Text mining methods:
Topic modeling & Graph-based keywords extraction

Jelena Jovanović

Email: jeljov@gmail.com

Web: http://jelenajovanovic.net

OVERVIEW

- Topic modeling methods
  - LDA
- Graph-based methods
  - TextRank
  - KeyGraph

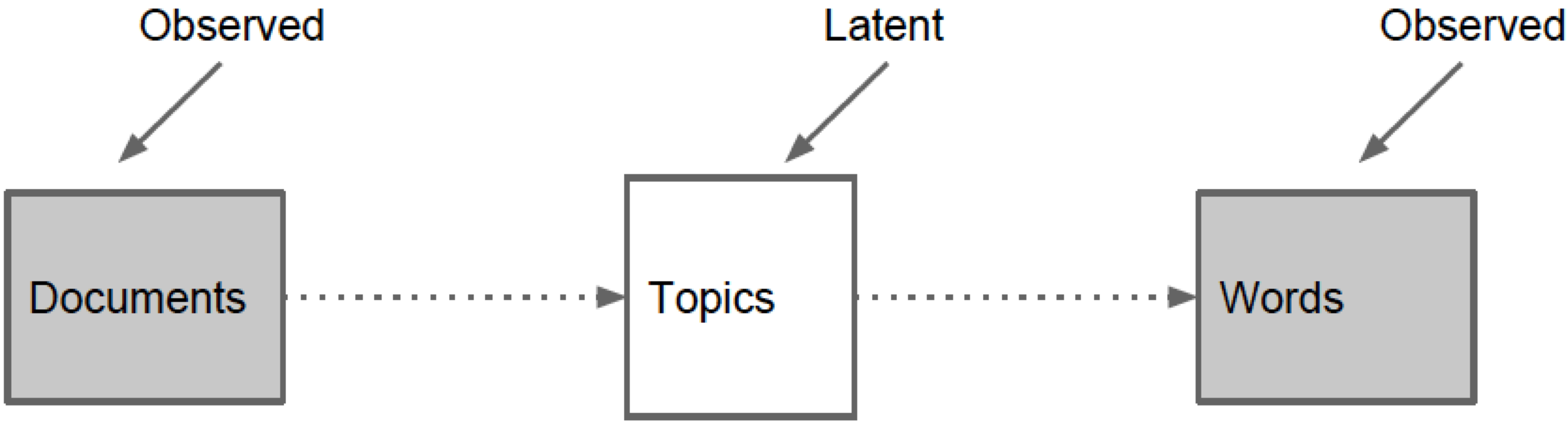
TOPIC MODELING

TOPIC MODELING METHODS

Topic modeling methods are statistical methods that analyze words of the given collection of documents to

- discover the underlying themes,
- how those themes are connected to each other, and
- how they change over time

TOPIC MODELING: THE BASIC CONCEPTS



- Documents can be about several topics at the same time
- Topics are expressed through the words used in the documents
- Documents and words are what we can observe, topics are latent (hidden) constructs

LATENT DIRICHLET ALLOCATION (LDA)

Latent Dirichlet allocation (LDA) is cited as the simplest topic modelling method

LDA assumptions:

- There is a *fixed* set of topics for a collection of documents
- Each topic is a *distribution over a fixed vocabulary*
- Each document in a collection has its own probability distribution over the given (fixed) set of topics
  - as a consequence, each document exhibits multiple topics
- Both topics and words are assumed to follow Dirichlet distributions – hence the name of the method

LDA – THE NAME ORIGIN

- **Dirichlet** comes from the name of the distribution (Dirichlet dist.) that is used to draw both
  - Distribution of topics per document
  - Distribution of words per topic
- **Latent** comes from the fact that topics (their distribution and structure) are *hidden*, *unobservable*, and have to be inferred / mined from the observable items (words)

LDA: EXAMPLE

A sample of topics detected in AP corpus

"Arts"	"Budgets"	"Children"	"Education"
NEW	MILLION	CHILDREN	SCHOOL
FILM	TAX	WOMEN	STUDENTS
SHOW	PROGRAM	PEOPLE	SCHOOLS
MUSIC	BUDGET	CHILD	EDUCATION
MOVIE	BILLION	YEARS	TEACHERS
PLAY	FEDERAL	FAMILIES	HIGH
MUSICAL	YEAR	WORK	PUBLIC
BEST	SPENDING	PARENTS	TEACHER
ACTOR	NEW	SAYS	BENNETT
FIRST	STATE	FAMILY	MANIGAT
YORK	PLAN	WELFARE	NAMPHY
OPERA	MONEY	MEN	STATE
THEATER	PROGRAMS	PERCENT	PRESIDENT
ACTRESS	GOVERNMENT	CARE	ELEMENTARY
LOVE	CONGRESS	LIFE	HAITI

Top 15 most probable words for each topic

The William Randolph Hearst Foundation will give $1.25 million to Lincoln Center, Metropolitan Opera Co., New York Philharmonic and Juilliard School. "Our board felt that we had a real opportunity to make a mark on the future of the performing arts with these grants an act every bit as important as our traditional areas of support in health, medical research, education and the social services," Hearst Foundation President Randolph A. Hearst said Monday in announcing the grants. Lincoln Center's share will be $200,000 for its new building, which will house young artists and provide new public facilities. The Metropolitan Opera Co. and New York Philharmonic will receive $400,000 each. The Juilliard School, where music and the performing arts are taught, will get $250,000. The Hearst Foundation, a leading supporter of the Lincoln Center Consolidated Corporate Fund, will make its usual annual $100,000 donation, too.

