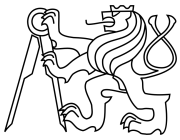
CZECH TECHNICAL UNIVERSITY IN PRAGUE
FACULTY OF INFORMATION TECHNOLOGY



ASSIGNMENT OF MASTER'S THESIS

Title:	Understanding Documents with Text Mining Methods
Student:	Bc. Sergii Stamenov
Supervisor:	Ing. Pavel Kordík, Ph.D.
Study Programme:	Informatics
Study Branch:	Knowledge Engineering
Department:	Department of Theoretical Computer Science
Validity:	Until the end of winter semester 2017/18

Instructions

Explore text mining methods and their applications to document mining.
Design and implement a text mining system that is able to extract labels from documents and cluster documents based on these labels. Explore machine learning methods capable of learning hierarchy of labels and model labels in time. Apply the text mining system to real-world dataset of articles provided by Predictable.ly and a set of documents provided by Profinit. Evaluate the predictive power of label attributes for popularity of articles and for document clustering.

References

Will be provided by the supervisor.

L.S.

doc. Ing. Jan Janoušek, Ph.D.
Head of Department

prof. Ing. Pavel Tvrdík, CSc.
Dean

Prague February 22, 2016

Czech Technical University in Prague
Faculty of Information Technology
Department of Information Technology

Master's thesis

Understanding Documents with Text Mining Methods

Bc. Stamenov Sergii

Supervisor: Kordik Pavel

19th January 2017

Acknowledgements

Many thanks to Geneea.com for NLP related consultations and annotations, Recombee.com for data from their recommendation engine. Many thanks to contributors of open source libraries used in this work.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as school work under the provisions of Article 60(1) of the Act.

In Prague on 19th January 2017 .....

Czech Technical University in Prague
Faculty of Information Technology

Citation of this thesis

Stamenov, Sergii. *Understanding Documents with Text Mining Methods.* Master's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2017.

Abstrakt

Klíčová slova jsou výrazy které popisují téma dokumentu. Používají se také pro shrnutí dokumentu nebo při optimalizaci jejich vyhledávaní. Cílem této práce je prozkoumat možnosti vylepšení extrakce klíčových slov a jejich použití při shlukování a klasifikaci dokumentu. Podařilo se prozkoumat možnosti vytváření ontologie na základě klíčových slov a modelování klíčových slov v čase. Navíc bylo v práci ukázáno, že metody text-miningu lze v omezené míře použít i pro predikci délky života článku. Dále byl navržen a implementován text miningový systém, který je prostřednictvím webových služeb schopný extrahovat klíčová slova z novinových článků a shlukovat je dle postupů, které byly vyhodnoceny jako nejúspěšnější.

Klíčová slova text mining, kličové slovo, word2vec, shluková analýza

Abstract

Keyword is a term that captures topic of a document. They can be used for quick article summarization or search optimization. In this work we test whether keywords can improve quality of document clustering and document classification compared to unigrams. Futher we describe text mining system that can extract keywords from a document and cluster documents as the

part of back-end of news web portal. We also explore possible methods of organizing keywords into hierarchy and visualize them in time.

Keywords keyword extraction, keyword hierarchy, word2vec, word embeddings, document clustering, regression, document visualization, tf-idf

Contents

List of Figures

List of Tables

Introduction

Text mining is the process of extraction information from a text. It includes text classification, document clustering, sentiment analysis, document summarization and named entity extraction. Large collections of text are hard to process manually. Text mining is used in many fields: to cluster documents in biomedical research, power chat-bots and personal assistants, to monitor opinions on various topics such as elections, legal initiatives or incidents. Text mining helps companies to keep track of the feedback on the quality of their services on social networks. It helps Amazon to understand better what user likes and improve recommendations, it helps Google provide users with personalized ads that meet their needs. As we can see, the area is very broad.

One of the fields in which text mining can be useful is online journalism. This field produces large amounts of text data every day and requires manual annotations to improve users reading experience. Most of the big players like BBC, Associated Press and New York Times are doing the same thing. we would like to address two tasks of text mining in this area: keywords extraction and document clustering. Keyword extraction can provide high level overview of an article and serve as summary of the text. Document clustering information may help readers to discover related articles on the same or related topic.

Keyword is phrase that consists of one or more terms describe what article is about. Keywords can explain article's content in few words. By reading only keywords readers can decide whether this article is in their area of interest. Given this descriptive property of keywords, we would like to check whether keywords can aid machine understanding of the material and improve quality of the document clustering and document classification tasks compared to more traditional text representation that uses all words in the document.

As we said keywords are a very useful way to summarize articles. Manual annotation of articles is time consuming, error prone and inconsistent. Because our language is so flexible and allows us to describe the same idea using different sets of words it is harder for readers to find related articles written

by different authors. The same is true for document clustering, it is hard and time consuming for editors to decide which cluster an article belongs to. To solve this problem we will design and implement text mining system that is able to automatically annotate articles with keywords and cluster information, with human feedback loop that will improve quality of the system results.

Text mining methods can help editors as well as readers. We can help writers to do their job more efficiently by providing them with tools for exploratory analysis of the topics in collection of documents. Writers explore how topics were changing through time to decide whether some story should be investigated further or there is no interest in it. Writers need the way of organizing topics into some structure that puts related topics together. We will explore methods of text visualization and ontology organization.

Chapter 1

Overview of natural language processing techniques for document processing

To achieve desired goals of this work we need to describe some text processing techniques used in this work. These techniques are standard across the fields of use and are used in various combinations to achieve desired goal.

1.1 Stemming

Stemming is a technique for removing endings from a word, reducing it to the stem. The stem is not identical to morphological root of the word. For example stemming algorithm can be used to reduce word *cars* to *car* or *walking* to *walk*. It allows to dramatically reduce size of the vocabulary in indexing task, instead of keeping every form of a word that we encountered in the corpus we'll keep only one.

One of the most-used stemming algorithm is Porter Stemming Algorithm, by Martin Porter. It is a set of linguistic rules designed to remove and replace well known English endings.

1.2 Named entities recognition

Named entity recognition (NER) is a subtask of information extraction that detects and classifies named entities in the text into categories like Location, Organization, Person and others. NER system takes a unannotated block of text as input and produces annotated block of text with highlighted named entities. For example in the sentence " Jim bought 300 shares of Acme Corp. in 2006." system may identify that Jim is a person and Acme Corp. is an organization and 2006 is time indicator.

1.3 Vector space model and Bag of words

To be able to work and compare documents of various length we need to develop some common representation of the documents. One of such representations is a vector space model. In this model document is represented by a fixed-length vector. Vector's length is determined by vocabulary we are using. *I-th* value of the vector indicates that *i-th* vocabulary word is present in the document and it is weight. We can use simple weighting and only take into account that word is present in the text or we can use raw counts how many times word occurs in the document.

1.4 Tf-Idf weighting

Another widely used weighting scheme is tf-idf. Tf-idf stand for term frequency-inverse document frequency, tf measures number of times that term t occurs in document d. inverse document frequency measures spread of term t, in other words measuring how popular the term is across whole corpus. For term t inverse document frequency is defined as:

$$idf_t = \log \frac{N}{df_t}$$

where N the total number of documents in the corpus and df_t is the number of documents that contain term t. And the tf-idf weighting is computed as:

$$tfidf_{t,d} = tf_{t,d} \times idf_t.$$

where $tf_{t,d}$ is the number of occurrences of term t in document d. [1].

1.5 LSA

Latent semantic analysis or "latent semantic indexing" (LSI) analysis uses singular-value decomposition. It takes large matrix of term-document association data and constructs a "semantic" space wherein terms and documents that are closely associated are placed near one another. Singular-value decomposition allows the arrangement of the space to reflect the major associative patterns in the data, and ignore the smaller, less important influences. As a result, terms that did not actually appear in a document may still end up close to the document, if that is consistent with the major patterns of association in the data. Position in the space then serves as the new kind of semantic indexing, and retrieval proceeds by using the terms in a query to identify a point in the space, and documents in its neighborhood are returned to the user. (Dumais et al.) [2]

1.6 Cosine similarity

Cosine similarity is the most common way to define similarity between two documents in the corpus. It is defined as:

$$sim(d_1, d_2) = \frac{\vec{V}(d_1) \cdot \vec{V}(d_2)}{|\vec{V}(d_1)||\vec{V}(d_2)|}$$

where the numerator represents the dot product, while the denominator is the product of their Euclidean lengths to length-normalize vectors. [1].

1.7 Keyword extraction algorithms

Keyword or key-phrase provide high-level information about main topics discussed in given article. Keyword may be single word or multi word phrase usually named key-phrase. The goal of this task is to automatically find and extract specific terms in the article that describe article's content best.

1.7.1 General keyword extraction algorithm

Any keyword extraction algorithm has following phases:

1. Candidate generation. Extraction of all possible candidates from source text. The simplest approach is to split text into tokens based on whitespace or punctuation. For the simplicity as candidates we will use unigrams, NEs and noun phrases. This approach gives good result and it is fast. It can be further extended using more complicated preprocessing steps.

