ D = $\{d_1, \cdots, d_N\}$— observed variables, |D|= N
Words: w $\in$ W = $\{w_1, \cdots, w_M\}$— observed variables, |W|=M
Topics: z $\in$ Z = $\{z_1, \cdots, z_K\}$—latent (or hidden) variables.
|Z|=K, has to be specified a priori.



- graphical model representation.
- generative process for each of the N documents.
- Nw: number of words in document d.
- Each word w has associated a latent topic z from which is generated.
- Shaded circles: observed variables,

PLSA – Generative process

- select a document d with probability P(d).
- for each word w_i,i $\in$ {1,⋯ ,N} in document d_n:
  Select a topic z_i from a multinomial conditioned as P(z|d_n).
  Select a word w_i from a multinomial conditioned as P(w|z_i).

Assuming
- bag-of-words model the joint distribution of the observed data factorize as a product:

$$P(\mathcal{D}, \mathcal{W}) = \prod_{(d,w)} P(d, w).$$

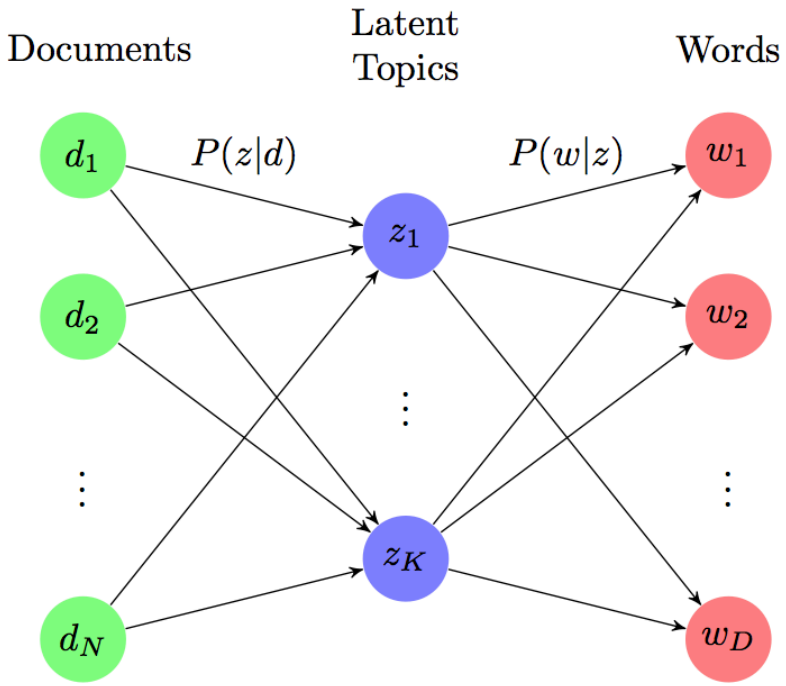
- Conditional independence

$$P(w|d) = \sum_{z \in \mathcal{Z}} P(w|z)P(z|d)$$

$$P(w, d) = \sum_{z \in \mathcal{Z}} P(z)P(d|z)P(w|z).$$

PLSA – mixture model

$$P(w|d) = \sum_{z \in \mathcal{Z}} P(w|z)P(z|d)$$



The general structure of pLSA model.

- intermediate layer of latent topics links documents to words
- each document is a mixture of topics weighted by the probability P(z|d)
- each word expresses a topic with probability P(w|z).

$$L = \prod_{(d,w)} P(w|d) = \prod_{d \in \mathcal{D}} \prod_{w \in \mathcal{W}} P(w|d)^{n(d,w)}$$

n(d, w) frequency of word w in d

PLSA – Log Likelihood Maximization

- Parameters can be estimated with Likelihood Maximization
- Find values maximizing predictive probability for observed word occurrences.
- predictive probability of pLSA mixture: *P(w|d)*, so the objective function is:

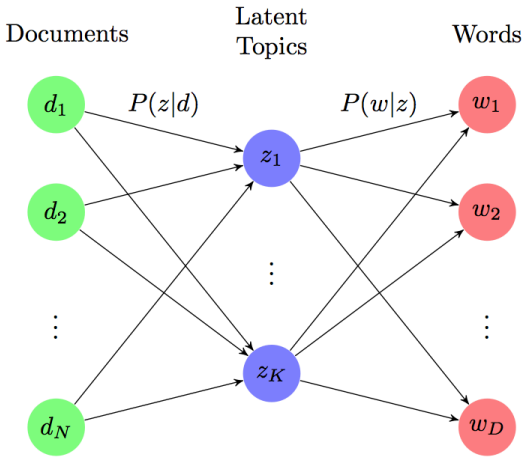
$$L = \prod_{(d,w)} P(w|d) = \prod_{d \in \mathcal{D}} \prod_{w \in \mathcal{W}} P(w|d)^{n(d,w)}$$

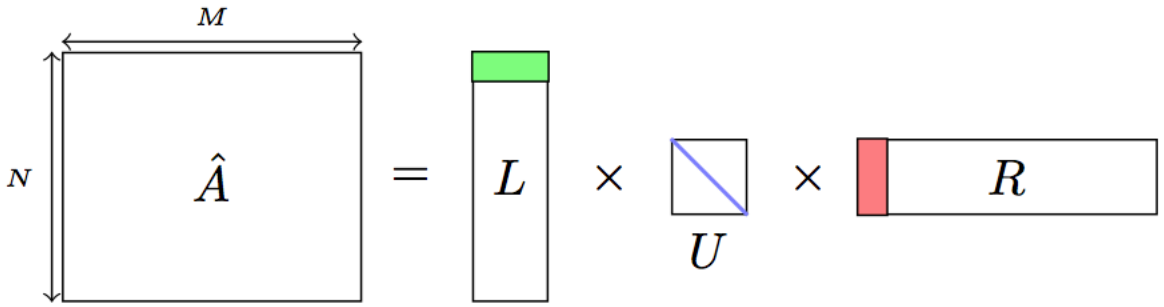
n(d, w) frequency of word w in d

Can be solved with Expectation-Maximization (EM) algorithm for the log-likelihood:

$$\mathcal{L} = \log L = \sum_{d \in \mathcal{D}} \sum_{w \in \mathcal{W}} n(d,w) \cdot \log \sum_{z \in \mathcal{Z}} P(w|z) P(z|d).$$

PLSA – as Matrix Decomposition





A: document-term matrix.

L: document probabilities P (d|z).

U: diagonal matrix - prior probabilities of the topics P (z).

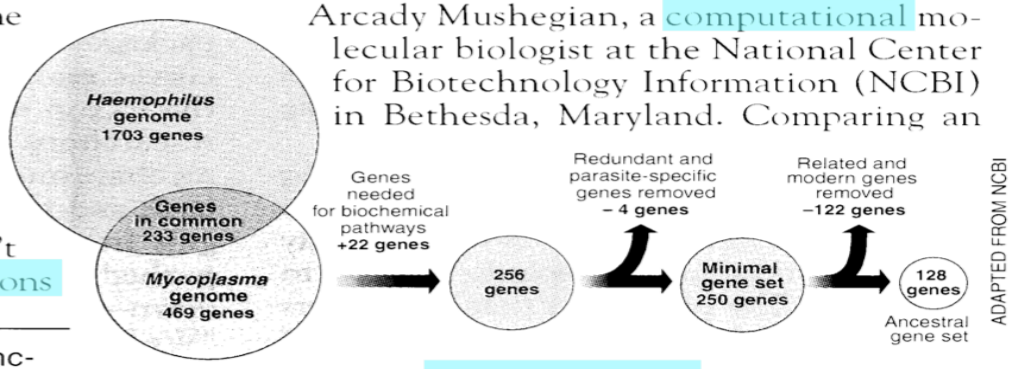
R: word probability P (w|z).

Latent Dirichlet Allocation

Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK—How many genes does an organism need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

Although the numbers don't match precisely, those predictions "are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an

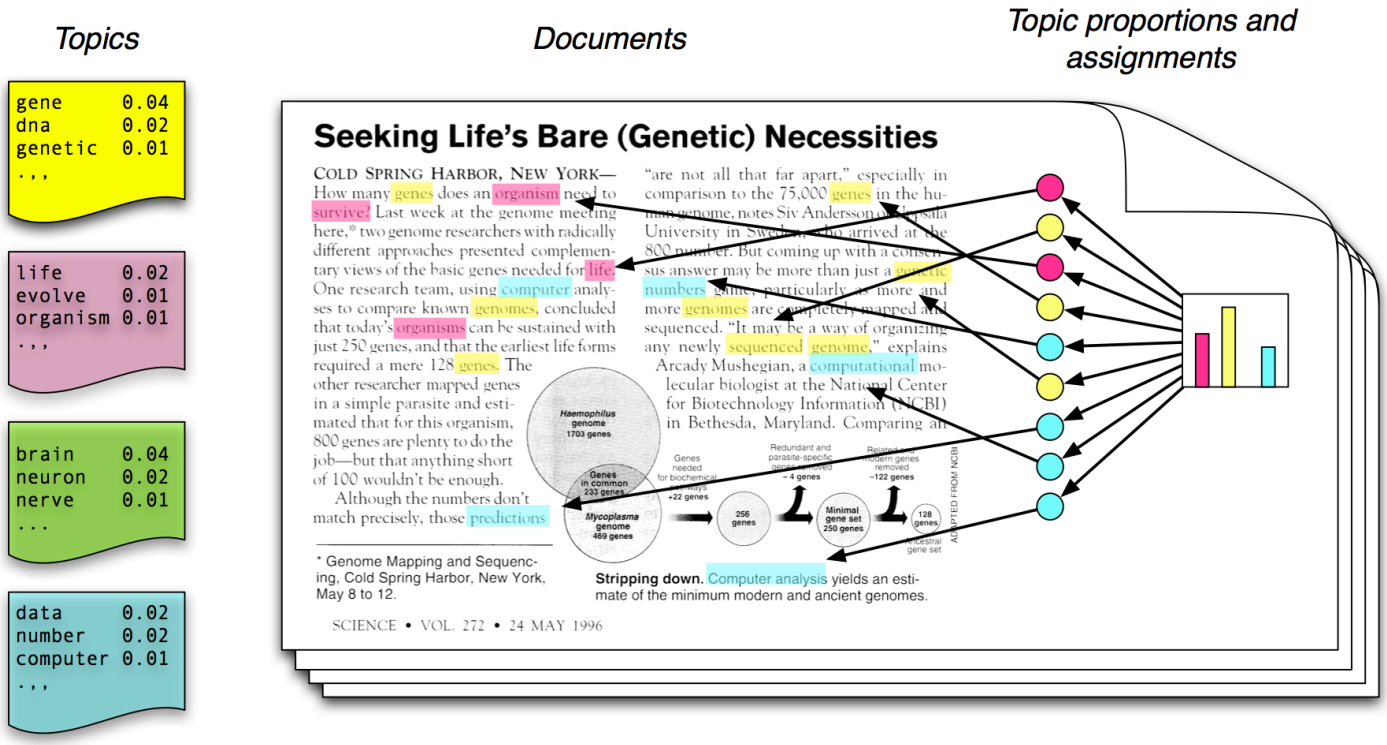


Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

SCIENCE • VOL. 272 • 24 MAY 1996

Latent Dirichlet Allocation

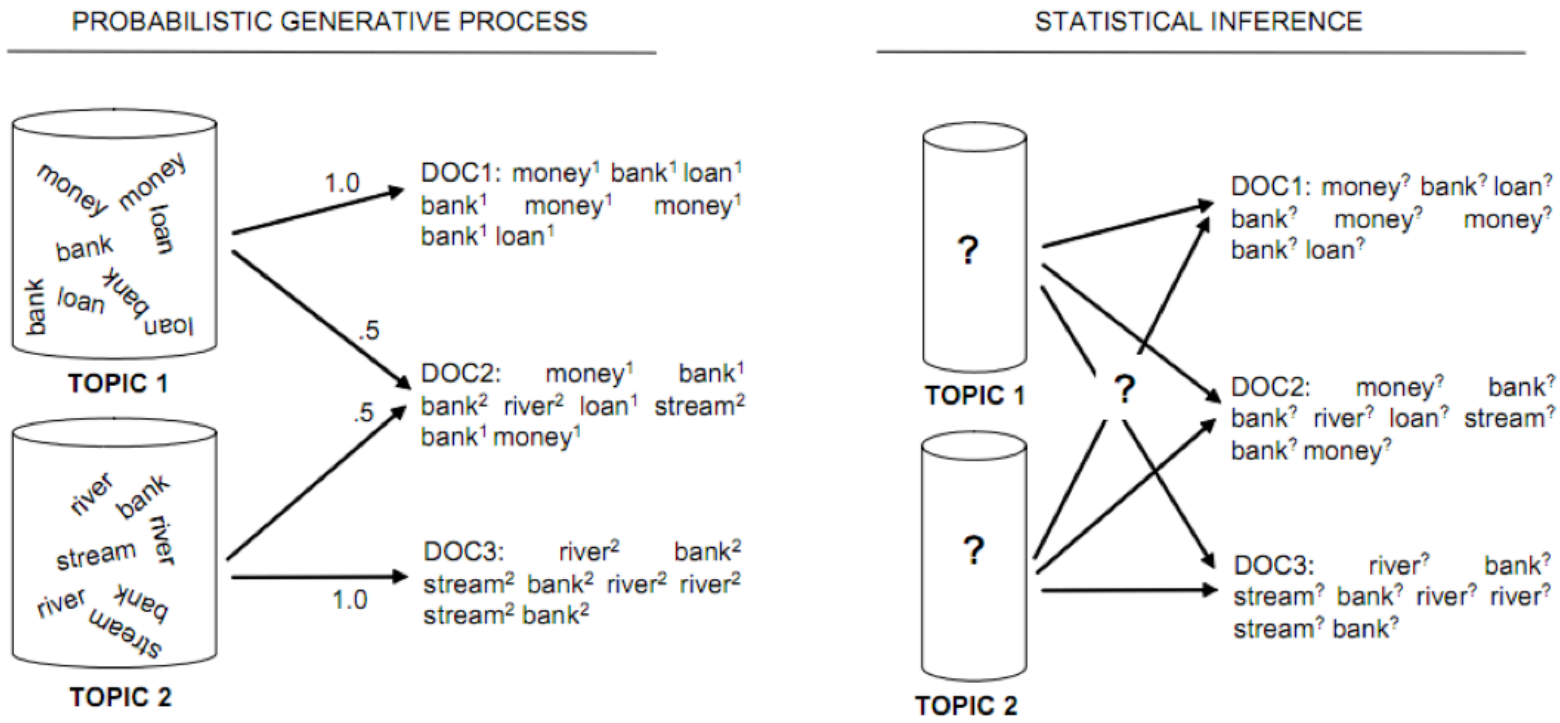


Topics: distribution over words

Documents: mixtures of corpus-wide topics

Words drawn probabilistically from one of those topics

Latent Dirichlet Allocation

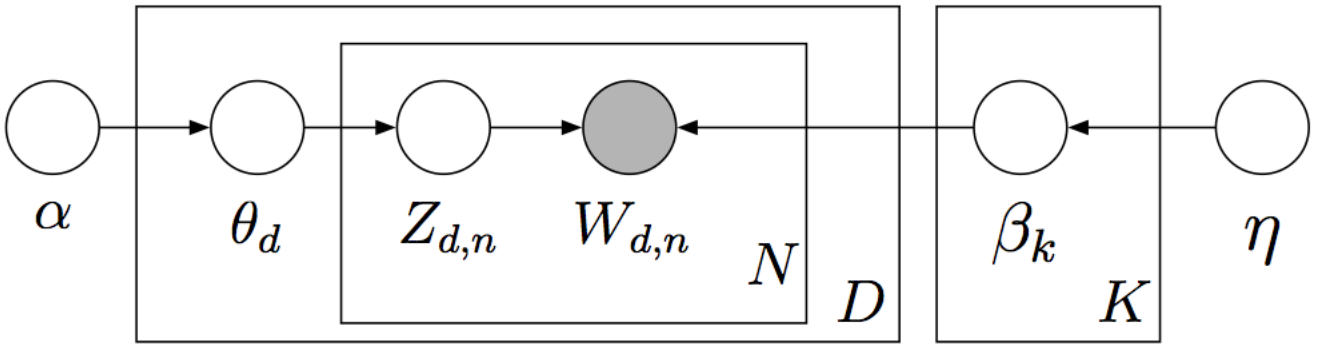


Latent Dirichlet Allocation: Towards a Deeper Understanding, Colorado Reed, 2012

LDA

K be the number of topics, **V** the size of the vocabulary,
α a positive K-vector, and **η** a scalar
Dirichlet Uniform Distribution
- DirV ((α), V-dimensional Dirichlet, DirK(η), a K dimensional symmetric Dirichlet

The hidden topical structure represented by hidden random variables:
- **β**: the K-topics distribution in the collection – probability of each topic in the collection
- **θ:** per-document topic proportions (NxK)–for each document the portion of each topic
- **z**: word - topic assignments



"Topic Models", David M. Blei, John D. Lafferty

LDA – Generative process

#words distribution over topics
For each topic
    draw a distribution over the words $\beta_{\varkappa} \sim Dir_V(\eta)$

#document-topic proportion & words topic assignment
For each document
  draw a vector of topic proportions $\theta_{\delta} \sim Dir_V(\alpha)$
  For each word
    draw a topic assignment $Z_{d,n} \sim Mult(\theta_{\delta}),\ Z_{d,n}\ in\ \{1,\ldots,K\}$
    draw a word $W_{d,n} \sim Mult(\beta_{Zd,n}),\ W_{d,n}\ \iota\nu\ \{1,\ldots,V\}$

LDA – fitting a posterior distribution

- joint distribution over the observed and hidden random variables. - hidden topic decomposition: *posterior distribution* of the hidden variables given the *D* observed documents $w_{1:D}$,

$$p(\vec{\theta}_{1:D}, z_{1:D,1:N}, \vec{\beta}_{1:K} \mid w_{1:D,1:N}, \alpha, \eta) = \frac{p(\vec{\theta}_{1:D}, \vec{z}_{1:D}, \vec{\beta}_{1:K} \mid \vec{w}_{1:D}, \alpha, \eta)}{\int_{\vec{\beta}_{1:K}} \int_{\vec{\theta}_{1:D}} \sum_{\vec{z}} p(\vec{\theta}_{1:D}, \vec{z}_{1:D}, \vec{\beta}_{1:K} \mid \vec{w}_{1:D}, \alpha, \eta)}$$

- this posterior can be thought as: given the observed corpus, the posterior is a distribution of the hidden variables that generated it.
LDA posterior intractable

- as hidden variables are dependent when conditioned on data.
- Specifically, this dependence yields difficulty in computing the denominator

  as one must sum over all configurations of the interdependent topic assignment variables $z_{1:N}$.

LDA – Posterior inference with variational inference

Mean field variational inference

- approximate an intractable posterior distribution over hidden variables, with a simpler distribution containing free *variational parameters*.
- parameters are fit: approximation is close to the true posterior.
- In contrast to true posterior, mean field variational distribution for LDA variables are *independent* of each other, each governed by a different variational parameter:

$$q(\vec{\theta}_{1:D}, z_{1:D,1:N}, \vec{\beta}_{1:K}) = \prod_{k=1}^{K} q(\vec{\beta}_k \mid \vec{\lambda}_k) \prod_{d=1}^{D} \left(q(\vec{\theta}_{dd} \mid \vec{\gamma}_d) \prod_{n=1}^{N} q(z_{d,n} \mid \vec{\phi}_{d,n}) \right)$$

- Each hidden variable is described by a distribution over its type:
- each topic $\beta_{1:K}$ by a *V*-Dirichlet distribution λ_k
- topic proportions θ by a K -Dirichlet distribution γ_d
- topic assignment $Z_{D,n}$ by a *K*-multinomial distribution $\phi_{d,n}$.

