

Keyword Extraction from News

Aleksandr Safin
Ivan Rodin
Maxim Kaledin

Skoltech

December 22, 2017

Introduction

In applications we want to know

- What goods to recommend given reading history? (advertisers)
- How to decide if news story is relevant for our newspaper? (journalists)
- ...

Introduction

In applications we want to know

- What goods to recommend given reading history? (advertisers)
- How to decide if news story is relevant for our newspaper? (journalists)
- ...

We can use keywords!

News are special

- Small texts (usually no more than 600-700 words)
- Lots of named entities (persons, places,...)
- Not much data with markup

Problem Formulation

Task Definition:

Consider some text T given as input. Then we should produce the subset K of the most 'significant' words.

Example:

*I am a person who works well under pressure.
In fact, I work so well under pressure that at
times, I will procrastinate in order to create
this pressure.*

Stephanie Pearl-McPhee

Problem Formulation

Task Definition:

Consider some text T given as input. Then we should produce the subset K of the most 'significant' words.

Example:

*I am a person who works well under pressure.
In fact, I work so well under pressure that at
times, I will procrastinate in order to create
this pressure.*

Stephanie Pearl-McPhee

Known Approaches

- Graph approaches (centrality, clustering, random walks)
- Latent semantic analysis (Matrix factorizations)
- Statistic approaches (tf-idf, tw-idf,...)
- Machine Learning

Our Approaches

Topic Modelling Methods

EM-algorithm with variations, matrix factorizations

Graph of Words Approaches

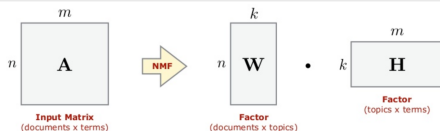
- Community detection based on modularity (Louvain method, Girvan-Newman) and other heuristic approaches (asynFluid communities, KCore, core decomposition)
- Text Rank

Matrix Factorizations

NMF

The task of Non-negative Matrix Factorization (NMF) could be formalized as:

$$\underset{W \geq 0, H \geq 0}{\text{minimize}} \|A - WH\|_F^2$$



Low-rank approximation

The task of low-rank approximation:

$$\underset{\text{rank}(A_r)=r}{\text{minimize}} \|A - A_r\|_F^2,$$

Matrix Factorizations

Consider sentence-term TF-IDF matrix:

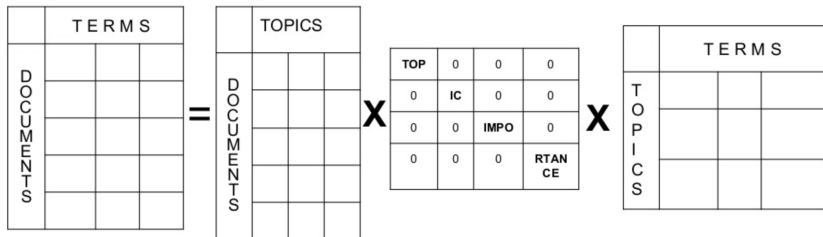
$$A = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{pmatrix}$$

where i^{th} row corresponds to the i^{th} sentence, j^{th} column – to the j^{th} word.

Consider SVD: $A = U\Sigma V^T$, therefore we choose the top words corresponding to the highest values in the first row of V^T .

SVD-based

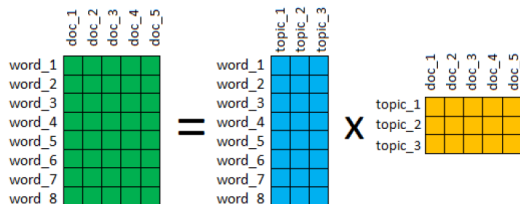
$$A = U\Sigma V^T$$



Main idea:

Choose the top words corresponding to the highest values in the first row of V^T

EM algorithm



$$P_{wd} = F_{wt} \times H_{td}$$

Optimization task for maximization of log-likelihood:

$$\left\{ \begin{array}{l} \sum_{d \in D} \sum_{w \in W} n_{dw} \ln \sum_{t \in T} f_{wt} h_{td} \rightarrow \max \\ s.t. \\ f_{wt} \geq 0, h_{td} \geq 0 \\ \sum_w f_{wt} = 1, \sum_t h_{td} = 1 \end{array} \right.$$

Implementation of EM algorithm

For our task we will use BigARTM - open library for topic modeling of big text document collection.