2. Properties calculation. Computing properties and statistics required for ranking.

3. Ranking. Score computed for each candidate, candidate list sorted in descending order and some limiting applied.

1.7.2 Corpus information based keyword extraction

This approach uses frequency information about terms extracted from all documents in the corpus and some heuristics at post-processing. As ranking score we will use tf-idf weighting. Algorithm described as follows:

1. For a given document generate list of candidates.

2. For each candidate term compute get its tf-idf score.

3. Sort candidates in the descending order.

4. Take top-N candidates and perform post-processing.

In the post-processing phase we remove duplicates by merging adjacent entities into multi-word expression. For example, if in the sentence *Google to Launch Mobile Network*, if both *mobile* and *network* are on the top-N list, then they will form new keyword *mobile network* instead of two separate keywords. We also will use lemmatization to remove possible duplicates caused by variations of the same word: to reduce *networks* into *network*.

We also must take into the account domain specific facts about dataset. News articles often contain names of locations, some famous persons or organizations. Those names are useful as keywords. User might be interested in other articles about same person or organization. For that reason we will include into keyword candidate list output of NER-tagger.

Here we define topic of the document as the set of keywords that capture high-level summary of the document. The alternative way to define a topic of the document is to use probabilistic model. In such model a topic is defined as a distribution over fixed vocabulary. And documents are assumed to have multiple topics. Each document then is represented a mixture of those topics. This method is called *Latent Dirichlet Allocation* (LDA) and was introduced by Blei et al. (2003) [3]. This is different approach than we take and is out of scope of this work.

1.7.3 Document based keyword extraction

RAKE is the keyword extraction algorithm that operates on single document proposed by Rose et al. (2010) [4]. This approach has many advantages: scalability and context awareness. RAKE is based on assumption that keywords rarely contain stop words, such as functional words *and*, *the* or other very frequent with minimal lexical meaning. The input to algorithm is the stop words list, list of phrase delimiters and words delimiters. Candidates are generated as words that are between those delimiters. Then co-occurrence graph is constructed and scores for each keyword are computed from the sum of the scores of an individual words. Possible metrics are: word frequency $freq(w)$, word degree $deg(w)$ and ratio of two $deg(w)/freq(w)$. To be able to identify keywords that might contain stop words algorithm also looks for a pair of words that occur in the same order in the text at least twice and adds them into candidate list.

1.7.4 Graph based approach aka TextRank

Graph based approach uses TextRank algorithm proposed by R Mihalcea et. al. [5] which is modification of famous PageRank algorithm developed in 90s by Larry Page and Sergej Brin [6], applied to document words. One of the proposed applications for the algorithm was keyword extraction.

Algorithm works as follows:

1. Identify application specific unit of text and add them as nodes to the graph.
2. Identify connections that will define the edge between nodes. Edges can be directed, undirected or weighted.
3. Run the iterative graph-based algorithm until it converges.
4. Sort values based on rank in descending order.

Specifically, for keyword extraction task nodes would be single words (to prevent graph's exploding) and multi-word keywords are reconstructed in the post-processing phase, edges are defined using co-occurrence relation in the window of size N, where N can be small number from 2 to 5-7. Empirical results suggest than the strength of co-occurrence influence is decreasing dramatically as the distance between words increases.

Further preprocessing tasks may be taken. For example one might filter out node with specific POS-tags. Usually verbs are considered as bad candidates for keywords. Best results observed in original paper are obtained using nouns and adjectives. Algorithm's implementation provided by gensim [7] open source library.

1.8 Keyword hierarchy and modeling time

The most popular method to organize entities into hierarchy is to use agglomerative clustering algorithms. The main challenge is to come up with distance metric between entities. In this section we will describe agglomerative clustering algorithms and two ways to define word distance using WordNet database and word embeddings.

1.8.1 Agglomerative clustering

Agglomerative clustering is the non-parametric clustering algorithm that starts with every data point as single cluster and merges the closest pairs of clusters to form new cluster until only one cluster remains. The question is how to compute distance between two clusters. There are two standard algorithms for doing so:

- Single linkage
- Complete linkage

In the single linkage mode we compute distance between the most similar members of each pair of clusters and merge the two clusters for which the distance between the most similar members is the smallest. The complete linkage approach is similar to single linkage but, instead of comparing the

most similar members in each pair of clusters, we compare the most dissimilar members to perform the merge. [1].

1.8.2 Wordnet

WordNet is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations. The resulting network of meaningfully related words and concepts can be navigated with the browser. WordNet is also freely and publicly available for download. WordNet's structure makes it a useful tool for computational linguistics and natural language processing. WordNet superficially resembles a thesaurus, in that it groups words together based on their meanings. However, there are some important distinctions. First, WordNet interlinks not just word forms—strings of letters—but specific senses of words. As a result, words that are found in close proximity to one another in the network are semantically disambiguated. Second, WordNet labels the semantic relations among words, whereas the groupings of words in a thesaurus does not follow any explicit pattern other than meaning similarity. [8]

1.8.3 Word embeddings

A word embedding method discovers distributed representations of words; these representations capture the semantic similarity between the words and reflect a variety of other linguistic regularities. [9]. Each term in a vocabulary is associated with two latent vectors, an embedding and a context vector. These two types of vectors govern conditional probabilities that relate each word to its context, i.e., the other words that appear around it. Specifically, the conditional probability of a word combines its embedding and the context vectors of its surrounding words. (Different methods combine them differently.) Given a corpus, we fit the embeddings and the context vectors by maximizing the conditional probabilities of the observed text.

1.8.4 Inverted index

The inverted index is a special data structure that for each token in the corpus contains references to all documents it is used in. The inverted index is used in many information retrieval tasks to speed up full text search. Index record itself may contain reference to a document or positions of the token in the document. It also can be used to visualize trends and popularity of the term, for example Google trends web portal shows recent history, figure 1.1 details example on that.

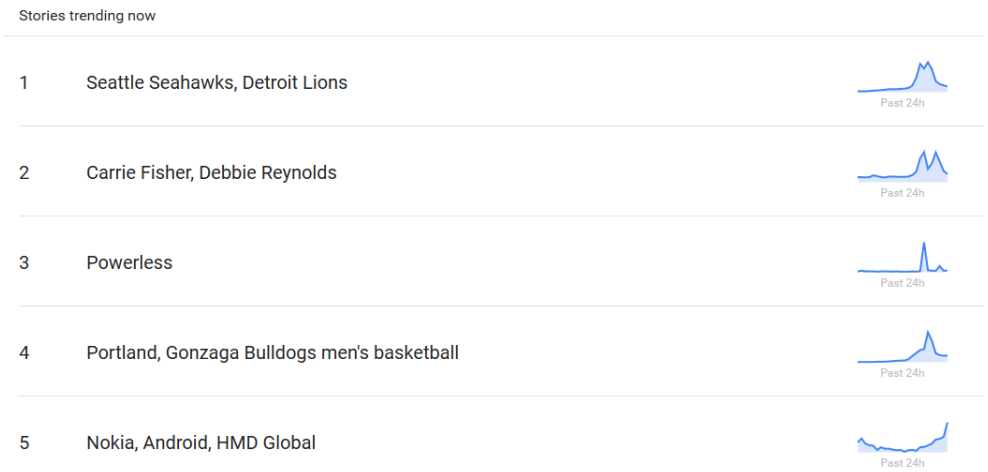


Figure 1.1: Example of the term trend visualization by Google Trends. Here we can see recent history of the term.

1.9 Document clustering algorithms

In the area of information retrieval K-means is de facto standard algorithms for document clustering. K-means is more efficient and popular algorithm in academia and industry. Algorithm minimizes the average squared distance of points in the dataset from cluster centers, also called *centroids*. K-means assumes that clusters have convex shape. K-means can be defined as simple optimization problem for minimizing the within-cluster sum of squared errors:

$$SSE = \sum_{i=1}^{n} \sum_{j=1}^{k} w^{(i,j)} \|x^{(i)} - \mu^{(j)}\|_2^2$$

Where $\mu^{(j)}$ is the centroid for cluster j, and $w^{(i,j)} = 1$ if sample $x^{(i)}$ is in cluster j, and 0 otherwise.

Algorithm has following steps:

- Randomly pick k centroids as initial guess for cluster centers.
- Assign each data point to nearest centroid.
- Move the centroids to the center of the data points that were assigned to it.
- Repeat the steps 2 and 3 until cluster assignment do not change or a maximum number of iteration is reached.

One of the biggest challenge with K-means is to pick up right number of clusters k. Since we don't have any information about documents classes or

categories we will use internal criterion of quality *SSE* metric as our estimate of clustering quality. [1]

1.10 Prediction algorithm

In this section we will briefly describe machine learning algorithms used for text classification and regression.

1.10.1 Naive Bayes

Algorithms widely used in production as base spam classifier in text processing. It is a probabilistic method that is based on naive assumption of conditional independence of features. The probability of a document d being in class c is computed as

$$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c)$$

where $P(t_k|c)$ is the conditional probability of term t_k occurring in a document of class c. $P(c)$ is a prior probability of a document occurring in class c. $t_1, t_2, ..., t_{n_d}$ are the tokens in d that are part of the vocabulary we use for classification and n_d is the number of such terms.