Source: David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. J. Mach. Learn. Res. 3 (March 2003), 993-1022.

LDA's GENERATIVE PROCESS

- LDA is based on a statistical model of how a set of documents have been created (generated)
  - this is known as *generative process*
- The objective is to find parameters of that model that best fit the observed data (document collection)

LDA's GENERATIVE PROCESS

LDA generative process is based on 2 assumptions:

1) Distribution of topics across documents follows Dirichlet distribution with parameter *alpha* (α)
   - Alpha is a K-dimensional vector, where K is the number of topics
   - Alpha determines how topics are associated with documents
   - Smaller alpha favours fewer topics strongly associated with a document
   - Alpha is an input to the LDA algorithm (i.e. generative process)

LDA'S GENERATIVE PROCESS

LDA generative process is based on 2 assumptions (cont.):

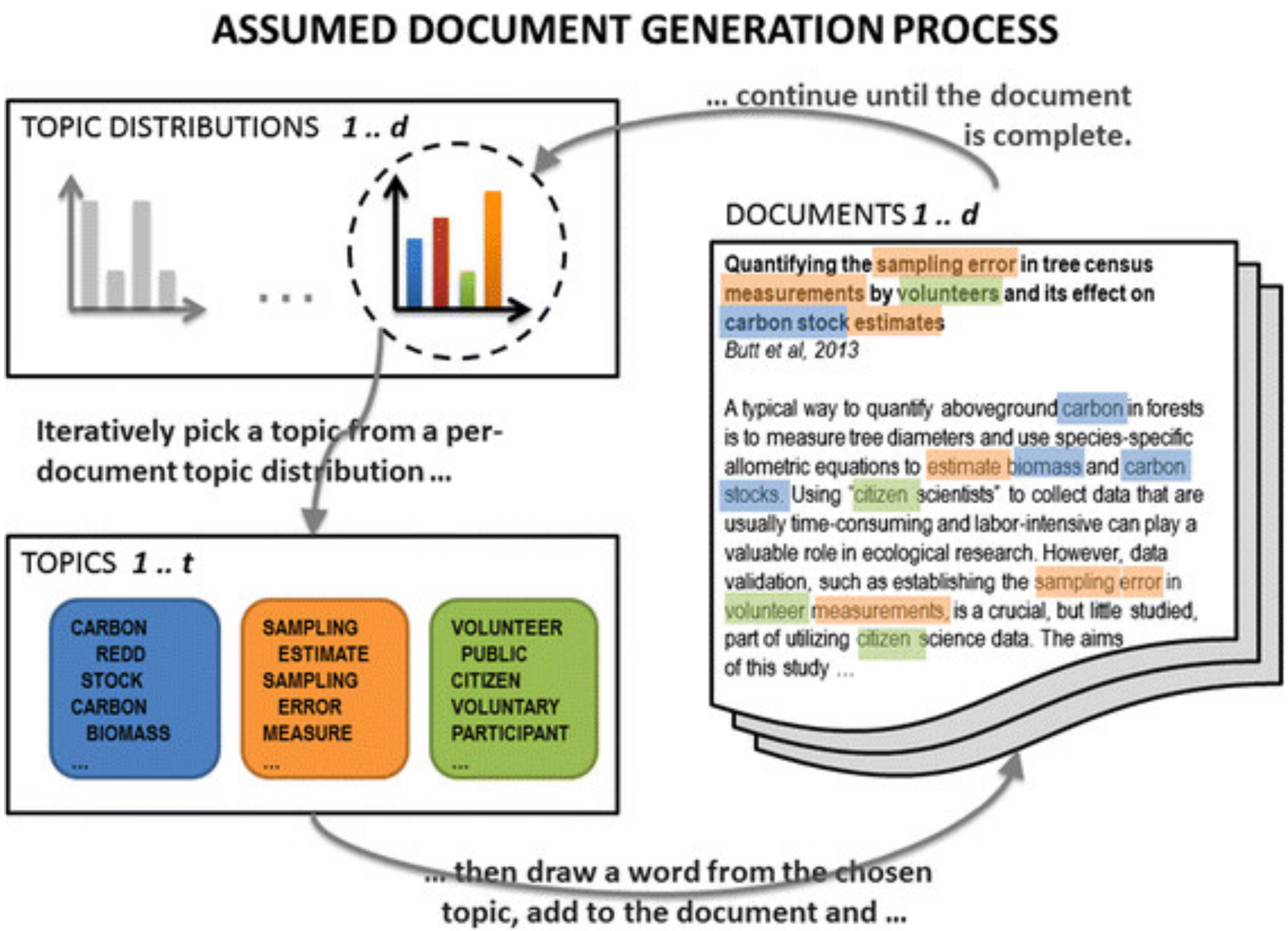
2) Distribution of words across topics also follows Dirichlet distribution with parameter *beta* (β)