LDA – Posterior inference with variational inference

We fit its variational parameters to minimize the Kullback-Leibler (KL) divergence to the true posterior:

$$\arg \min_{\vec{\gamma}_{1:D}, \vec{\lambda}_{1:K}, \vec{\phi}_{1:D,1:N}} \mathrm{KL}(q(\vec{\theta}_{1:D}, z_{1:D,1:N}, \vec{\beta}_{1:K}) || p(\vec{\theta}_{1:D}, z_{1:D,1:N}, \vec{\beta}_{1:K} \mid w_{1:D,1:N}))$$

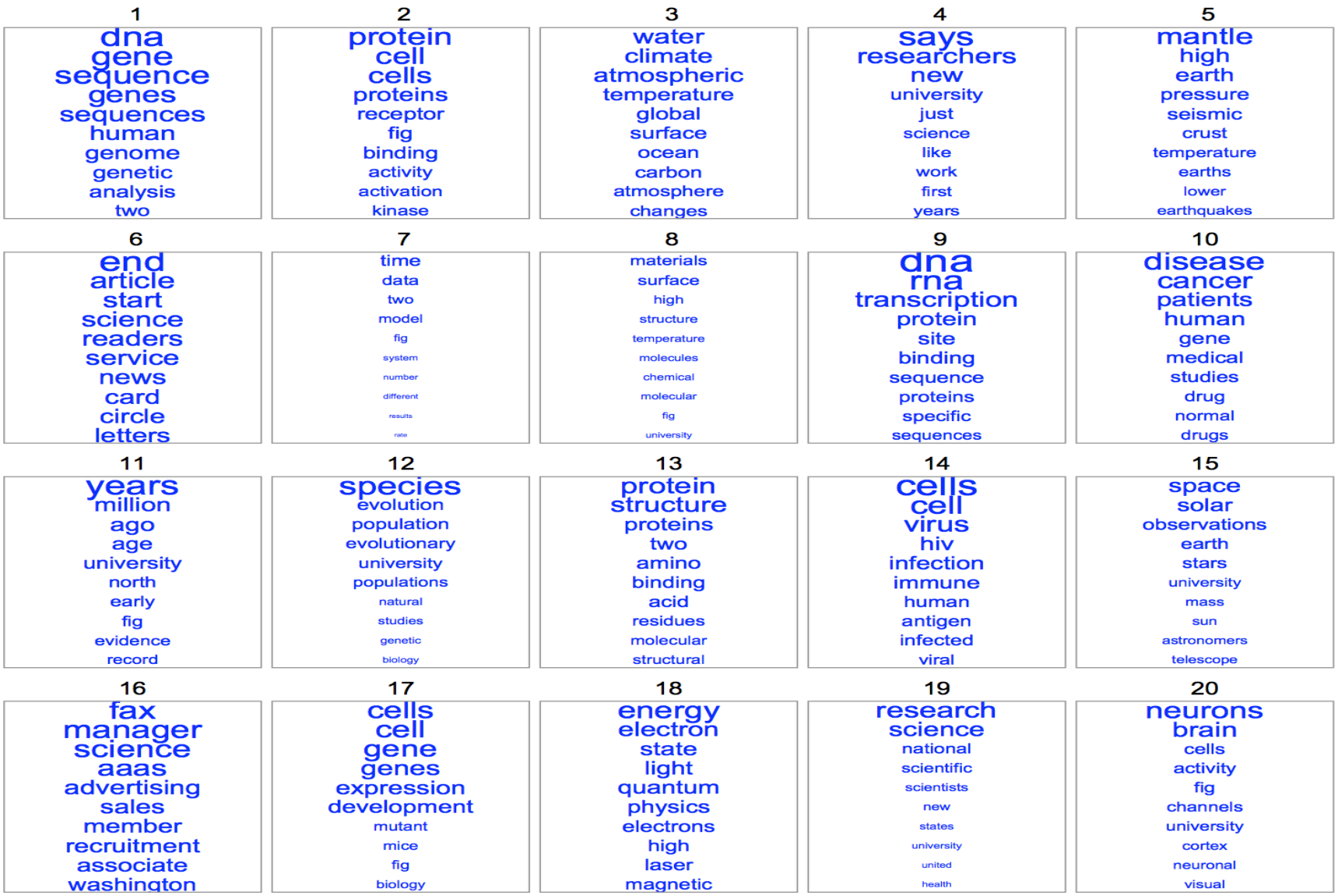
Aim to maximize the *likelihood* of the data under the model:

$$\mathcal{L} = \sum_{k=1}^{K} \mathrm{E}[\log p(\vec{\beta}_k \mid \eta)] + \sum_{d=1}^{D} \mathrm{E}[\log p(\vec{\theta}_d \mid \vec{\alpha})] + \sum_{d=1}^{D} \sum_{n=1}^{N} \mathrm{E}[\log p(Z_{d,n} \mid \vec{\theta}_d)]$$
$$+ \sum_{d=1}^{D} \sum_{n=1}^{N} \mathrm{E}[\log p(w_{d,n} \mid Z_{d,n}, \vec{\beta}_{1:K})] + \mathrm{H}(q),$$

H the entropy and all expectations are taken with respect to the variational distribution

Solution is obtained via an Expectation Maximization process.

Examples of LDA topics inference



References

- “Latent Dirichlet Allocation: Towards a Deeper Understanding Colorado Reed January 2012
- D. Blei. Introduction to probabilistic topic models. Communications of the ACM, 2011.
- Probabilistic Topic Models, David M. Blei, Department of Computer Science Princeton University, September 2, 2012
- Probabilistic Latent Semantic Analysis, Dan Oneaţă
- Thomas Hofmann. Probabilistic latent se- mantic indexing. In Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR ’99, pages 50–57, New York, NY, USA, 1999. ACM

Outline.

Topic modelling

Advanced keyword extraction and summarization

- Gow visualization & summarization - gowis – ACL 2016
- Heuristics and Submodularity based keyword extraction – EMNLP 2016 + EACL 2017
- keyword extraction for summarization - off line – IEEE-ICASSP(submitted)
- Abstractive summary (fillipova + openpaas)

Advanced GoW topics

- Collection level graph weights - tw-icw (under-submission)
- Shortest-Path Graph Kernels for Document Similarity -
- Graph based regularization for text classification – EMNLP2016

GoWvis

https://safetyapp.shinyapps.io/GoWvis/

A method for solution of systems of linear algebraic equations with m-dimensional lambda matrices. A system of linear algebraic e
matrices is considered. The proposed method of searching for the solution of this system lies in reducing it to a numerical system c

Parameters

Text | Graph | building | mining

Window size

2 3 16

2 4 6 8 10 12 14 16

Build on processed text?

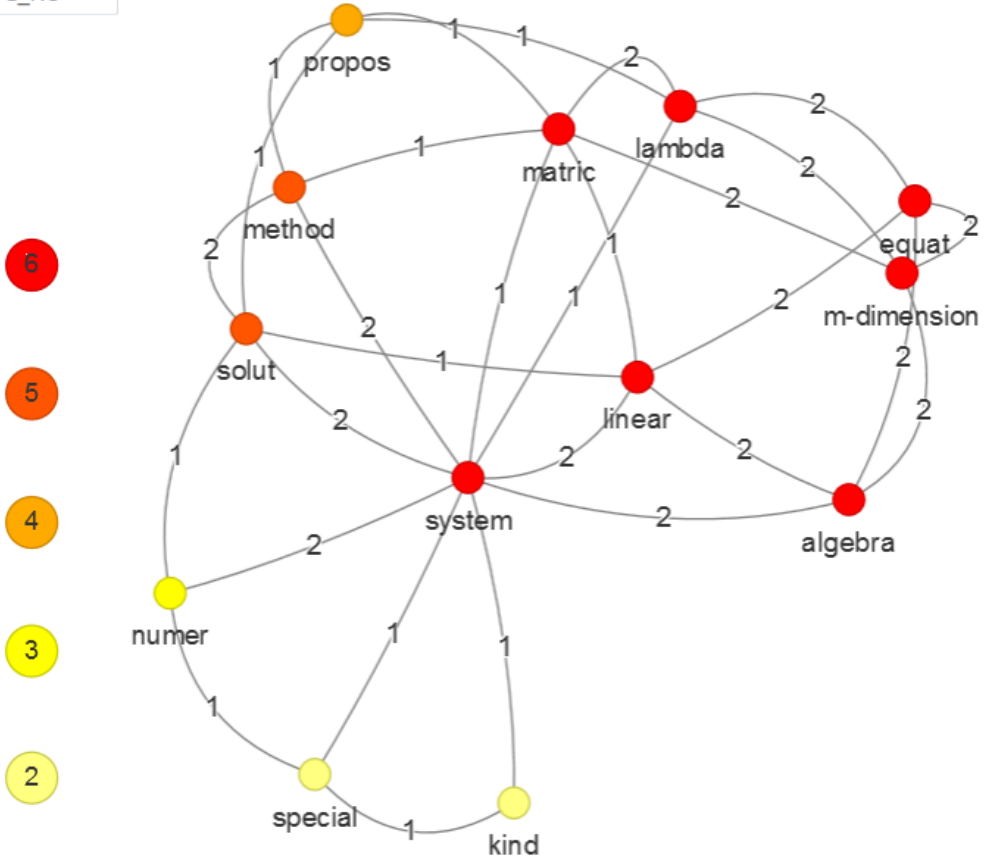
Yes

Overspan sentences?

Yes

Color

Select by core_no



GoWvis

https://safetyapp.shinyapps.io/GoWvis/

- Builds a graph-of-words and displays an interactive representation of any text pasted by the user
- Allows the user to tune many parameters:
  - text pre-processing (stopword removal, …)
  - graph building (window size, …)
  - graph mining (node ranking and community detection algorithms, …)
- Extracts keyphrases and generates a summary of the input text
- Built in R Shiny with the visNetwork library
- Presented to ACL 2016 as a demo paper

ACL 2016
AUGUST 7 – 12 | BERLIN, GERMANY

Outline.

Topic modelling

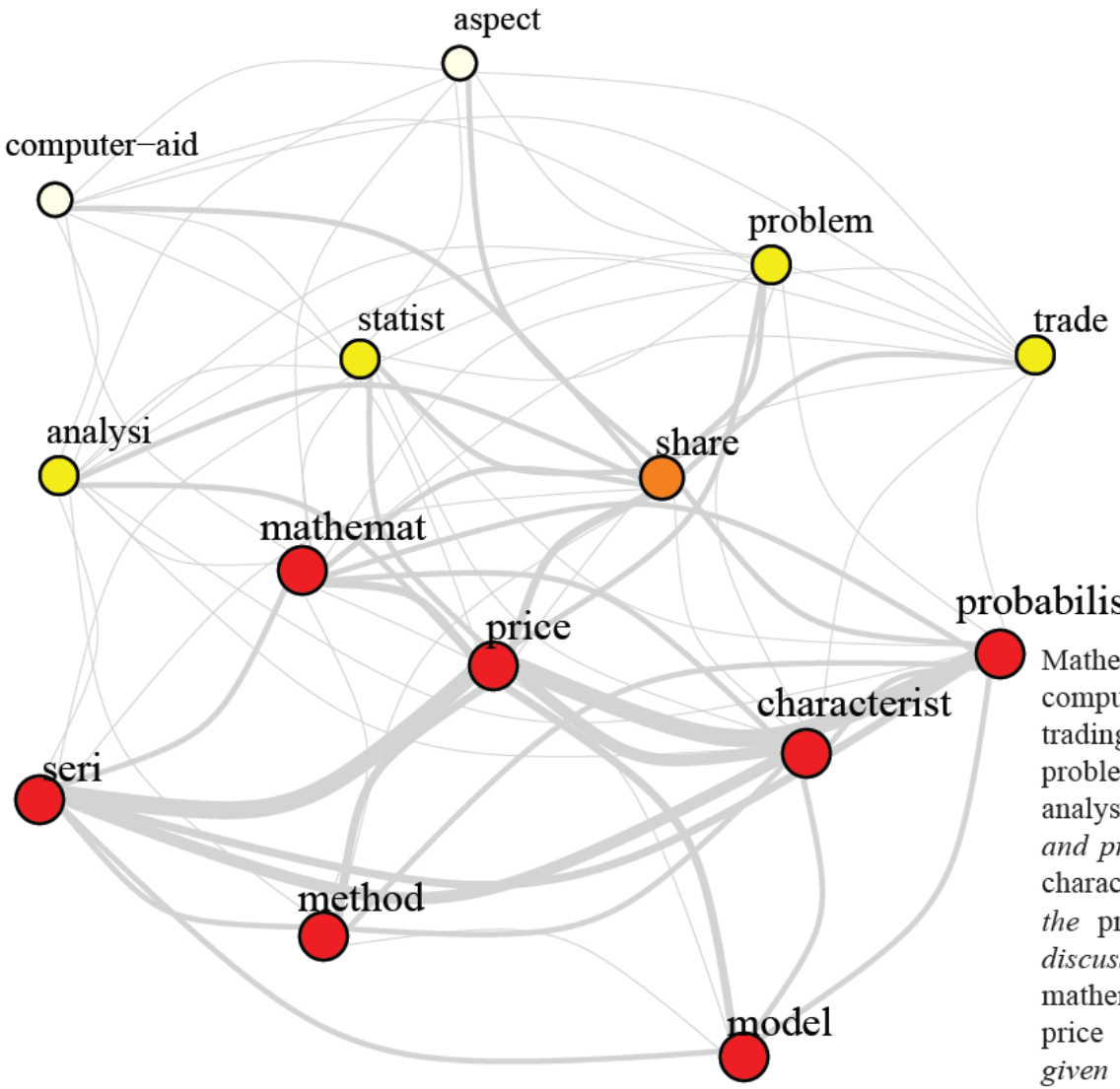
Advanced keyword extraction and summarization

- Gow visualization & summarization - gowis – ACL 2016
- Heuristics and Submodularity based keyword extraction – EMNLP 2016 + EACL 2017
- keyword extraction for summarization - off line – IEEE-ICASSP(submitted)
- Abstractive summary (fillipova + openpaas)

Advanced GoW topics

- Collection level graph weights - tw-icw (under-submission)
- Shortest-Path Graph Kernels for Document Similarity -
- Graph based regularization for text classification – EMNLP2016

Graph degeneracy for keyword extraction



Mathematical aspects *of* computer-aided share trading. *We consider* problems *of* statistical analysis *of* share prices *and propose* probabilistic characteristics *to describe the* price series. *We discuss three* methods *of* mathematical modelling *of* price series *with given* probabilistic characteristics.

Graph degeneracy for keyword extraction

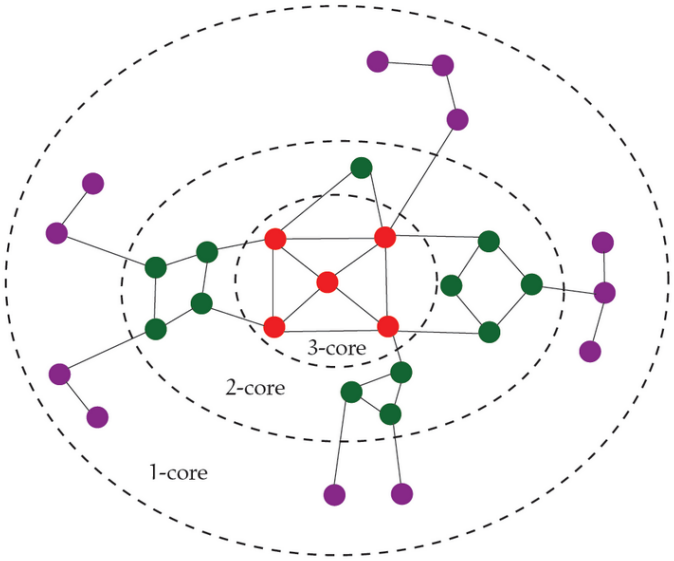
Graph-of-words:

- each unique term in the processed document is a node in the graph
- there is an edge between two nodes if they co-occur within a sliding window
- GoW can capture:
  - dependence strength (via edge weights)
  - term order (via edge directions)
- enables **graph theory** to be applied to **NLP**

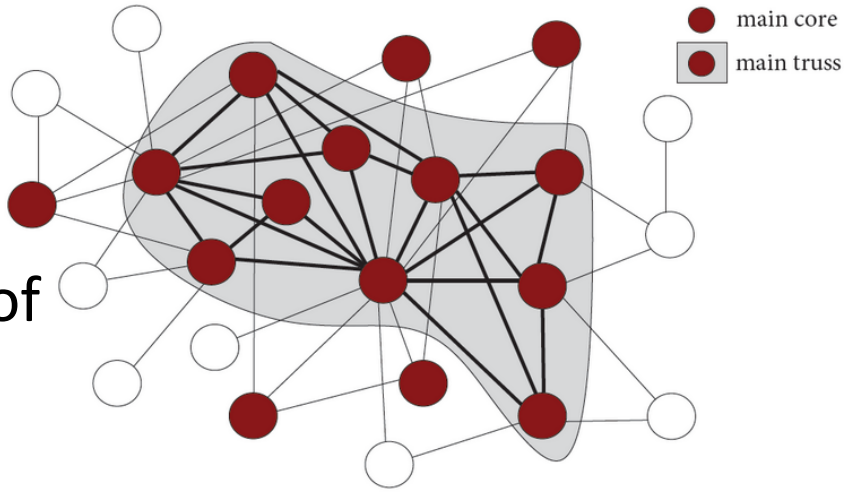
Graph degeneracy for keyword extraction

Graph degeneracy:

- **k-core** and **k-truss** algorithms [1,2]
- recursively prune out nodes of lowest degree
- k-core is based on traditional (un)weighted degree
- k-truss uses a **triangle**-based degree (core of the k-core)
- both can be **computed efficiently**
- -> hierarchy of nested subgraphs of ↘ size and ↗ cohesiveness



k-core decomposition

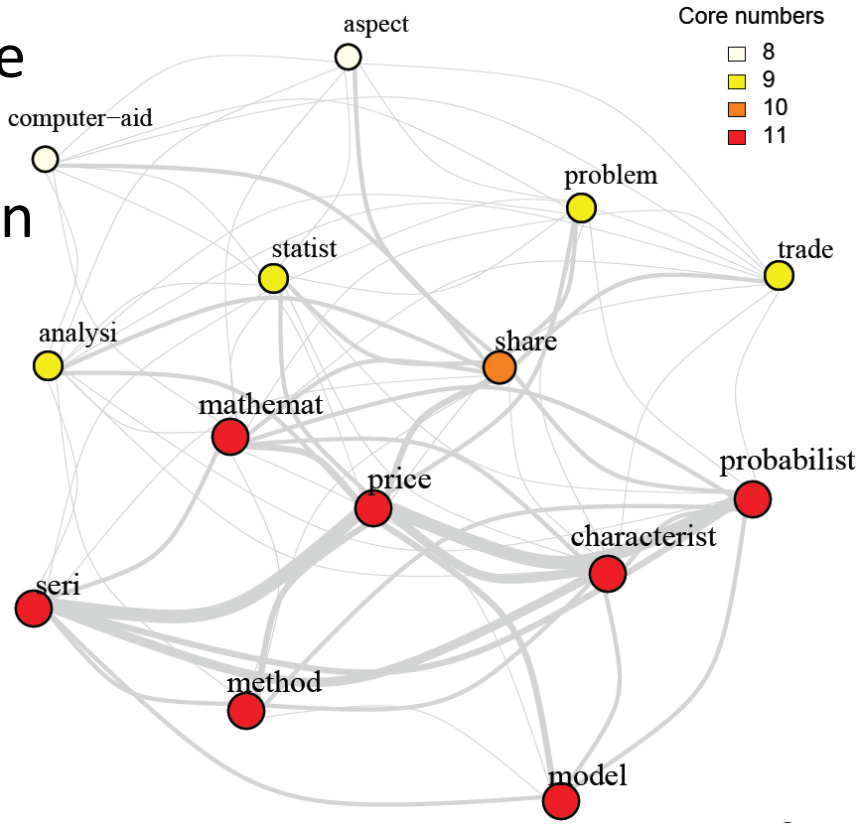


k-core versus *K*-truss

Graph degeneracy for keyword extraction

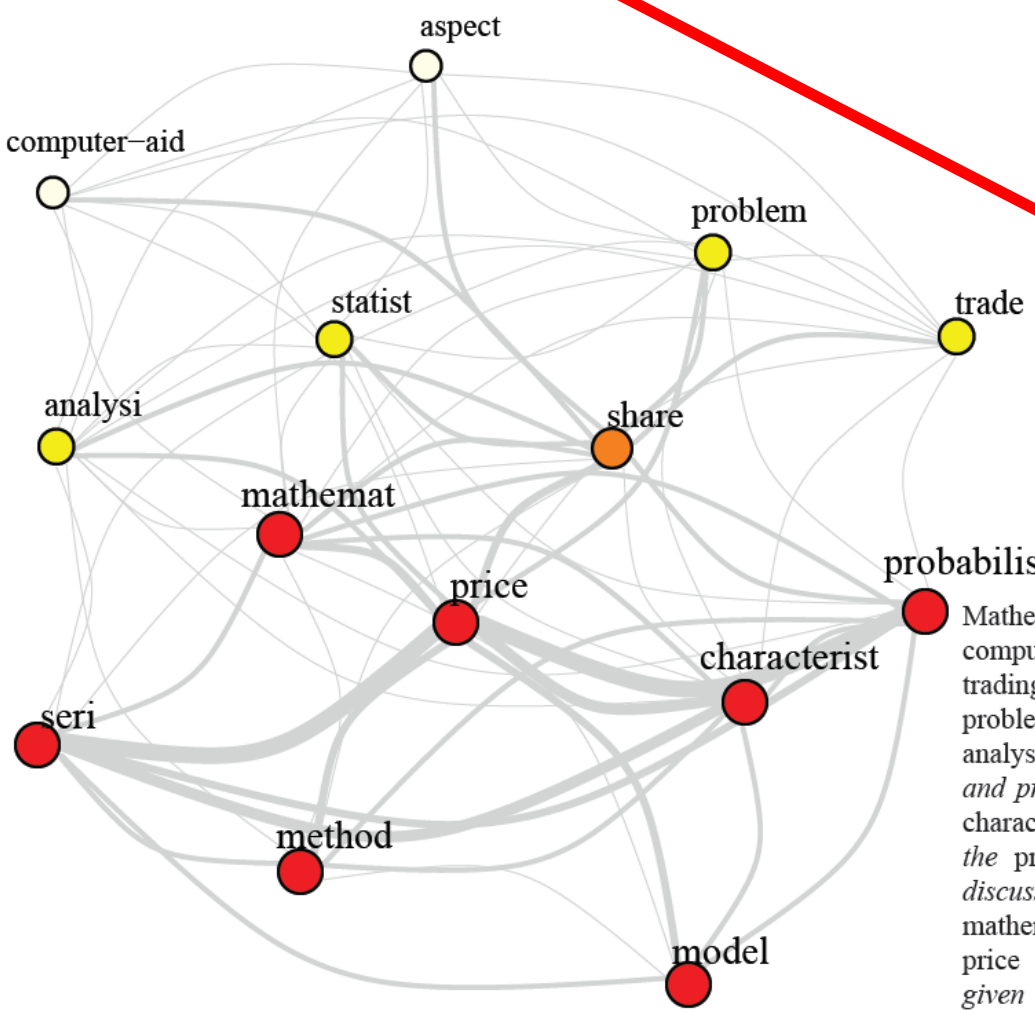
Graph degeneracy:

- in social networks, nodes part of the highest levels of the hierarchy are **better spreaders** than nodes high on **PageRank** [3]
- nodes with high truss numbers are **even more influential** than nodes with high core numbers [4]
- -> **spreading influence** may be a better “keywordness” metric than **prestige** (captured by PageRank)



Graph degeneracy for keyword extraction

Retaining the **top level** like in [5] may be an appealing initial idea



Edge weights: 1, 2, 3, 4, 5

Core numbers: 8, 9, 10, 11

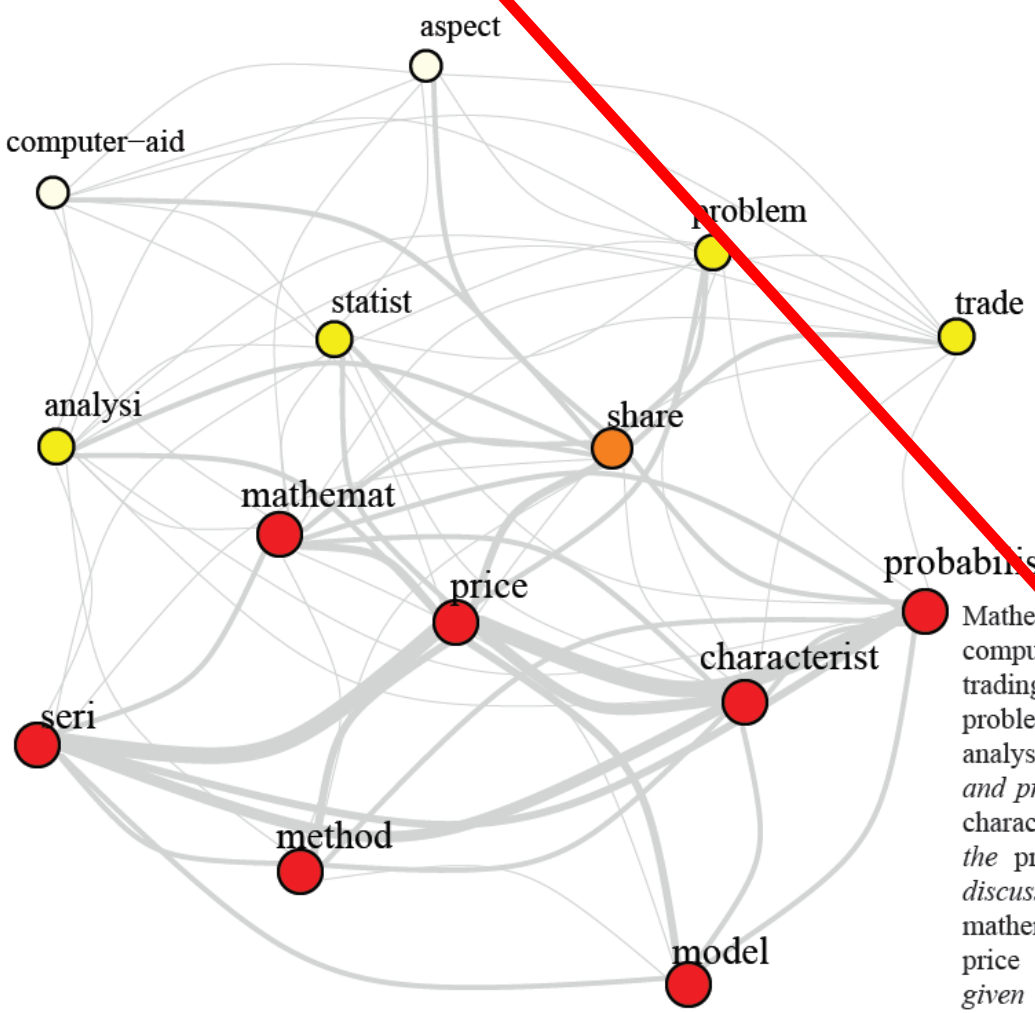
Mathematical aspects *of* computer-aided share trading. *We consider* problems *of* statistical analysis *of* share prices *and propose* probabilistic characteristics *to describe the* price series. *We discuss three* methods *of* mathematical modelling *of* price series *with given* probabilistic characteristics.

Core numbers			
	P	R	F1
MAIN	0.86	0.55	0.67
INF	**0.83**	**0.91**	**0.87**
DENS	0.88	0.64	0.74

mathemat		11
price		11
probabilist		11
characterist		11
seri		11
method		11
model	*MAIN*	11
share	*DENS*	10
trade		9
problem		9
statist		9
analysi	*INF*	9
aspect		8
computer-aid		8

Graph degeneracy for keyword extraction

But many keywords live **below the top** level -> good **precision**, poor **recall**



Edge weights
1
2
3
4
5

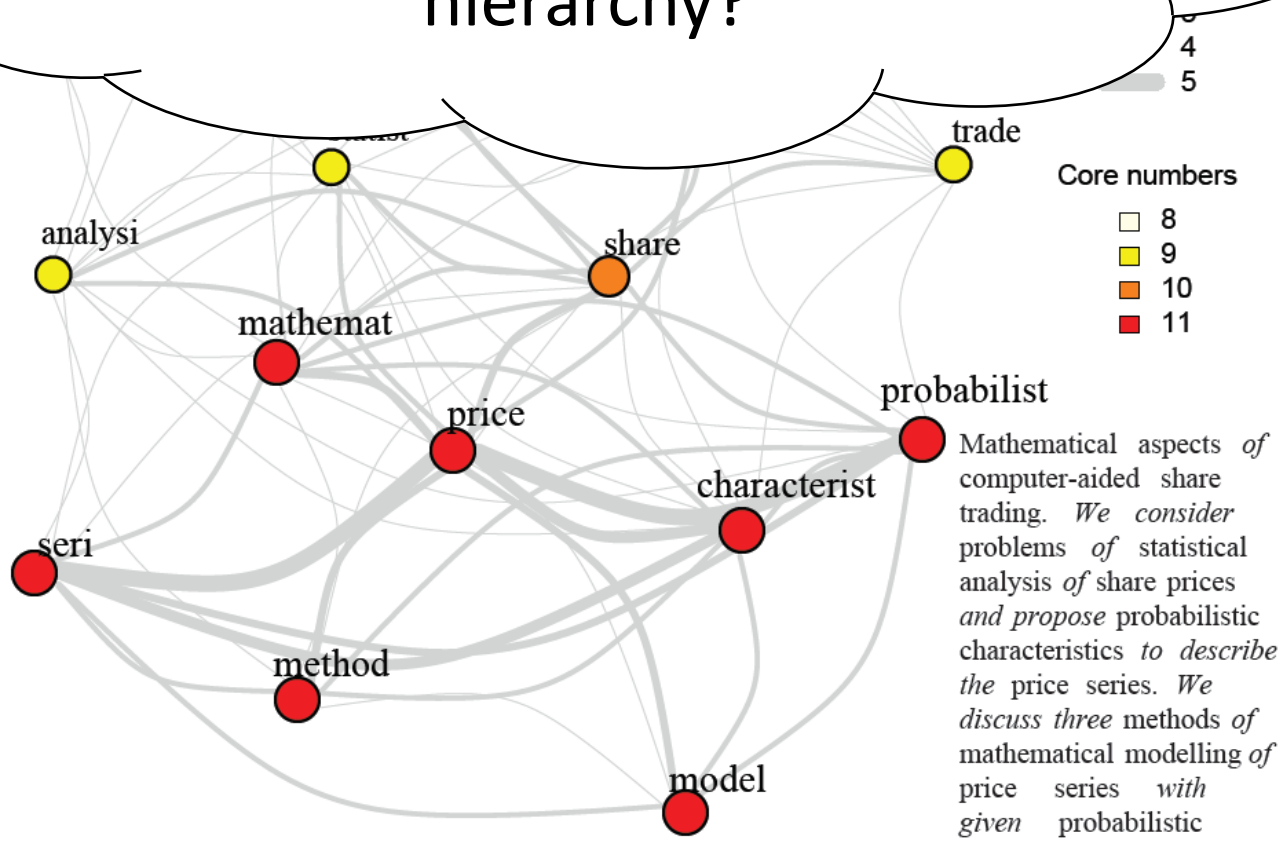
Core numbers
8
9
10
11

Mathematical aspects *of* computer-aided share trading. *We consider* problems *of* statistical analysis *of* share prices *and propose* probabilistic characteristics *to describe the* price series. *We discuss three* methods *of* mathematical modelling *of* price series *with given* probabilistic characteristics.

Core numbers			
	P	R	F1
MAIN	0.86	0.55	0.67
INF	**0.83**	**0.91**	**0.87**
DENS	0.88	0.64	0.74
mathemat			11
price			11
probabilist			11
characterist			11
seri			11
method			11
model		*MAIN*	11
share		*DENS*	10
trade			9
problem			9
statist			9
analysi		*INF*	9
aspect			8
computer-aid			8

Graph degeneracy for keyword extraction

How to automatically select the best level in the hierarchy?

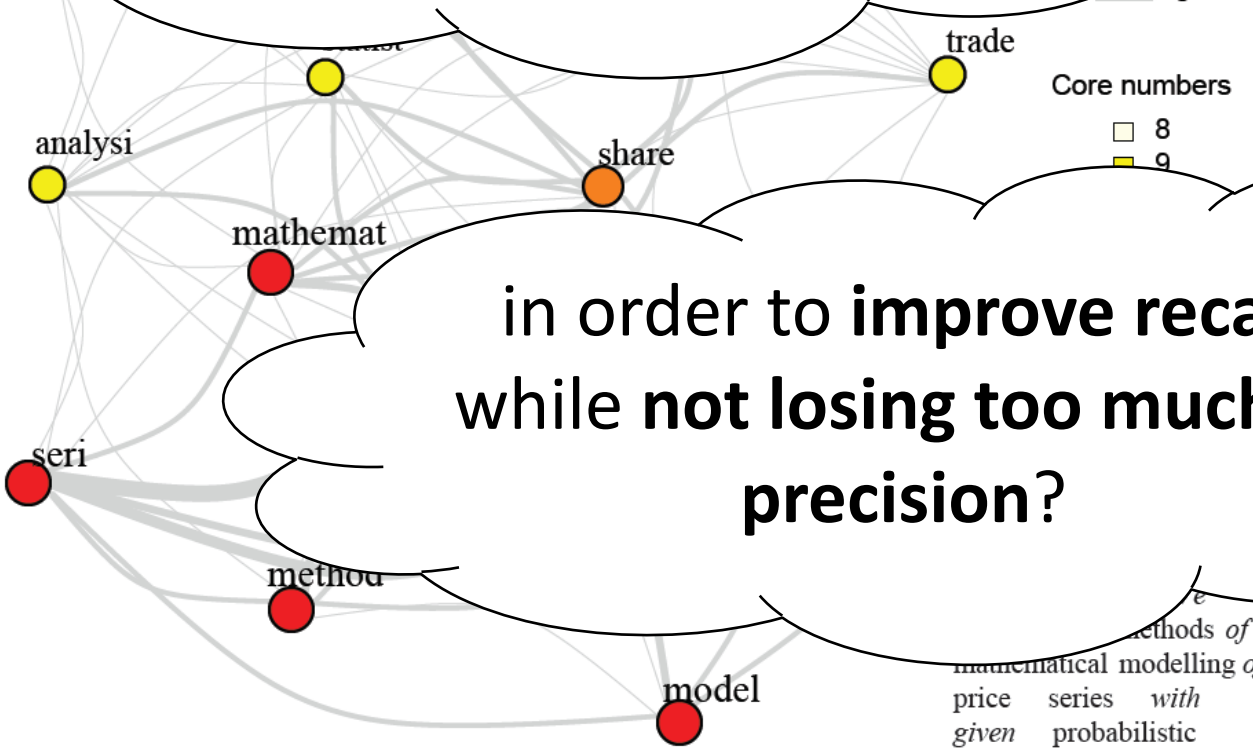


Mathematical aspects *of* computer-aided share trading. *We consider* problems *of* statistical analysis *of* share prices *and propose* probabilistic characteristics *to describe the* price series. *We discuss three* methods *of* mathematical modelling *of* price series *with given* probabilistic characteristics.

Core numbers			
	P	R	F1
MAIN	0.86	0.55	0.67
INF	**0.83**	**0.91**	**0.87**
DENS	0.88	0.64	0.74
mathemat			11
price			11
probabilist			11
characterist			11
seri			11
method			11
model		*MAIN*	11
share		*DENS*	10
trade			9
problem			9
statist			9
analysi		*INF*	9
aspect			8
computer-aid			8

Graph degeneracy for keyword extraction

How to automatically select the best level in the hierarchy?

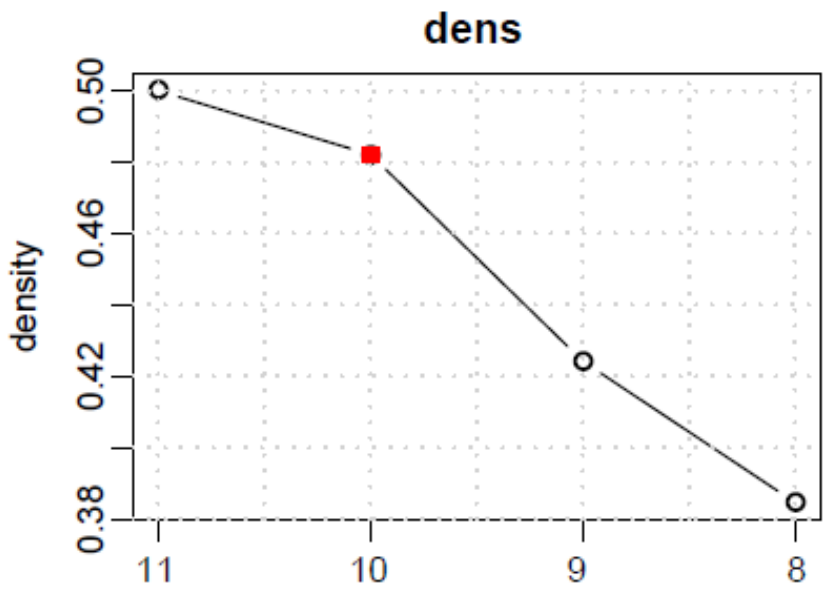


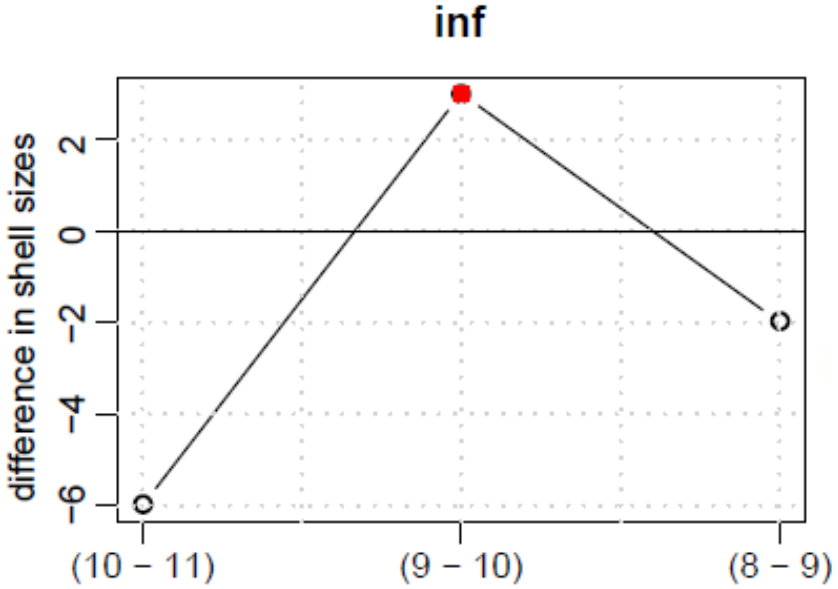
in order to **improve recall** while **not losing too much** in **precision**?