We classify document into class that has maximum probability. We don't know true probabilities $P(c)$ and $P(t_k|c)$ but estimate them from test set using maximum likelihood estimate. For the priors we use:

$$\hat{P}(c) = \frac{N_c}{N}$$

Where N_c is the number of documents in class c and N is the total number of documents in the corpus. For conditionals we use:

$$\hat{P}(t|c) = \frac{T_{ct} + 1}{\sum_{t' \in V}(T_{ct'} + 1)}$$

where T_ct is the number of occurrences of t in training documents from class c. 1 is used for smoothing and protecting from having zero-probability for yet unseen words and is called Laplace smoothing. Prediction is done using maximum a posteriori class c_{map}:

$$c_{map} = \arg\max_{c \in C} [\log \hat{P}(c) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k|c)]$$

Logarithm is introduced to improve computational stability. It is better to sum logarithms than multiply probabilities. [1]

1.10.2 Random Forest

A random forest is an ensemble of decision trees. The main idea is to combine weak learners to build more robust model, a strong learner, that has better generalization and is less subjected to over-fitting. The random forest algorithm can be described in four steps:

1. Draw a random bootstrap sample of size n (randomly choose n sample from training set with replacement).
2. Grow a decision tree from bootstrap sample. At each node:
   - Randomly select d features without replacement.
   - Split the node using the feature that provides the best split according to the objective function.
3. Repeat the steps 1 to 2 k times.
4. Aggregate the prediction by each tree to assign class label by majority vote.

Where K is the number of weak learners (trees) and d is feature subset size for each tree to consider. Reasonable default for d is $\sqrt{m}$ where m is the total number of features in the training set.
[10]

1.11 Metrics

In this section we will define metrics used to evaluate quality of prediction models. We will describe metrics for regression and classification tasks. One of the metrics used to evaluate regression model is the coefficient of determination or R^2, it is a number that indicates the proportion of the variance in the dependent variable that is predictable from the independent variable. For real valued target variable y, R^2 is defined as:

$$R^2 = 1 - \frac{\sum_i (y_i - \bar{y})}{\sum_i (y_i - f_i)^2}$$

where $\bar{y}$ is the mean of the observed data and f_i is the i-th predicted value.

To evaluate classification task with unbalanced classes in the dataset we will use F measure. The F measure (F score) is defined as weighted average of the precision and recall. For classification task precision is defined as:

$$\text{Precision} = \frac{\text{True positive}}{\text{True positive} + \text{False positive}}$$

Recall is defined as:

$$\text{Recall} = \frac{\text{True positive}}{\text{True positive} + \text{False negative}}$$

And F measure is defined as:

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Chapter 2

Data

Through this work we will use two news data sets. One is from anonymous US news portal, this dataset contains articles in English language. Second dataset is from Czech news portal and is in Czech language. First dataset contains 8500 news articles for six month period from March to August 2015, second dataset contains 10000 annotated news articles from July to November 2016. For each article dataset contains title, main text, human annotated keywords, date and time of the article's publication. First dataset also contains page views information for each article. Table 2.1 shows summary statistics about datasets used in this work including number of documents n, the bag-of-words dimensionality for unigrams and for keywords, average number of words per document and average number of human annotated keywords per document.

Table 2.1: Dataset characteristics, used in evaluation.

Name	n	BOW unigrams	BOW keywords	WORDS (AVG)	KEYWORDS (AVG)
Crowd500	450	-	-	9	50
English news	8847	22972	10402	312	-
Czech news	10799	25317	8992	212	30

Chapter 3

Keyword extraction

Keyword or key-phrase provide high-level information about main topics discussed in given article. Keyword may be single word or multi word phrase usually named key-phrase. The goal of this task is to automatically find and extract specific terms in the article that describe article's content best. Keywords can serve multiple goals:

- Summarization - keywords enable the reader to understand whether the given article is in his field of interest.
- Indexing - keywords speed up searching in catalogs and enable automatic catalog construction.
- Search optimization - keyword search is faster than full text search.
- Key-phrase highlighting in the text - provide the reader with visual hints on most important terms in the documents. Whether in **bold**, *italics*, by underlining text or using some background highlight.

In this section we will evaluate keyword extraction algorithms on real-world data sets. We extracted keywords from article's title and main text. Since first dataset does not have human labeled keywords for benchmarking purpose we will use similar academic news dataset with knows keywords by Luis Marujo et al. (2012) [11]. We will use F measure for evaluation of keywords extraction algorithms. For this task we need to redefine precision and recall metrics. In the area of information retrieval precision is defined as the fraction of retrieved keywords that are relevant to the document:

$$precision = \frac{\{\text{relevant keywords}\} \cap \{\text{retrieved keywords}\}}{\{\text{retrieved keywords}\}}$$

And recall is defined the fraction of the keywords that are relevant to the document that are successfully retrieved:

Table 3.1: Keyword extraction algorithm's performance comparison on Crowd500 dataset.

Algorithm	Precision	Recall	F-measure
RAKE	21.7	32.2	25.93
TextRank	19.6	18.2	18.88
Corpus based	19.7	34.5	25.07

Table 3.2: Keyword extraction algorithm's performance comparison on Czech news dataset.

Algorithm	Precision	Recall	F-measure
RAKE	14.53	17.96	16.06
TextRank	10.0	13.42	11.45
Corpus based	13.58	22.04	16.8

$$recall = \frac{\{\text{relevant keywords}\} \cap \{\text{retrieved keywords}\}}{\{\text{relevant keywords}\}}$$

One big issue of all algorithms is that they can not do inference of new keywords or generalization, they can not come up with some new keywords that are not explicitly mentioned in text but have some relation to it. That is why perfect recall is not possible, because human annotators can use words that do not appear in the text.

Tables 3.2 and 3.1 compare results of keyword extraction algorithms. As we can see Rake algorithms performs better than TextRank algorithm and it is preferable than corpus based approach because it does not require additional global information. We also can see that all algorithms perform worser on Czech language corpus than English this is because current implementations have some language-specific logic built into it that can not be changed using parametric calls. Further in this work we will use RAKE algorithm for keyword extraction tasks.

Sample keyphrases extracted from article 'will the next iphone have a 3d camera?'

apple has filed for a patent that would involve taking multiple pictures and meshing them together to create a 3-dimensional image - a process that would include the use of two cameras instead of one - according to appleinsider. it remains to be seen as to whether apple will jump on the 3d bandwagon, but many companies have already embraced the technology. there are actually a number of phones on the market that already use 3d cameras and displays. the evo 3d, which debuted at ctia wireless 2011, uses a pair of cameras on the back of the phone to capture high-definition 3d video. the patent indicates that the technology would allow the iphone to shoot still images in 3d and also record high-definition 3d video, according to the report. both the evo 3d and nintendo's latest handheld gaming device, the 3ds, uses a technology called a parallax barrier - which basically only lets each eye see a certain set of pixels ont he screen. that means that each eye sees something different, and the brain combines the images. the net effect is an illusion of depth and a 3d image without needing cumbersome 3d glasses. the patent doesn't indicate whether apple's next iteration of the iphone would carry a similar screen using parallax barrier technology. but 3d has not emerged as the dominant new trend in display technology, as there are some complaints about whether it places undue strain on eyes. the nintendo 3ds, for example, warns users to take a break from time to time so they do not strain their eyes. apple is expected to delay the release of its next iphone until the fall. that would give the company more than enough time to introduce a number of new pieces of technology to the device, including access to the next generation of wireless networks. tags: 3d, 3ds, apple iphone, evo 3d, iphone, nintendo 3ds companies: apple, nintendo copyright 2011 venturebeat. all rights reserved. venturebeat is an independent technology blog.

Keywords assigned by Rake: definition 3d video; places undue strain; independent technology blog; parallax barrier technology; nintendo 3ds companies; parallax barrier; 3d camera; 3d image; evo 3d; 3d bandwagon; including access; rights reserved; brain combines; net effect; wireless networks; pixels ont; record high; ctia wireless 2011; dimensional image; warns users; capture high; nintendo 3ds; display technology; 3d cameras; eye sees; similar screen; apple iphone; nintendo'; companies; strain; 3ds; technology; screen; eye; cameras; apple; iphone; apple'; lets; device; include; dominant; means; read; introduce; break; fall; report; release; set; generation; phones; back; jump; displays; images; emerged; appleinsider; tags; number; pair; embraced; carry; market; remains; create; trend; venturebeat; expected; eyes; company; phone; basically; filed; meshing; shoot; iteration; give; process; pieces; patent; debuted; delay; illusion; complaints; depth; time

Keywords assigned by human annotators: eyes; iphone; 3d; video; nintendos; apple; glasses; strain; pictures; parallax; technology; patent; display; highdefinition; phones; cameras; illusion; wireless networks; involve; break; time; basically; doesnt indicate; 3d camera; bandwagon; next generation; screen; companies; number; certain set; create; include; meshing; venturebeat; delay; embraced; capture high-definition; give; pair; multiple pictures; use; time to time; including access; handheld gaming; warns users; appleinsider; pixels; evo; process; independent; next iphone; brain combines; something different; debuted; filed; phone; two cameras; introduce; iteration; parallax barrier; net effect; eye; shoot still images; without needing cumbersome; depth; jump; market; company; release; allow; new pieces; 3dimensional image; expected; ctia wireless 2011

Chapter 4

Learning hierarchy of keywords and time modeling

Depending on the algorithm, the size of the text and cutoff threshold, we might get too much keyword candidates to display to the user at once. We can sort keywords for relevancy and apply paging, but then related terms may be pages of each other just because scoring function was not taking into account similarities between keywords. Instead it is good idea to display all keyword candidates to the user at once, but organized in hierarchy with more general terms at the top and more specific at the bottom. In this section we will have a look on a few approaches that allows us to do so. To organize keywords into hierarchy we will use bottom-top agglomerative clustering algorithm. For algorithm to work we have to decide hot to define distance between words and phrases. For this we will use word embeddings. In this work we will use pre-trained word embeddings model trained on English Wikipedia by Jeffrey Pennington et al. (2014) [12]. By comparing two dense vectors we can say how 2 words are similar in embedding space which captures semantic relations between words. In our experiments we will stick to the cosine similarity (although other metrics are available Euclidean or Manhattan).

