ZÁPADOČESKÁ
UNIVERZITA
V PLZNI

University of West Bohemia in Pilsen
Department of Computer Science and Engineering
Univerzitni 8
30614 Pilsen
Czech Republic

Stance detection and summarization in social networks

PhD Study Report

Ing. Peter Krejzl

Technical Report No. DCSE/TR-2018-01
February, 2018

Technical Report No. DCSE/TR-2018-01
February 2018

Stance detection and summarization in social networks

Ing. Peter Krejzl

Abstract During recent years, there have been a lot of research in the area of Natural Language Processing (NLP) related to the sentiment analysis. Stance detection goes even further and tries to detect whether the author of the text is in favor or against a given target. The main difference to sentiment analysis is that in stance detection, systems are to determine the author's favorability towards a given target and the target may not even be explicitly mentioned in the text. Moreover, the text may express positive opinion about an entity contained in the text, but one can also infer that the author is against the defined target (an entity or a topic).

This thesis is focused on the two main tasks: identifying the stance and its summarization and outlines the state-of-the-art approaches to stance detection and summarization.

Copies of this report are available on
http://www.kiv.zcu.cz/publications/
or by surface mail on request sent to the following address:

University of West Bohemia in Pilsen
Department of Computer Science and Engineering
Univerzitni 8
30614 Pilsen
Czech Republic

Contents

Chapter 1

Introduction

Understanding the meaning of the text is crucial task of natural language processing (NLP). Internet is full of various opinions, every single blog post or tweet expresses an author's opinion regarding a given topic. Such an opinion can be valuable information for companies regarding their product, politicians and many others.

The issue in processing and understanding the opinionated text is its extreme size as it is nearly impossible to manually process such a huge corpus of data. Using NLP techniques from data mining and information retrieval researchers are trying to build systems that can automatically process and understand such opinionated data.

In this thesis we are going to investigate semantic approaches to stance detection and summarization of the text, with focus on tweets and news commentaries.

Thesis is organized as follows, background and related work is in Chapter 2, in Chapters 3 and 4 we discuss current state-of-the-art in Stance Detection and Summarization respectively.

Chapter 5 shows preliminary results of our work and ideas and future aims of doctoral thesis.

This thesis is focused on the two main tasks: identifying the stance and its summarization. In the two following sections we describe both these areas in more detail.

1.1 Motivation

In the recent years many researchers approached the problem of stance detection but mostly for English. The stance detection for Czech language haven't been targeted yet. Moreover, current results show existing solutions are still not mature and there is a huge space for an improvement.

Although, the text summarization is a task existing for a long time and covered from many various aspects, to the best of our knowledge there is no a system summarizing a stance data in the way users are easily able to see the most important stances for a given target.

The aim of the doctoral thesis is to study stance detection for multiple languages, with focus on English and Czech and extend it with the multilingual stance summarization.

1.2 Stance Classification

Stance detection has been defined as automatically detecting whether the author of a piece of text is in favor of the given target or against it. In the third class, there are cases, in which neither inference is likely [Krejzl and Steinberger, 2016]. Classifying stance involves identifying subjective disposition of an author towards a given topic [Anand et al., 2011]. It can be viewed as a subtask of opinion mining and it stands next to the sentiment analysis.

The significant difference is that in the case of sentiment analysis, systems determine whether a piece of a text is positive, negative or neutral. However, in stance detection, systems are to determine the author's *favorability* towards a given target and target may not even be explicitly mentioned in the text. Moreover, the text may express positive opinion about an entity contained in the text, but one can also infer that the author is against the defined target (an entity or a topic). This makes the task more difficult, compared to the sentiment analysis, but it can often bring complementary information [Krejzl and Steinberger, 2016].

Formally, in [Sobhani, 2017], the problem is defined as given a set of texts (tweets, news, blog posts) D related to target entity T and the goal is to determine mapping:

$$s_t : D \to \{\text{in favor, against, neither}\} \tag{1.1}$$

for any element $d \in D$.

The example of a twitter tweet expressing a stance might look like following:

Target: *Donald Trump*

Text: *@realDonaldTrump you're the man for the job*

This particular tweet should be automatically identified as being *in favor* of the given target *Donald Trump.*

The target might be a person, or a product, or an organisation. Moreover, from the text one can infer the author's stance towards multiple various targets.

If we consider the following example:

Target: *Hillary Clinton*

Text: *The only real candidate is Jebb.*

One can immediately see the author is *against* Hillary Clinton but *in favor* of Jebb Bush.

Target is not explicitly mentioned in the tweets, so the system needs to infer that author's preferred candidate is *Jeb Bush* and so the author is very likely *against* the given target *Hillary Clinton.*

The third example:

Target: *Climate Change is a Real Concern*

Text: *It doesn't simply rain in Houston anymore, it storms.*

requires additional context in order to successfully assign a stance. One needs to have a knowledge the author thinks the change from rains to storms is due to the global climate change.

[Sobhani, 2017] also defines **multi-target stance classification task**, which maps a post to a multiple stances based on the multiple targets.

Targets: *Apple iPhone, Samsung Galaxy S8*

Text: *iPhone is extremely expensive, S8 beats it in many aspects.*

System then has to determine that the text is *in favor* of Samsung Galaxy S8 but *against* of Apple iPhone.

There are many applications which could benefit from the automatic stance detection, including information retrieval, textual entailment, or text summarization, in particular opinion summarization.

1.3 Stance Summarization

Stance summarization is a new task that tries to aggregate and summarize the most important information about a given target and a stance.

topic = Hillary Clinton	
FAVOR	**AGAINST**
• Hillary can help this country • Hillary supports LGBT rights • She supports equality (jews, women, marriage) • Best choice to continue being a progressive nation	• Hillary supports war • Hilary has lied, deleted Benghazi emails, and betrayed the trust of Americans • Did not create any jobs

Table 1.1: Summary of stances related to target Hillary Clinton

By summarizing the information by the target as well as by the stance, one can immediately see most relevant opinions for each of the stances (*favor, against*). The neutral stance (*neither / neutral*) is usually ignored here. For example, for a target *Hillary Clinton*, the task is to summary the most relevant opinions split into *against* and *favor* categories. Such a summary can be found in the table 1.1.

The stance containing posts are usually very informal, opinionated and in the case of Twitter also very short, so the automatic system has to deal with all of these issues which are not so common for existing summarization systems. It's also worth of noting, the input texts fed into a summarization system are already classified into in *favor* and *against* classes.

Chapter 2

Background and Related Work

In this chapter, we first describe some necessary background in order to fully understand the problem of stance detection and summarization. We also provide a description of the methods commonly used in the sentiment analysis and stance detection as deep neural networks and word embeddings as well as the approaches for the topic modelling.

2.1 Topic Modelling

A topic model is a statistical approach that discovers the abstract topics existing in a collection of documents. Topic modelling is often used to discover hidden (latent) semantic structures in the text.

Each document can be associated with multiple topics, not just one. Topic modelling provides a relatively simple way to analyse large volumes of unlabelled textual data. A topic consists of a cluster of words that frequently occur together.

Generally, topic modelling can help in [Nair, 2016]:

- discovering hidden topical patterns that are present across the collection,
- annotating documents according to these topics,
- using these annotations to organize, search and summarize texts.

Here, we briefly describe the most important and used algorithms for topic modelling, which are Latent Semantic analysis (LSA) (subsection 2.1.3) and

Latent Dirichlet Allocation (LDA) (subsection 2.1.4), we also briefly describe Vector Space Model in subsection 2.1.2.

2.1.1 Term Frequency - Inverse Document Frequency (TF-IDF)

TF-IDF is a statistical method that reflects the importance of a word in a document or a set of documents [Ullman, 2011].

TF-IDF consists of two parts - **Term Frequency (TF)**, which measures how frequently a word occurs in a document and **Inverse Document Frequency (IDF)**, which measures how important a word is.

Final score is then computed as:

$$TF_w = \frac{\text{number of times word } w \text{ appears in document}}{\text{total number of words in the document}}$$

$$IDF_w = \log_e \frac{\text{total number of documents}}{\text{number of documents with word } w \text{ in it}}$$

and the final score is computed as $TF_w \times IDF_w$.

Second part of the equation (IDF) will effectively zero the probability for words occurring in most of the documents. And then the final TF-IDF score for words is a good measure of importance.

2.1.2 Vector Space Model

In **Vector Space Model (VSM)** documents are represented as vectors, for example $d = (w_1, w_2, \ldots, w_n)$. Each dimension corresponds to a particular term - usually a word but it can be a longer phrase as well. If a term exists in the document, its value is higher than zero. One of the mostly used approaches for computing these values is term frequency - inverse document frequency (TF-IDF) [Manning et al., 2008], in detail discussed in section 2.1.1.

Similarity of two documents than can be calculated as a *cosine* of the angle between the vectors. For example for the vectors

$$d_i = (w_{i,1}, w_{i,2}, w_{i,3}, \ldots, , w_{i,n})$$

$$d_j = (w_{j,1}, w_{j,2}, w_{j,3}, \ldots, , w_{j,n})$$

the similarity would be

$$\cos\theta = \frac{d_i \cdot d_j}{||d_i||\,||d_j||}$$

$d_i \cdot d_j$ is a dot product of two vectors and $||d_i||$ and $||d_j||$ are the norms of vectors d_i and d_j that can be calculated (for d_i) as

$$||d_i|| = \sqrt{\sum_{i=1}^{n} d_i^2}$$

VSM is relatively simple but still very powerful model. To its disadvantages belongs worse representation of long documents - dot product of many (small or even zero) values and larger dimensionality. And also its semantic sensitivity when documents represented by different but semantically close words are not deemed similar.

These issues are addressed in the following models that are either based on or extending VSM.