Core numbers			
	P	R	F1
MAIN	0.86	0.55	0.67
INF	**0.83**	**0.91**	**0.87**
DENS	0.88	0.64	0.74
mathemat			11
price			11
probabilist			11
statist			9
analysi	*INF*		9
aspect			8
computer-aid			8

...ethods *of* mathematical modelling *of* price series *with given* probabilistic characteristics.

Graph degeneracy for keyword extraction



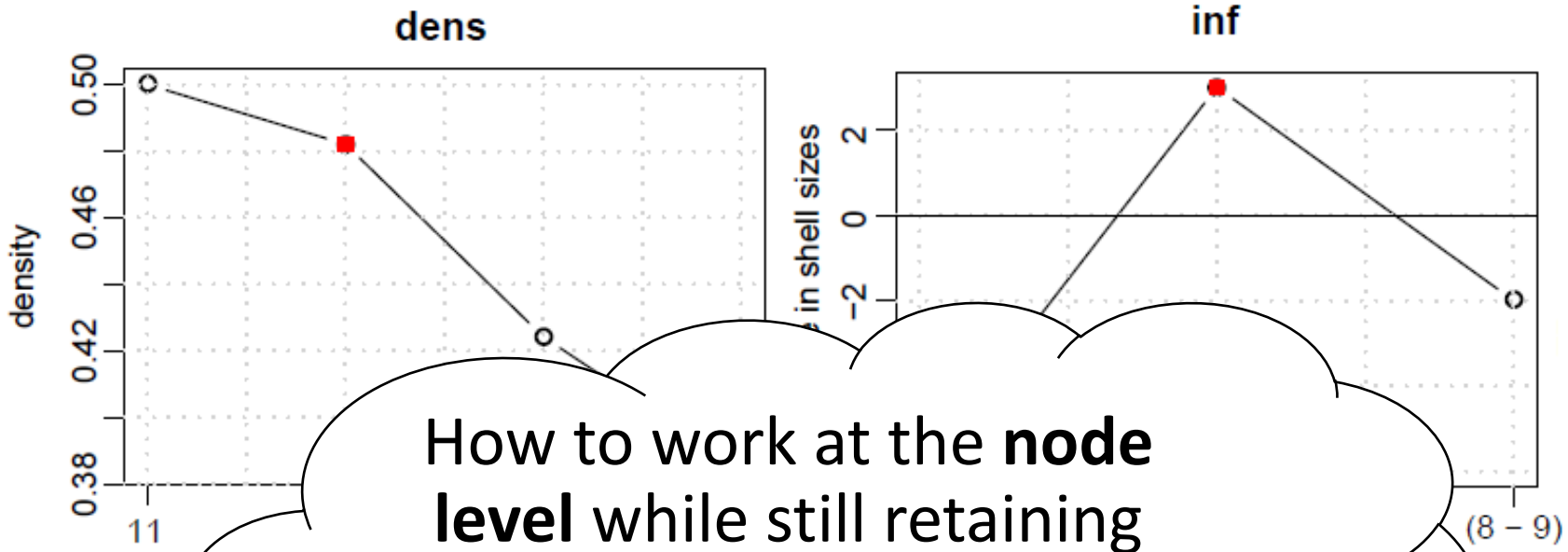


Heuristics:

- ***dens*****:** go down the hierarchy until a drop in k-core (or truss) density is observed, i.e., as long as the desirable cohesiveness properties are kept
- ***inf*****:** go down the hierarchy as long as the shells increase in size (starting at the *main – 1* level)

-> problem: both methods work at the **subgraph level** -> lack flexibility for large graphs (adding an entire group of nodes or not)

Graph degeneracy for keyword extraction



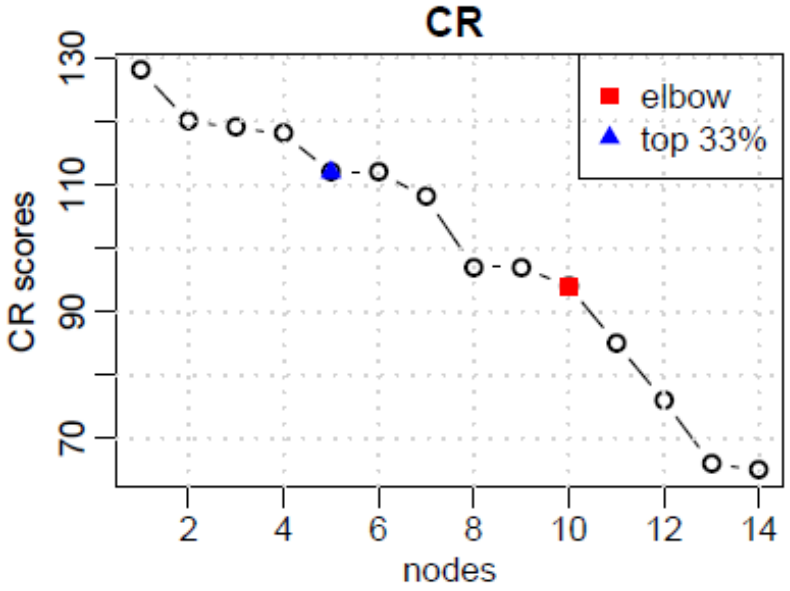
How to work at the **node level** while still retaining the valuable cohesiveness information captured by degeneracy?

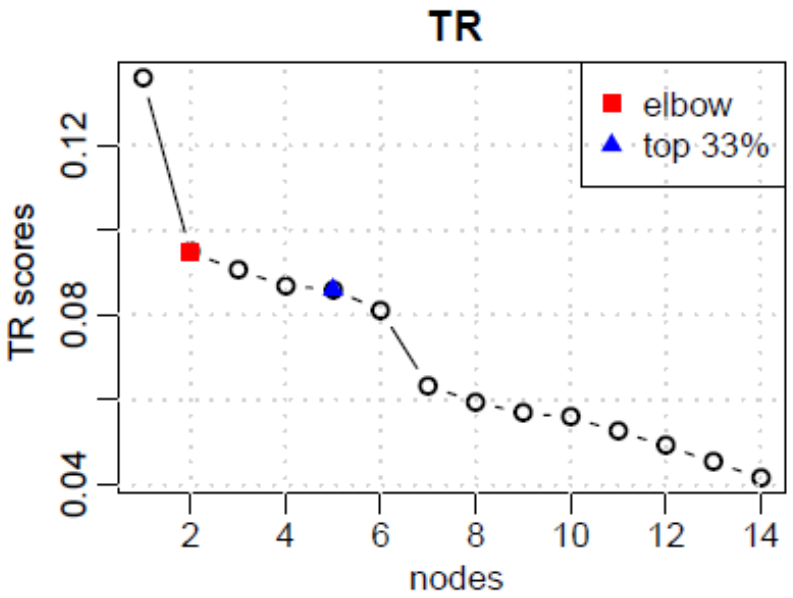
Heuristi

- ***dens***: go … nsity is observe
- ***inf***: go down th… se in size (starting at the *main – 1* level)

-> problem: both methods work at the **subgraph level** -> lack flexibility for large graphs (adding an entire group of nodes or not)

Graph degeneracy for keyword extraction





CoreRank (CR):

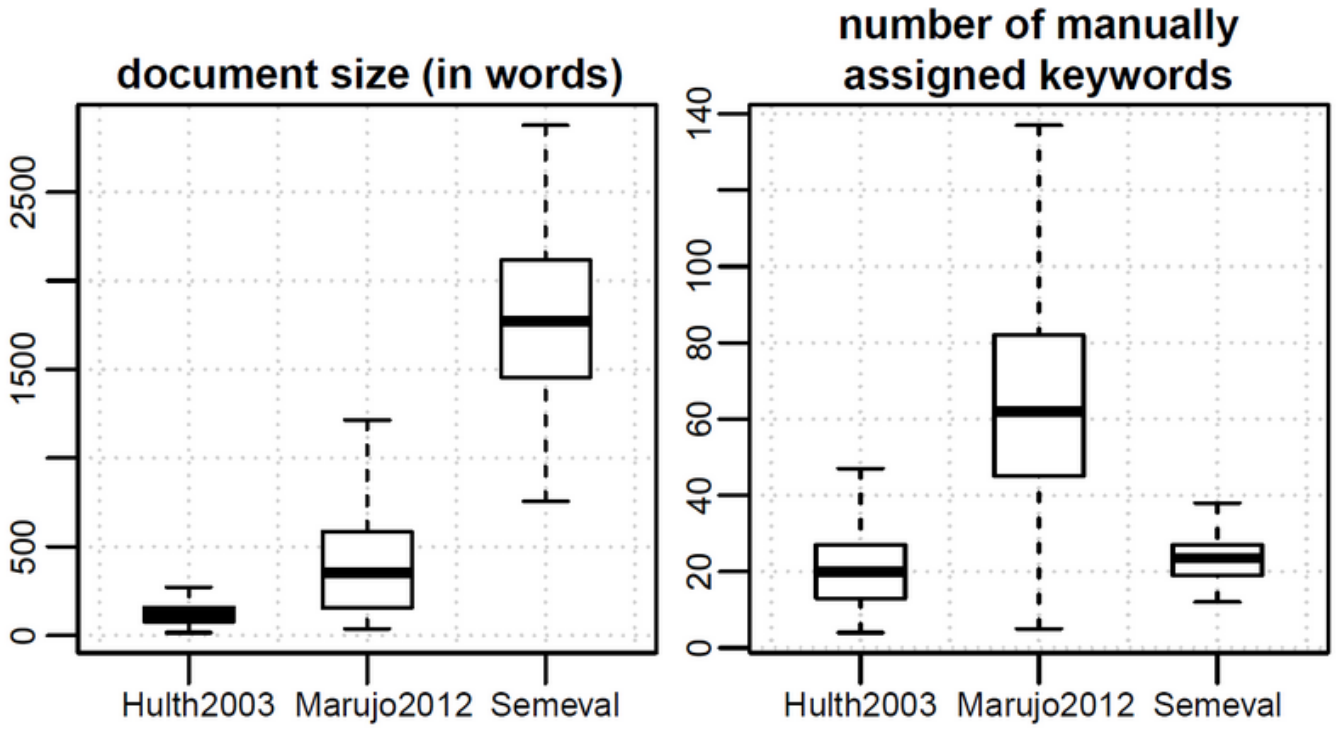
- assign to each node the sum of the core (or truss) numbers of its **neighbors**

-> granularity is **much finer** and allows for **much flexible selection**

-> comparable to applying **PageRank** to the graph-of-words (aka TextRank or TR, [6]) but taking into account **cohesiveness** concerns rather than individual **prestige** only

Heuristics: nodes can be selected based on the **elbow** or **top p%** method

Graph degeneracy for keyword extraction



Datasets:

- Hulth2003: 500 abstracts from the Inspec physics & engineering database
- Marujo2012: 450 web news stories covering 10 different topics
- Semeval: 100 scientific papers from the ACM

Graph degeneracy for keyword extraction

	precision	recall	F1-score
dens	**48.79**	**72.78**	**56.09***
inf	48.96	72.19	55.98*
CRP	61.53	38.73	45.75
CRE	65.33	37.90	44.11
main†	51.95	54.99	50.49
TRP†	65.43	41.37	48.79
TRE†	71.34	36.44	45.77

Hulth2003, K-truss, $W = 11$. *stat. sign. ($p < 0.001$) w.r.t. all baselines†

	precision	recall	F1-score
dens	**47.62**	**71.46**	**52.94***
inf	53.88	57.54	49.10*
CRP	54.88	36.01	40.75
CRE	63.17	25.77	34.41
main†	64.05	34.02	36.44
TRP†	55.96	36.48	41.44
TRE†	65.50	21.32	30.68

Marujo2012, k-core, $W = 13$. *stat. sign. ($p < 0.001$) w.r.t. all baselines†

	precision	recall	F1-score
dens	8.44	79.45	15.06
inf	17.70	65.53	26.68
CRP	**49.67**	**32.88**	**38.98***
CRE	25.82	58.80	34.86
main†	25.73	49.61	32.83
TRP†	47.93	31.74	37.64
TRE†	33.87	46.08	37.55

Semeval, K-truss, $W = 20$. *stat. sign. ($p < 0.001$) w.r.t. *main*

Datasets:

- for small documents (i.e., small graphs), the **subgraph-level** heuristics significantly outperform main core retention (main) and TextRank (TRP, TRE)
- **recall** is **drastically improved**, **precision** is **maintained** (especially with ***inf***)
- for long documents (Semeval), the **node-level** heuristics are better. Especially, CoreRank with top p% retention (**CRP**) reaches best performance.

DaScim

Graph degeneracy for keyword extraction

	precision	recall	F1-score
dens	**48.79**	**72.78**	**56.09***
inf	48.96	72.19	55.98*
CRP	61.53	38.73	45.75
CRE	65.33	37.90	44.11
main†	51.95	54.99	50.49
TRP†	65.43	41.37	48.79
TRE†	71.34	36.44	45.77

Hulth2003, K-truss, $W = 11$. *stat. sign. ($p < 0.001$) w.r.t. all baselines†

	precision	recall	F1-score
dens	**47.62**	**71.46**	**52.94***
inf	53.88	57.54	49.10*
CRP	54.88	36.01	40.75
CRE	63.17	25.77	34.41
main†	64.05	34.02	36.44
TRP†	55.96	36.48	41.44
TRE†	65.50	21.32	30.68

Marujo2012, k-core, $W = 13$. *stat. sign. ($p < 0.001$) w.r.t. all baselines†

	precision	recall	F1-score
dens	8.44	79.45	15.06
inf	17.70	65.53	26.68
CRP	**49.67**	**32.88**	**38.98***
CRE	25.82	58.80	34.86
main†	25.73	49.61	32.83
TRP†	47.93	31.74	37.64
TRE†	33.87	46.08	37.55

Semeval, K-truss, $W = 20$. *stat. sign. ($p < 0.001$) w.r.t. *main*

Outline.

Topic modelling

Advanced keyword extraction and summarization

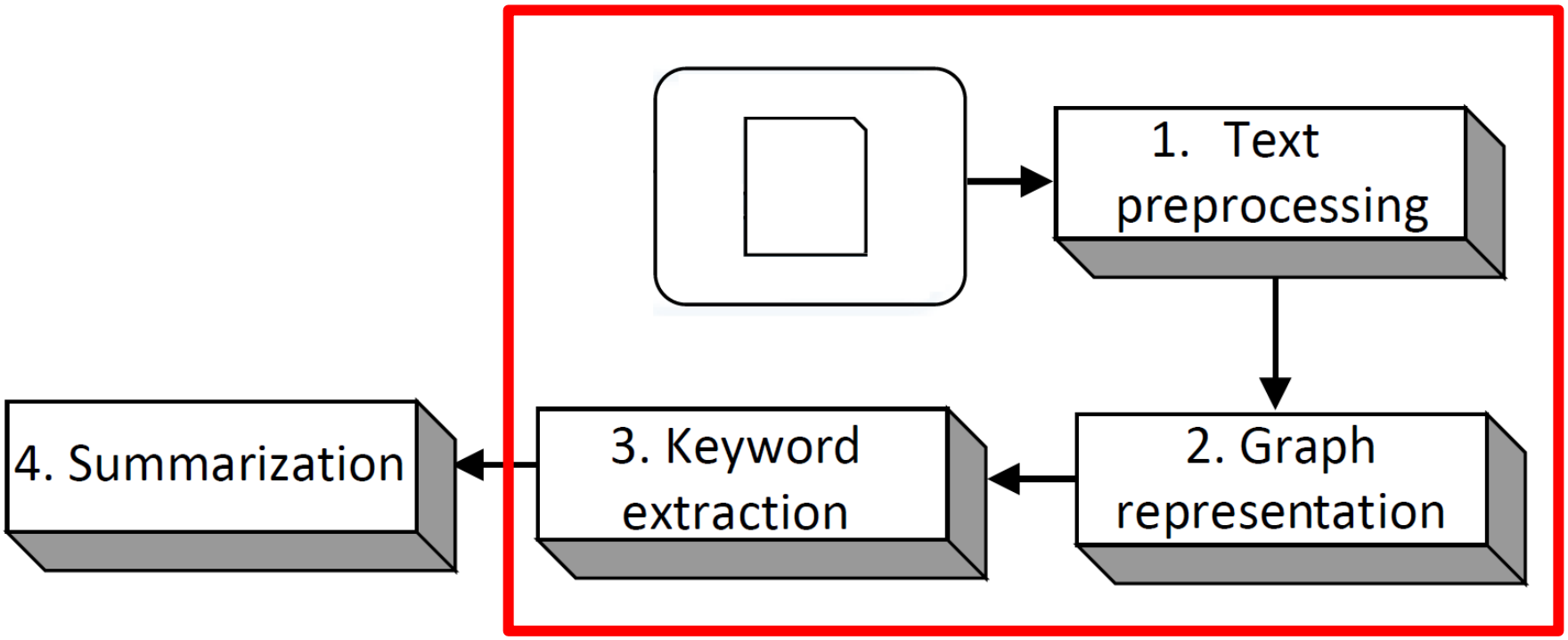
- Gow visualization & summarization - gowis – ACL 2016
- Heuristics and Submodularity based keyword extraction – EMNLP 2016
- keyword extraction for summarization - off line – IEEE-ICASSP(submitted)
- Abstractive summary (fillipova + openpaas)

Advanced GoW topics

- Collection level graph weights - tw-icw (under-submission)
- Shortest-Path Graph Kernels for Document Similarity -
- Graph based regularization for text classification – EMNLP2016

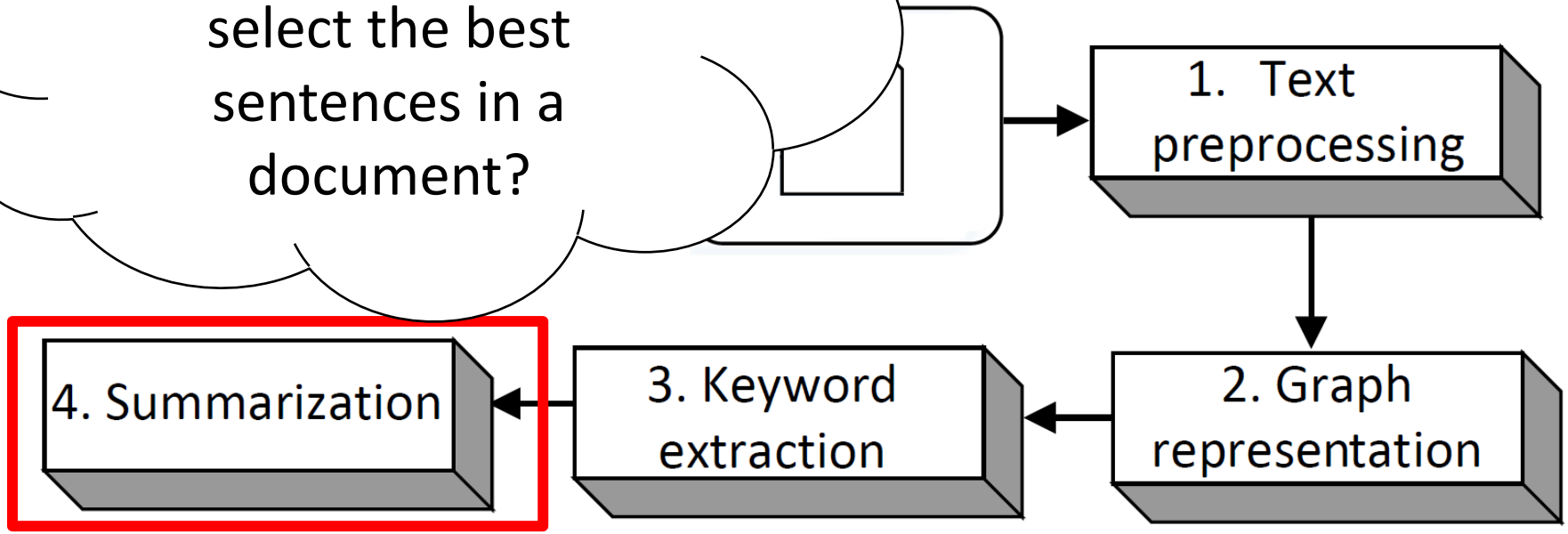
Extension to extractive document summarization

Same as before



Extension to extractive document rization

How to use keywords (and their scores) to select the best sentences in a document?