We also have to decide how we can assign names to the intermediate nodes in the hierarchy. We will compare two approaches:

- We can use a few closest words from embedding space to the centroid of the subtree. In this case we do not guarantee that selected term will be more general, but a few terms may describe subtree better than one general term.

- We can use closest ancestor from WordNet tree. This approach works good for known words that are already in the WordNet taxonomy, but if we will face new word that is not in the vocabulary there will not be any common word.

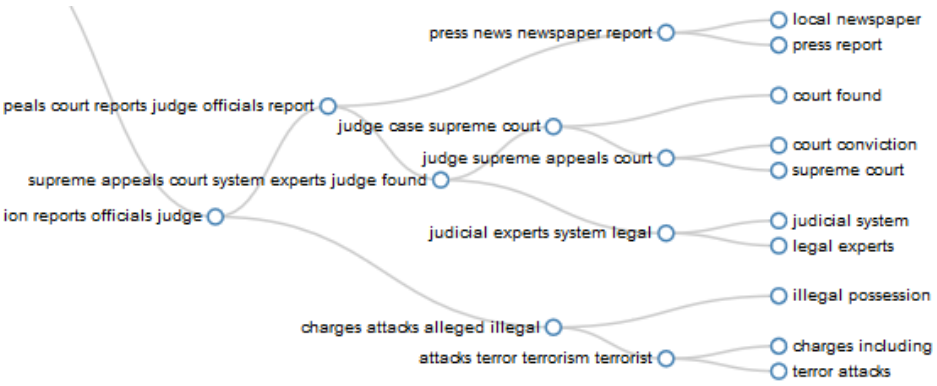


Figure 4.1: Keyword hierarchy obtained using Word2Vec relations.

We will run hierarchical clustering only on English news corpus because we do not have enough data to train our own word embeddings model for Czech keywords to compare its performance with Czech WordNet. Figures 4.1 and 4.2 show built hierarchies. As we can see word vectors do place similar terms closer in the hierarchy but they difficulties with naming intermediate nodes, because word embeddings do not capture general-specific terms relations. WordNet does not have this limitation. We have ignored terms that are not in WordNet taxonomy. If two terms do not have common predecessor, we use ROOT placeholder that means most general term. It was manually built to have this properties. WordNet has one downside which makes it useless in the context of on-line media. Since it is constructed manually it lacks many term that arise every day and new terms are added with delay. Word must be examined by linguist, manually annotated a put into right place, then the new version of WordNet will be released when there will be enough new words. From the practical point of view it is better to use word embeddings but the embedding model also has to updated regularly and because model update does not requires supervision this is preferable option for real world systems.

The second question we want to address in this chapter is keyword time modeling. By this we mean how does change keyword usage during some period of time. This information gives us overall impression about how popular some topic is. Whether it has positive or negative trend. Possibility to visually explore and compare usage of keywords is more useful for exploratory analysis and allows users to find correlations between topics and decide what is going to be the main topic of the next issue. One of the possible use cases is to plot a couple of related keywords next to each other for interactive exploratory analysis. Possible GUI must have components that allow user to find and plot one or more keyword time series. To do so we split each article into terms and create row in the special data structure that contains information about term, document id, position in the document and publication date. Now for each

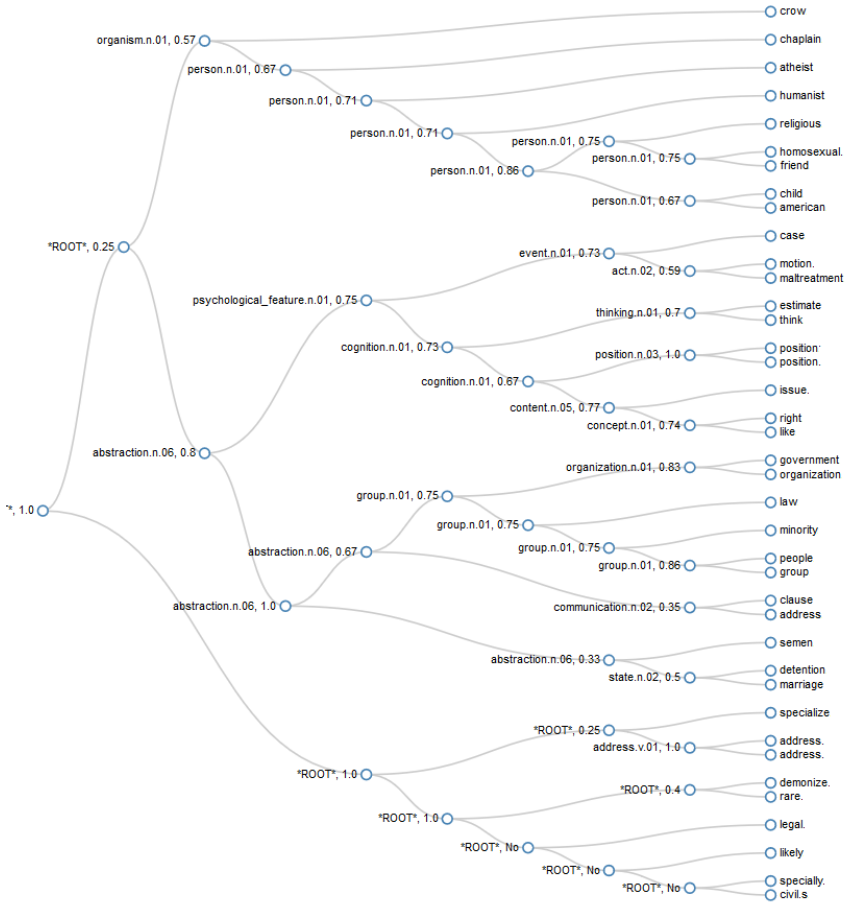


Figure 4.2: Keyword hierarchy obtained using WordNet relations.

term we have a time-series than we can visualize. Examples of visualizations are shown on figures 4.3 and 4.4. Those graphs were constructed from data from the first dataset and show how changed in time a few related terms. From this graph we can see changes in the terms itself and also relative change to other related terms.

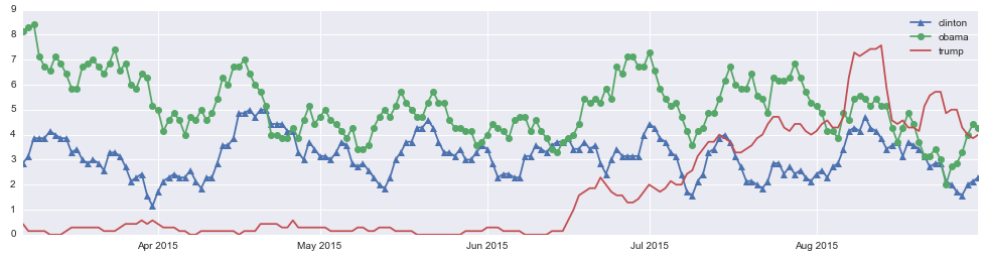


Figure 4.3: Time series for 6 month period for Trump, Obama and Clinton terms. Shows rolling mean of number of articles that mention keyword per week. Donald Trump was not going for president till Jun 2015, this can explain relatively low number of articles for the preceding period.

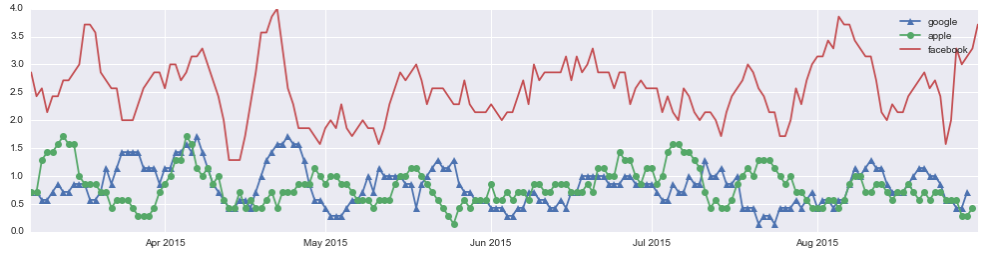


Figure 4.4: Time series for 2 month period for Facebook, Google and Apple terms. Shows rolling mean of number of articles that mention keyword per week. Higher volume of articles mentioning Facebook term can be explained by the fact that if article contains Facebook term it does not mean that the whole articles is about Facebook, often it does mean that somebody posted something on Facebook.