2.1.3 Latent Semantic Analysis

Latent Semantic Analysis (LSA) ([Deerwester et al., 1990], [Landauer and Dumais, 1997], [Landauer et al., 1998], [Gong and Liu, 2001], [Steinberger and Jezek, 2004]) analyzes relations between documents and their terms and build a low-rank semantic space out of a collection of documents. The key idea is that words with similar meaning occur in similar documents.

First, given documents $d_1, d_2, \ldots, d_n$ and terms $w_1, w_2, \ldots, w_m$, we need to create a **term-document matrix** $X \in \mathbb{R}^{m \times n}$, where x_{ij} describes the term occurrence of term w_i in document d_j. There are multiple ways how to calculate x_{ij}, it can be binary value (exist / not exist), count or TF-IDF score.

In term-document matrix, rows represent terms, while columns represent documents. Term-document matrix dimension is quite high, it contains as many rows as terms in vocabulary and as many columns as documents.

The *Singular Value Decomposition (SVD)* is applied in order to reduce the dimensionality and to expose latent relationship between words in the document.

SVD finds the approximation of term-document matrix as

$$X \approx U_k \Sigma_k V_k^T$$

where U and V are orthogonal matrices, Σ is a diagonal matrix and k is rank.

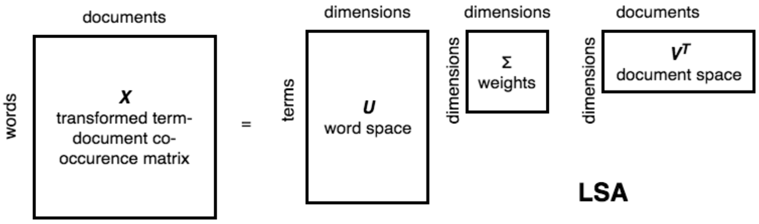

Figure 2.1: Latent Semantic Analysis

With this low-rank representation, we found an approximation of matrix X that emphasises the most important relationships and ignores the noise. The important variable in the process is defining how many dimensions (concepts) is needed. Too many dimensions increase the noise, while too few dimensions ignores some important patterns. The typical size of k is usually between 100 and 500.

With such approximation, we can easily compare two documents or terms, typically by computing their cosine similarity.

2.1.4 Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) is a generative probabilistic model for collections of discrete data such as text corpora. It is based on the Distributionl Hypothesis and the Bag-of-words Hypothesis, i.e. that the word order does not matter and there is some latent relation between the words within the same document (within the same content). It allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar.

Each document is a mixture of various topics where each document has a set of topics assigned to it via LDA.

The name Dirichlet came from generating the topic distribution for a document from a Dirichlet prior. In this model, each word in a document is sampled from the multinomial distribution of a topic where this topic is also generated from the multinomial topic distribution for the document [Blei et al., 2003] [Nair, 2016] [Sobhani, 2017].

More formally, we have a vocabulary V consisting of v terms, a set T consisting of K topics and M documents. For every topic $k \in T$ a distribution φ_k on V is sampled from $\mathrm{Dir}(\beta)$, where $\mathrm{Dir}(\beta)$ is a v-dimensional Dirichlet distribution, $\beta \in R^V$ is a smoothing parameter. Then, for every document d a distribution Θ_d on T is sampled from $\mathrm{Dir}(\alpha)$, where $\mathrm{Dir}(\alpha)$ is a K-dimensional Dirichlet distribution, $\alpha \in R^K$ is a smoothing parameter.

The words of the documents are drawn in the following steps [Materna, 2012, Bíró, 2009]:

- for every word position i of document j a topic $z_{ij} = k$ is drawn from Θ_i,
- a term w_{ij} is drawn from φ_k.

The graphical visualisation of LDA is presented in figure 2.2.

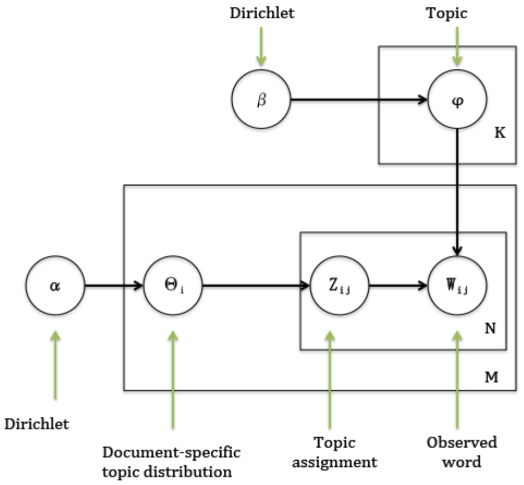

Figure 2.2: Graphical visualisation of LDA as a Bayesian Network

To infer the topic assignments, the original paper used a variational Bayes approximation, other methods such as Gibbs sampling or expectation propagation can be used too [Griffiths and Steyvers, 2004]. From the mentioned methods, Gibbs sampling is often used because its performance is comparable to others two but is more tolerant to local optima. The whole method belongs to a group of sampling methods known as Markov Chain Monte Carlo.

2.2 Deep Neural Networks for Natural Language Processing

In recent years, neural networks are widely used for almost all supervised machine learning tasks from computer vision to natural language processing (NLP). Neural networks, inspired by the human brain, are the computing approach allowing to learn from the past data without the need of various hand-crafted features, so usual in the ”classical” machine learning systems.

It is not true that non-neural machine learning algorithms couldn't achieve same goals in NLP but when some conditions like a lot of training data are fulfilled, neural networks usually performs better [Goodfellow et al., 2016a, LeCun et al., 1998, Raschka and Mirjalili, 2017].

Neural network is highly parametrised model sometimes also called an universal approximator, meaning a computer can learn (with enough data) any relationship in data. The basic neural network is feed-forward neural network, containing neurons, layers and connections. Neurons are grouped into layers and connected between layers. Each connection has a weight that changes over time. During the network training these weights are tuned in order to make a neural network adaptive to inputs and capable of learning. Important concept is an activation function - a function that converts the weight of particular neuron to some output activation. Most widely used activation functions are sigmoid and rectified linear unit (ReLU). The simple network architecture is displayed at Figure 2.3.

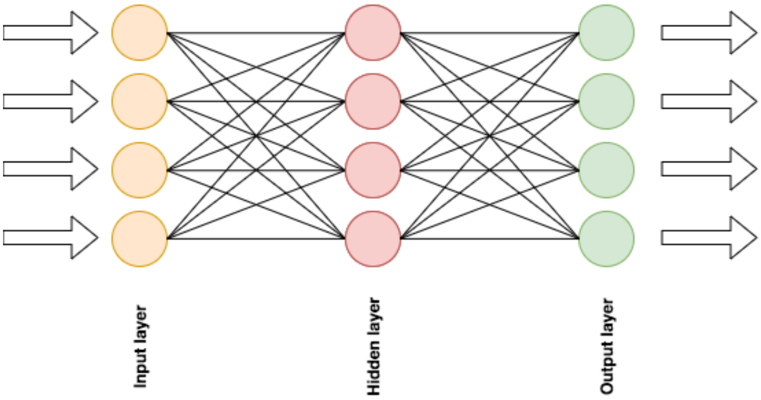

Figure 2.3: Feed forward neural network with one hidden layer.

2.2.1 Recurrent Neural Networks

The key concept of **Recurrent Neural Networks (RNNs)** is the idea of using sequence information. While traditional neural network assumes all input are independent, which might not be a true, RNN executes same task on every element of the sequence, so the output is dependent on the previous outputs. Another way of thinking about RNN is that it has a memory which keeps the information seen before. Theoretically, RNNs can handle long sequences but in practice they are quite limited only to a couple of steps [Hochreiter et al., 2001].

Recurrent Neural Networks (RNN) does not take as the input only the input data but also consider their previous states. This the key difference to feed-forward neural networks, RNN have a feedback loop connected to their previous states. It helps them to reuse their previous outputs as a new inputs.

RNNs have been extremely successful in many tasks [Boulanger-Lewandowski et al., 2012, Eck and Schmidhuber, 2002, Sutskever et al., 2009, 2011]. For NLP related tasks we will focus on their versions called Convolutional Neural Networks, Long Short Term Memory Networks and GRUs, which we discuss in the following sections.

Convolutional Neural Networks

Convolutional Neural Networks (CNN) ([LeCun et al., 1989], [Goodfellow et al., 2016b]) are widely used especially in the field of computer vision. However, in recent years researches started applying CNNs to NLP problems as well and with the high rate of success.

There are four main steps in CNN - convolution, non-linearity (ReLU), pooling or sub-sampling and classification.

Convolution extracts relationship between data elements using small squares of input data.

A small matrix is called a kernel or a filter. The process does element wise multiplication of filter and particular area of the input matrix. Resulting matrix is called Feature Map. Different filters will provide different feature maps for the same input data. In the image processing, one filter can detect edges, while other can for example blur the image. Also, values of these filters are learnt during the training phase of the network. However, there are still some parameters like:

- Depth - the number of filters,
- Stride - number of positions (e.g. pixels) by which a filter is slide over the input matrix,
- Zero padding - sometimes we need to pad input matrix with zeros around the border.

Non-linearity. Rectified Linear Unit (ReLU) is widely used as an activation function. The function is applied after the convolution step and is defined as:

$$\text{ReLU} = \max(0, \text{input}) \tag{2.1}$$

Feature map is then called recritified feature map. Other activation functions are *tanh* or *sigmoid*.

Pooling step. Subsampling (spatial pooling) reduces the dimensionality of each feature map while preserving the most important information. It can be of different types - max, average, sum.

Feature map, using spatial neighbourhood 2x2 with stride = 2

$$\begin{bmatrix} 1 & 1 & 2 & 4 \\ 5 & 6 & 7 & 8 \\ 3 & 2 & 1 & 0 \\ 1 & 2 & 3 & 4 \end{bmatrix} \Longrightarrow \begin{bmatrix} 6 & 8 \\ 3 & 4 \end{bmatrix}$$

In the example above, the pooling step reduces the dimensionality of the feature map from 4x4 to 2x2. Pooling is applied separately to each feature map (product of each filter). In practice pooling:

- reduces the dimensionality and reduces the number od parameters,
- makes the network more robust and prone to small changes in the input data.