Extension to extractive document summarization

Generating a summary in an ***extractive*** way is akin to selecting the best sentences in the document under a ***budget constraint*** (max number of words allowed).

-> **Combinatorial optimization** task:

$$\arg\max_{S \subseteq V} F(S) \mid \sum_{v \in S} c_v \leq B$$

- S is a given summary (a subset of the set of sentences V)
- F is the objective function to maximize (measuring summary quality)
- C_v is the cost of sentence v (number of words it contains)
- B is the budget (in words)

Extension to extractive document summarization

$$\arg\max_{S \subseteq V} F(S) \mid \sum_{v \in S} c_v \leq B$$

- Solving this task is **NP-complete**
- But, [7] showed that if F is **non-decreasing** and **submodular**, a greedy algorithm can approach the best solution with factor (e – 1)/e
- At each step, the algorithm selects the sentence v that **maximizes**:

objective function gain → $\frac{F(G \cup v) - F(G)}{c_v^r}$

scaled cost →

- r is a tuning parameter

Extension to extractive document summarization

$$\arg\max_{S \subseteq V} F(S) \mid \sum_{v \in S} c_v \leq B$$

- The choice of F, the **summary quality** objective function, is what matters
- A good summary should cover all the important topics in the document, while not repeating itself:
  -> maximize **coverage**
  -> penalize **redundancy** (reward **diversity** to ensure monotonicity)

$$F(S) = L(S) + \lambda R(S)$$ see [7]

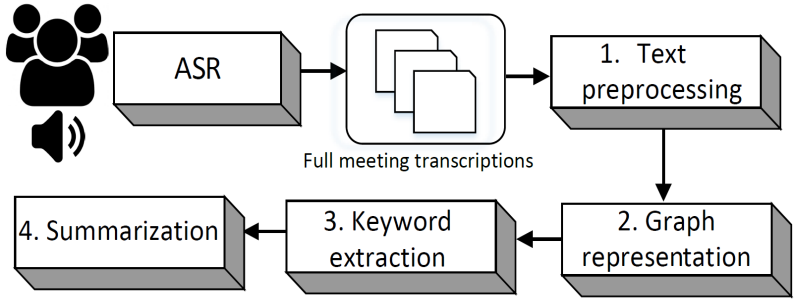
$$L(S) = \sum_{i \in S} n_i w_i \qquad R(S) = N_{keywords \in S} / N_{keywords}$$

weighted sum of the keywords contained in the summary

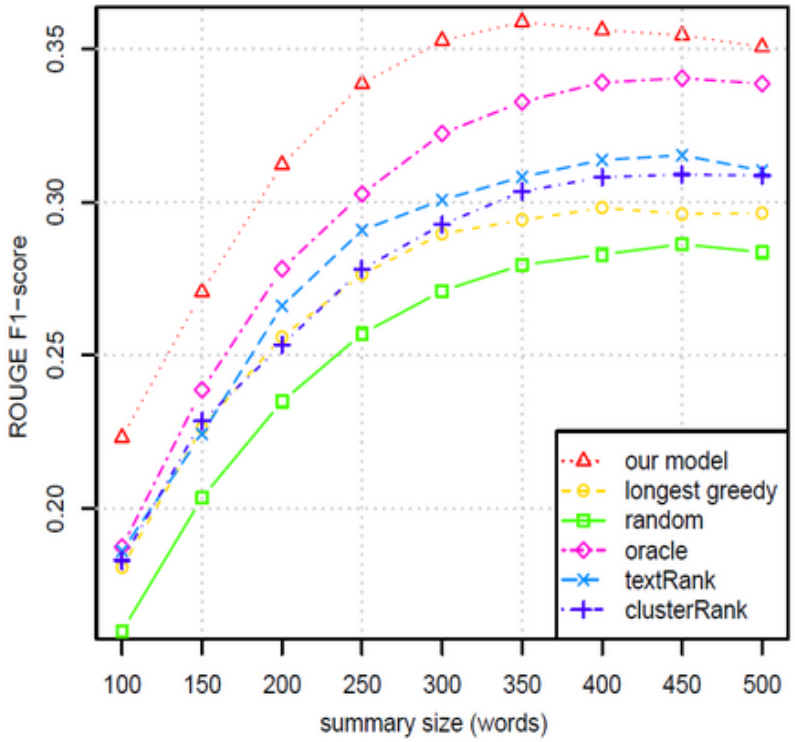
proportion of unique keywords contained

Extension to extractive document summarization

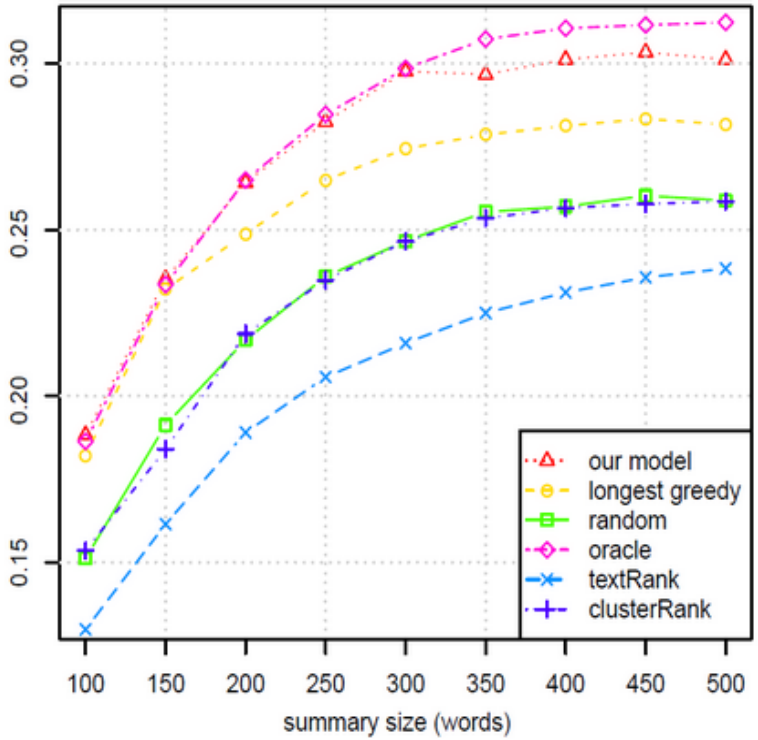
Tested for multiparty virtual meetings summarization:



AMI corpus



ICSI corpus



References

[1] Seidman, S. B. (1983). Network structure and minimum degree. *Social networks*, *5*(3), 269-287.

[2] Cohen, J. (2008). Trusses: Cohesive subgraphs for social network analysis. *National Security Agency Technical Report*, 16.

[3] Kitsak, M., Gallos, L. K., Havlin, S., Liljeros, F., Muchnik, L., Stanley, H. E., & Makse, H. A. (2010). Identification of influential spreaders in complex networks. *Nature physics*, *6*(11), 888-893.

[4] Malliaros, F. D., Rossi, M. E. G., & Vazirgiannis, M. (2016). Locating influential nodes in complex networks. *Scientific reports*, *6*.

[5] Rousseau, F., & Vazirgiannis, M. (2015, March). Main core retention on graph-of-words for single-document keyword extraction. In *European Conference on Information Retrieval* (pp. 382-393). Springer International Publishing.

[6] Mihalcea, R., & Tarau, P. (2004, July). TextRank: Bringing order into texts. Association for Computational Linguistics.

[7] Lin, H., & Bilmes, J. (2010, June). Multi-document summarization via budgeted maximization of submodular functions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (pp. 912-920). Association for Computational Linguistics.

[8] Lin, H., & Bilmes, J. (2011, June). A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1* (pp. 510-520). Association for Computational Linguistics.

Multi sentence compression/fusion

Setting: we are presented with a group of similar sentences (e.g., first sentence of each article on a Google News cluster). Each sentence contains important bits of information. Collectively, the sentences cover everything, but none single sentence 'gets it all'.

Goal: *fuse* the sentences into a single, compact one, that contains *as much information as possible* while being *fluent* and *grammatical*.

Method: many approaches can be used. However, it is possible to produce excellent results in a *fully unsupervised way*, with only a list of stopwords and a part-of-speech tagger.

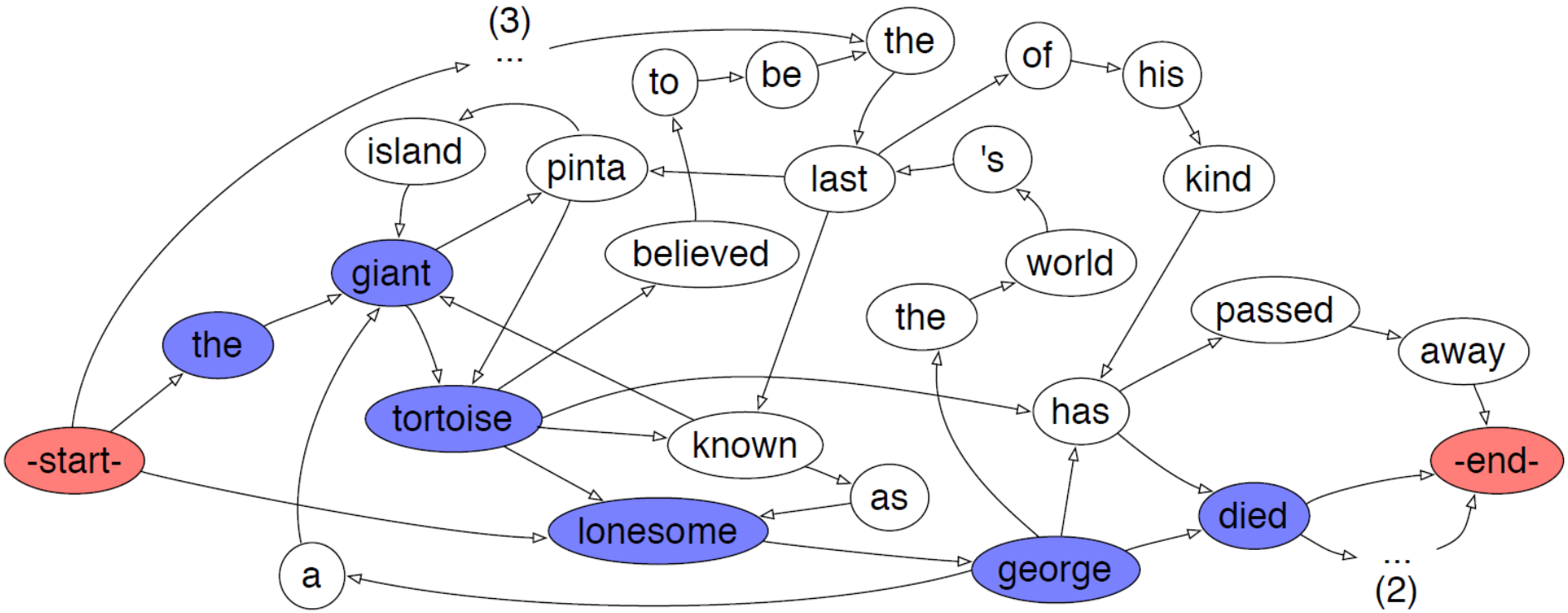
=> Multi-sentence compression: *Finding shortest paths in word graphs.* Katja Filippova (from Google), ACL 2010

Research at Google

Research Area(s)
Human-Computer Interaction
Machine Intelligence
Natural Language Processing

Word-graph sentence compression

1) Lonesome George, the world's last Pinta Island giant tortoise, has passed away
2) The giant tortoise known as Lonesome George died Sunday at the Galapagos National Park in Ecuador
3) He was only about a hundred years old, but the last known giant Pinta tortoise, Lonesome George, has passed away
4) Lonesome George, a giant tortoise believed to be the last of his kind, has died



Word-graph construction

Build a directed graph from the first sentence, with 'start' and 'end' nodes. Then, consider each word in the remaining sentences.

i. if the word is not a stopword, and if there is already a node in the graph for it (with same lowercased spelling and POS tag), and assuming that no word from the same sentence has already been mapped onto the node **=>** map word to the node

Otherwise:

ii. if the word is not a stopword, but there are more than one candidate in the graph or multiple occurrences of the word in the sentence

iii. if the word is a stopword

=> select the candidate which has larger overlap in context (preceding and following words in sentence and neighbors in the graph), or the node which has more words mapped onto it

Word-graph construction

Edge weights (the smaller the better):

$$w''(e_{i,j}) = \frac{w'(e_{i,j})}{\texttt{freq}(i) \times \texttt{freq}(j)} \qquad (1)$$

Where:

$$w'(e_{i,j}) = \frac{\texttt{freq}(i) + \texttt{freq}(j)}{\sum_{s \in S} \texttt{diff}(s,i,j)^{-1}} \qquad (2)$$

- *freq(i)* is the number of words that have been mapped to node i.
- *diff(s,i,j)* is the distance between word i and word j in sentence s

Intuition for (2): edges between strongly associated words are given more importance, taking into account the overall freq. of the nodes (edge freq. of 3 should count more if the edge connects 2 nodes with freq. 3 rather than with freq. >>3)
Connections between nodes between which there are multiple paths are also given more importance, proportionally to the lengths of the paths

Word-graph construction

Edge weights (the smaller the better):

$$w''(e_{i,j}) = \frac{w'(e_{i,j})}{\texttt{freq}(i) \times \texttt{freq}(j)} \quad (1)$$

Where:

$$w'(e_{i,j}) = \frac{\texttt{freq}(i) + \texttt{freq}(j)}{\sum_{s \in S} \texttt{diff}(s,i,j)^{-1}} \quad (2)$$

freq(i) is the number of words that have been mapped to node i.
diff(s,i,j) is the distance between word i and word j in sentence s

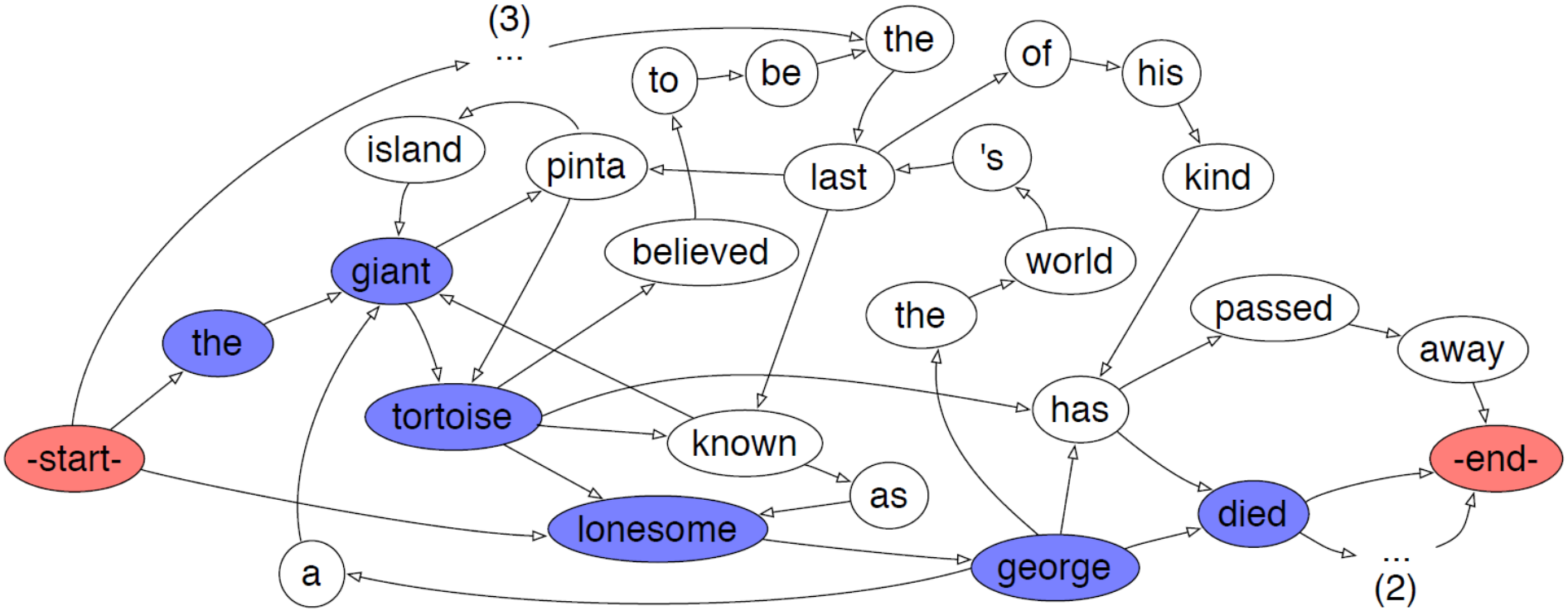
Intuition for (1): eq. (2) is a measure of **cohesion** between 2 words, but disregards the individual importance of the words => we need to take saliency into account. Edges connecting two important words are thus favored.

