Unsupervised learning for news summarization

how to automatically extract the most important information

Jakub Kubajek

Agenda

1. Introduction
2. Words' importance
3. Embeddings
4. Clustering
5. Summarization

Introduction

Why unsupervised learning?

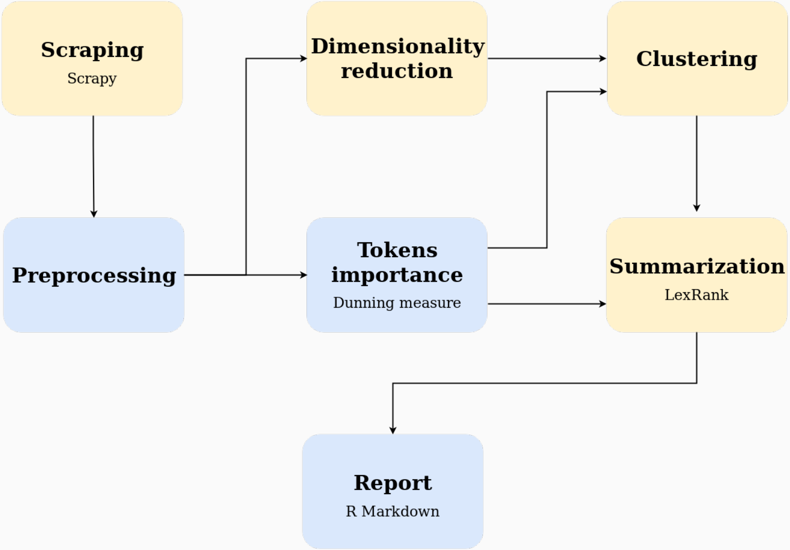
- Lack of pre-trained models for **Polish**
- Shortage of training data in Polish
- Low computing resources
- Multi-document problem - hundreds of articles every day

Definitions of key words

- **Token** - word
- **Lemmatization** - reducing word to its basic grammar form
- **TF matrix** - a matrix that describes the frequency of terms occuring in a set of documents
- **IDF** - Inverse Document Frequency
- **Embedding** - numeric (vector) representation of a word
- **Cosine similarity** - a measure of similarity between two vectors

$$cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \, \|\mathbf{B}\|}$$

Model's overview



Words’ importance

Dunning's measure

General information

- It measures whether token's **frequency** on a particular day is statistically different from that in the reference period
- Likelihood ratio test
- No normality assumption - **binomial** distribution
- Proper measure for **rare** events

Dunning's measure

General information

- It measures whether token's **frequency** on a particular day is statistically different from that in the reference period
- Likelihood ratio test
- No normality assumption - **binomial** distribution
- Proper measure for **rare** events

Equation

$$\lambda = \frac{L(p, k_0, n_0)\, L(p, k_1, n_1)}{L(p_0, k_0, n_0)\, L(p_1, k_1, n_1)} \tag{1}$$

$$L(p, k, n) = p^k (1 - p)^{n-k} \tag{2}$$

where, $p = \frac{k_0 + k_1}{n_0 + n_1}$, $p_i = \frac{k_i}{n_i}$ and **-2*log*$\boldsymbol{\lambda}$** has χ^2 distribution with one degree of freedom. k_0 is the number of a word's occurrences **except for those on** a particular day, k_1 is the word's count **on** a particular day.

Model's implementation

Modification of Dunning measure

- Multiplication by -1, when $p_1 < p_0$

Selecting words for clustering

- $-2log\lambda \geq 10$
- Token count (k_i) larger than the value of 90^{th} percentile of all counts

Embeddings

Latent Semantic Analysis (LSA)

- Singular Value Decomposition (**SVD**) of TF matrix composed of both **paragraphs** and **articles**
- No need for training - algebra
- Dimensionality reduction
- Capturing **semantic** relation between words

Clustering

Clustering

Agglomerative hierarchical clustering

- Start with N topics
- Find the two most similar topics - **cosine** between embeddings
- Merge the two topics - set the new embedding as a **sum** of the two
- Stop when there is 1 topic
- Return the optimal clustering determined by the **silhouette** algorithm

Silhouette algorithm

It aims to find clusters in which objects inside the groups are the most **similar** to each other and **dissimilar** to objects from other clusters.

Equation

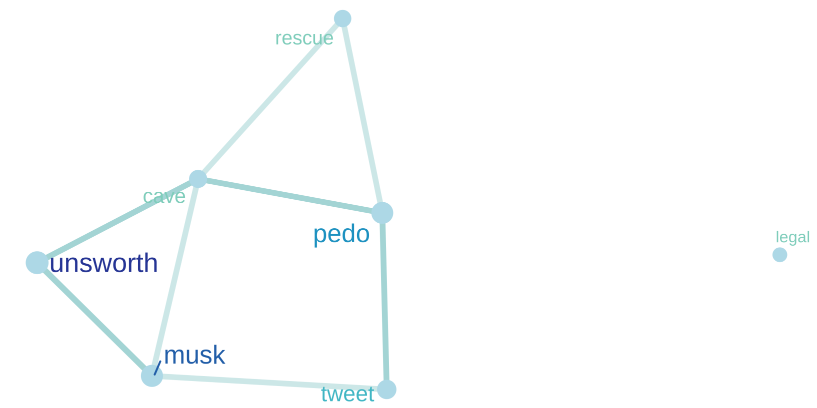
$$a(i) = cos(e_i,\, e_{C_k} - e_i)$$

$$b(i) = \min_{l \neq k} cos(e_i,\, e_{C_l})$$

$$s(i) = \begin{cases} \frac{b(i)-a(i)}{max\{b(i),a(i)\}}, & \text{if } |C_k| > 1 \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

where $a(i)$ and $b(i)$ are inner and outer similarity respectively, e_i is an embedding of i^{th} token, e_{C_k} is the topic's embedding where $i \in C_k$.