Convolution, activation functions and pooling are basic elements of the network and can be further combined into e.g. multiple hidden convolutional laters. At the end, one or more convolutional layers are usually followed by at least one fully connected layer (every neuron in the previous layer is connected to each neurone in the following layer). Fully connected layer is used to classify the input data into pre-defined set of classes based on the training data.

LSTM and GRU

One of the key problems of standard RNNs is the vanishing gradient. RNNs generally have problems learning long-term dependencies, in case of NLP it can be a relations between words that are many words apart. It is because the gradient contributions of these words are (close to) zero and thus the state of those steps is not considered in the learning process. As this gap grows, RNNs are no more able to learn this relation. This problem is being addresses in **Long Short Term Memory (LSTM) Network.** LSTMs [Hochreiter and Schmidhuber, 1997] are special kind of RNN, designed to learn long-term dependencies. The difference is in the computation of the hidden state. While vanilla RNNs compute the hidden state as $s_t = \tanh(Ux_t + Ws_{t-1})$, where x_t are inputs, s_t is the current input step, s_{t-1} is the previous state. LSTM computes this slightly differently using *input, forget* and *output* gates. Gated cells preserving information outside the normal flow. The cells can also make a decision what to store.

Gated Recurrent Unit (GRU) is a type of RNN very similar to already discussed LSTM. A GRU has two gates, a reset gate and an update gate. Reset gate is combining the new input with the previous memory and update gates specifies the amount of information in memory that is kept for further processing. There are a couple of key differences to LSTM - a GRU has only two gates, while LSTM has three. An update gate in GRU is in fact a combination of input and forget gates in LSTM [Cho et al., 2014].

2.3 Word Embeddings

Many NLP systems approach words as a separate symbols with no relations between them. It leads to data sparsity, requiring a lot more data in order to train machine learning models based on the statistics of occurrence. Moreover, such a system can't transfer the knowledge learnt in one area into other but similar area.

A word embedding formally maps a word using a vocabulary

$$W : words \rightarrow R^n \tag{2.2}$$

from some language into a high-dimensional vectors (can be 100 - 500 dimensions).

A simple example of the distributional representation is one-hot encoding. A particular word is encoded into the vector that is as long as the dictionary size and only one value is set to 1, others are 0. Obviously, each word has its position in the vector. The vector might look like: $\begin{bmatrix}0 & 1 & 0 & 0 & 0\end{bmatrix}$ with the meaning of columns as [*python java C# perl sql*]. The mentioned vector $\begin{bmatrix}0 & 1 & 0 & 0 & 0\end{bmatrix}$ encodes the word *java*.

One-hot encoding is simple but it has some problems. It is nearly impossible to calculate similarity between words. For example if we calculate similarity of *python*, i.e. $\begin{bmatrix}1 & 0 & 0 & 0 & 0\end{bmatrix}$ with sql, i.e. $\begin{bmatrix}0 & 0 & 0 & 0 & 1\end{bmatrix}$, the element-wise product is zero vector.

The other issue of the model is that with bigger dictionary it tends to be high dimensional.

There are generally two types of word embeddings - **frequency based** and **prediction based** embeddings. Into the category of frequency based embeddings belong models like TF-IDF or Count Vectors.

On the other hand, distributed representation encodes a word as a low-dimensional vector of real numbers. Very often the size of the vector is between 50 and 300. With such a model, we can for example represent *python* as a vector $v1 = \begin{bmatrix}0.01 & 0.23 & -0.5 & \dots & 0.4\end{bmatrix}$, *sql* as vector $v2 = \begin{bmatrix}0.0 & 0.34 & -0.44 & \dots & 0.9\end{bmatrix}$ and easily compute the similarity between those two words. With the fixed vector dimension (e.g. 300) the vocabulary can increase with no impact.

2.3.1 Word2Vec

One of the most important methods in prediction based embeddings is Word2Vec [Mikolov et al., 2013].

Word2Vec is efficient model for learning embeddings from the text. It comes as a combination of two algorithms - Continuous Bag-of-Words (CBOW) and Skip-Gram models. CBOW is used to predict target words based on the previous words and Skip-Gram predicts source words based on the target words.

Word2Vec is in fact three layer neural network with one input layer, one hidden layer and one output layer. Input words are fed into the network in order to predict their neighbouring words. The input as well as the output vectors are one-hot encoded words from vocabulary.

For example, with vocabulary size $|V| = 10000$, each input and output vectors would have 10000 components. The neural network then outputs the probability that for every word from vocabulary randomly selected nearby word is that vocabulary word. Hidden layer neurons don't use any activation function, the output layer uses softmax function.

The simplest version (with window size = 1, i.e. considering only the word right next to the given word) is depicted at figure 2.4.

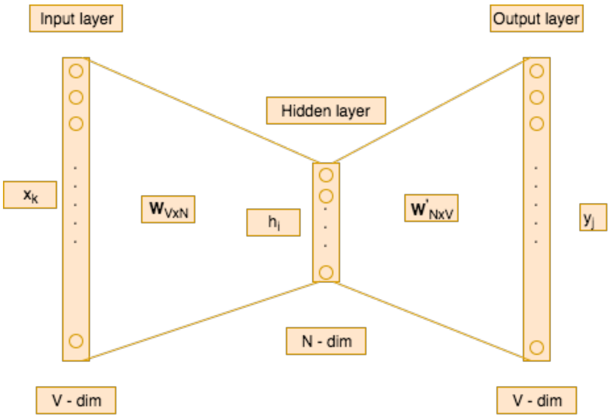

Figure 2.4: Skip-Gram model with window size of 1.

The task of the neural network is to predict neighbours. Then the last output layer is removed and only input and hidden layers are kept. As a result, the output from the hidden layer is the embedding [McCormick, 2016].

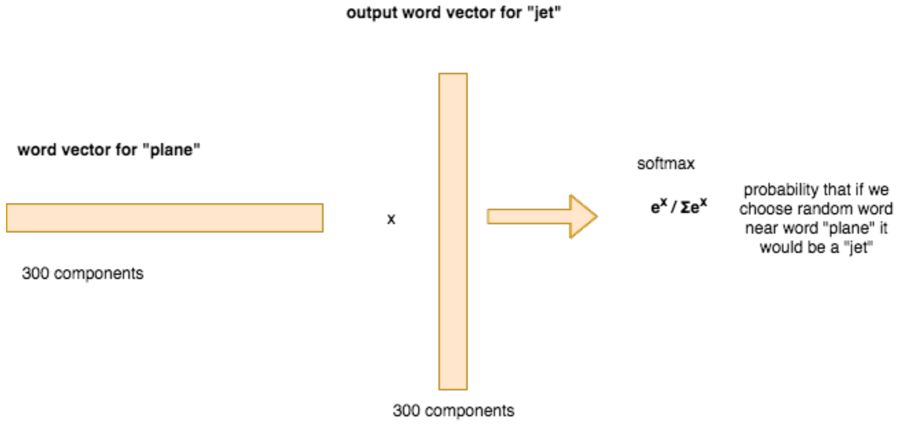
output word vector for "jet"
word vector for "plane"
300 components
x
300 components
softmax
e^x / Σe^x
probability that if we choose random word near word "plane" it would be a "jet"

Figure 2.5: Example of word2vec probability calculation.

Interesting and very useful features of such embedding is that if effectively captures semantic meanings of words and synonyms tend to have similar vectors. Moreover, with embedding vectors it has been shown that they follow rules of analogy. For example a sentence

”woman is to queen as man is to king”

can be encoded as

$$v_{\text{queen}} - v_{\text{woman}} + v_{\text{man}} \approx v_{\text{king}}$$

where $v_{\text{queen}}, v_{\text{woman}}, v_{\text{man}}, v_{\text{king}}$ are word vectors for *queen, woman, man* and *king* respectively.

More formally, Skip-Gram model's goal is to create a vector representation of words that best predicts a surrounding windows of words. The cost function is defined as:

$$J(\Theta) = \frac{1}{T} \sum_{t=1}^{T} \sum_{-c \leq j \leq c, j \neq 0} \log(p(w_{t+1}|w_t))$$

where c is the window size. Softmax function is used to compute the probability as

$$p(w_{t+1}|w_t) = \frac{e^{(v_{w_O} \tilde{v}_{wI}^T)}}{\sum_{k=1}^{V} e^{(w_i^T \tilde{w}_k)}}$$

where v_w and $\tilde{v}_w$ are input and output vectors, representing w and V is the vocabulary size. Gradient descent is then used to minimize the expression.

Important thing is that it is unsupervised problem, we don't specify the target values and thus the network has to figure it out [Turian et al., 2010].

2.3.2 GloVe

GloVe [Pennington et al., 2014] is also an algorithm to learn geometrical encoding of words based on their co-occurence. The difference is that GloVe is based on dimensionality reduction on the co-occurence matrix.

The cost function is defined as

$$J(\Theta) = \frac{1}{2}\sum_{i,j=1}^{W} f(P_{ij})(u_i^T v_j - logP_{ij})^2$$

and goes over all pairs of words in the co-occurence matrix. P_{ij} is position in the co-occurence matrix and u, v are row and column vectors for particular word, f is a weighing function that should be relative small for large values of x, so that frequent co-occurences are not overweighted. The GloVe authors defined f as

$$f(x) = (\frac{x}{x_{max}})^{\alpha} \text{ if } x < x_{max} \text{ or 1 otherwise.}$$

Although, Word2Vec and GloVe are different algorithms, the embeddings generated using these two methods perform similarly. The advantage of GloVe is better parallelization of the training process.