Path ranking and selection

- a K-shortest paths algorithm is applied on the graph to find the 50 paths with ***smallest*** edge weights
- all the paths which are shorter than eight words and do not contain a verb are **filtered out**
- the survivors are re-ranked by normalizing the total path weight over its length
- the path which has the **lightest average edge weight** is finally considered as the best compression

Examples

1) Lonesome George, the world's last Pinta Island giant tortoise, has passed away
2) The giant tortoise known as Lonesome George died Sunday at the Galapagos National Park in Ecuador
3) He was only about a hundred years old, but the last known giant Pinta tortoise, Lonesome George, has passed away
4) Lonesome George, a giant tortoise believed to be the last of his kind, has died



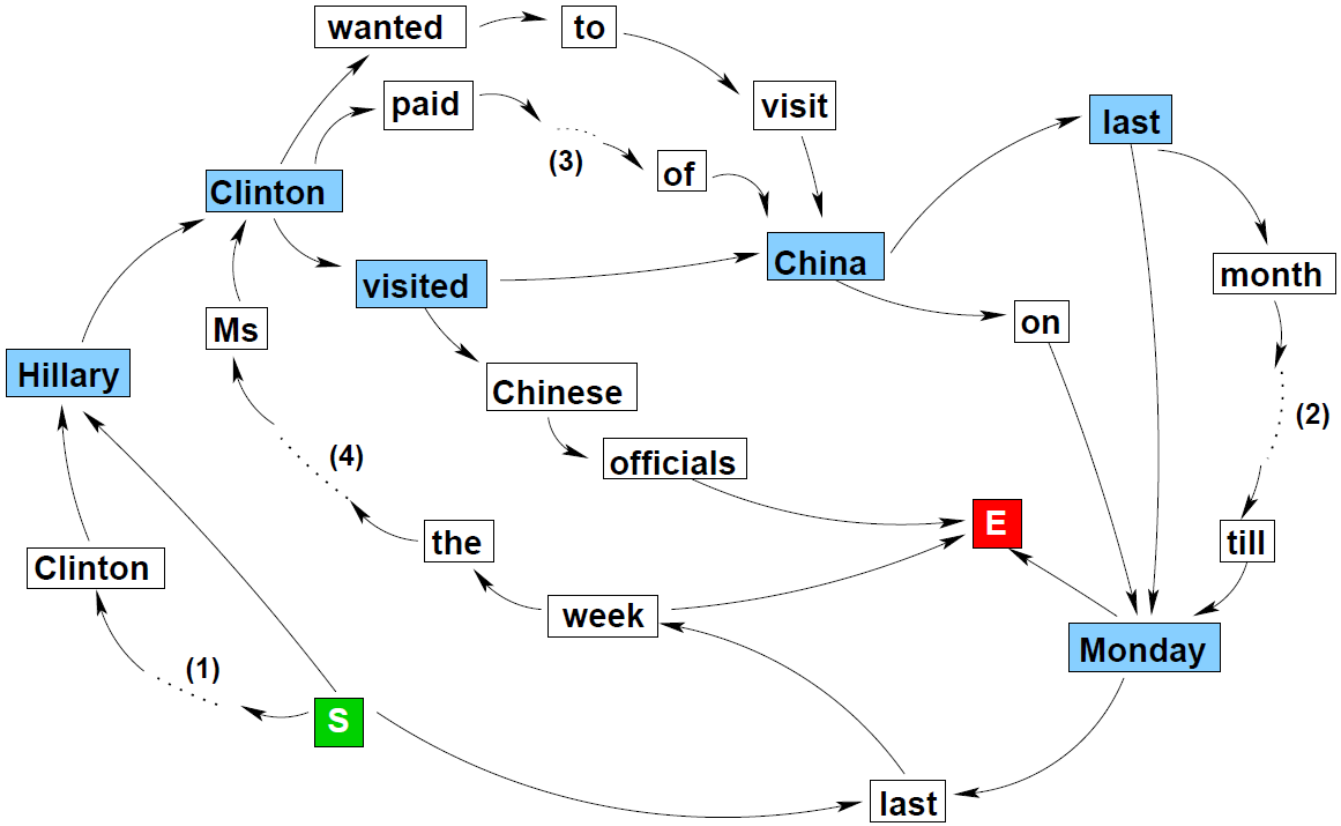
Examples

1) The wife of a former U.S. president Bill Clinton Hillary Clinton visited China last Monday

2) Hillary Clinton wanted to visit China last month but postponed her plans till Monday last week

3) Hillary Clinton paid a visit to the People Republic of China on Monday

4) Last week the Secretary of State Ms. Clinton visited Chinese officials



Lessons learned

syntactic parsers, language models, and/or handcrafted rules are not the only way of controlling the grammaticality of the output

=> redundancy provides a reliable way of generating grammatical sentences

References

Filippova, K. (2010, August). Multi-sentence compression: Finding shortest paths in word graphs. In *Proceedings of the 23rd International Conference on Computational Linguistics* (pp. 322-330). Association for Computational Linguistics.

Boudin, F., & Morin, E. (2013, June). Keyphrase Extraction for N-best reranking in multi-sentence compression. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Mehdad, Y., Carenini, G., Tompa, F. W., & Ng, R. T. (2013, August). Abstractive meeting summarization with entailment and fusion. In *Proc. of the 14th European Workshop on Natural Language Generation* (pp. 136-146).

DaScim

Dascim – Tex mining topics

Advanced keyword extraction and summarization

- gowis - acl 2016
- keyword extraction - emnlp 2016
- keyword extraction for summarization - off line
- keyword extraction for summarization - on line

Advanced GoW topics

- tw-icw
- graph kernels for document similarity
- graph based regularization for text classification

Motivation

1. Keywords provide a snapshot of the ongoing topics
2. Can be used to improve participants experience
3. Utterances are often ill-formed
4. Speakers dilute information by pausing
5. Errors by the ASR
6. Traditional keyword extraction algorithms may not apply

Contribution

- A novel method to select representative words from automatic speech transcriptions in real-time.
- A thorough performance evaluation framework on the standard AMI and ICSI meeting corpora.

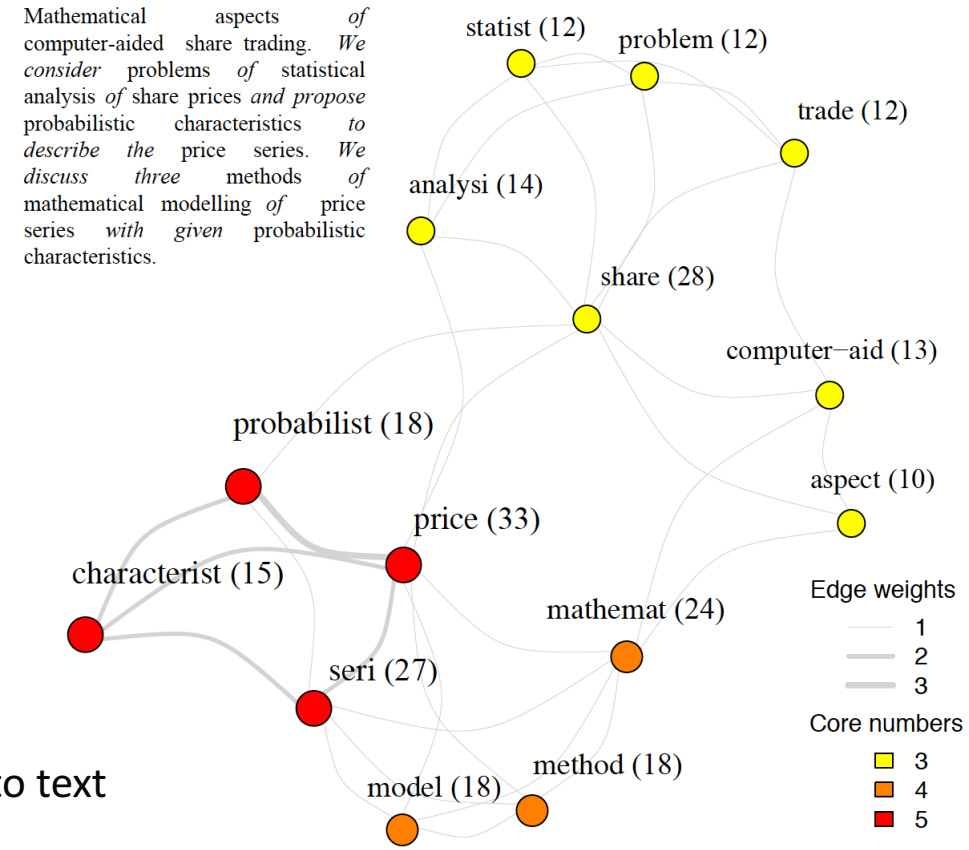
start	end	role	text
			Um well as the kick-off meeting for our our
55.415	60.35	PM	project i
60.35	64.16	PM	And um this is what we're
			this is what we're gonna be doing as an extra five
64.16	67.55	PM	minutes um
			um so of course we'll just a kind of make sure
67.55	72.79	PM	that we all know each other if i'm right
67.21	67.45	ME	Uh-huh
...	...	...	...

Example of utterances from the AMI corpus

Representation

Graph-of-Words

1. Document as an *undirected*, *weighted* graph
2. Nodes are unique terms
3. Edge between two nodes if they co-occur within a fixed-size sliding window
4. Edge weights match co-occurrence counts



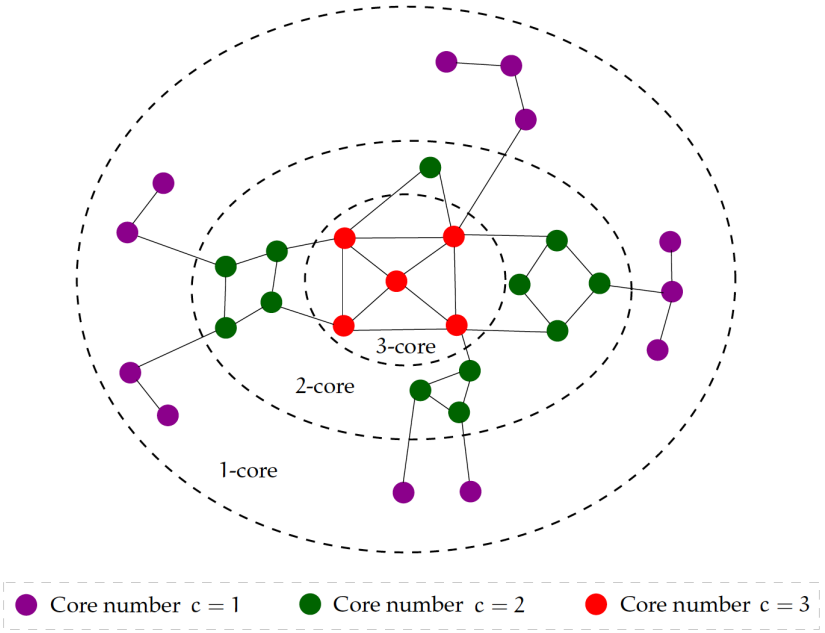
Graph-of-words example for a window of size 3.

Pros

- powerful algorithms from graph theory to text
- Encodes the relative position of words

Graph Decomposition

- A core of order k (or k-core) of G is a maximal connected subgraph of G in which every vertex v has at least degree k
- Degree of v is the sum of the weights of its incident edges
- k-core decomposition: Hierarchy of nested subgraphs of decreasing size and increasing cohesiveness
- Intuition: Cohesive components include keywords



CoreRank

Goal:

Why CoreRank?

$$cr(v) = \sum_{u \in \mathcal{N}(v)} core(u)$$

- Coarse granularity of the k-core decomposition
- Small Interval values worsen the problem
- Many nodes may end up sharing the exact same core number
- Work at the node level
- Solves the ties

Keyword extraction Quality function

Goal: Select the top keywords that also form dense subgraphs

f(S): objective function to optimize

S: set of nodes

cr(v): CoreRank of node v

h(S): penalty function

$$f(S) = \sum_{v \in S} cr(v) - \lambda h(S)$$

$$h(S) = \binom{|S|}{2} - |E(S)|$$

- f satisfies the property of diminishing returns

$$f(A \cup v) - f(A) \geq f(B \cup v) - f(B)$$

for $$A \subseteq B \subseteq V \setminus v$$

- f is a submodular function

- Goal: maximize f

$$S^* = \underset{S \subseteq V, \sum_{v \in S} c_v \leq B}{\arg\max} f(S)$$

- Solution NP-Complete
- Greedy Algorithm approximation in polynomial time
- Performance Bound

$$(1 - 1/e) \approx 0.63$$

If $$|S| \ll |V|$$

Evaluation

Datasets

- AMI -136 short meeting
- ICSI -56 long meetings

Evaluation Scenario 1:

- Streaming evaluation
- Ground truth: extractive summary
- For each time interval, keywords are compared to the part of the of the same time interval
- Metric: Cosine Similarity

$$cos(\vec{d_1}, \vec{d_2}) = \frac{\vec{d_1} \cdot \vec{d_2}}{\|\vec{d_1}\| \times \|\vec{d_2}\|}$$

Evaluation Scenario 2:

- Summarization evaluation
- Ground truth: abstractive summary
- Group together the keywords from all intervals
- Concise keyword-based summary of the meeting
- Metrics
  - ROUGE

$$\text{ROUGE-N}(S) = \frac{\sum_{e \in R_i} \min(c_e(S), r_{e,i})}{\sum_{e \in R_i} r_{e,i}}$$

  - WMD

$$\text{WMD}(\vec{d_1}, \vec{d_2}) = \min \sum_{i,j=1}^{n} T_{i,j} c_{i,j}$$

Word Mover's Distance as Evaluation Metric

$$\text{WMD}(\vec{d_1}, \vec{d_2}) = \min \sum_{i,j=1}^{n} T_{i,j} c_{i,j}$$

- minimum weighted cumulative cost needed for all words of d_1 to travel to d_2
- word-word traveling cost is computed as the distance in a highly-dimensional euclidean word embedding space
- Distributed representations of words encode syntactic and semantic regularities
- WMD can accurately measure the true dissimilarity between two documents

Baseline Methods

1. **Random**: terms are selected at random from the interval text. To reduce variance, we average results over 10 runs.
2. **Frequency**: this baseline uses the bag-of-words representation. The words with the greatest term-frequency (TF) are selected from the vocabulary.
3. **Degree**: the vertices associated with the highest degree centrality in the graph-of-words are returned.
4. **PageRank**: this method applies PageRank to the graph-of-words and returns the highest scoring nodes as keywords.
5. **RAKE**: The Rapid Automatic Keyword Extraction assigns scores deg(v)/freq(v) to terms. Since utterances cannot be considered to be well-formed sentences, we use a sliding window like in our system.
6. **Oracle**: this last baseline is similar to Frequency, but works from the part of the extractive summary corresponding to the current time interval

Results -Scenario 1

Method \ Dataset	AMI	ICSI
CoreRank	**0.474***	**0.259***
Frequency	0.460	0.231
PageRank	0.469	0.250
Random	0.365	0.190
Degree	0.470*	0.245
RAKE	0.384	0.196
Oracle	0.849	0.758

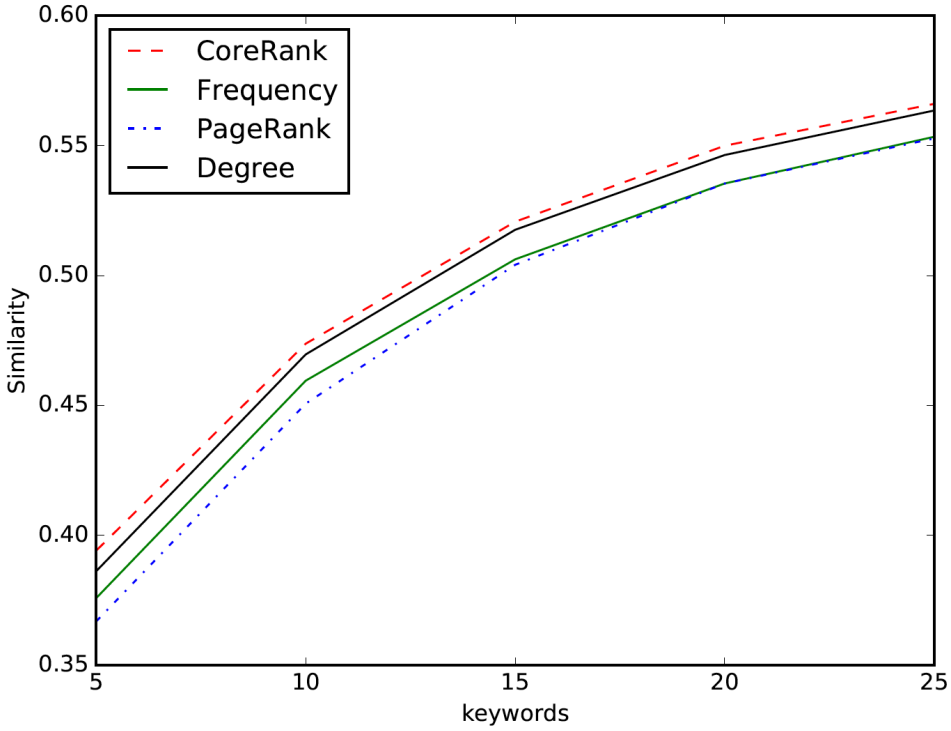
Real-time keyword extraction performance of

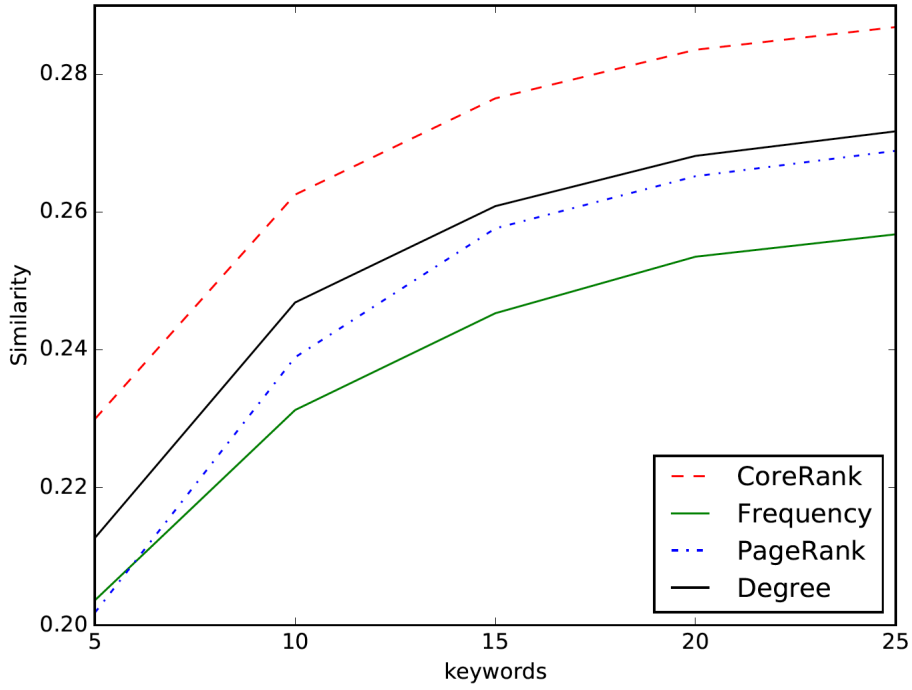
using **cosine similarity**.

*indicates statistical significance

against the Frequency baseline of the same column.

Results -Scenario 1





Performance vs number of keywords using cosine similarity for AMI corpus(left) and ICSI corpus(right)

Results -Scenario 1

Method \ Dataset	AMI	
	$\lambda = 0$	optimal λ
CoreRank	0.470	0.474
PageRank	0.466	0.469
Degree	0.467	0.470

Real-time keyword extraction performance with

λ= 0 (left) and the optimal value of λ(right),

found using a small development set

Results -Scenario 2

Method \ Dataset	AMI		ICSI	
	Rouge	WMD	Rouge	WMD
CoreRank	**0.237**	**1.6528**	**0.134**	**1.6986**
Frequency	0.214	1.6610	0.121	1.7085
PageRank	0.219	1.6569	0.133	1.7011
Random	0.161	1.7610	0.077	1.7723
Degree	0.213	1.6570	0.130	1.7123
RAKE	0.195	1.7242	0.108	1.7050
Oracle	0.227	1.5816	0.136	1.0516

Real-time keyword extraction performance using

ROUGE (greater is better) and WMD (lower is better)

Outline.

Topic modelling

Advanced keyword extraction and summarization

- Gow visualization & summarization - gowis – ACL 2016
- Heuristics and Submodularity based keyword extraction – EMNLP 2016 + EACL 2017
- keyword extraction for summarization - off line – IEEE-ICASSP(submitted)
- Abstractive summary (fillipova + openpaas)

Advanced GoW topics

- Shortest-Path Graph Kernels for Document Similarity -
- Collection level graph weights - tw-icw (under-submission)
- Graph based regularization for text classification – EMNLP2016

Shortest-Path Graph Kernels for Document Similarity

Bag Of Words

Traditionally, documents are represented as bag of words (BOW) vectors

- I entries correspond to terms
- I non-zero for terms appearing in the document

Example

- corpus vocabulary: {the, quick, brown, cat, fox, jumped, went, over, lazy, lion, dog}
- BOW representation of sentence: “the quick brown fox jumped over the lazy dog“

1	1	1	0	1	1	0	1	1	0	1

- However, BOW representation disregards word order!!!