Chapter 5

Document clustering

When dealing with large amount of documents it is hard to organize documents into subsets or collections related to the same topic. Clustering is a method of unsupervised learning that can help it. In unsupervised learning no prior information is given to the algorithm and its task to find some structure in the data. One big advantage of this approach is that we do not need human labels, which are time consuming, expensive and biased. In the section 2.2 we compared keyword extraction algorithms to pick the best algorithm that works best with available datasets. In this section we will evaluate predictive power of keywords to document clustering. To do so we will run K-means clustering on datasets and will compare quality of clustering using different text representations. Here they are:

- Tf-Idf on unigrams
- Tf-Idf on keywords

Documents in the corpus do not have any label assigned. That is why we don not know right number of clusters. We will suppose that number of clusters is small 10 - 20. Which should correspond to the basic news categories: cinema, music, sports, politics e.t.c. Bigger number of clusters will correspond to subcategories and topics but it is much harder for the end used to deal with them. Instead it is more common to run clustering iteratively inside categories discovering smaller and smaller subsets of documents. User can start with some high level category like *sports*. Then he will drill down to explore subcategories like *soccer, basketball, tennis, archery*. And there user may want to go even deeper to the level of stories like EURO football championship or world hockey cup. On this level we expect for user to deal with relatively small amount of articles in cluster (around ten). This method is often called Scatter-Gather in the literature.

To gain some intuition about quality of the clustering lets create some visualization of the corpus. We will use Gephi graph software to visualize

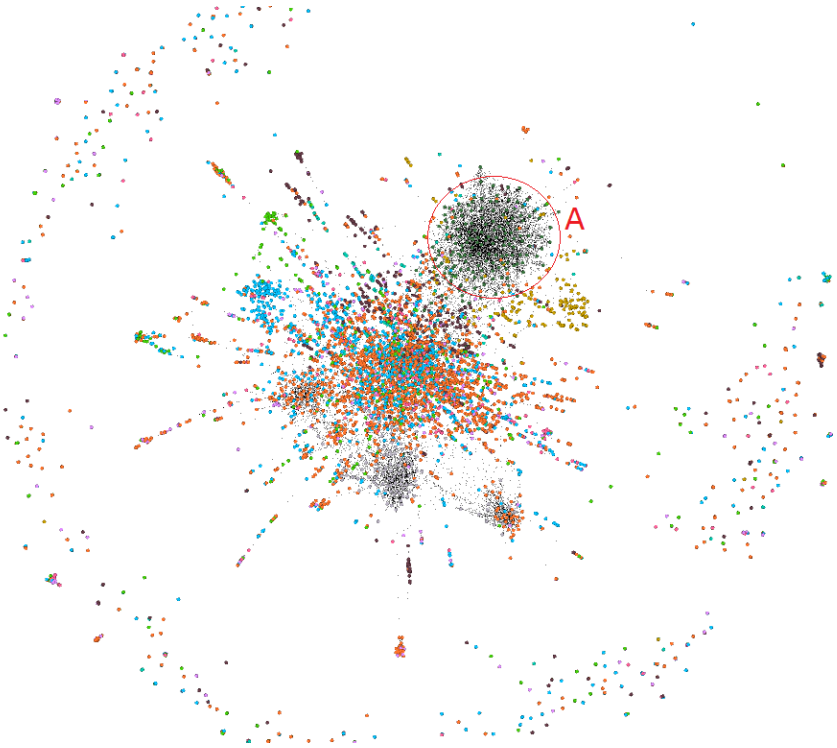


Figure 5.1: Graph constructed from documents of Czech news dataset using keyword features. Each node represents article, edge exists between two articles if their cosine similarity is higher than 0.3

document graph. We will use documents as nodes of the graph and we will add an edge between two nodes if their cosine similarity is higher than 0.4. Each node will be colored according to its cluster number. As we can see on the figure 5.3 graph is not homogeneous. Second observations is that documents that were assigned to the same cluster may not be connected, but if we will zoom in and take a closer look on them we will realize that they are indeed belong to the same cluster or category. For example blue articles are about sports and pink articles are about politics. Figures 5.4 and 5.5 demonstrate closer look on a few smaller clusters of documents.

Subjectively keywords representation provides worser separation of the clusters and generates more sparse graph. Keyword based graph created dense region of live broadcasts, in the middle of the graph there is large noisy region that consist of the noise.

Figure 5.2: Cluster A from keywords graph that consists of live-broadcast articles.

Figure 5.3: Graph constructed from documents of Czech news dataset using Tf-Idf features. Each node represents article, edge exists between two articles if their cosine similarity is higher than 0.4

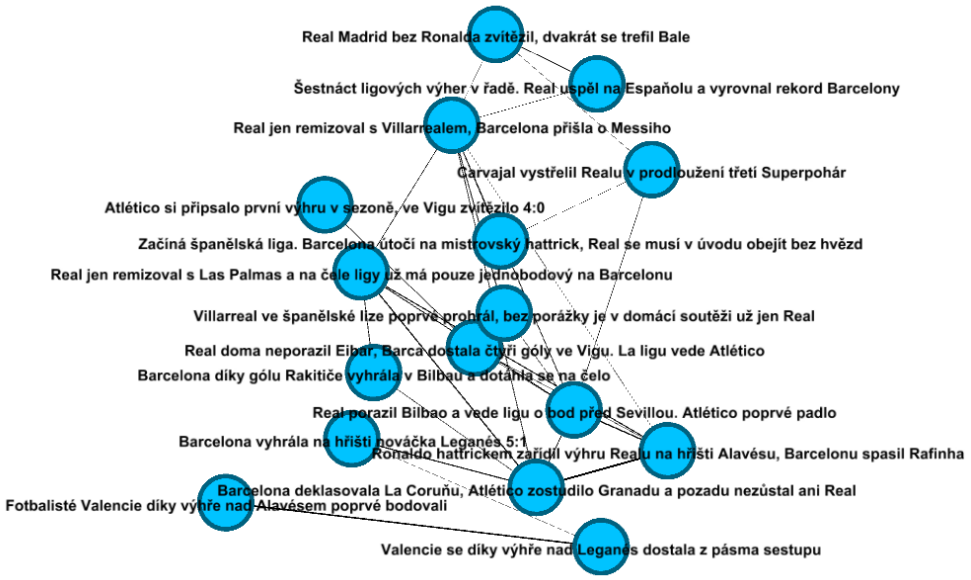


Figure 5.4: Cluster A of documents related to the same topic from sports category.

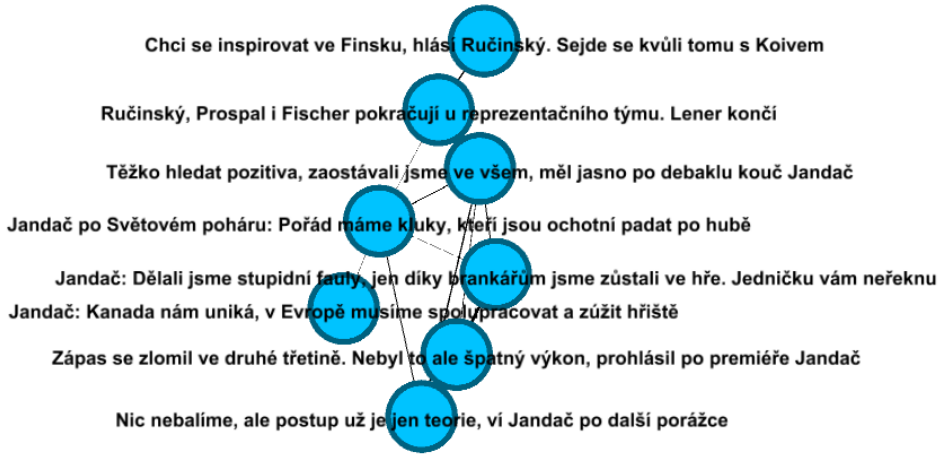


Figure 5.5: Cluster B of documents related to the same topic from sports category.

Chapter 6

Prediction

In the last section we will have a look on most promising and discussed in media area of data science - predictive modeling. Machine learning has proven itself as a good tool for automation of many routine problems, giving people ability and time to work on more intellectual and creative tasks. Machine learning has been successfully applied to computer vision: automatic sorting products in production, post code and car number recognition from photos and even lip-reading. In many of this areas machine learning achieved quality and precision comparable to the human or even went beyond human precision. It is also common to augment existing expert systems with machine learning, when model evaluates current situation and provides human operator with reasonable insights into current situation and a list of possible actions. This approach is more productive for both human and machine. Machine can process large amounts of different data much quicker than human can do, but since machine lacks common sense there a need to have a human as last frontier. Success stories and hype that is out there in media makes an impression that machine is capable to solve any task, we just need to give it enough data. Besides biases that are hidden in the data, we have to remember about biases that come from developers of those systems. In this chapter we will evaluate machine capabilities to solve difficult real-world problem from publishing domain. We will try to solve two tasks: forecast number of article's page views and classify article as long-living. Most of the news lose their relevance next day they have been published. Ability to distinguish short-living articles from long-living can benefit recommendation engine. Even humans do have difficulties with this kind of tasks. It is almost impossible to predict what is going to be popular on the internet. That is why it is even more interesting to check how machine will perform on this tasks using only text content.