Example



Summarization

LexRank

An extractive summarization algorithm, based on Google's **PageRank**, aiming to select sentences that are highly linked (by common words) to other highly linked sentences.

- PageRank
  - Used to rank web pages
  - Ranking equal to the **steady state** of Markov chain
  - Markov matrix obtained as a weighted mean of a normalized matrix of **links** between pages and a **random** matrix ($\frac{1}{N}$ in every cell)
- LexRank
  - Cosine similarity between sentences
  - Uses the **TF-IDF** matrix

Implementation in the model

Modifications

- Filtering articles - minimal topic words' frequency

Implementation in the model

Modifications

- Filtering articles - minimal topic words' frequency
- Filtering sentences - x% of the most similar sentences to the topic

Implementation in the model

Modifications

- Filtering articles - minimal topic words' frequency
- Filtering sentences - x% of the most similar sentences to the topic
- Sentences' **embeddings** used to calculate similarity

Implementation in the model

Modifications

- Filtering articles - minimal topic words' frequency
- Filtering sentences - x% of the most similar sentences to the topic
- Sentences' **embeddings** used to calculate similarity
- Weights of words
    - **TF**
    - Modified **Dunning** measure (D_i^m)
        - $\sqrt{D_i + 3}$ when $D_i \geq -2$
        - 0 otherwise

Implementation in the model

Modifications

- Filtering articles - minimal topic words' frequency
- Filtering sentences - x% of the most similar sentences to the topic
- Sentences' **embeddings** used to calculate similarity
- Weights of words
  - **TF**
  - Modified **Dunning** measure (D_i^m)
    - $\sqrt{D_i + 3}$ when $D_i \geq -2$
    - 0 otherwise
- Scaling the ranking - $F_j = \sqrt{\frac{W_j \cdot D_T^m}{\sum D_T^m}}$
  - where F_j is a scaling factor, W_j is the vector indicating which topic words appear in the j^{th} sentence
  - **Upscale** when a sentence has many important topic words

Implementation in the model

Modifications

- Filtering articles - minimal topic words' frequency
- Filtering sentences - x% of the most similar sentences to the topic
- Sentences' **embeddings** used to calculate similarity
- Weights of words
  - **TF**
  - Modified **Dunning** measure (D_i^m)
    - $\sqrt{D_i + 3}$ when $D_i \geq -2$
    - 0 otherwise
- Scaling the ranking - $F_j = \sqrt{\frac{W_j \cdot D_T^m}{\sum D_T^m}}$
  - where F_j is a scaling factor, W_j is the vector indicating which topic words appear in the j^{th} sentence
  - **Upscale** when a sentence has many important topic words
- Non-duplicated sentences ($cos(\theta) > 0.5$)

Example

Summary

- Two days later, Mr Musk wrote a series of tweets including one describing Mr Unsworth as a **”pedo guy”**.
- Mr Unsworth’s legal team have described Mr Musk’s now-deleted **tweet** as ”vile and false” and are seeking unspecified punitive damages.
- On Thursday, Mr Unsworth told the **court** that Mr Musk’s tweet had left him feeling ”humiliated”.

GitHub

jkubajek/News_Selector

Bibliography i

E. Altszyler, M. Sigman, and D. F. Slezak.
Comparative study of LSA vs word2vec embeddings in small corpora: a case study in dreams database.
CoRR, abs/1610.01520, 2016.

T. Dunning.
Accurate methods for the statistics of surprise and coincidence.
COMPUTATIONAL LINGUISTICS, 19(1):61–74, 1993.

G. Erkan and D. R. Radev.
Lexrank: Graph-based lexical centrality as salience in text summarization.
Journal of artificial intelligence research, 22:457–479, 2004.

Bibliography ii

P. J. Liu, M. Saleh, E. Pot, B. Goodrich, R. Sepassi, L. Kaiser, and N. Shazeer.
Generating wikipedia by summarizing long sequences.
arXiv preprint arXiv:1801.10198, 2018.

R. Mihalcea and P. Tarau.
Textrank: Bringing order into text.
In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411, 2004.

L. Page, S. Brin, R. Motwani, and T. Winograd.
The pagerank citation ranking: Bringing order to the web.
Technical report, Stanford InfoLab, 1999.

Bibliography iii

G. Rossiello, P. Basile, and G. Semeraro.
Centroid-based text summarization through compositionality of word embeddings.
In *Proceedings of the MultiLing 2017 Workshop on Summarization and Summary Evaluation Across Source Types and Genres*, pages 12–21, 2017.