Advantages of BigARTM:

- Fast and memory-efficient implementation of EM-algorithm
- Built-in quality metrics
- Regularizers to sparse/smooth matrices F and H .

For purpose of finding keywords we have done the following procedure:

- 1 Find F and H
- 2 For document d_i find topic t_{j^*} such that $h_{j^*,i} \geq h_{j,i}$
- 3 Find top k words for topic j^* from matrix F .

We will try this approach for two cases - default EM algorithm and EM algorithm with smooth regularizers.

GoW: definition

Graph of Words:

Let V be a set of words from text (dictionary) and w be a sliding window size.

Graph of words (GoW) $G = (V, E)$ is a graph where $u, v \in V$ are connected if v follows u within a sliding window in the text.

The edge e_{uv} reflects the number of times which word u occurs before the word v within the window of size w .

It is possible to consider directed and undirected versions of this graph.

Our Approaches: TextRank

Denote by r_i the importance of the i -th word:

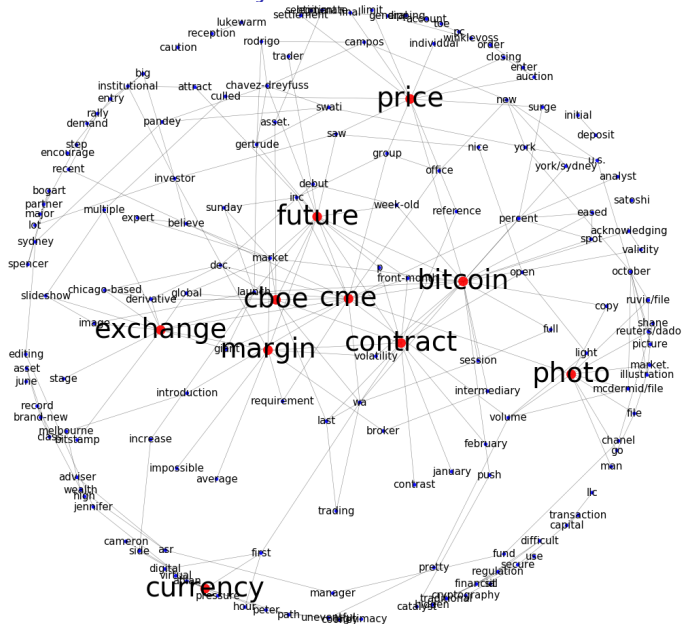
$$r_i = \sum_{j \in N(i)} \frac{r_j}{\# \text{outgoing_edges_of}(j)},$$

where $N(i)$ is the set of output neighbours of i . We want to find the vector r , such that:

$$Ar = r, \quad r \geq 0, \quad \sum_i r_i = 1,$$

where A is the matrix of this system of equations.