The biggest problem when working with text is the question how to present text data to the machine so it can deal with texts of arbitrary length. The most popular method based on bag of words can represent any article in unified representation but has some drawbacks. First of all it completely

ignores word ordering which may be crucial in some applications, for example in sarcasm detection. Secondly bag of words ignores semantic meaning of the words which is very important for document clustering and recommendation engines. In this representation two articles will be similar if they share most of the words. But natural language is very complex and has many ways to say the same thing. There are such complex phenomenas as synonyms, antonyms, metaphors, neologisms and sarcasm. So in extreme case we can have two sentences that consist from the same set of words but have opposite meaning. There are some approaches that are trying to tackle this problem. One of them is bases on mathematical decomposition called SVD of tf-idf matrix into low-dimensional space by preserving most variability. This approach is sometimes called latent semantic analysis or latent semantic indexing. This approach assumes that words with the similar meaning occur in the similar texts. And again we assume that keywords can represent article's content better and thus can potentially lead to a better precision. So we will compare predictive power of keywords against baseline approach and provide some alternative view on problematics.

We will use same document representations as in clustering chapter.

- Tf-Idf on unigrams
- Tf-Idf on keywords
- LSA

To interpret results correctly we will perform cross-validation on our data set. Because data has natural ordering we will use sliding cross-validation. We will use one week data as test set and previous weeks as train data set. We will do this 3-5 times for different weeks. To measure quality of predictions for regression task we will use R squared metric and for classification task we will use F measure.

6.1 Popularity prediction

First task that we are going to address is popularity prediction problem. Here we need to predict how many page views given article will have using only text features. Page views is a continuous variable, that is the regression task. In this experiment we want to compare which text representation produces best results. We will compare two algorithms: linear regression as the baseline algorithm and random forest regression and an example of non-linear algorithm. Cross-validation scores are presented in the table 6.1.

As we can see, more complex algorithm can better fit training data but fails to generalize and has high test error as simple linear model. Which is a sign that text data does not contain enough information to explain news popularity. In order to gain some intuition on how model performs we will explore residual plot (differences between actual value and predicted versus predicted values). If our model generalizes well we should see that residuals are randomly distributed. Residual plot is shown in figure 6.1. As we can see model fits training data very well and residuals are very close to zero. But for the test data errors are much higher and they have some pattern above zero.

Low performance mainly caused by unpredictable nature of the task itself. It is very hard to tell whether some article will or will not be popular even for editors. Content is not the only factor that affects amount of page views. Page views number is also dependent on news context. Whether this is some local story or big investigation the number of page views will vary. And it is very hard for the machine to infer scope of the article from the article's text. Simply predicting mean value is (on average) better estimate than any machine learning technique used in this work. Therefore we conclude that we have not discovered evidence that keywords might improve quality of predictions for articles popularity. From out experiments follows the opposite, keywords actually hurt regressor performance in terms of R squared measure.

Table 6.1: R square measure for cross validation on first dataset.

Algorithm	Train	Test
Linear regression tokens	-0.18	-0.19
Linear regression LSA	0.02	0.01
Linear regression keywords	-0.41	-0.43
RF tokens	0.99	0.03
RF keywords	0.66	-0.16
RF LSA	0.99	0.06

Figure 6.1: Residual plot for Random forest regressor

6.2 Text classification

Article's life cycle is short and on average is 24 hours. After that articles loses its actuality and nobody has interest in reading old news. In the second task we need to determine whether there will be an interest to the articles after 24 hours from news publication. This problem is from area of text classification. Naive Bayes algorithm is considered as baseline algorithm in this area. As in first problem we will compare results with non-linear model of random forests. Results of cross-validation are presented in table 6.2.

Table 6.2: Cross-validation scores for F1 score, precision and recall for long-living detection.

Algorithm	Precision	Recall	F1 score
RF tf-idf	0.33	0.01	0.01
RF LSA	0.64	0.07	0.11
RF keywords	0.50	0.05	0.08
NB tf-idf	0.35	0.41	0.34
NB LSA	0.25	0.81	0.34
NB keywords	0.26	0.33	0.25

As we can see non-linear model has very poor performance on test data. Clearly non-linear models over-fits data, this can be explained by two facts:

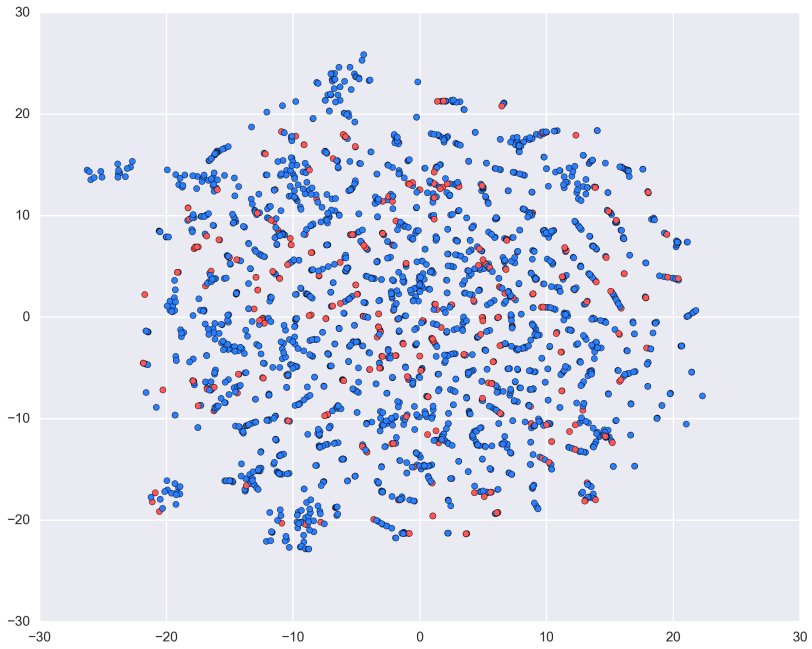


Figure 6.2: Two dimensional projection of Czech news articles corpus. Long-living articles are colored in red.

dataset is relatively small for such complex algorithm and very skewed. Naive Bayes has better results, but still results are poor and model with such performance cannot be put into production environment.

Fig 6.2 shows low-dimensional projection of the articles, long-living articles are depicted in red. As we can see on the figure there is no region that contains only long-living articles, they are mixed with ephemeral ones.

But lets have a closer look on the results that it produces. Tables 6.4 and 6.3 show titles for articles that were selected as long-living and ephemeral with highest confidence. As we can see algorithm is consistent about what he thinks is an ephemeral article, most of the articles are about sports and live broadcasts, which are not relevant to the reader after 24 hours they were published. Long-living articles are obvious and have worse separation, we can see that this category has broader scope: finance, society, politics, household and science.

Lets also have a look on which words have highest probability being in ephemeral article: *zapas, stat, volba, prezident, turnaj, soud, misto, proti, tym, uvest, sezona, bod.* And again we clearly see that sports topic dominates

in this class. In contrast here are the words from long-living articles: *cesky, procento, cesko, stat, koruna, prace, svet, auto, cena, dite, firma, spolecnost, praha*. Again, they come from broader topic scope, that is why it is harder for algorithm to tell that article is long-living.

Table 6.3: Top 10 of most probable ephemeral articles with the probability score of being ephemeral equals to 1.

Živě: Plzeň - Razgrad, zvládne Viktoria dotáhnout dvoubrankové manko z Bulharska?
Živě: Světový pohár startuje. V prvním zápase skupiny B Rusko vyzve tým Švédska.
Živě: Levadia Tallinn - Slavia 3:1, slávisté si do Prahy nevezou příznivý výsledek
Živě: Berdych - Bellucci 6:2, 7:5, Berdych je po roce ve finále, v Šen-čenu vyzve Gasqueta
Šafářová s Mattekovou-Sandsovou si poradily s Ruskami a postoupily do finále US Open
Živě: Sönderjyske - Sparta, Pražané hrají v Dánsku proti evropskému nováčkovi
Živě: Kvitová vs. Ostapenková, česká dvojka zahajuje US Open
Kvůli nevěře manželky zavraždil tchyni a tchána. Byl jsem v amoku, omlouval se u soudu
Berdych podával na vítězství, ale Veselý protáhl zápas do tmy, Šafářová padla s 96. hráčkou světa
Živě: Sparta vs. Fenerbahce 0:0, utkání dvou rozdílných poločasů skončilo remízou

Table 6.4: Top 10 of most probable long-living articles with the probability score of being long-living equals to 1.