- Beta is V-dimensional vector, where V is the number of unique words in the document collection
- Beta determines how words are associated with topics
- Smaller beta favours fewer words strongly associated with a topic
- Beta is an input to the LDA algorithm (i.e. generative process)

LDA's GENERATIVE PROCESS

1) Set the number of topics K and parameters α and β that capture general associations between documents and topics (α), and topics and words (β)
2) For *each document*, pick one sample from a Dirichlet distribution parametrized by α, to obtain the document's distribution over topics
3) For *each topic*, pick one sample from a Dirichlet distribution parametrized by β, to obtain the topic's distribution over the words
4) For each position in each document:
   - Pick a topic from the document's topic distribution
   - Pick a word from a selected topic's word distribution

LDA's GENERATIVE PROCESS – AN ILLUSTRATION

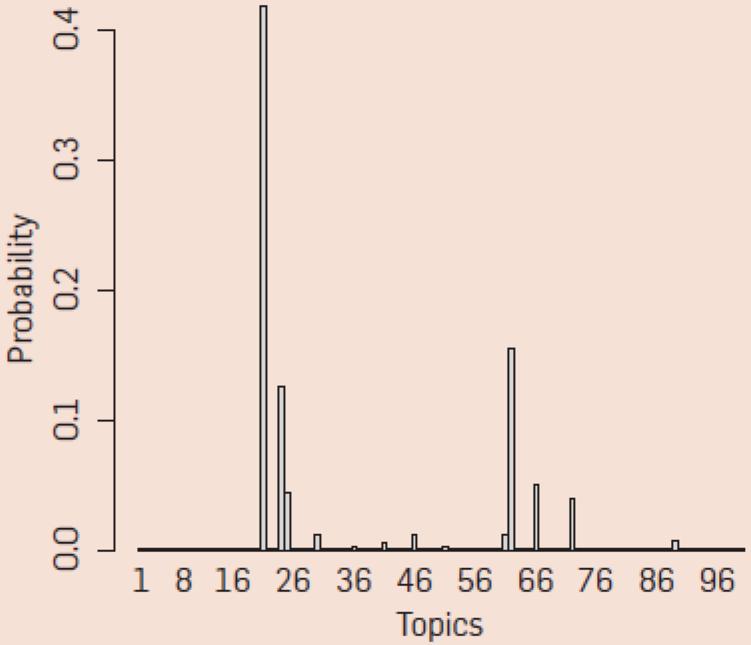


Source:
https://www.researchgate.net/publication/264656298_Assessing_citizen_science_opportunities_in_forest_monitoring_using_probabilistic_topic_modelling

LDA'S GENERATIVE PROCESS

- Computation of the model parameters is intractable, so parameters are estimated typically using:
  - Variational Bayesian methods
  - Gibbs sampling
- An easy to follow explanation of the Gibbs sampling method is given in the Introduction to Latent Dirichlet Allocation blog post

LDA RESULTS



"Genetics"	"Evolution"	"Disease"	"Computers"
human	evolution	disease	computer
genome	evolutionary	host	models
dna	species	bacteria	information
genetic	organisms	diseases	data
genes	life	resistance	computers
sequence	origin	bacterial	system
gene	biology	new	network
molecular	groups	strains	systems
sequencing	phylogenetic	control	model
map	living	infectious	parallel
information	diversity	malaria	methods
genetics	group	parasite	networks
mapping	new	parasites	software
project	two	united	new
sequences	common	tuberculosis	simulations

Real results for the previous example article, obtained by fitting a 100-topic LDA model over 17,000 articles from the Science journal

Source: Blei, David. 2012. Probabilistic topic models. Communications of the ACM 55(4):77–84.

INTERPRETATION OF LDA INFERRED TOPICS

- Topics inferred by LDA are not always easily interpretable by humans
- Several attempts at facilitating the task of topic interpretation
  a) Visualization of the LDA results
  b) Alternative ways for ranking terms within topics
  c) Combination of a) and b)
- An example (of approach (a))
  – Interactive visualization of LDA results (topics, terms) and documents, such as this Wikipedia browser

INTERPRETATION OF LDA INFERRED TOPICS

Alternative measures for ranking terms within a topic

- Lift
  - the ratio of a term's probability within a topic to its marginal probability across the corpus
  - decreases the rankings of globally frequent terms; but, might introduce some noise, by highly ranking very rare terms

- Pointwise Mutual Information (PMI)
  - combines frequency ranking and ranking based on co-occurrence of the frequent terms
  - each of the 10 most probable terms within a topic is ranked in decreasing order of how often they occur in close proximity to the 9 other most probable terms from that topic in some large, external "reference" corpus, such as Wikipedia or Google n-grams