TextRank in All Its Glory



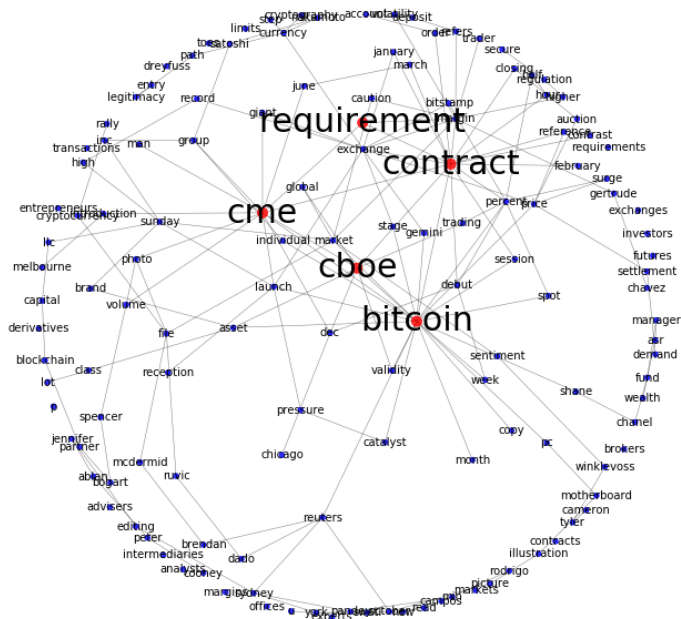
Core-shell cluster

Let us try to find cluster consisting of two parts - the shell S and the core R :

$$\begin{cases} \sum_{i,j}^N \left(A_{ij} - \lambda s_i s_j - (\lambda + \mu) r_i r_j \right)^2 \min \\ s.t. \\ \lambda \geq 0, \mu \geq 0 \\ s_i = \{0, 1\}, r_i = \{0, 1\}, i = 1, \dots, N \end{cases}$$

The problem is NP-hard \Rightarrow we will use heuristic algorithm *AddRemAdd(j)* proposed in [1] to find local optima.

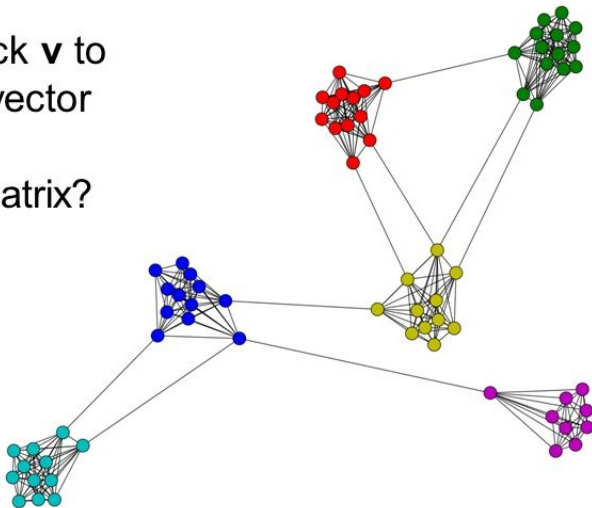
Core-shell cluster



Spectral clustering

$\mathbf{W} \cdot \mathbf{v} = \lambda \mathbf{v} : \mathbf{v}$ is an eigenvector with eigenvalue λ

How do I pick \mathbf{v} to be an eigenvector for a block-stochastic matrix?



Community Detection

Consider an undirected graph (of words) $G = (V, E)$.

"Definition"

Communities are graphs which has more inside edges between its members than outside ones with other vertices.

History

These are "communities" because they have a nice interpretation in Social Network Analysis.

Optimization of modularity

Let c_u denotes the assignment of $u \in V$ to community. Define

$$\delta(c_u, c_v) = \delta_{uv} = \begin{cases} 0 & \text{if } c_u = c_v \\ 1 & \text{otherwise} \end{cases}.$$

The problem of community detection (general) is to choose $c_u, c_v \in N$ in order to maximize *modularity*:

$$Q = \sum_{u,v \in V: u \neq v} \left(A_{ij} - \frac{k_u k_v}{2m} \right) \delta(c_u, c_v),$$

where $m = |E|$, k_u, k_v are vertex degrees. This can be solved as LP with $O(n^3)$ transitivity constraints:

$$\begin{aligned} & \max_{\delta_{uv}} Q, \text{ s.t.} \\ & \delta_{uv} + \delta_{vw} \leq 2\delta_{uw} \quad \forall u, v, w \in V, \end{aligned}$$

which is very hard even if $n \sim 10^2 \Rightarrow$ approximations!

Approximation Algorithms

Girvan-Newman Algorithm

Hierarchically divide the graph into two communities by deleting the *most central* edges and recalculating the centralities of remaining edges. In original version, it is edge *betweenness centrality*: number of shortest paths which contain the edge. The result is a partition with best modularity among produced ones.