5 mýtů o kvalitním krmivu pro psy, které změní vaše úvahy
Jak ušetřit starosti a vyhnout se havárii kotle?
Bydlíte v paneláku? Vybírejte vstupní dveře pečlivě
Začínala bez peněz a práce, teď má obří pěstírnu hub. Velkoobchodům nedodává, hlídá si radši kvalitu
Telefonní ústředna v Dejvicích se bude bourat. Je mi to líto, říká architekt, který ji navrhl
Vědci poprvé identifikovali zkamenělinu mozku dinosaura
Námořní kontejnery slouží v Kodani jako levné plovoucí koleje pro studenty
Tohle je nejmenší Velorex na světě. Startuje na dálku a má nevyčíslitelnou hodnotu
Američané vynalezli umělý roh nosorožce. Věří, že vzácná zvířata uchrání před pytláky
Unikátní proměna maringotky. Dřív sloužila Berouskům, architektky ji změnily v rekreační objekt

Chapter 7

Design and Implementation of the text mining system

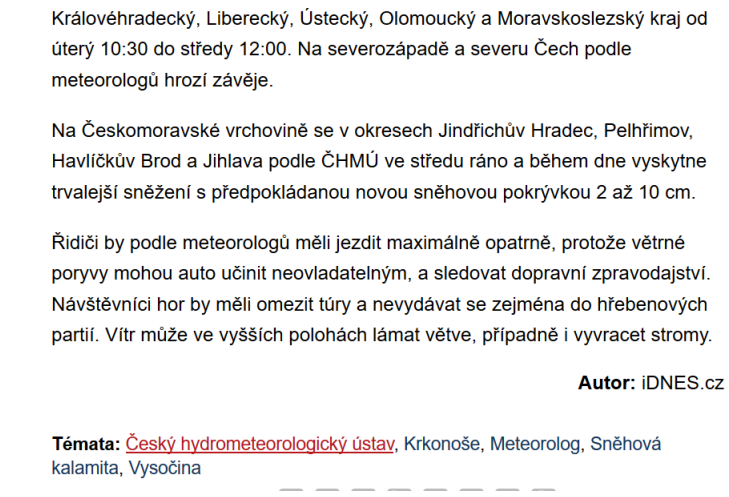
7.1 Design

We dedicate this chapter to design and implementation details of the text mining system. We will start with formal requirements for the system. We need to design and implement system that is able to extract keywords and cluster documents. This system will be a part of existing infrastructure at recombee lab and will support recommendation engine for Czech news news portal and will serve as additional source of meta data for articles for the web portal. Since we do not have access to the source codes of the main portal and we can not make improvements on the existing system we have to build a new system that is located nearby exposes REST API for communication with main portal. Keywords will be shown to the users on the frond-end side and will allow users to search for articles related to the keyword. Examples of such approach are shown on figures 7.1 and 7.2. Clustering information will be used in recommendation engine to recommend related articles that are from similar area but do not share same keywords. Example of recommendation block in shown on the figure 7.3.

The main use case: an editor writes an article and saves it into portal information system. Before saving an article into database backend will call text mining system REST API for keywords and clustering annotations. Then it will store keywords and cluster number in the database alongside with article's text and metadata. High level architecture of the system is shown on figure 7.4.

We can define following functional requirements:

- System will assign list of keywords to each article in the batch.
- System will assign cluster number to each article in the batch.



Královéhradecký, Liberecký, Ústecký, Olomoucký a Moravskoslezský kraj od úterý 10:30 do středy 12:00. Na severozápadě a severu Čech podle meteorologů hrozí závěje.

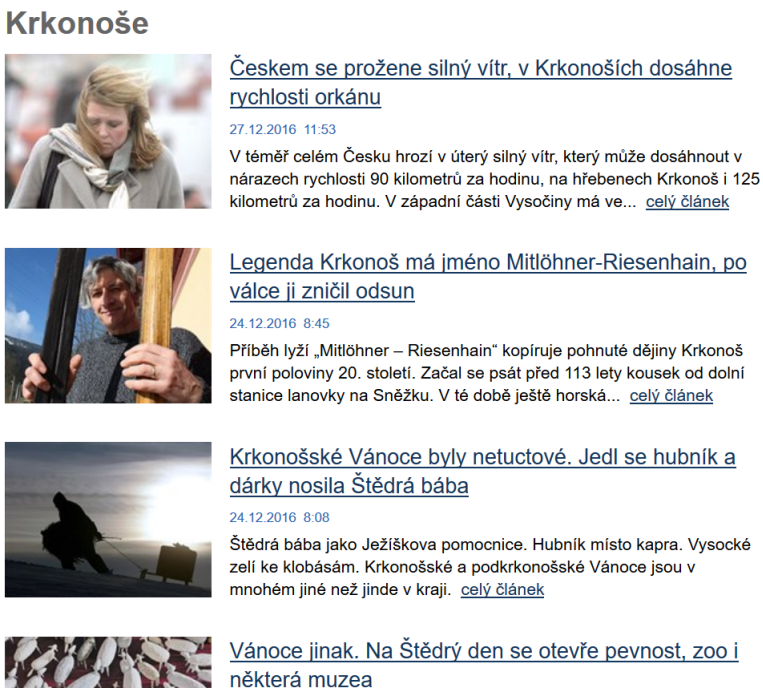
Na Českomoravské vrchovině se v okresech Jindřichův Hradec, Pelhřimov, Havlíčkův Brod a Jihlava podle ČHMÚ ve středu ráno a během dne vyskytne trvalejší sněžení s předpokládanou novou sněhovou pokrývkou 2 až 10 cm.

Řidiči by podle meteorologů měli jezdit maximálně opatrně, protože větrné poryvy mohou auto učinit neovladatelným, a sledovat dopravní zpravodajství. Návštěvníci hor by měli omezit túry a nevydávat se zejména do hřebenových partií. Vítr může ve vyšších polohách lámat větve, případně i vyvracet stromy.

Autor: iDNES.cz

Témata: Český hydrometeorologický ústav, Krkonoše, Meteorolog, Sněhová kalamita, Vysočina

Figure 7.1: News article annotated with search-able keywords. Snapshot from Idnes.cz news portal.



Krkonoše

Českem se prožene silný vítr, v Krkonoších dosáhne rychlosti orkánu

27.12.2016 11:53

V téměř celém Česku hrozí v úterý silný vítr, který může dosáhnout v nárazech rychlosti 90 kilometrů za hodinu, na hřebenech Krkonoš i 125 kilometrů za hodinu. V západní části Vysočiny má ve... celý článek

Legenda Krkonoš má jméno Mitlöhner-Riesenhain, po válce ji zničil odsun

24.12.2016 8:45

Příběh lyží „Mitlöhner – Riesenhain" kopíruje pohnuté dějiny Krkonoš první poloviny 20. století. Začal se psát před 113 lety kousek od dolní stanice lanovky na Sněžku. V té době ještě horská... celý článek

Krkonošské Vánoce byly netuctové. Jedl se hubník a dárky nosila Štědrá bába

24.12.2016 8:08

Štědrá bába jako Ježíškova pomocnice. Hubník místo kapra. Vysocké zelí ke klobásám. Krkonošské a podkrkonošské Vánoce jsou v mnohém jiné než jinde v kraji. celý článek

Vánoce jinak. Na Štědrý den se otevře pevnost, zoo i některá muzea

Figure 7.2: Keyword search. Snapshot from Idnes.cz new portal.

Figure 7.3: Recommendation section on the news portal. Snapshot from Idnes.cz new portal.

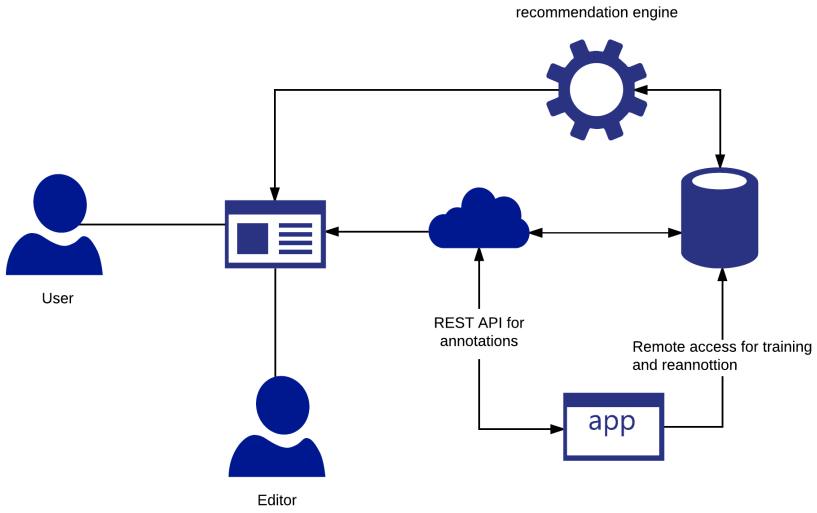


Figure 7.4: High level architecture of the text mining system and its integration into existing infrastructure.

System will provide web services to work with following entities:

- keywords
- clustering
- annotate
- stopwords

For each service system will expect input data to be an array of article objects in JSON format. Each object must contain article's id, title of the article and text.

Keywords endpoint will take input and will return an array of objects containing article's id and list of keywords. Service endpoint is :

```
POST /keywords/
```

Clustering endpoint will take input and will return an array of objects containing article's id and number of the cluster. Service endpoint is :

```
POST /cluster/
```

Annotate endpoint will take input and will return an array of objects containing article's id, list of keywords and number of the cluster. Service endpoint is :

```
POST /annotate/
```

The purpose of the stop-words endpoint is to give to editors ability to ban some words from being selected as candidates for keywords. On the front-end editor may hide some keywords that he thinks are wrong or blacklist it. He will click "ban" button and request will be sent to text mining system. System will add word to its internal stop-words list.

```
GET /stopwords/
POST /stopwords/
DELETE /stopwords/
```

The API will allow editors to customize stop-words list. Each method will take an array of one or more keywords to be processed.

GET method will return the list of currently banned keywords. POST will add words to the stop-words list. DELETE will remove words from the stop-words list.