INTERPRETATION OF LDA INFERRED TOPICS

- LDAVis:
  - URL: https://github.com/cpsievert/LDAvis
  - Combines interactive visualization and alternative ways of term ranking
  - Introduces the measure of term *relevance*:

$$r(w,k|\lambda) = \lambda * \log(\phi_{kw}) + (1-\lambda) * \log\left(\frac{\phi_{kw}}{p_w}\right)$$

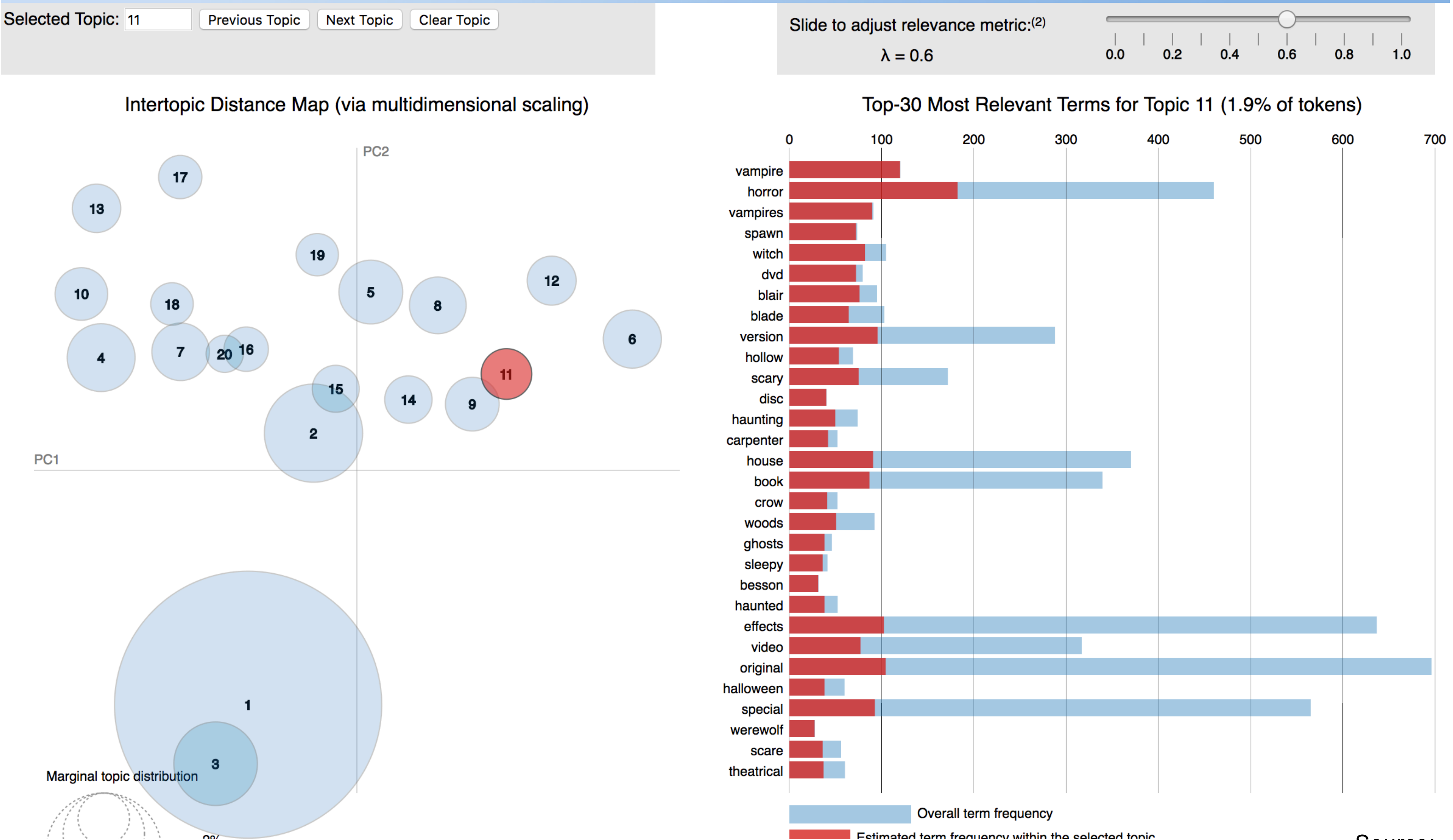
ϕ_{kw} - probability of the term w in the topic k

p_w - probability of the term w in the overall corpus (marginal prob.)

λ - the parameter (0-1); the authors' study found 0.6 to be the best value

Sievert, C. & Shirley, K. (2014). LDAvis: A method for visualizing and interpreting topics. Proc. of the Workshop on Interactive Language Learning, Visualization, and Interfaces. URL: http://nlp.stanford.edu/events/illvi2014/papers/sievert-illvi2014.pdf

LDAVIS EXAMPLE



Source:
http://cpsievert.github.io/LDAvis/reviews/vis/

Check this short talk on LDAVis:
https://speakerdeck.com/bmabey/visualizing-topic-models

Limitations of LDA

Limitations of LDA are rooted in its assumptions:

- bag of words assumption: the order of words in a document does not matter
- the order of documents (in the corpus) does not matter
- the number of topics is known and fixed
- topics are mutually unrelated