2.3.3 FastText

FastText is a library created by the Facebook [Bojanowski et al., 2016] for efficient learning of word representations and sentence classification. There is a significant difference to Word2Vec in the way it approaches the data. While Word2Vec considers a word as a minimal text unit, FastText assumes a word is a set of character n-grams. There are several benefits over Word2Vec or Glove like:

- it can find vector space representations for rare words as they can share n-grams with more common words,
- it can also represent a word that is not in the vocabulary but its n-grams are,
- it appears character n-grams performs better on small datasets than Word2Vec and Glove.

Otherwise, this algorithm is similar to Word2Vec and also uses CBOW and Skip Gram algorithms.

2.4 Context Polarity and Sentiment Analysis

Stance detection stands next to the sentiment analysis. The task of sentiment analysis is usually formulated as determining whether the text is positive or negative or neutral. And by analysing the sentiment, we're trying to find positive or negative opinions, emotions, and evaluations.

Sentiment Analysis and Opinion Mining is the area that is being recently heavily researched mostly due to the high demand for such a research supported by the abundance of data coming especially from social networks such as Facebook or Twitter. The key information for the stance detection as well as sentiment analysis and related fields is to distinguish whether the text presents factual information or whether presents opinions and evaluations [Turney, 2002, Yu and Hatzivassiloglou, 2003, Wilson et al., 2005a].

There are three main features of an opinion. It is an **opinion target**, the object the opinion is referring to such as person or product. Then it is an **opinion polarity**, either positive or negative or neutral. And finally, an **opinion intensity**, which defines the strength of the intensity on the predefined range, e.g. at the scale from 1 to 5, where 1 can be strongly negative and 5 strongly positive.

[Liu, 2012] defines opinion as a quadruple *(G, S, H, T)*, where G is the sentiment target, S is the sentiment about target, H is the opinion holder and T is the time when the opinion was expressed.

There are two main approaches when computationally analysing sentiment (opinions). **Lexicon-based** methods are using sentiment lexicons, list of words with pre-defined polarity, while **statistical** methods use semantic or syntactic features extracted from the text in order to correctly predict the polarity. [Pang et al., 2002, Ding et al., 2008, Kiritchenko et al., 2014a].

In lexicon-based methods, [Ding et al., 2008] suggested four step process starting with finding affective terms in the text that are in one or more lexicons. The process continues with detecting sentiment shifters, clauses that change the polarity of the sentiment such as negations. Next step is to find contrary phrases and the final step is to aggregate all previous steps.

In the statistical-based approach all currently known supervised learning algorithms can be used (SVM, Naïve Bayes, Random Forests, . . .). The main issue with supervised approach is a need of (large) dataset of labelled examples in order to train a classifier. Especially, for large datasets it can be quite time consuming to manually label the data.

The import information is also on what level is opinion analyzed. It can vary from the whole document, through sentence or phrase until word-level opinion mining. Each of these levels has it own challenges and usage. For example a Twitter tweet is usually considered as a sentence, thus **sentence-level** opinion mining, while another key area of our research, an online news is usually considered to be on the **document-level**. The issues with the document-level opinion mining are that a single document can express multiple (opposite) opinions, while word- or sentence-level lack the context that might be crucial for a correct understanding. **Aspect-level** sentiment analysis targets particular aspects (features) and extracts the sentiment towards them [Liu, 2012]. The task in **word-level** sentiment analysis is to extract semantic orientation of a subjective term. This is also often called word polarity detection.

For **word-level** sentiment analysis, widely used are methods based on the several dictionaries containing sentiment word polarity, such as *General Inquirer*[1] or *SentiWordNet* [Baccianella et al., 2010]. They are especially used in the lexicon (dictionary) based approaches or as a features for the statistical (machine learning) methods. Other approaches are computing *Pointwise Mutual Information* between the given word and a set of positive and negative paradigm words such as *good, nice, excellent, ...* and *bad, nasty, poor, ...* [Turney and Littman, 2003].

Sentiment words can be also gathered iteratively by expanding small initial set with synonyms and antonyms [Kim and Hovy, 2004, Hu and Liu, 2004].

In [Rao and Ravichandran, 2009], they approached the sentiment polarity problem as a graph task with nodes representing words and edges relations between words. Each word (node) can have either positive or negative label. Authors use WordNet and OpenOffice thesaurus and their system works for English, French and Hindi.

On the higher level (**document- or sentence-level**) of sentiment analysis, [Socher et al., 2013] created the *Sentiment Treebank*, containing sentiment labels for 215,154 phrases in the parse trees of 11,855 sentences. Then they trained a *Recursive Neural Tensor Network* against the Sentiment Treebank that outperforms previous methods in single sentence positive/negative classification by more than 5% and in sentiment labelling of phrases by almost 10%. Moreover, their system is able to capture the effects of negation in the sentence.

[Kiritchenko et al., 2014b] created a system that detects the sentiment of short informal textual messages and a sentiment of words or phrases within a message. This supervised classification system leverages many different features such as semantic, sentiment or surface-form features. They automatic-

[1] http://www.wjh.harvard.edu/inquirer/

ally generate sentiment lexicons from tweets with sentiment-word hashtags and tweets with emoticons.

Aspect-level (also called Aspect Based Sentiment Analysis (ABSA)) sentiment analysis is getting more attention recently, especially because it is bridging the academic research and the commercial needs. While majority of other approaches detects the overall sentiment of the whole document or sentence, aspect-based analysis focuses on entities mentioned in the text (e.g. phones, cars) and their aspects (battery life, screen size, consumption,...). As a first step, we need to extract aspects, usually by finding frequent nouns and noun phrases [Liu et al., 2005, Blair-Goldensohn et al., 2008, Moghaddam and Ester, 2010, Long et al., 2010].

[Hu and Liu, 2004] extended this by extracting most frequent nouns and noun phrases and removing meaningless phrases and redundant single-word features.

Extracting aspects can be also approached as an information extraction problem by using sequential learning methods such as *Hidden Markov Models* [Rabiner, 1989] or *Conditional Random Fields (CRF)* [Lafferty et al., 2001, Hercig et al., 2016]. *CRF* based approach was also successfully used in recent shared task on Aspect Based Sentiment Analysis as a part of SemEval 2016 [Pontiki et al., 2016, Hercig et al., 2016].

Popular methods for extracting aspects are also based on the topic modelling, such as sentence-level LDA (subsection 2.1.4) [Brody and Elhadad, 2010] or formulating the task as a joint model, assuming every opinion has a target, and predicting sentiment of words and topics at the same time [Mei et al., 2007, Titov and McDonald, 2008, Xianghua et al., 2013].

In terms of aspect-level sentiment classification similar methods as for other levels can be used, i.e. lexicon based or machine learning based. Lexicon based methods leverage a list of aspect-sentiment phrases [Ding et al., 2008, Xianghua et al., 2013]. [Jo and Oh, 2011] proposed a probabilistic generative model that assumes the whole sentence is related to one topic and then extend the model to *Aspect and Sentiment Unification Model (ASUM)* which incorporates aspect and sentiment together to classify sentiment for different aspects.

2.5 Argumentation Mining

Argumentation mining (sometimes also called argument mining) is a relatively new task aiming at automatic extraction of arguments from unstructured text. It automatically identifies the areas within discourse containing argumentative structures like premises or conclusions. It also prepares and feeds data into computational models of argument and reasoning engines [Lippi and Torroni, 2016].

Although, an argument mining is a new field of study, the arguments and their structure has been researched for a long time. The argument itself is viewed as a sequence of opinions that leads to a conclusion. If the argument is correct than the conclusion is correct as well [Fox et al., 1993].

There are several types of argument models. [Bentahar et al., 2010] distinguish among *monological*, *dialogical* and *rhetorical* models. Monological models focus on the relations between particular components of the argument, while dialogical models focus on the relations between arguments. Rhetorical models stand apart as they do not consider argument structures but rather their rhetorical structure like rhetorical patterns or schemas [Bentahar et al., 2010]. Several examples of dialogical models can be found in [Atkinson et al., 2006, Bentahar et al., 2003, 2004] and rhetorical models in [Grasso, 2002, O'keefe, 2002, Pasquier et al., 2006, Gordon et al., 2007].

One of the most import models belonging to monological models is Toulmin's [Toulmin, 1958], depicted in figure 2.6.

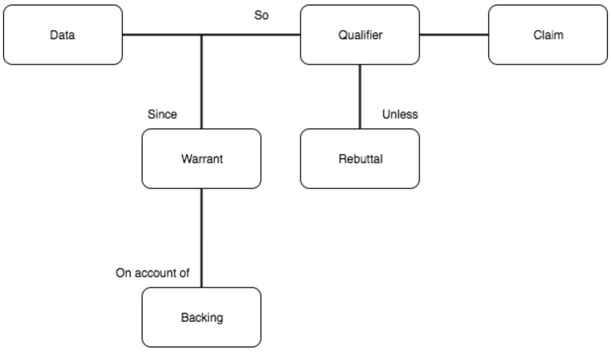

Figure 2.6: Toulmin's model.

Toulmin noticed that good and realistic arguments typically have six parts. Those are *data (D)*, facts or evidences to prove the argument; *claim (C)*, the statement put for acceptance; *warrants (W)*, hypothetical statements that bridges the claim and the data; *qualifier (Q)* limits the strength of the argument; *rebuttal (R)*, a situation when the argument is not true and *backing (B)*, statements to support the warrants [Toulmin, 1958].

This argument structure can be also represented as [Bentahar et al., 2010]:

Given D (and Since W), Therefore, C, unless R.

W Because of B.

However, because of its complexity, the Toulmin's model is not widely used.

More often used is *Claim-Premises* model [Habernal et al., 2017] containing an argument as a single claim and then the claim is either supported or attacked by several of the premises [Besnard and Hunter, 2008].

Argument mining from the social networks is more difficult than from more formal documents like legal or scientific papers. It is mostly because of informality of such texts with many grammar errors, typos and slang words.