Bag of n-grams

n-gram: a contiguous sequence of n words

For example, the previous sentence: “the quick brown fox jumped over the lazy dog“ contains the following tri-grams:

1. the quick brown
2. quick brown fox
3. brown fox jumped
4. fox jumped over
5. jumped over the
6. over the lazy
7. the lazy dog

n-grams distinguish word order, however, they are too strict

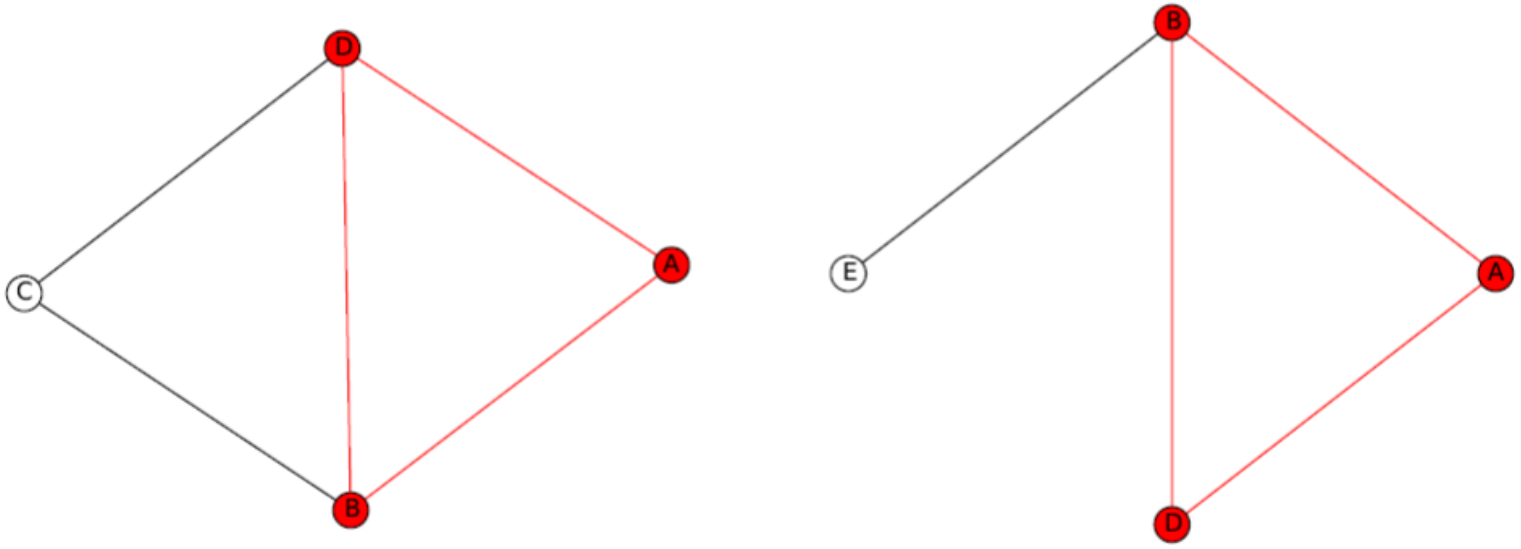
- I unlikely that the same sequence of n word appears in independent documents

Graph Comparison

Given two graphs G1 and G2 from the space of graphs G, the problem of graph comparison is to find a mapping

$$s : \mathcal{G} \times \mathcal{G} \rightarrow \mathcal{R}$$

such that $s(G_1, G_2)$ quantifies the similarity of the two graphs



Applications of Graph Comparison

- Function prediction of chemical compounds
- Structural comparison and function prediction of protein structures
- Comparison of social networks
- Comparison of UML diagrams
- Document similarity

Approaches for Comparing Graphs

1. Graph isomorphism: find a mapping of the vertices of G1 to the vertices of G2 s.t. G1 and G2 are identical
   - No polynomial-time algorithm is known
   - Neither is it known to be NP-complete

2. Subgraph isomorphism: find if any subgraph of G1 is isomorphic to a smaller graph G2
   - I NP-complete

3. Graph edit distance: count necessary operations to transform G1 into G2
   - Contains subgraph isomorphism check as one intermediate step

4. Graph kernels: compare substructures of graphs
   - Computable in polynomial time

What is a Kernel?

Definition (Kernel Function)

The function k : $\mathcal{X} \times \mathcal{X} \to \mathcal{R}$ is a kernel if it is:

1. symmetric: k(x,y) = k(y,x)
2. positive semi-definite: the Kernel Matrix K defined by $\mathbf{K}_{ij} = k(x_i, x_j)$ is positive semi-definite $\forall x_1, x_2, \ldots, x_n \in \mathcal{X}$

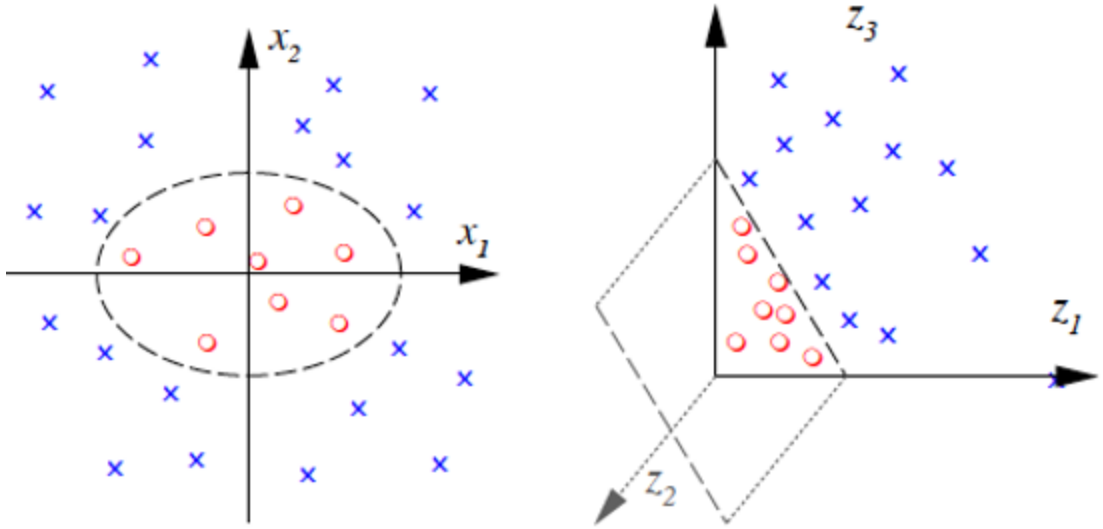
If a function satisfies the above two conditions on a set X, it is known that there exists a map $\phi : \mathcal{X} \to \mathbb{H}$ into a Hilbert space H, such that: $k(x, y) = \langle \phi(x), \phi(y) \rangle$

for all (x,y) where $\langle \cdot, \cdot \rangle$ is the inner product in $\mathbb{H}$

Kernel Trick

The kernel trick avoids the explicit mapping that is needed to get linear learning algorithms to learn a nonlinear function or decision boundary

$$\Phi : R^2 \to R^3$$
$$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{(2)} x_1 x_2, x_2^2)$$



Kernels for Graphs

Graph representation of objects is advantageous compared to vectorial approaches :

- able to model relations between different parts of an object as well as the values of object properties
- The dimensionality of graphs is not fixed

Problem: How to compute kernels for structured data:

- Sequences
- Graphs

Shortest-Path Kernels on Graphs

- Compute all-pairs-shortest-paths for G1 and G2 via Floyd-Warshall
- Define a kernel by comparing all pairs of shortest path lengths from G1 and G2:

$$k(G_1, G_2) = \sum_{v_i, v_j \in G_1} \sum_{v_k, v_l \in G_2} k_{length}(d(v_i, v_j), d(v_k, v_l))$$

  - $d(\cdot,\cdot)$: length of the shortest path between two nodes
  - k_{length}: kernel that compares the lengths of two shortest paths.
    - linear kernel

$$k_{length}(d(v_i, v_j), d(v_k, v_l)) = d(v_i, v_j) \cdot d(v_k, v_l)$$

    - delta kernel

$$k_{length}(d(v_i, v_j), d(v_k, v_l)) = \begin{cases} 1 & \text{if } d(v_i, v_j) = d(v_k, v_l) \\ 0 & \text{otherwise} \end{cases}$$

Shortest-Path Kernels on Document Graphs

$$k(d_1, d_2) = \frac{\left(\begin{array}{l} \sum_{v_1 \in \mathcal{V}_1, v_2 \in \mathcal{V}_2} k_{node}(v_1, v_2) \\ \quad + \sum_{e_1 \in \mathcal{E}_1, e_2 \in \mathcal{E}_2} k^{(1)}_{walk}(e_1, e_2) \end{array} \right)}{norm} \tag{1}$$

$$k_{node}(v_1, v_2) = \begin{cases} 1 & \text{if } label(v_1) = label(v_2), \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

$$k_{walk}(e_1, e_2) = k_{node}(u_1, u_2) \times k_{edge}(e_1, e_2) \times k_{node}(v_1, v_2) \tag{3}$$

$$k_{edge}(e_1, e_2) = \begin{cases} \min[label(e_1), & \\ \qquad label(e_2)]^2 & \text{if } e_1 \in \mathcal{E}_1 \wedge e_2 \in \mathcal{E}_2, \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

$$norm = \|\mathbf{M}_1\|_F \times \|\mathbf{M}_2\|_F$$

Evaluation

Document classification

- Goal: classify documents in a predefined set of categories
- Compute kernel matrix for all pairs of documents
- Support Vector Machine Model using the kernel matrices
- Compute Accuracy and F1 score

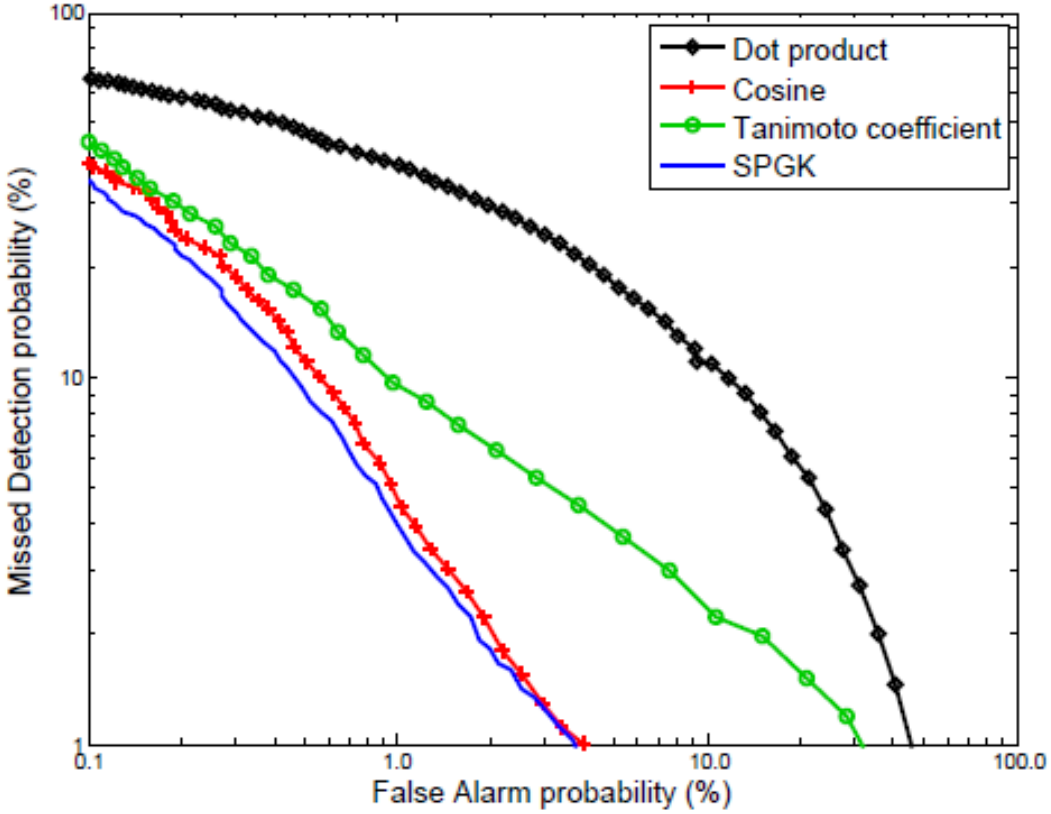
Link Detection

- Goal: find if two documents are linked
- Compute kernel distances for all pairs of documents
- Calculate Detection and Miss probabilities for every possible detection threshold (DET curve)

Classification results

Similarity measure		WebKB		20 Newsgroups		Reuters-21578		Amazon (category)		Amazon (sentiment)	
		Accuracy	F1-score	Accuracy	F1-score	Accuracy	F1-score	Accuracy	F1-score	Accuracy	F1-score
Dot product		0.8850	0.8687	0.7439	0.7423	0.9460	0.8188	0.9125	0.9128	0.7918	0.7968
Cosine		0.9356	0.9304	0.8118	0.8073	0.9497	0.8091	0.9375	0.9375	0.8100	0.8112
Tanimoto coefficient		0.9103	0.8926	0.7883	0.7845	0.9310	0.7264	0.9368	0.9370	0.8100	0.8112
SPGK	$d = 1$	**0.9428**	**0.9380**	0.8138	0.8093	0.9634	0.8944	0.9412	0.9413	0.8175	0.8179
	$d = 2$	0.9399	0.9352	0.8148	0.8101	**0.9648**	**0.8991**	**0.9425**	**0.9425**	**0.8194**	**0.8200**
	$d = 3$	0.9399	0.9353	0.8148	0.8101	**0.9648**	**0.8991**	0.9412	0.9413	0.8181	0.8184
	$d = 4$	0.9392	0.9343	**0.8150**	**0.8105**	0.9643	0.8990	0.9412	0.9413	0.8169	0.8177

Link Detection results



Dascim – Text mining topics

Advanced keyword extraction and summarization

- gowis - acl 2016
- keyword extraction - emnlp 2016
- keyword extraction for summarization - off line
- keyword extraction for summarization - on line

Advanced GoW topics

- graph kernels for document similarity
- **tw-icw**
- graph based regularization for text classification

Outline.

Topic modelling

Advanced keyword extraction and summarization

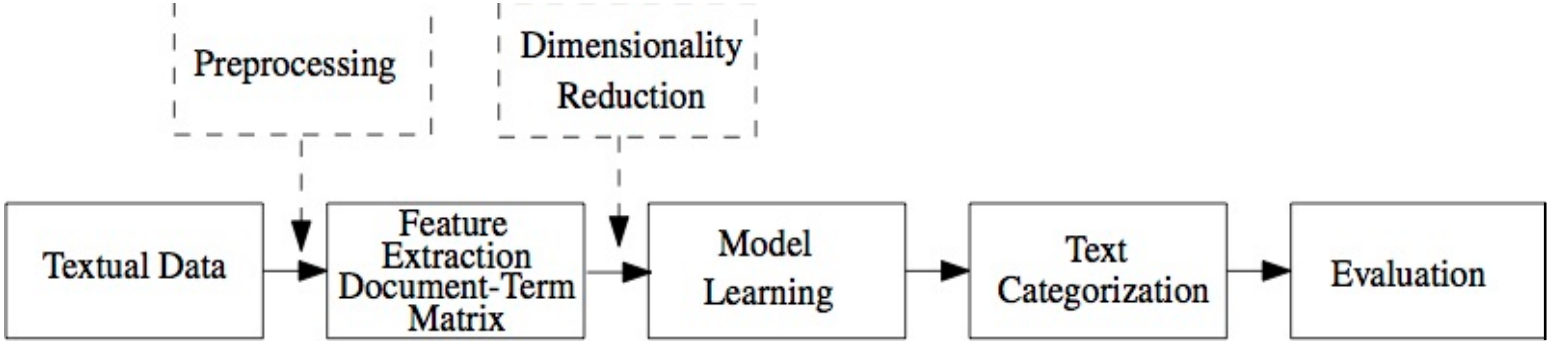
- Gow visualization & summarization - gowis – ACL 2016
- Heuristics and Submodularity based keyword extraction – EMNLP 2016 + EACL 2017
- keyword extraction for summarization - off line – IEEE-ICASSP(submitted)
- Abstractive summary (fillipova + openpaas)

Advanced GoW topics

- Shortest-Path Graph Kernels for Document Similarity -
- Collection level graph weights - tw-icw (under-submission)
- Graph based regularization for text classification – EMNLP2016

Text Categorization or Classification

- Supervised learning: training and test data, knowing the class a document belongs
- TC: assign a document to a set of two or more predefined categories (or classes)
- Sentiment analysis, automated news classification, spam detection
- Multiclass, single-label text categorization



Features: Bag-of-words

- Each document is modeled wit the Vector Space Model
- We need to find appropriate weights regarding the importance of each term in a document
- A document is represented as a multiset of its terms, disregarding dependencies between the terms
- The importance of a term in a document is mainly determined by the frequency of the term
- **Term independence assumption**

Weakness: the order of the terms within a document is completely disregarded!

TF-IDF for TC

- Let $D = \{d_1, d_2, \ldots, d_m\}$ be a collection of documents
- Let $C = \{c_1, c_2, \ldots, c_{|C|}\}$ be the set of predefined categories
- Vector Space Model, i.e., the spatial representation in which each document is represented by a vector in the **n-dimensional space,** where n is the number of words in the collection level
- Terms $T = \{t_1, t_2, \ldots, t_n\}$ of the overall vocabulary of the collection
- Term frequency of term *t* in a document *d*: tf(t.d)
- Inverse Document Frequency: $idf(t, \mathcal{D}) = \log\left(\frac{m+1}{|\{d \in \mathcal{D} : t \in d|\}}\right)$

- TF-IDF: |d| is the length of document, lambda is the average document length

$$tf\text{-}idf(t,d) = \frac{1 + \ln\left(1 + \ln(tf(t,d))\right)}{1 - b + b \times \frac{|d|}{\ell}} \times idf(t, \mathcal{D})$$

Graph-based Text Categorization

Collection level graph

- Merge document graphs
- Relative importance of the node in the graph based on specific criteria
- Each term is represented as a node in the graph and the edges capture co-occurrence relationships of terms with a specified distance in the document.

- **Nodemetrics**
  - **Replace tf with some graph besed metric either:**
    - **Local information of the graph** (e.g., degree centrality, in-degree/out-degree centrality in directed networks, weighted degree in weighted graphs, clustering coefficient) (Easley and Kleinberg, 2010)
    - **Global information**, in the sense that the importance of a node is determined by the properties of the node globally in the graph (e.g., PageRank, eigenvector, betweeness, closeness centralities).

Node Centrality Criteria

- **Degree centrality**: local node importance criteria, which captures the number of neighbors that each node has.

$$\text{degree\_centrality}(i) = \frac{|\mathcal{N}(i)|}{|V|-1}$$

- I**n-degree and out-degree centrality:** extensions of the degree centrality in directed networks, in-degree (number of incoming edges) and out-degree (number of out-going edges) of each node.

- **Closeness centrality:** how close a node is to all other nodes in the graph. *dist(i,j)* is the shortest path distance between nodes *i* and *j.*

$$\text{closeness}(i) = \frac{|V|-1}{\sum_{j \in V} dist(i,j)}$$

- **Eigenvector centrality:** Let A be the adjacency matrix of an undirected graph G. The eigenvector centrality of node *i* is defined as:

$$u(i) = \frac{1}{\lambda} \sum_j A_{ij} u_j$$

u is the eigenvector that corresponds to the largest eigenvalue λ of A.

TW-IDF

- Let *tw(t, d)* be the centrality score of term (node) t in the graph representation G_d of document d.

- Similar to TF of Eq. (1), we can define the basic weighting scheme denoted as Term Weight (TW), as follows:

$$tw(t,d) = \frac{\tilde{tw}(t,d)}{1 - b + b \times \frac{|d|}{\ell}}$$

- Consider information about the inverse document frequency (IDF factor) of the term t in the collection D.

$$tw\text{-}idf(t,d) = tw(t,d) \times idf(t,\mathcal{D})$$

- Very effective in graph-based Information Retrieval (Rousseau and Vazirgiannis 2013) and keyword extraction (Mihalcea and Tarau, 2004; Rousseau and Vazirgiannis, 2015)

- Avoid “exploding the feature space” cause by n-grams

Graph-of-Word Construction

- Each document $d \in D$ is represented by a graph $G_d = (V_d , E_d)$, where the nodes correspond to the terms t of the document and the edges capture co-occurrence relationships between terms within a fixed-size sliding window of size w.