Other, more complex topic modeling methods relax these assumptions

TOPIC MODELS BEYOND LDA

- *Dynamic topic model* respects the ordering of the documents in a collection
- *Correlated topic model* allows the occurrence of topics to exhibit correlation
- *Spherical topic model* allows words to be unlikely in a topic
- *Structural topic model* includes document metadata as covariates that might affect
  - topical prevalence - how much a document is associated with a topic
  - topical content – the words used within a topic

SOFTWARE LIBRARIES FOR TOPIC MODELING

- A variety of options in R:
  - lda: https://cran.r-project.org/package=lda
  - topicmodels: https://cran.r-project.org/package=topicmodels
  - stm: http://www.structuraltopicmodel.com/

- Also, several Python libraries:
  - Gensim: https://radimrehurek.com/gensim/
  - lda: http://pythonhosted.org//lda/

- In Java:
  - MALLET Topic Modeling lib: http://mallet.cs.umass.edu/topics.php

GRAPH-BASED METHODS: TEXTRANK

Mihalcea, R. & Tarau, P. (2004). TextRank: Bringing order into texts. In D. Lin & D. Wu (Eds.), Proc. of Empirical Methods in Natural Language Processing (EMNLP) 2004 (pp. 404–411), Barcelona, Spain, July. Association for Computational Linguistics.

GRAPH-BASED RANKING METHODS

- TextRank is a *graph-based ranking method*
- The basic idea behind such methods is that of ‘voting’ or ‘recommendation’:
  - when node A links to the node B, it is basically casting a vote for B
  - the higher the number of votes a node receives, the higher is its importance (in the graph)
  - the importance of the node casting the vote (A) determines how important the vote itself is

TEXTRANK METHOD

- It is based on the Google's original PageRank model for computing a node's importance score:

$$S(N_i) = (1-d) + d * \sum_{j \in In(N_i)} \frac{1}{|Out(N_j)|} S(N_j)$$

$S(N_i)$ – score for node i

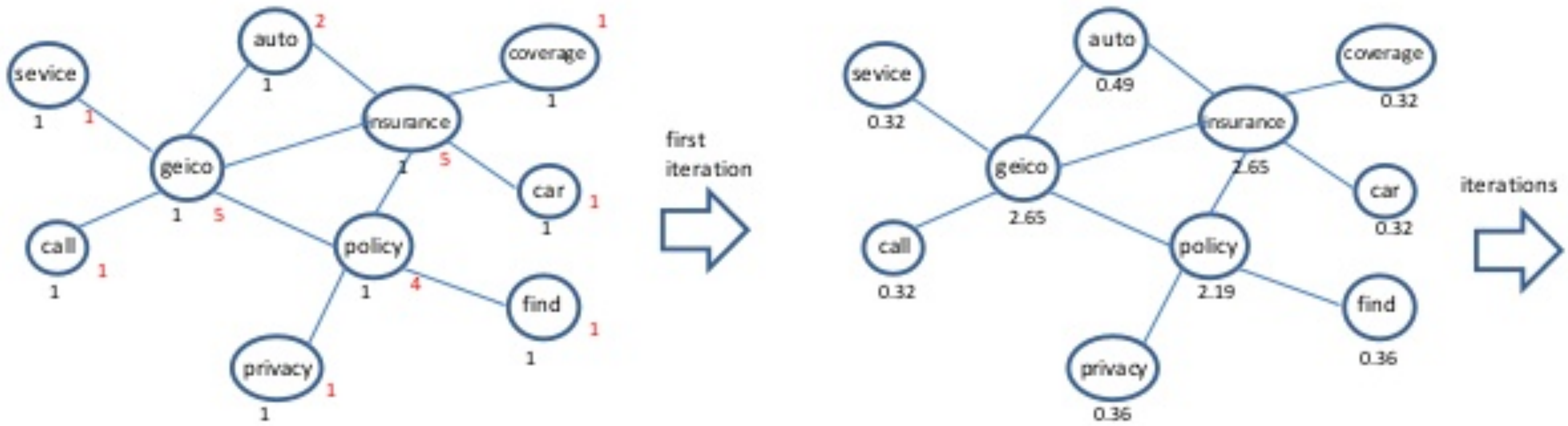
$Out(N_j)$ – the set of nodes that node N_j points to

$In(N_i)$ – the set of nodes that point to N_i

d – the prob. of going from N_j to N_i; 1-d is the prob. of jumping to a random node in the graph (the random surfer model)

TEXTRANK METHOD

$$S(V_i) = (1 - d) + d * \sum_{j \in ngbr(V_i)} \frac{1}{|degree(V_j)|} S(V_j)$$



$$Score(geico) = 0.15 + 0.85 * \left(\underbrace{\frac{1}{1} * 1}_{\text{service}} + \underbrace{\frac{1}{1} * 1}_{\text{call}} + \underbrace{\frac{1}{2} * 1}_{\text{auto}} + \underbrace{\frac{1}{5} * 1}_{\text{insurance}} + \underbrace{\frac{1}{4} * 1}_{\text{policy}} \right) = 2.65$$

$$Score(policy) = 0.15 + 0.85 * \left(\underbrace{\frac{1}{1} * 1}_{\text{find}} + \underbrace{\frac{1}{1} * 1}_{\text{privacy}} + \underbrace{\frac{1}{5} * 1}_{\text{insurance}} + \underbrace{\frac{1}{5} * 1}_{\text{geico}} \right) = 2.19$$

$$Score(service) = 0.15 + 0.85 * \left(\underbrace{\frac{1}{5} * 1}_{\text{geico}} \right) = 0.32$$