The key difference to sentiment analysis is while sentiment is mainly working with emotions or feelings or stances, arguments are working with convictions and persuasions. And being complementary to a sentiment analysis, argument mining can provide additional information about the author or authors of the text [Somasundaran et al., 2007, Schneider et al., 2012].

2.6 Summarization

The task for every summarization system is to provide a short summary of the source document(s). Moreover, the summary should be coherent, contain only important information and obviously be grammatically correct. With the massive growth of the web, where a large amount of documents is created every day, the task of summarization is necessary in order to avoid information overload and to provide a reasonable size digest for an user. Summarization systems usually process either single document or a cluster of documents and produce a summary. There are multiple types of summarization, in the following text we will go through some necessary terminology.

Extractive summarization creates a summary using sentences from the source text in the form exactly as they appear there. On the other hand *abstractive* summarization uses different words or phrases in order to describe the document(s) being summarized.

Usually, the summarization consists of three key steps, that are more or less independent [Nenkova et al., 2011]:

Intermediate representation - every system has to first identify the important parts of a document. There are several ways how to create this representation; *topic representation* converts a raw text into a set of topics existing in the text. Here we can mention methods based on frequency or TF-IDF representation, where words with higher weights are more important. Another approach uses *latent semantic analysis*. These methods are more discussed in subsection 2.6.1.

Methods based on *lexical chains* utilize knowledge-base systems such as WordNet in order to find topics containing semantically similar words. In *graph* methods the whole document is modelled as a graph with sentences as a nodes and edges between sentences describe their similarity.

Sentence scoring - after creating an intermediate representation, each sentence is scored based on its importance. For example how much is the sentence close to a topic (in case of topic models).

Selecting sentences for the summary is a final step where a system has to choose best n sentences based on the score in order to create a summary.

Already in 1969 in [Edmundson, 1969] author suggested the approach not based on a single topic but on many different indicators that can be combined. Followed by [Kupiec et al., 1995], they framed the future machine learning approach to automatic text summarization. In a supervised learning approach, summarization can be considered as a binary classification task of of including / not including a particular sentence into the final summary. Unfortunately, a supervised approach also requires a human-labelled training data set. This is quite labour-intensive process and very often, when multiple annotators provide the reference summary, the agreement is low [Rath et al., 1961].

Classifier then scores each sentence with its confidence whether a sentence belongs to a summary or not. As described in [Hovy and Lin, 1999, Osborne, 2002, Zhou and Hovy, 2003, Leskovec et al., 2005, Fuentes et al., 2007, Wong et al., 2008] many different classifiers and wide range of various features like sentence position in the document, similarity with the document title and many others can be used in the machine learning approach to automatic summatization.

2.6.1 Intermediate Representation

Topic Representations were present already in [Luhn, 1958], where authors suggested to identify key words for the summary based on their frequency. They also excluded some stop words such such determiners, prepositions or generally very frequent words. Followed by [Harabagiu and Lacatusu, 2005, Conroy et al., 2006], topics were defined as *"words that occur often in the text but are rare in the other texts"*[Nenkova et al., 2011]. While topic representations tend to be binary (exists or not exists), **frequency-based** methods go further and add weights into the process. The simplest method is to include word probabilities such as TF-IDF.

In **Lexical Chains** approach a semantic similarity between words is considered. These methods are based on knowledge-base such as *WordNet* [Miller et al., 1990] and explore situations where cooccurrence of multiple words forms a topic better than each of those words separately.

The principle of **Latent Semantic Analysis (LSA)** algorithm is discussed in the subsection 2.1.3. In original usage, only as many topics as needed sentences were defined. The system then kept only the sentence with the highest score for each of the topics. Following research showed more promising ways such as weighing each topic and thus possible having multiple sentences per topic or the finding that sentences covering multiple topics are good candidates [Steinberger et al., 2007].

2.6.2 Final Summary Selection

Usually, some sentence-by-sentence selection method is used for the final summary. A system processes sentence by sentence sorted by their score and add most relevant ones into the summary. [Carbonell and Goldstein, 1998] suggested **Maximal Marginal Relevance (MMR)** method using greedy approach that picks a sentence with the highest score and minimal redundancy with sentences already in the summary. MMR considers both relevance as well as novelty and linearly combines them into a single score, using cosine similarity.

If we formulate summarization as an optimization task of finding the best overall summary, considering some natural constrains such as summary length or no redundancy, although it has been proved as a NP-hard problem in [Filatova and Hatzivassiloglou, 2004], we can approximate solution using integer linear programming (ILP).

[Gillick et al., 2009] works with the idea of *concept* - a minimal independent piece of information and by summing the values of a unique concept set gives a global score. As opposite to a summary of values of utterances it contains. Using ILP they seek for a summary that maximizes a global objective function

$$\text{maximize} \sum_i w_i c_i \tag{2.3}$$

where w_i is the weight of concept i and c_i is a binary variable indicating the presence of that concept in the summary. Length constraint is defined as $\sum_j l_j u_j < L$, where l_j is the utterance of j, L is the desired summary length and u_j is binary variable representing the selection of utterance j for summary. Other constraints are related to the consistency. Formally,

$$\sum_j u_j o_{ij} \geq c_i \ \forall i$$

$$u_j o_{ij} \leq c_i \ \forall i, j$$

which means a concept can be selected only if it refers to at least one selected utterance and utterance can be selected if all concepts it refers are selected. To finish the formulation of ILP task we need to define c_i and u_j as

$$c_i = 0 \text{ or } 1, \forall i; \ \ u_j = 0 \text{ or } 1, \forall j$$

But even such a methods use word frequency, TF-IDF or other type of word frequency approach [Yih et al., 2010, Filatova and Hatzivassiloglou, 2004, McDonald, 2007]. These methods tend to outperform greedy-based algorithms (MMR) and were particularly effective in certain domains such as summarization of meetings [Riedhammer et al., 2008, Gillick et al., 2009].

2.6.3 Summarization Evaluation

Manual summary evaluation is labour intensive task and that's why researchers suggested several methods how to automatically evaluate a summarization system. High quality summarization does not only covers the important parts of the source document but is also easily readable and with no grammar errors.

ROUGE (Recall-Oriented Understudy Gisty Evaluation) [Lin and Hovy, 2003] is widely used metric for evaluation of summaries. It is recall-based in order to enforce a focus on the inclusion of all important parts. ROUGE methods are comparing n-grams between a summary and one or more human summaries. There exist several versions of ROUGE, for example ROUGE-n comparing n-grams, ROUGE-L comparing longest common sequence or ROUGE-s/su comparing skip-bigrams and skip-bigrams and unigrams respectively.

ROUGEn can be computed as follows:

$$ROUGEn = \frac{\Sigma_{s \in ReferenceSummaries} \Sigma_{gram_n \in S} Count_{match}(gram_n)}{\Sigma_{s \in ReferenceSummaries} \Sigma_{gram_n} Count(gram_n)} \tag{2.4}$$

where n stands for length of the n-gram, $gram_n$ and $Count_{match}(gram_n)$ is the maximum number of n-grams co-ocuring in a automatic summary as well as reference summary [Lin and Hovy, 2003].

In [Nenkova and Passonneau, 2004] a pyramid evaluation approach is proposed. It uses Summarization Content Units (SCUs) to compute weighted scores, organised in a pyramid. In this pyramid, SCUs with higher weight, i.e. occurring more frequently are in the higher tiers of pyramid and indicate higher importance. The order of pyramid (i.e. the number of tiers) refers to the number of manual summaries. Image 2.7 shows pyramid constructed out of 4 manual summaries. SCUs of weight 4 are in all 4 summaries and thus are in the top tier of the pyramid. Then the optimal summary should contain all the SCUs from the top tier, if we need more then SCUs from the tier below the top and so on.

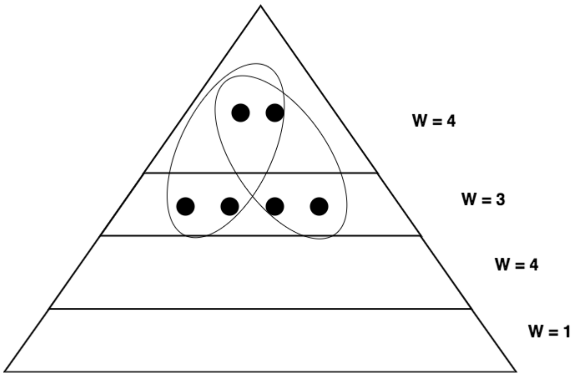

Figure 2.7: Pyramid Evaluation Method

More formally, the total SCU weight is

$$D = \sum_{i=1}^{n} i \times D_i$$

where D_i is the number of SCUs in the summary that are in tier T_i. A pyramid has n tiers, T_n is the top and T_1 the bottom tier. The optimal score for a summary is then calculated as

$$\text{Max} = \sum_{i=j+1}^{n} i \times |T_i| + j \times (X - \sum_{i=j+1}^{n} |T_i|)$$

$$j = \max_i(\sum_{t=i}^{n} |T_t \geq X)$$

where X is the number of SCUs, $|T_i|$ is the number of SCUs in tier T_i and j is the index of the lowest tier an optimal summary will draw from. Finally, the pyramid score P is then the ratio of D to Max [Nenkova and Passonneau, 2004, Cohan and Goharian, 2016].

Pyramid evaluation and the need to speed up the evaluation process were also the inspiration for [Steinberger et al., 2017]. This approach automatize the evaluation process by comparing abstract meaning representations (AMR) [Banarescu et al., 2013] of SCUs and summary sentences.

Described approaches can help evaluating the result of summarization but there are still some aspects out of scope of these methods. Usually, some human judges are involved in order to evaluate a clarify or a coherence of summarization.