- **Directed vs. Undirected graph**
  - Directed graphs are able to preserve the actual flow of a text
  - In undirected, an edge captures co-occurrence of two terms whatever the respective order between them is

- **Weighted vs. Unweighted graph**
  - The higher the number of co-occurences of two terms in the document, the higher the weight of the corresponding edge

- **Size w of the sliding window**
  - Add edges between the terms of the document that co-occur within a sliding window of size ***w***
  - Larger window sizes produce graphs that are relatively dense

TW - Inverse Collection Weight

- A graph-based criterion to penalize the weight of terms that are “important” across the whole collection of documents.
- The main concept behind ICW is the ***collection level*** graph G – an extension of Graph-of-Words in the collection of documents D.
- Collection Level Graph G: *Let* $\{G_1, G_2, \ldots, G_d\}_{|D|}$ be the set of graphs that correspond to all documents $d \in D$.
- The collection level graph G is defined as the union of graphs $G_1 \cup G_2 \cup \ldots \cup G_d$ over all documents in the collection.

$$icw(t, \mathcal{D}) = \log \left(\frac{\sum_{v \in \mathcal{D}} tw(v, \mathcal{D})}{tw(t, \mathcal{D})} \right)$$

TW-ICW results

Methods	20NG				IMDB			
	$w = 2$		$w = 3$		$w = 2$		$w = 3$	
	F1	**Acc**	**F1**	**Acc**	**F1**	**Acc**	**F1**	**Acc**
TF	0.7985	0.8040	0.7985	0.8040	0.8397	0.8380	0.8397	0.8380
TF binary	0.8173	0.8226	0.8173	0.8226	0.8481	0.8480	0.8481	0.8480
TW (degree)	0.7660	0.7827	0.7954	0.8078	0.8193	0.8185	0.8414	0.8415
TW (closeness)	0.8090	0.8171	0.7984	0.8065	0.8313	0.8335	0.8285	0.8305
TF-IDF	0.8222	0.8275	0.8222	0.8275	0.8434	0.8425	0.8434	0.8425
TW-IDF (degree)	0.8261	0.8377	**0.8362**	**0.8454**	0.8249	0.8265	0.8385	0.8390
TW-IDF (closeness)	0.8241	0.8321	0.8257	0.8333	0.8480	0.8495	0.8416	0.8450
TF-ICF	0.8328	0.8291	0.8328	0.8291	0.8318	0.8340	0.8318	0.8340
TW-ICW (deg, deg)	0.8291	0.8374	0.8231	0.8301	**0.8526**	**0.8525**	0.8478	0.8470
TW-ICW (deg, eig)	**0.8354**	**0.8426**	0.8203	0.8279	0.8486	0.8485	**0.8507**	**0.8505**
TF-ICW (degree)	0.7862	0.7898	0.7856	0.7892	0.8386	0.8400	0.8384	0.8400

Methods	REUTERS				WEBKB			
	$w = 2$		$w = 3$		$w = 2$		$w = 3$	
	F1	**Acc**	**F1**	**Acc**	**F1**	**Acc**	**F1**	**Acc**
TF	0.9139	0.9666	0.9139	0.9666	0.8837	0.8954	0.8837	0.8954
TF binary	0.8691	0.9566	0.8691	0.9566	0.8686	0.8818	0.8686	0.8818
TW (degree)	0.9130	0.9671	**0.9409**	**0.9707**	0.8647	0.8839	**0.8906**	**0.8989**
TW (closeness)	0.8979	0.9602	0.9105	0.9634	0.8935	0.9025	0.8876	0.8982
TF-IDF	0.8950	0.9639	0.8950	0.9639	0.8665	0.8789	0.8665	0.8789
TW-IDF (degree)	**0.9410**	**0.9748**	0.9396	0.9716	0.8480	0.8753	0.8789	0.8875
TW-IDF (closeness)	0.8930	0.9597	0.8940	0.9584	0.8871	0.8946	0.8729	0.8832
TF-ICF	0.8848	0.9570	0.8848	0.9570	0.8661	0.8825	0.8661	0.8825
TW-ICW (deg, deg)	0.9335	0.9721	0.9194	0.9661	**0.8976**	**0.9040**	0.8684	0.8839
TW-ICW (deg, eig)	0.9313	0.9712	0.9201	0.9661	0.8916	0.8982	0.8633	0.8782
TF-ICW (degree)	0.8797	0.9479	0.8797	0.9479	0.8651	0.8775	0.8681	0.8796

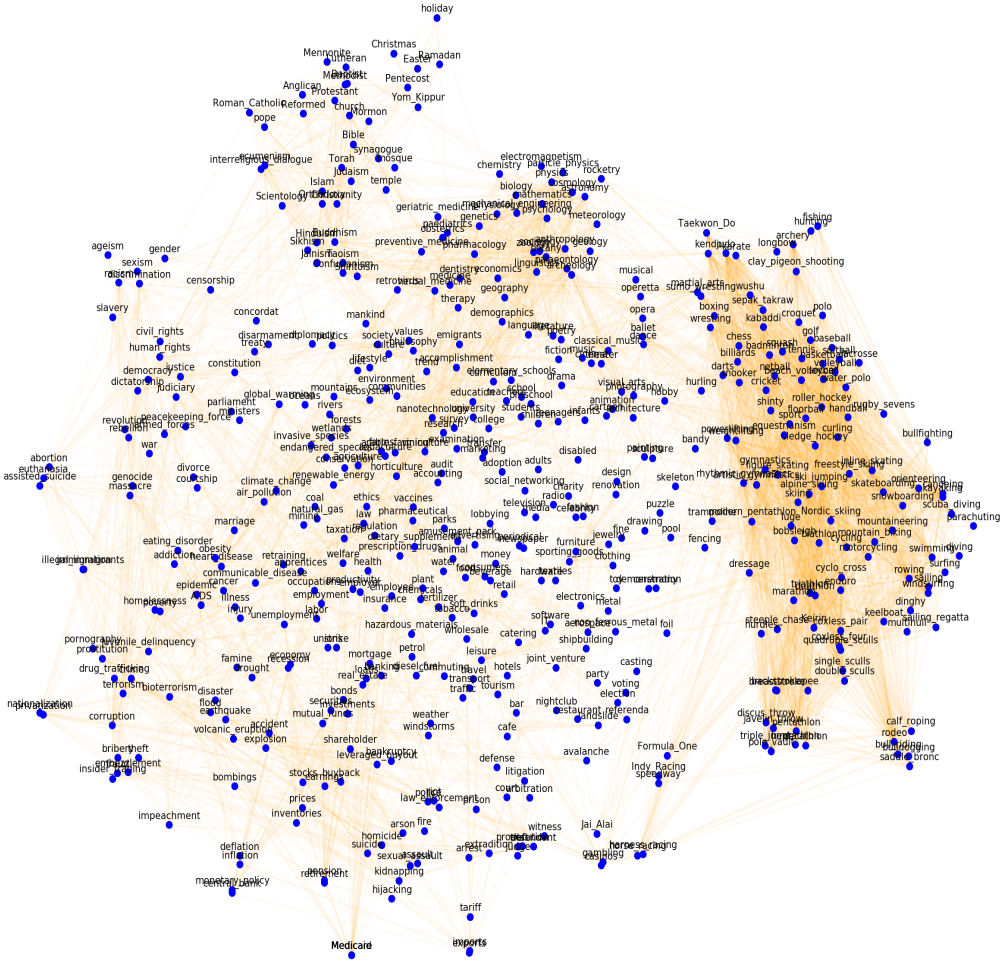
Future work - Edge Weighting

- Can use **word2vec** similarities between words in documents as edge weights
- Cosine similarity ranges between -1 and 1
- Valid distance metric needs to be bounded between 0 and 1

- Angular distance with positive or negative vector elements:

$$\text{distance} = \frac{cos^{-1}(\text{similarity})}{\pi}$$

- *# of windows where node/word co-ocurr*
- *Distance* in the window



Outline.

Topic modelling

Advanced keyword extraction and summarization

- Gow visualization & summarization - gowis – ACL 2016
- Heuristics and Submodularity based keyword extraction – EMNLP 2016 + EACL 2017
- keyword extraction for summarization - off line – IEEE-ICASSP(submitted)
- Abstractive summary (fillipova + openpaas)

Advanced GoW topics

- Shortest-Path Graph Kernels for Document Similarity -
- Collection level graph weights - tw-icw (under-submission)
- Graph based regularization for text classification – EMNLP2016

Regularization for Text Categorization

- Why regularization:
  - Address overfitting: high training score, low test score
  - Better accuracy

- We want our model to generalize in new unseen test instances

- Harvest the full potential hidden in the rich textual data

- **Feed meaningful group of words to group lasso for regularization**

Objective function + Loss

- Text categorization as a loss minimization problem:

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \underbrace{\sum_{i=1}^{N} \mathcal{L}(\boldsymbol{x}^i, \boldsymbol{\theta}, y^i)}_{\textbf{empirical risk}}$$

- Logistic regression with binary predictions (y={-1,1}), $h_{\theta,b}(x) = \theta^T x + b$ and $L(x,\theta,y) = e^{-yh(x)}$ (log loss)
- Only minimizing the empirical risk can lead to overfitting.

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \underbrace{\underbrace{\sum_{i=1}^{N} \mathcal{L}(\boldsymbol{x}^i, \boldsymbol{\theta}, y^i)}_{\textbf{empirical risk}} + \underbrace{\lambda \Omega(\boldsymbol{\theta})}_{\textbf{penalty term}}}_{\textbf{expected risk}}$$

- L1, L2 regularization aka lasso (Tibshirani, 1996) and ridge (Hoerl, Kennard, 1970)

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \sum_{i=1}^{N} \mathcal{L}(\boldsymbol{x}^i, \boldsymbol{\theta}, y^i) + \lambda \sum_{j=1}^{p} |\boldsymbol{\theta}_j|$$

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \sum_{i=1}^{N} \mathcal{L}(\boldsymbol{x}^i, \boldsymbol{\theta}, y^i) + \lambda \sum_{j=1}^{p} \boldsymbol{\theta}_j{}^2$$

Learning

- A constrained optimization problem is formed that can be solved as an augmented Lagrangian problem:

$$\Omega_{las}(\boldsymbol{\theta}) + \Omega_{glas}(\mathbf{v}) + \mathcal{L}(\boldsymbol{\theta}) + \mathbf{u}^\top(\mathbf{v} - M\boldsymbol{\theta}) + \frac{\rho}{2}\|\mathbf{v} - M\boldsymbol{\theta}\|_2^2$$

- The problem becomes the iterative update of θ, v and u:

$$\min_{\boldsymbol{\theta}} \Omega_{las}(\boldsymbol{\theta}) + \mathcal{L}(\boldsymbol{\theta}) + \mathbf{u}^\top M\boldsymbol{\theta} + \frac{\rho}{2}\|\mathbf{v} - M\boldsymbol{\theta}\|_2^2$$

$$\min_{\mathbf{v}} \Omega_{glas}(\mathbf{v}) + \mathbf{u}^\top\mathbf{v} + \frac{\rho}{2}\|\mathbf{v} - M\boldsymbol{\theta}\|_2^2$$

$$\mathbf{u} = \mathbf{u} + \rho(\mathbf{v} - M\boldsymbol{\theta})$$

- Yogatama and Smith(2014) proved that ADMM for sparse overlapping group lasso converges. A good approximate solution is reached in a few tens of iterations.

Structured regularization

- In L1 and L2 regularization, features are considered as independent
- Group lasso: Bakin(1999), Yuan and Lin (2006) introduced group sparsity in the model:

$$\Omega(\boldsymbol{\theta}) = \lambda \sum_{g} \lambda_g \|\boldsymbol{\theta}_g\|_2$$

Algorithm 1 ADMM for overlapping group-lasso

Require: augmented Lagrangian variable ρ, regularization strengths λ_{glas} and λ_{las}

1: **while** update in weights not small **do**

2: $\boldsymbol{\theta} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, \Omega_{las}(\boldsymbol{\theta}) + \mathcal{L}(\boldsymbol{\theta}) + \frac{\rho}{2} \sum_{i=1}^{V} N_i(\boldsymbol{\theta}_i - \mu_i)^2$

3: **for** $g = 1$ to G **do**

4: $\mathrm{v}_g = \mathrm{prox}_{\Omega_{glas}, \frac{\lambda_g}{\rho}}(z_g)$

5: **end for**

6: $u = u + \rho(\mathrm{v} - M\boldsymbol{\theta})$

7: **end while**

Structured regularization in NLP

Statistical regularizers:

- Sentence regularizer (state-of-the-art) (overlapping)
  - Large numebr of groups but small group sizes.

Semantic regularizers:

- LDA and LSI regularizers
  - groups are considered the LDA/LSI results – keep top10 words in each group

GoW regularizers

- Graph-of-words regularizer: community detection on collection graph
- Word2vec regularizer: k-means clustering in word2vec space (overlapping)

Trying to extract groups of words that talk about similar topics.

ÉCOLE POLYTECHNIQUE
UNIVERSITÉ PARIS-SACLAY

DaScim

Regularizing Text Categorization with Clusters of Words

Konstantinos Skianis, Francois Rousseau, Michalis Vazirgiannis

Results

	dataset	no reg.	lasso	ridge	elastic	group lasso				
						LDA	LSI	sentence	GoW	word2vec
20NG	science	0.946	0.916	0.954	0.954	**0.968**	**0.968***	0.942	0.967*	**0.968***
	sports	0.908	0.907	0.925	0.920	0.959	0.964*	**0.966**	0.959*	0.946*
	religion	0.894	0.876	0.895	0.890	0.918	0.907*	**0.934**	0.911*	0.916*
	computer	0.846	0.843	0.869	0.856	0.891	0.885*	0.904	0.885*	**0.911***
Sentiment	vote	0.606	0.643	0.616	0.622	**0.658**	0.653	0.656	0.640	0.651
	movie	0.865	0.860	0.870	0.875	**0.900**	0.895	0.895	0.895	0.890
	books	0.750	0.770	0.760	0.780	0.790	0.795	0.785	0.790	**0.800**
	dvd	0.765	0.735	0.770	0.760	0.800	**0.805***	0.785	0.795*	0.795*
	electr.	0.790	0.800	0.800	**0.825**	0.800	0.815	0.805	0.820	0.815
	kitch.	0.760	0.800	0.775	0.800	0.845	**0.860***	0.855	0.840	0.855*

Table 2: Accuracy results on the test sets. Bold font marks the best performance for a dataset. * indicates statistical significance of improvement over lasso at $p < 0.05$ using micro sign test for one of our models LSI, GoW and word2vec (underlined).

	dataset	no reg.	lasso	ridge	elastic	group lasso				
						LDA	LSI	sentence	GoW	word2vec
20NG	science	100	1	100	63	19	20	86	19	21
	sports	100	1	100	5	60	11	6.4	55	44
	religion	100	1	100	3	94	31	99	10	85
	computer	100	2	100	7	40	35	77	38	18
Sentiment	vote	100	1	100	8	15	16	13	97	13
	movie	100	1	100	59	72	81	55	90	62
	books	100	3	100	14	41	74	72	90	99
	dvd	100	2	100	28	64	8	8	58	64
	electr.	100	4	100	6	10	8	43	8	9
	kitch.	100	5	100	79	73	44	27	75	46

Table 3: Fraction (in %) of non-zero feature weights in each model for each dataset: the smaller, the more compact the model.

Results(2)

	dataset	GoW	word2vec
20NG	science	79	691
	sports	137	630
	religion	35	639
	computer	95	594

Table 4: Number of groups.

	dataset	lasso	ridge	elastic	group lasso				
					LDA	LSI	sentence	GoW	word2vec
20NG	science	10	1.6	1.6	15	11	76	12	19
	sports	12	3	3	7	20	67	5	9
	religion	12	3	7	10	4	248	6	20
	computer	7	1.4	0.8	8	6	43	5	10

Table 5: Time (in seconds) for learning with best hyperparameters.

$= 0$	piscataway combination jil@donuts0.uucp jamie reading/seeing chambliss left-handedness abilities lubin acad sci obesity page erythromycin bottom
$\neq 0$	and space the launch health for use that medical you space cancer and nasa hiv health shuttle for tobacco that cancer that research center space hiv aids are use theory keyboard data telescope available are from system information space ftp

Table 6: Examples with LSI regularizer.

$= 0$	village town edc fashionable trendy trendy fashionable points guard guarding crown title champion champions
$\neq 0$	numbness tingling dizziness fevers laryngitis bronchitis undergo undergoing undergoes undergone healed mankind humanity civilization planet nasa kunin lang tao kay kong

Table 7: Examples with word2vec regularizer.

$= 0$	islands inta spain galapagos canary originated anodise advertises jewelry mercedes benzes diamond trendy octave chanute lillienthal
$\neq 0$	vibrational broiled relieving succumb spacewalks dna nf-psychiatry itself commented usenet golded insects alternate self-consistent retrospect

Table 8: Examples with graph-of-words regularizer.

Discussion & Future Work

- Find and extract semantic and syntactic structures that lead to sparser feature spaces → faster learning times
- Linguistic prior knowledge in the data can be used to improve categorization performance for baseline bag-of-words models, by mining inherent structures
- No significant change in results with different loss functions as the proposed regularizers are not log loss specific

INTERESTING QUESTIONS?

- How can we create and cluster graphs, i. e. covering weighted and/or signed cases?
- Find better clusters in word2vec? (+overlapping with GMM)
- Explore alternative regularization algorithms diverging from group-lasso?

References

[1] Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of Workshop at International Conference on Learning Representations*, ICLR '13.

[2] François Rousseau and Michalis Vazirgiannis. 2013. Graph-of-word and tw-idf: New approach to ad hoc ir. In *Proceedings of the 22nd ACM international con- ference on Information and knowledge management*, CIKM '13, pages 59–68.

[3] Konstantinos Skianis, Francois Rousseau, Michalis Vazirgiannis. Regularizing Text Categorization with Clusters of Words. EMNLP 2016

[4] Robert Tibshirani. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288.

[5] Vladimir Naumovich Vapnik. 1991. Principles of Risk Minimization for Learning Theory. In *Advances in Neural Information Processing Systems 4*, NIPS '91, pages 831–838.

[6] Dani Yogatama and Noah A. Smith. 2014a. Linguistic structured sparsity in text categorization. In *Proceed- ings of the 52nd Annual Meeting of the Association for Computational Linguistics*, ACL '14, pages 786–796.

[7] Dani Yogatama and Noah A. Smith. 2014b. Making the most of bag of words: Sentence regularization with al- ternating direction method of multipliers. In *Proceed- ings of the 31st International Conference on Machine Learning*, volume 32 of *ICML '14*, pages 656–664.

[8] Malliaros, F. D., Rossi, M. E. G., & Vazirgiannis, M. (2016). Locating influential nodes in complex networks. *Scientific reports*, *6*.

[9] Rousseau, F., & Vazirgiannis, M. (2015, March). Main core retention on graph-of-words for single-document keyword extraction. In *European Conference on Information Retrieval* (pp. 382-393). Springer International Publishing.

[10] Mihalcea, R., & Tarau, P. (2004, July). TextRank: Bringing order into texts. Association for Computational Linguistics.

[11] Lei Yuan, Jun Liu, and Jieping Ye. 2011. Efficient methods for overlapping group lasso. In *Advances in Neural Information Processing Systems 24*, NIPS '11, pages 352–360.