Source: https://pt.slideshare.net/JingwenJessicaWang1/clipboards/textrank

TEXTRANK METHOD

- Starting from arbitrary values assigned to each node, the computation iterates until convergence is achieved
  - that is, until $|S^{k+1}(N_i) - S^{k}(N_i)| < \mu$
- After running the algorithm, the score associated with each node represents the node's "importance" within the graph

TEXTRANK FOR WEIGHTED GRAPHS

- In case of weighted graphs, where weights represent the strength of the connection between node pairs, weighted node score is:

$$WS(N_i) = (1-d) + d * \sum_{j \in In(N_i)} \frac{w_{ji}}{\sum_{N_k \in Out(N_j)} w_{kj}} WS(N_j)$$

$WS(N_i)$ – weighted score for node *i*

w_{ij} – weight (strength) of the connection between nodes *i* and *j*

TextRank for keywords extraction

- The input text is pre-processed
  - tokenization, part-of-speech tagging, and stemming/lemmatization
- Co-occurrence (undirected) graph is created
  - a node is created for each unique noun and adjective of the input text
  - an edge is added between nodes (i.e. words) that co-occur within a window of N words ($N \in \{2,10\}$)*
- The ranking algorithm is run
  - initial score for all the nodes is set to 1
  - the algorithm is run until the conversion (typically 20-30 iterations) at the chosen threshold (e.g. $\mu = 10^{-4}$)

*The authors' experiments showed that the larger the window, the lower the precision; N=2 proved the best.

TextRank for keywords extraction (cont.)

- Nodes are sorted based on their final score, and top *T* (or *T%* of) words are taken as potential keywords
- Post-processing: potential keywords are matched against the input text, and sequences of adjacent keywords are collapsed into multi-word keywords
  - E.g. in the text "Matlab code for plotting functions", if both *Matlab* and *code* are among the potential keywords, they would be collapsed into *Matlab code*

TextRank for text summarization

TextRank method can be also used for extracting relevant sentences from the input text, thus, effectively enabling automated text summarization

In this application case:

- nodes of the graph are whole sentences
- edges are established based on the sentence *similarity*

TEXTRANK FOR TEXT SUMMARIZATION (CONT.)

- The intuition:
  - the similarity relation between two sentences can be seen as a act of “recommendation”: a sentence recommends other sentences that address similar concepts
  - the sentences that are highly recommended by other sentences in the text are likely to be more informative for the given text

TextRank for text summarization (cont.)

- Sentence similarity can be measured in many different ways
  - E.g., cosine similarity, longest common subsequence, various string metrics

- The authors' original proposal is based on the content (word) overlap of two sentences S_i and S_j:

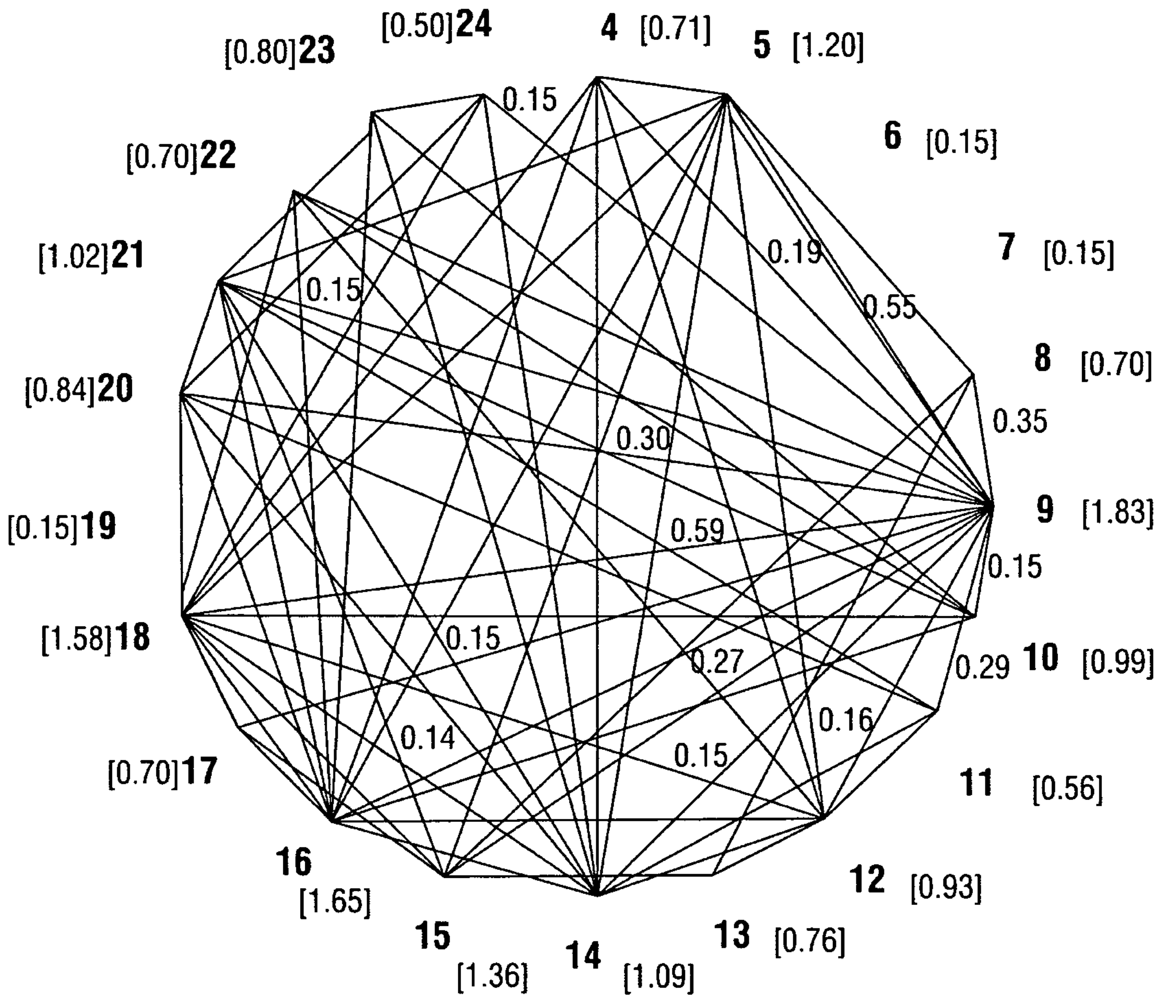
$$Similarity(S_i, S_j) = \frac{|\{w_k | w_k \in S_i \,\&\, w_k \in S_j\}|}{\log(|S_i|) + \log(|S_j|)}$$

The similarity measure uses the length of the sentences as the normalization factor to avoid promotion of long sentences

TEXTRANK FOR TEXT SUMMARIZATION (CONT.)

- The resulting graph is weighted and highly connected
  - edge weights correspond to the computed similarities of the text sentences
  - graph density can be reduced by setting the minimum similarity value for establishing a connection
- The (weighted) ranking algorithm is run on the graph
- Sentences are sorted based on their score
- The top ranked sentences are selected for the summary

EXAMPLE WEIGHTED SENTENCE GRAPH



Source: https://www.google.com/patents/US7809548

Implementation of TextRank

- TextRank method is patented:
  https://www.google.com/patents/US7809548

- No 'official' implementation, but several implementations in different programing languages (Java, Python, R,…)
  - Easy to find by googling it

GRAPH-BASED METHODS: KEYGRAPH

H. Sayyadi, L. Raschid. "A Graph Analytical Approach for Topic Detection", ACM Transactions on Internet Technology (TOIT), 2013