2.7 Summary

In this chapter we discussed all necessary background needed to understand modern NLP approaches to stance detection and summarization. We highlighted the differences between sentiment analysis and stance detection, we showed how significant topics in the text can be uncovered and how the text can be summarized. We also touched modern machine learning approaches to process nature language texts using (deep) neural networks.

Chapter 3

Stance Detection In Online News

In the recent years several researchers approached a task of stance detection, originally as an extension of sentiment analysis. In this chapter we would like to review current methods of researching and modelling stance in the textual information.

Existing works focused mostly on three areas: congressional debates, internal discussion sites and online social and public forums and networks.

[Thomas et al., 2006] investigated whether it is possible from the transcripts of U.S. Congressional floor debates infer the support or opposition to proposed legislation. They approached the task as a sentiment analysis problem, concretely document-level sentiment classification.

Other direction of research focused on online forums dedicated to various types of discussions. These debate sites usually contain two-sided debates where authors always add their opinion under one or the other opinion. For researchers it is quite convenient as they don't need to label the data manually but immediately have an access to gold labels. [Somasundaran and Wiebe, 2009, Anand et al., 2011, Walker et al., 2012a, Hasan and Ng, 2014] analyzed data from these online discussion websites[1].

[Somasundaran and Wiebe, 2009] approached the task as an unsupervised classification problem formulated as an Integer Linear Programming problem. They first use web corpus to learn preferences associated with a particular side. These preferences are later used to identify the side for a particular post. Their approach involves using an MPQA subjectivity lexicon (Wiebe et al. [2005]).

[Anand et al., 2011] utilized data about 14 different debates on topics such

[1] `http://www.convinceme.net` or `http://www.createdebate.com/`

as *cats vs dogs, Firefox vs IE, 2nd Ammendment, Climate Change* and others. Interestingly, human annotators achieved accuracies from 66% to 94%. Their system used unigrams, features generated from Linguistic Inquiry Word Count Tool (LIWC) [Pennebaker et al., 2001], generalized dependency features containing MPQA terms [Wiebe et al., 2005] and part-of-speech tags and Naïve Bayes as a classifier. For the topic where human annotators achieved an accuracy 94% their system achieved 62.31%, showing there is not a significant difference between unigram based features and more advanced lexicon-based features. It also shows there is still a huge space for improvement of automatic systems.

As most of the researchers use some version of either supervised or unsupervised machine learning system, features definition is the crucial and important task. Features commonly used can be divided into multiple groups:

- **Lexical features**
  - n-grams,
  - initial n-grams (i.e. first n-gram of a post),
  - number of sentences, words and characters,
  - repeated characters (exclamations, question marks, punctuations, . . .).
- **Morphological Features**
  - part-of-speech,
- **Semantic Features**
  - sentence polarity,
  - sentiment,
  - Linguistic Inquiry Word Count Tool (LIWC) [Pennebaker et al., 2001],
- **Syntactic Features**
  - dependency tree,
- **Non-Linguistic Features**
  - author constraints,
  - user-interaction constraints or ideology constraints,

[Anand et al., 2011, Sridhar et al., 2014, Somasundaran and Wiebe, 2010, 2009, Walker et al., 2012a, Hasan and Ng, 2013, Joshi and Penstein-Rosé, 2009, Lin and He, 2009, Lu et al., 2012].

3.1 Existing Datasets

There are several existing datasets related to the stance detection. Due to the abundance of data coming from various discussion groups and forums such as *4forums.com* and *createdebates.com*, researchers were able to create a datasets mostly reflecting currently discussed ideological or controversial topics related to guns, gay rights or abortion. Usually, these discussions forums provide comments labelled as pro or cons by the posts' authors.

[Somasundaran and Wiebe, 2010] prepared a dataset based on the multiple debating websites and containing topics such as *Gun Rights, Gay Rights or Abortion.* Similarly, [Hasan and Ng, 2013], aggregated comments about *Abortion, Obama, Marijuana and Gay Rights.* In [Walker et al., 2012b] a large corpus was created based on the debates from *4forums.com*. Their corpus named the Internet Argument Corpus (IAC) is a collection of almost 400,000 posts from over 3,000 authors in almost 12,000 discussions. Additionally, they annotated about 6,000 posts about 10 topics, using labels as *pro, con* and *other.*

More recently, [Ferreira and Vlachos, 2016] prepared an Emergent, dataset for stance classification. It contains 300 rumored claims and over 2,500 related news articles.

One of the mostly used datasets for stance detection was created to support shared task at SemEval 2016 [Mohammad et al., 2016b]. The corpus contains over 4,000 tweets about following topics: *Atheism, Climate Change is Concern, Feminist Movement, Hillary Clinton* and *Legalization of Abortion.* Apart from the tweets labelled for stance detection with labels *favor, against* and *neutral*, the authors provided also all tweets labelled with the sentiment. Moreover, they collected about 78,000 unlabelled tweets related to the topic *Donald Trump* in order to support weakly-supervised task B.

3.2 Czech Stance Corpus

We understand the importance of having manually labelled stance corpus for the Czech language and thus we created in [Krejzl et al., 2016] a corpus of 1,560 manually annotated comments from a Czech news server[2] related to two topics - "**Miloš Zeman**" (the Czech president) and "**Smoking ban in restaurants**".

Later, in [Hercig et al.] we extended this dataset almost four times. Statistics of the Czech corpora in terms of the number of news comments and stance labels is in the table 3.2 and detailed annotation procedure in [Hour-

[2] http://www.idnes.cz

ova, 2017].

Target Entity	Total	*In favor*	*Against*	*Neither*
"Miloš Zeman"	2,638	691 (26%)	1,263 (48%)	684 (26%)
"Smoking ban" - Gold	1,388	272 (20%)	485 (35%)	631 (45%)
"Smoking ban" - All	2,785	744 (27%)	1,280 (46%)	761 (27%)

The target entity "**Miloš Zeman**" part of the dataset was annotated by one annotator and then 302 comments were also labeled by a second annotator to measure inter-annotator agreement. The target entity "**Smoking Ban in Restaurants**" part of the dataset was independently annotated by two annotators. To resolve conflicts a third annotator was used and then the majority voting scheme was applied to the gold label selection.

The inter-annotator agreement (Cohens κ) was calculated between two annotators on 2,203 comments. The final κ is 0.579 for "**Miloš Zeman**" (2,638 comments) and 0.423 for "**Smoking Ban in Restaurants**" (2,785 comments).

The inter-annotator agreement for the target "**Smoking Ban in Restaurants**" was quite low, thus we selected a subset of the "**Smoking Ban in Restaurants**" part of dataset, where the original two annotators assigned the same label as the gold dataset (1,388 comments).

The corpus is available for research purposes at http://nlp.kiv.zcu.cz/research/sentiment#stance.

3.3 SemEval 2016 - Detecting Stance in Tweets

Currently, a lot of attention was brought to stance detection thanks to SemEval 2016 and shared task *Detecting Stance in Tweets*[3][Mohammad et al., 2016a].

The task was, given a tweet and a target, classify whether the author of the tweet is *in favor, against* or *neither* of the target.

The data corresponding to five of the targets (*Atheism, Climate Change is a Real Concern, Feminist Movement, Hillary Clinton*, and *Legalization of Abortion*) was used in a standard supervised stance detection task - Task A. About 70% of the tweets per target were used for training and the remaining for testing.

All of the data corresponding to the target *Donald Trump* was used as test set in a separate task - weakly supervised Task B [Mohammad et al., 2016a].

[3] http://alt.qcri.org/semeval2016/task6/

Team	$F_{\mathbf{avg}}$	F_{favour}	F_{against}
MITRE	**67.82**	59.32	76.33
pkudlab	67.33	61.98	72.67
TakeLab	66.83	60.93	72.73
...	...	...	...
Our system (ranked 9th)	63.42	57.41	69.42
...	...	...	...
Last system	46.19	30.16	62.23

Table 3.1: SemEval 2016 Task A Results. 19 teams participated.

The description of all systems is beyond the scope of this paper, so we just briefly describe some of the systems, including our approach.

Total number of 19 teams participated in task A and 9 teams in task B. The highest achieved F-score was 67.82 for Task A and 56.28 for task B.

The systems can be divided into two separate groups. Systems in the first group were using common text classification features (n-grams, word embeddings) as well as features related to sentiment analysis. Systems in the second group utilised various neural networks architectures such as autoencoders, recursive neural networks or convolutional neural networks.

All top three teams used word embeddings [Zarrella and Marsh, 2016, Wei et al., 2016, Tutek et al., 2016] and many teams also used available sentiment lexicons such as NRC Emotion Lexicon [Mohammad and Turney, 2010], Lexicon described in [Hu and Liu, 2004],the MPQA Subjectivity Lexicon [Wilson et al., 2005b] or NRC Hashtag Lexicons [Kiritchenko et al., 2014b].

The best results for task A achieved [Zarrella and Marsh, 2016] with an overall $F_{\text{avg}} = 67.82$. Authors used a recurrent neural network initialised with features learned via distant supervision on two large unlabelled datasets. They trained a neural network to predict relevant hashtags on a large Twitter corpus. Outputs of the first network were used to feed into the second neural network in order to provide final stance classification. For the first neural network they trained word embeddings with the word2vec skip-gram model (more in subsection 2.3.1).

In the tables 3.1 and 3.2 we present shortened version of the full results in order to demonstrate results of our system as well the difference between the teams.

Our approach was based on a maximum entropy classifier which uses surface-level, sentiment and domain-specific features. After initial text preprocessing (removing stop-words, replacing urls, twitter user names, multiple exclamations and question marks with constant strings), we defined

Team	$F_{\mathbf{avg}}$	F_{favour}	F_{against}
pkudlab	56.28	57.39	55.17
LitisMind	44.66	30.04	59.28
INF-UFRGS	42.32	32.56	52.90
Our system (ranked 4th)	42.02	34.26	49.78
...	...	...	...
Last system	25.73	16.59	34.87

Table 3.2: SemEval 2016 Task B Results. 9 teams participated.

text features. Unigrams were already proved to work well [Somasundaran and Wiebe, 2009] in this case. Twitter hashtags are also important, so we built unigram and bigram features out of them, using TF-IDF weighing. [Anand et al., 2011] showed that initial n-grams are useful features. Our system supported initial unigrams to initial trigrams. However, from our experiments with the training dataset, we found useful only initial unigrams, and initial bigrams for the Hillary Clinton target. Another surface feature was tweet length (in words) after preprocessing.