7.2 Implementation

To implement such system we will need two programs. First program will be a simple web service that waits for request, does the document processing

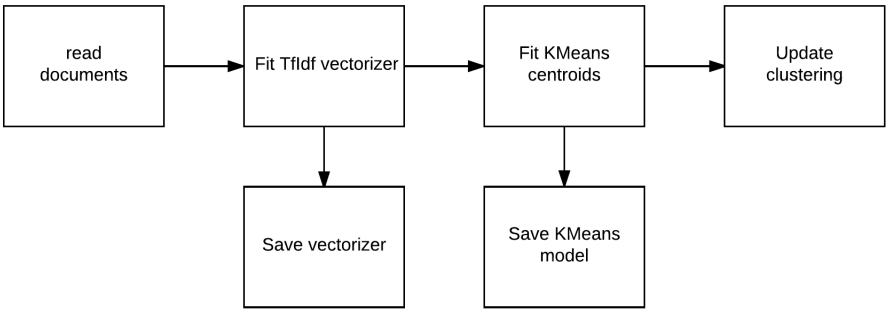


Figure 7.5: Data flow diagram for model update process.

and keyword extraction and returns then in the response. Second program will perform clustering model update. Since our corpus is not static and we have new documents every day, after a while we need to adjust our models to express and fit data better. Figure 7.5 describes this process.

Both programs are implemented using python language. They are using pandas [13], scikit-learn [14] and rake [4] libraries to extract keywords, build models and save data. We use nltk [15] python library to split text into tokens and stem them. We use Flask framework to implement REST API.

7.2.1 Document processing

When web server starts it will load serialized models into global variables shared between requests on the server. This is mainly to reduce disk I/O. When web server receives request for each article it will run keyword extraction algorithm and will store result in the list for each article. It will also run KMeans cluster prediction for each article. Then thread will serialize response into JSON and will send in the response.

7.2.2 Model update

Second program will have the following stages

- init
- processing
- finalization

At init stage program will read historical article's for the specified number of days from DB. Then it will run vocabulary builder procedure for TfIdf vectorization. Once TfIdf model is built, it fit KMeans centroids and it will

reassign cluster numbers for historical articles. In the last stage program will store TfIdf and KMeans in the binary serialized form.

Models will be stored on disk in binary serialized format. All parameters for model training and text processing will be stored separately from source code in configuration directory.

7.3 Testing

Testing is the important part of every software project. Testing helps to answer the question "Does system work as expected at current stage of the development process?". Automatic and manual tests help to find errors in the new functionality and find bugs in already implemented and tested functionality. Depending on goal and methods : unit testing, performance testing, manual testing, integration testing, usability testing and others.

Unit testing is a piece of code created by developer to ensure that individual parts of the software system, typically classes and modules, meet the design requirements and behaves as intended.

Performance testing is generally executed to determine how a system or sub-system performs in terms of responsiveness and stability under a particular workload. It can also serve to investigate, measure, validate or verify other quality attributes of the system, such as scalability, reliability and resource usage. In terms of large number of users or large number of requests.

Manual testing is the process of manually testing software for defects. It requires a tester to play the role of an end user and use most of all features of the application to ensure correct behavior. To ensure completeness of testing, the tester often follows a written test plan that leads them through a set of important test cases.

Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing. [16]

Usability testing is a technique used in user-centered interaction design to evaluate a product by testing it on users. This can be seen as an irreplaceable usability practice, since it gives direct input on how real users use the system. [17]

The most way to test web applications is unit testing. To test our system we will use *pytest* library for python language. *Pytest* is a testing framework that makes it easy to write simple test. An example of such test is presented below.

```
def test_stopwords():
  import requests

  response = requests.get('http://127.0.0.1:5000/stopwords')
  assert response.status_code == 200
```

To ensure that implemented system behaves as intended we will write three test cases for each of annotation services and three tests for stopwords service. There cases will test correctness of services on valid and invalid or incomplete data. Each test case will check HTTP code of the response and payload of the response.

Results of the test run are presented below.

```
$ pytest
============================= test session starts =============================
platform win32 -- Python 2.7.12, pytest-3.0.5, py-1.4.32, pluggy-0.4.0
rootdir: C:\Users\sstamenov\dp\text_mining_dp\sources\web_app, inifile:
collected 30 items

api_test.py ..

========================== 30 passed in 0.04 seconds ==========================
```

Conclusion

In the third chapter we evaluated three keyword extraction algorithms on the collection of news articles in English and Czech languages. RAKE algorithm has the best F1 measure performance. It does not require additional preprocessing steps and information about whole corpus. Therefore keyword extraction using this algorithm can be runned in parallel.

We have conducted a series of experiments to determine whether keyword based representation of documents produces better document clustering. We have examined document graphs and discovered that keyword representation produces more sparse graphs with many clusters and fails to connect related articles. It also mixes up documents from different categories into one cluster.

In the area of predictive modeling all representations failed to produce good results. It seems that the nature of those tasks is unpredictable and text itself does not capture enough information for the machine to make a correct decision on both how article is going to be popular or whether readers will have interest in the article in the future. Popularity prediction does not predict page views count better than simple mean prediction. Classification task showed some promising results but its performance is still worse than random guessing.

We explored ways to construct keyword hierarchy using agglomerative clustering. WordNet ontology has built-in parent-predecessor relationship but has a limited vocabulary. On the contrary, word embeddings can organize any set of words and phrases into hierarchy, but the parent-predecessor relationship is much weaker that in WordNet because word vectors were not trained to capture this kind of information. In fact, to label an intermediate node we show a few terms instead of one more general term which will allow users to guess what sub-cluster is about. Time series visualization of keywords popularity allows editors to detect anomalies and evaluate hypothesis about correlation of the events.

In the last chapter we designed, implemented and tested text mining system that extracts keywords from articles and does the clustering. System is

designed as REST API service that can be integrated into existing infrastructure at online website. Further work may involve scaling the system up to bigger volumes of data by using distributed computational framework (like Apache Spark).

The main difficulty that we faced during this work is impossibility of the quantitative analysis of the results. The nature of the clustering task is very subjective. This approach is limited by time we are able to spend exploring the results and volume of data we are capable to process. We need to develop unsupervised metrics to evaluate and compare clustering results on a larger scale.

Tf-Idf and bag of words are powerful tools for document processing, but we need to develop new text representations that can capture document similarity better to improve quality of clustering and classification. Automatic text processing is a complex area of research. Quality of results depends on preprocessing steps one takes, chosen representation of the text and availability of human annotated data to improve performance of the machine learning models.

Bibliography

[1] Christopher D. Manning, H. S., Prabhakar Raghavan. An Introduction to Information Retrieval. In *An Introduction to Information Retrieval*, 2009.

[2] Dumais, S. T. Latent semantic analysis. *Annual Review of Information Science and Technology*, volume 38, no. 1, 2004: pp. 188–230, ISSN 1550-8382, doi:10.1002/aris.1440380105. Available from: `http://dx.doi.org/10.1002/aris.1440380105`

[3] Blei, D. M.; Ng, A. Y.; et al. Latent Dirichlet Allocation. *J. Mach. Learn. Res.*, volume 3, Mar. 2003: pp. 993–1022, ISSN 1532-4435. Available from: `http://dl.acm.org/citation.cfm?id=944919.944937`

[4] Stuart Rose, N. C. W. C., Dave Engel. Automatic keyword extraction from individual document. 2010.

[5] Mihalcea, R.; Tarau, P. TextRank: Bringing order into texts. In *In Proceedings of EMNLP*, Poster, 2004, pp. 404–411.

[6] Page, L.; Brin, S.; et al. The PageRank citation ranking: Bringing order to the web. Technical report, 1998.

[7] Řehůřek, R.; Sojka, P. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, Valletta, Malta: ELRA, May 2010, pp. 45–50, `http://is.muni.cz/publication/884893/en`.

[8] Miller, G. A. WordNet: A Lexical Database for English. *COMMUNICATIONS OF THE ACM*, volume 38, 1995: pp. 39–41.

[9] Tomas Mikolov, G. C., Kai Chen; Dean, J. Efficient Estimation of Word Representations in Vector Space. In *In Proceedings of Workshop at ICLR*, 2013.

[10] Raschka, S. In *Python Machine Learning*, 2015.

[11] Marujo, L.; Gershman, A.; et al. Supervised Topical Key Phrase Extraction of News Stories using Crowdsourcing, Light Filtering and Coreference Normalization. In *Proceedings of the LREC 2012*, 2012.

[12] Pennington, J.; Socher, R.; et al. GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543. Available from: `http://www.aclweb.org/anthology/D14-1162`

[13] McKinney, W. Data Structures for Statistical Computing in Python. In *Proceedings of the 9th Python in Science Conference*, edited by S. van der Walt; J. Millman, 2010, pp. 51 – 56.

[14] Buitinck, L.; Louppe, G.; et al. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, pp. 108–122.

[15] Bird, S.; Loper, E. NLTK: The Natural Language Toolkit. In *Proceedings of the ACL demonstration session*, 2004, pp. 214–217.

[16] (ed), M. A. O. . C. U. Testing in Software Development. *BCS*, 1986.

[17] Nielsen, J. Usability Engineering. *Academic Press Inc*, 1994.

Appendix A

Acronyms

NLP Natural language processing

CSV Comma separated value

nltk The Natural Language Toolkit

LDA Latent Dirichlet Allocation

LSA Latent Semantic Analysis

SVD Singular Value Decomposition

TF-IDF Term frequency-Inverse Document Frequency

DB Database

NER Named entity recognition

NE Named entity

Appendix B

Contents of enclosed CD

- `readme.txt` ..... short content description
- `src`
  - `webapp` ..... text mining system sources
  - `thesis` ..... thesis source code in LaTeX
- `text` ..... thesis text
  - `thesis.pdf` ..... thesis text in PDF format