Louvain Method

Assign each vertex to its own community. Merge communities, if it increases modularity. Then perform the same on communities: consider them as nodes, which are connected if there are edges between the vertices of the communities. One of the most popular unsupervised algorithms.

Other Algorithms

K-core Decomposition

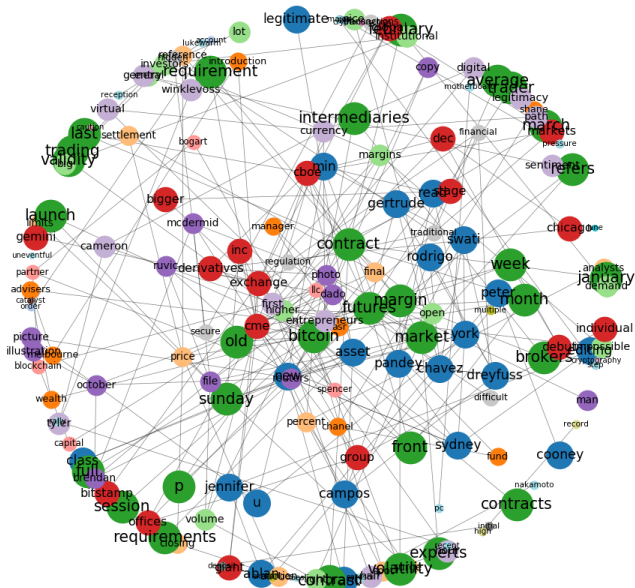
The k -core is a maximal subgraph of G where all vertices have degree at least k . The core corresponding to largest k is called the main core. This is our set of keywords.

Asynchronous Fluid Communities

Set the number of communities k , assign start points at random vertices. Then perform the updates according to defined flow dynamics until maximum number of iterations is achieved. Works only on connected graphs so it is not used in further analysis.

Example (Louvain method)

Reuters news story "Bitcoin hits bigger stage as exchange giant CME launches futures", December 17, 2017



Dataset

We have used **500N-KeyPhrasesCrowdAnnotated-Corpus** dataset for testing GoW approach. This dataset consists of:

- 500 news articles divided into 10 categories
- keywords manually assigned by annotators (each keyword was accepted if it was assigned by at least 90% of annotators phrase)
- name of the article

Results







Quality Measures

- Jaccard index: $K_J = \frac{|A \cap B|}{|A \cup B|}$, where A is set of extracted keywords, B is set of keywords from data;
- Average matches: average number of guessed words.

Approach	Average Matches	Jaccard Index
SVD-based	3.35	0.069
NMF-based	3.59	0.076
TextRank	3.395	0.125
Louvain	4.0133	0.0702
Girvan-Newman	3.9444	0.0694
KCore	6.5444	0.0948
EM	4.15	0.05
EM regularised	4.475	0.054
Core-shell	0.96	0.023
Spectral	6.131	0.073

Table: Comparison of the results for different approaches

References

-  I. Rodin and B. Mirkin, “Supercluster in statics and dynamics: An approximate structure imitating a rough set,” in *International Joint Conference on Rough Sets*, pp. 576–586, Springer, 2017.
-  R. Mihalcea and P. Tarau, “TextRank: Bringing order into texts,” in *Proceedings of EMNLP-04 and the 2004 Conference on Empirical Methods in Natural Language Processing*, July 2004.
-  M. Girvan and M. E. J. Newman, “Community structure in social and biological networks,” *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.
-  C. Peng, T. G. Kolda, and A. Pinar, “Accelerating Community Detection by Using K-core Subgraphs,” *ArXiv e-prints*, Mar. 2014.
-  V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, “Fast unfolding of communities in large networks,” *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, p. P10008, 2008.
-  T. Suzek, “Using latent semantic analysis for automated keyword extraction from large document corpora,” vol. 25, pp. 1784–1794, 01 2017.

Thank you for your attention!

Time performance

Approach	Average time (secs / text)
SVD-based	0.0047
NMF-based	0.0237
TextRank	0.418

Table: Comparison of the results for different approaches