Part-of-speech tags were generated from the preprocessed tweet and we built unigram and bigram data model. General Inquirer[4] (General-Inquirer, 1966) provides dictionaries useful for example for sentiment analysis. We used a subset of the dictionary, in particular columns: *Positiv, Negativ, Hostile,Strong, Pleasure, Pain.*

We also used another resource borrowed from the sentiment analysis: dictionaries created mainly for the purpose of entity-related polarity detection [Steinberger et al., 2012]. Based on the training data analysis of each topic, we created a list of key words that tend to indicate a particular stance. We first generated a list of candidates: for each topic, we took words with ratio $frequency - in - topic/frequency -in -the - training - data > 0.6$ and $frequency - in - topic > 1$. If a word occurred at least four times more frequently in *IN FAVOR* tweets than in *AGAINST*, it was added to the *IN FAVOR* candidates' list. We repeated the same approach to produce *AGAINST* candidates. The lists were then filtered manually and it resulted in strong stance-predictive keywords lists.

Our system performed well for *Abortion* (2nd), *Climate* (3rd) and *Hillary Clinton* (4th) targets in comparison with the other participating systems, we received an average rank for *Atheism* and *Feminism*. The overall rank was 9th. In the weakly supervised subtask (*Donald Trump*), we were ranked 4th, only the top system was significantly better. More details about our approach can be found in [Krejzl and Steinberger, 2016].

[4] http://www.wjh.harvard.edu/inquirer/

Chapter 4

Stance Summarization

Automatic summarization deals with the problem of producing a succinct gist for a document (or a set of documents about the same topic) [Steinberger, 2013] or a summary is defined as a text that is produced from one or more texts, that conveys important information in the original text(s), and is no longer than half of the original text(s) and usually, significantly less than that [Radev et al., 2002].

The task of summarization here is to prepare most relevant opinions related to a given target for each of two (in favor, against) stance categories. Neutral class is usually ignored but the approach would be the same, if one would decide to add it. In the context of stance summarization, we are talking about multi-document summarization.

The difference to the existing summarization tasks is that stance summarization is in fact extractive summarization applied to the whole dataset of stance-related documents split into two subsets - one for documents *"in favor"* of a given topic and the other of documents *"against"* a given topic.

4.1 Multiling's OnForumS Task

The Multiling shared tasks were traditionally linked to summarization of news articles [Giannakopoulos et al., 2011, Giannakopoulos, 2013]. However, the increasing amount of user-supplied comments in most major online news portals suggests the need for automatic summarization methods, which brings a novel challenge for the summarization community.

In 2015 organisers brought a new task: Online Forum Summarization (OnForumS). The purpose of the OnForumS track is to set the ground for investigating how such a mass of comments can be summarized. An important initial step in developing reader comment summarization systems is to determine what comments relate to, be that either specific points within the text of the article, the global topic of the article, or comments made by other users. This constitutes a linking task. Furthermore, a set of link types or labels may be articulated to capture whether, for example, a comment agrees with, elaborates, disagrees with, etc., the point made in the commented-upon text. Solving this labelled linking problem should facilitate the creation of reader comment summaries by allowing, for example, that comments relating to the same article content can be clustered, points attracting the most comment can be identified, representative comments can be chosen for each key point, and the implications of labelled links can be digested (e.g., numbers for or against a particular point), etc.

The OnForumS task is a particular specification of the linking task, in which systems take as input a news article with a reduced set of comments (sifted, according to predefined criteria, from what could otherwise be thousands of comments) and are asked to link and label each comment to sentences in the article (which, for simplification, are assumed to be the appropriate units here) or to preceding comments. The labels include agreement/disagreement and sentiment indicators. The data cover two languages (English and Italian) [Kabadjov et al., 2015].

Our approach consists of computing the *similarity score* between sentences based on two models: Vector Space Model (VSM) and Latent Dirichlet Allocation (LDA). Both models are described in chapter 2.1. The score is calculated as a average of these two models. After computing the similarity score a list of link candidates is produced - they are either comments to article sentence or comment to another comment. The final output, containing two runs, consist of one and two percents of links respectively ordered by the similarity score. For each of these links, we calculated also a sentiment. The final agreement label is calculated based on the sentiment of both sentences in the pair.

For English, in 5 of the 10 English articles, all the links proposed by our system were correct and system was ranked 4th (out of 9). All prediction of argument structure were correct in 8 articles. Our system was ranked 3rd with a very large precision (0.974). More details of our approach in [Krejzl et al., 2015].

Chapter 5

Preliminary Results and Future Work

This chapter describes preliminary results and plans for the future work.

We have done a deep research of the current state-of-the-art approaches for the stance detection. We have built a system for stance detection that successfully participated in SemEval 2016 shared task [Krejzl and Steinberger, 2016] and later we extended it for Czech language in [Krejzl et al., 2016].

In [Krejzl et al., 2016, Hercig et al.] we have created an annotated Czech corpus for the stance detection containing over 5,000 manually labelled news comments for the further research.

In [Krejzl et al., 2015] we attacked the problem of multilingual online forums summarization and in [Steinberger et al., 2017] we proposed a novel metric for evaluating summary content coverage.

5.1 Overall Aims of the PhD Thesis

The goal of doctoral thesis is to propose novel methods for improving performance of stance detection and summarization with the emphasis on multilingual online forum approach. The following work will be focused on the following tasks:

- Explore and extend existing state-of-the-art systems for stance detection.
- Propose a novel approach for stance summarization.
- Explore multilinguality in both stance detection and stance summarization with focus on English and Czech.

Bibliography

Pranav Anand, Marilyn Walker, Rob Abbott, Jean E. Fox Tree, Robeson Bowmani, and Michael Minor. Cats rule and dogs drool!: Classifying stance in online debate. In *Proceedings of the 2Nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis*, WASSA '11, pages 1–9, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics. ISBN 9781937284060. URL http://dl.acm.org/citation.cfm?id=2107653.2107654.

Katie Atkinson, Trevor Bench-Capon, and Peter McBurney. Computational representation of practical argument. *Synthese*, 152(2):157–206, 2006.

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *LREC*, volume 10, pages 2200–2204, 2010.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, 2013.

Jamal Bentahar, Bernard Moulin, and Brahim Chaib-draa. Commitment and argument network: a new formalism for agent communication. In *Workshop on Agent Communication Languages*, pages 146–165. Springer, 2003.

Jamal Bentahar, Bernard Moulin, John-Jules Ch Meyer, and Brahim Chaib-draa. A computational model for conversation policies for agent communication. In *International Workshop on Computational Logic in Multi-Agent Systems*, pages 178–195. Springer, 2004.

Jamal Bentahar, Bernard Moulin, and Micheline Bélanger. A taxonomy of argumentation models used for knowledge representation. *Artificial Intelligence Review*, 33(3):211–259, 2010.

Philippe Besnard and Anthony Hunter. *Elements of argumentation*, volume 47. MIT press Cambridge, 2008.

István Bíró. Document classification with latent dirichlet allocation. *Unpublished Doctoral Dissertation, Eotvos Lorand University*, 4, 2009.

Sasha Blair-Goldensohn, Kerry Hannan, Ryan McDonald, Tyler Neylon, George A Reis, and Jeff Reynar. Building a sentiment summarizer for local service reviews. In *WWW workshop on NLP in the information explosion era*, volume 14, pages 339–348, 2008.

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. *arXiv preprint arXiv:1206.6392*, 2012.

Samuel Brody and Noemie Elhadad. An unsupervised aspect-sentiment model for online reviews. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 804–812. Association for Computational Linguistics, 2010.

Jaime Carbonell and Jade Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336. ACM, 1998.

Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

Arman Cohan and Nazli Goharian. Revisiting summarization evaluation for scientific articles. *arXiv preprint arXiv:1604.00400*, 2016.

John M Conroy, Judith D Schlesinger, and Dianne P O'Leary. Topic-focused multi-document summarization using an approximate oracle score. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 152–159. Association for Computational Linguistics, 2006.

Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391, 1990.

Xiaowen Ding, Bing Liu, and Philip S Yu. A holistic lexicon-based approach to opinion mining. In *Proceedings of the 2008 international conference on web search and data mining*, pages 231–240. ACM, 2008.

Douglas Eck and Juergen Schmidhuber. A first look at music composition using lstm recurrent neural networks. *Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale*, 103, 2002.

Harold P Edmundson. New methods in automatic extracting. *Journal of the ACM (JACM)*, 16(2):264–285, 1969.

William Ferreira and Andreas Vlachos. Emergent : a novel data-set for stance classification. *Naacl2016*, pages 1163–1168, 2016. doi: 10.18653/v1/N16-1138.

Elena Filatova and Vasileios Hatzivassiloglou. A formal model for information selection in multi-sentence text extraction. In *Proceedings of the 20th international conference on Computational Linguistics*, page 397. Association for Computational Linguistics, 2004.

John Fox, Paul Krause, and Morten Elvang-Gøransson. Argumentation as a general framework for uncertain reasoning. In *Proceedings of the Ninth international conference on Uncertainty in artificial intelligence*, pages 428–434. Morgan Kaufmann Publishers Inc., 1993.

Maria Fuentes, Enrique Alfonseca, and Horacio Rodríguez. Support vector machines for query-focused summarization trained and evaluated on pyramid data. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 57–60. Association for Computational Linguistics, 2007.