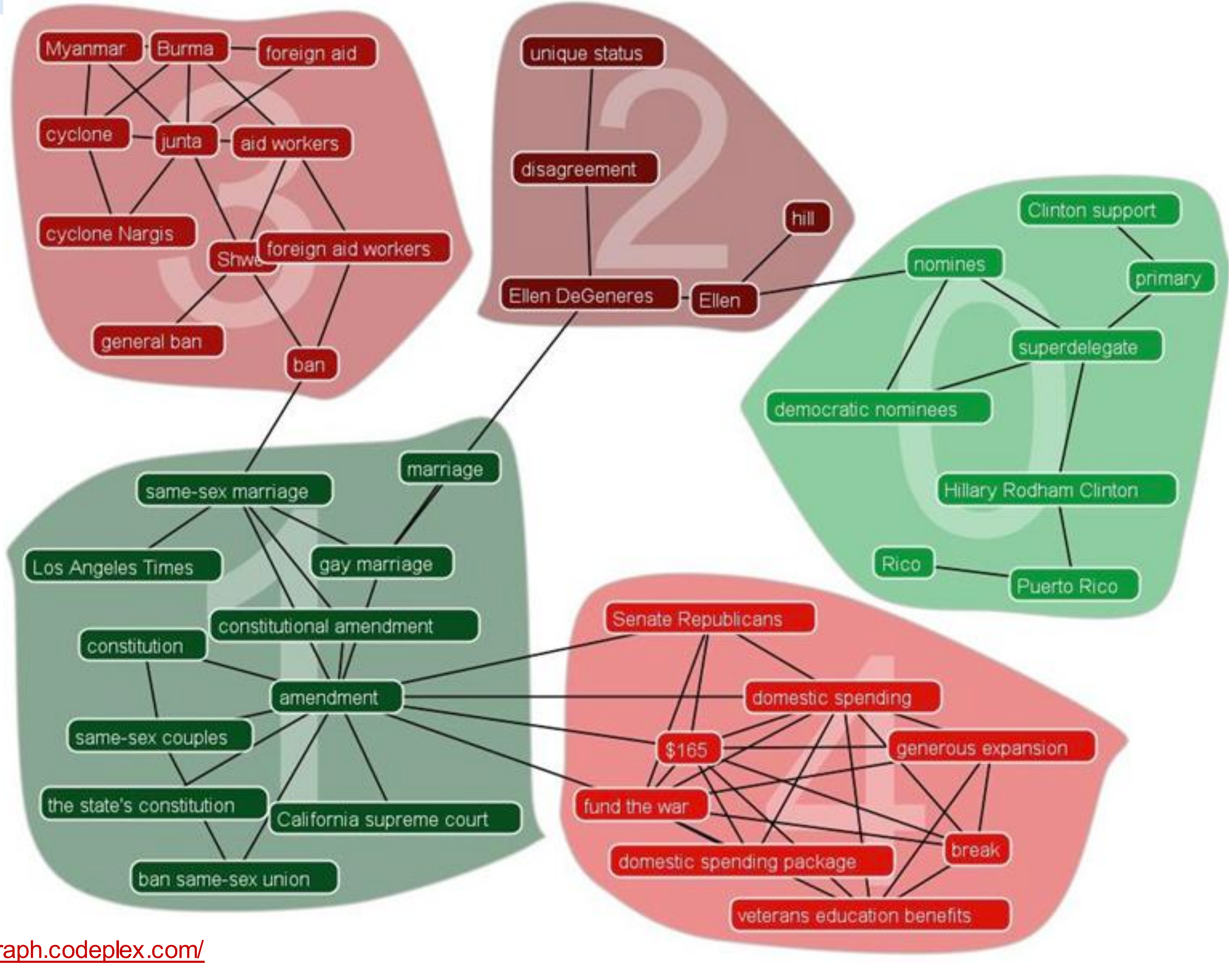
KEYGRAPH IN A NUTSHELL

- Represents a collection of documents as a keyword co-occurrence graph
- Uses an off-the shelf community detection algorithm to group highly co-occurring keywords into "communities" (clusters)
- The detected communities prove to be good proxies for document topics

KEYGRAPH: THE INTUITION

- Keywords co-occur when there is a meaningful topical relationship between them
- Making an analogy to real-world social networks - where people connect if they share a common ‘topic’ (interest, activity, affiliation, etc.) - KeyGraph is modelled as a social network of keywords

ILLUSTRATION OF KEYGRAPH RESULT



Source: http://keygraph.codeplex.com/

KeyGraph Algorithm

1) Build a keywords co-occurrence graph for the given document collection
2) Community detection and extraction of topic features
3) Assigning topics to documents (based on the detected topic features)
4) Merging topics with significant document overlap

KEYGRAPH ALGORITHM: STEP 1

- Create the initial keywords co-occurrence graph
  - nodes are keywords (nouns, noun phrases, named entities) extracted from the corpus
  - an edge is established between two nodes if the corresponding keywords co-occur in at least one document;
  - edges are weighted by the count of the co-occurrences
- The initial graph is filtered based on
  - the document frequency (df) of individual keywords
  - the probability of co-occurrence of each pair of keywords

$$p(k_i|k_j) = \frac{df_{i \cap j}}{df_j} \quad ; \quad p(k_j|k_i) = \frac{df_{i \cap j}}{df_i}$$

KeyGraph Algorithm: Step 2

- Community detection
  - relies on an off-the shelf algorithm for community detection (relational clustering) based on the *edge betweenness centrality (Bc)* metric
  - *Bc* for an edge is defined as the count of the shortest paths, for all pairs of nodes in the network, that pass through that edge
  - in an iterative process, all edges with high *Bc* are removed, thus cutting all inter-community connections and splitting the graph into several components, each corresponding to one (topical) community
- Extraction of topic features
  - the highly co-occurring keywords in each component of the KeyGraph form the features for the corresponding topic

KEYGRAPH ALGORITHM: STEP 3

- Each community of keywords forms a *feature document* f_t, for the corresponding topic t
- The likelihood of the topic t for a document *d* is determined as the cosine similarity of *d* and the feature document ft :

$$p(t|d) = \frac{cosine(d, f_t)}{\sum_{t \in T} cosine(d, f_t)}$$

- Each document can be associated with multiple topics (each with a different likelihood)

KEYGRAPH ALGORITHM: STEP 4

- If case multiple documents are assigned to a pair of topics, it is assumed that those two topics are sub-topics of the same parent topic, and they are merged
- The allowed level of overlap between any two topics is controlled by a parameter (threshold)

ADVANTAGES OF THE KEYGRAPH METHOD

- Comparable performance (precision, recall, F1) to state of the art topic modelling methods
- Capable of filtering noisy irrelevant (social media) posts, thus creating smaller clusters of relevant documents for each topic
- Its running time is linear in the size of the document collection
  - it significantly outruns LDA method on large datasets (>50,000 documents)
- It is robust with respect to the parameters, that is, its performance does not vary much with the change in parameter values

FIND MORE ABOUT KEYGRAPH

- Implementation in Java and further information available at: https://keygraph.codeplex.com/