George Giannakopoulos. Multi-document multilingual summarization and evaluation tracks in acl 2013 multiling workshop. In *Proceedings of the MultiLing 2013 Workshop on Multilingual Multidocument Summarization*, pages 20–28, 2013.

George Giannakopoulos, Mahmoud El-Haj, Benoit Favre, Marianna Litvak, Josef Steinberger, and Vasudeva Varma. Tac 2011 multiling pilot overview. 2011.

Dan Gillick, Korbinian Riedhammer, Benoit Favre, and Dilek Hakkani-Tur. A global optimization framework for meeting summarization. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 4769–4772. IEEE, 2009.

Yihong Gong and Xin Liu. Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th annual* *international ACM SIGIR conference on Research and development in information retrieval*, pages 19–25. ACM, 2001.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016a. http://www.deeplearningbook.org.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep learning (2016). *Book in preparation for MIT Press. URL: http://www. deeplearningbook. org*, 2016b.

Thomas F Gordon, Henry Prakken, and Douglas Walton. The carneades model of argument and burden of proof. *Artificial Intelligence*, 171(10): 875–896, 2007.

Floriana Grasso. Towards a framework for rhetorical argumentation. In *EDILOG 02: Proceedings of the 6th workshop on the semantics and pragmatics of dialogue*, pages 53–60, 2002.

Thomas L Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National academy of Sciences*, 101(suppl 1):5228–5235, 2004.

Ivan Habernal, Henning Wachsmuth, Iryna Gurevych, and Benno Stein. The Argument Reasoning Comprehension Task. 2017. URL http://arxiv.org/abs/1708.01425.

Sanda Harabagiu and Finley Lacatusu. Topic themes for multi-document summarization. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 202–209. ACM, 2005.

Kazi Saidul Hasan and Vincent Ng. Extra-linguistic constraints on stance recognition in ideological debates. In *ACL (2)*, pages 816–821, 2013.

Kazi Saidul Hasan and Vincent Ng. Why are you taking this stance? identifying and classifying reasons in ideological debates. In *EMNLP*, volume 14, pages 751–762, 2014.

Tomáš Hercig, Peter Krejzl, Barbora Hourová, Josef Steinberger, and Ladislav Lenc. Detecting stance in czech news commentaries.

Tomáš Hercig, Tomáš Brychcín, Lukáš Svoboda, and Michal Konkol. Uwb at semeval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th international workshop on semantic evaluation (SemEval-2016)*, pages 342–349, 2016.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, Jürgen Schmidhuber, et al. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

Barbora Hourova. Automatic detection of argumentation. Master's thesis, Faculty of Applied Sciences, University of West Bohemia, Czech Republic, 2017.

Eduard Hovy and Chin-Yew Lin. Automated text summarization in summarist. advances in automatic text summarization, 81-94, 1999.

Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM, 2004.

Yohan Jo and Alice H Oh. Aspect and sentiment unification model for online review analysis. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 815–824. ACM, 2011.

Mahesh Joshi and Carolyn Penstein-Rosé. Generalizing dependency features for opinion mining. In *Proceedings of the ACL-IJCNLP 2009 conference short papers*, pages 313–316. Association for Computational Linguistics, 2009.

Mijail Kabadjov, Josef Steinberger, Emma Barker, Udo Kruschwitz, and Massimo Poesio. Onforums: The shared task on online forum summarisation at multiling'15. In *Proceedings of the 7th forum for information retrieval evaluation*, pages 21–26. ACM, 2015.

Soo-Min Kim and Eduard Hovy. Determining the sentiment of opinions. In *Proceedings of the 20th international conference on Computational Linguistics*, page 1367. Association for Computational Linguistics, 2004.

Svetlana Kiritchenko, Xiaodan Zhu, Colin Cherry, and Saif Mohammad. Nrc-canada-2014: Detecting aspects and sentiment in customer reviews. In *SemEval@ COLING*, pages 437–442, 2014a.

Svetlana Kiritchenko, Xiaodan Zhu, and Saif M Mohammad. Sentiment analysis of short informal texts. *Journal of Artificial Intelligence Research*, 50:723–762, 2014b.

Peter Krejzl and Josef Steinberger. Uwb at semeval-2016 task 6: Stance detection. *Proceedings of SemEval*, pages 408–412, 2016.

Peter Krejzl, Josef Steinberger, Tomáš Hercig, and Tomáš Brychcín. Online Forum Summarization. *Proceedings of the DaZ 2015,34th Conference on Data and Knowledge, Prague, Czech Republic*, 2015.

Peter Krejzl, Barbora Hourová, and Josef Steinberger. Stance detection in online discussions. *Proceedings of the WIKT & DaZ 2016, 11th Workshop on Intelligent and Knowledge Oriented Technologies,35th Conference on Data and Knowledge, Smolenice, Slovakia*, 2016.

Julian Kupiec, Jan Pedersen, and Francine Chen. A trainable document summarizer. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 68–73. ACM, 1995.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

Thomas K Landauer and Susan T Dumais. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, 104(2):211, 1997.

Thomas K Landauer, Peter W Foltz, and Darrell Laham. An introduction to latent semantic analysis. *Discourse processes*, 25(2-3):259–284, 1998.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Yann LeCun et al. Generalization and network design strategies. *Connectionism in perspective*, pages 143–155, 1989.

Jure Leskovec, Natasa Milic-Frayling, and Marko Grobelnik. Impact of linguistic analysis on the semantic graph coverage and learning of document extracts. 2005.

Chenghua Lin and Yulan He. Joint sentiment/topic model for sentiment analysis. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 375–384. ACM, 2009.

Chin-Yew Lin and Eduard Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 71–78. Association for Computational Linguistics, 2003.

Marco Lippi and Paolo Torroni. Argumentation mining: State of the art and emerging trends. *ACM Transactions on Internet Technology (TOIT)*, 16(2):10, 2016.

Bing Liu. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167, 2012.

Bing Liu, Minqing Hu, and Junsheng Cheng. Opinion observer: analyzing and comparing opinions on the web. In *Proceedings of the 14th international conference on World Wide Web*, pages 342–351. ACM, 2005.

Chong Long, Jie Zhang, and Xiaoyan Zhut. A review selection approach for accurate feature rating estimation. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 766–774. Association for Computational Linguistics, 2010.

Yue Lu, Hongning Wang, ChengXiang Zhai, and Dan Roth. Unsupervised discovery of opposing opinion networks from forum discussions. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 1642–1646. ACM, 2012.

Hans Peter Luhn. The automatic creation of literature abstracts. *IBM Journal of research and development*, 2(2):159–165, 1958.

Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. Scoring, term weighting and the vector space model. *Introduction to information retrieval*, 100:2–4, 2008.

Jiří Materna. LDA-Frames: An Unsupervised Approach to Generating Semantic Frames. In Alexander Gelbukh, editor, *Proceedings of the 13th International Conference CICLing 2012, Part I*, volume 7181 of *Lecture Notes in Computer Science*, pages 376–387. Springer Berlin / Heidelberg, 2012. ISBN 978-3-642-28603-2.

C. McCormick. Word2vec tutorial - the skip-gram model, 2016. URL http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/.

Ryan McDonald. A study of global inference algorithms in multi-document summarization. *Advances in Information Retrieval*, pages 557–564, 2007.

Qiaozhu Mei, Xu Ling, Matthew Wondra, Hang Su, and ChengXiang Zhai. Topic sentiment mixture: modeling facets and opinions in weblogs. In *Proceedings of the 16th international conference on World Wide Web*, pages 171–180. ACM, 2007.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

George A Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J Miller. Introduction to wordnet: An on-line lexical database. *International journal of lexicography*, 3(4):235–244, 1990.

Samaneh Moghaddam and Martin Ester. Opinion digger: an unsupervised opinion miner from unstructured product reviews. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 1825–1828. ACM, 2010.

Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiao-Dan Zhu, and Colin Cherry. Semeval-2016 task 6: Detecting stance in tweets. In *SemEval@ NAACL-HLT*, pages 31–41, 2016a.

Saif M Mohammad and Peter D Turney. Emotions evoked by common words and phrases: Using mechanical turk to create an emotion lexicon. In *Proceedings of the NAACL HLT 2010 workshop on computational approaches to analysis and generation of emotion in text*, pages 26–34. Association for Computational Linguistics, 2010.

Saif M Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. SemEval-2016 Task 6: Detecting Stance in Tweets. pages 31–41, 2016b.

Goutam Nair. Text mining 101: Topic modeling, 2016. URL http://www.kdnuggets.com/2016/07/text-mining-101-topic-modeling.html.

Ani Nenkova and Rebecca J Passonneau. Evaluating content selection in summarization: The pyramid method. In *HLT-NAACL*, volume 4, pages 145–152, 2004.

Ani Nenkova, Kathleen McKeown, et al. Automatic summarization. *Foundations and Trends® in Information Retrieval*, 5(2–3):103–233, 2011.

Daniel J O'keefe. *Persuasion: Theory and research*, volume 2. Sage, 2002.

Miles Osborne. Using maximum entropy for sentence extraction. In *Proceedings of the ACL-02 Workshop on Automatic Summarization-Volume 4*, pages 1–8. Association for Computational Linguistics, 2002.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics, 2002.

Philippe Pasquier, Iyad Rahwan, Frank Dignum, and Liz Sonenberg. Argumentation and persuasion in the cognitive coherence theory. *COMMA*, 144:223–234, 2006.

James W Pennebaker, Martha E Francis, and Roger J Booth. Linguistic inquiry and word count: Liwc 2001. *Mahway: Lawrence Erlbaum Associates*, 71(2001):2001, 2001.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammed AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, et al. Semeval-2016 task 5: Aspect based sentiment analysis. In *ProWorkshop on Semantic Evaluation (SemEval-2016)*, pages 19–30. Association for Computational Linguistics, 2016.

Lawrence R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

Dragomir R. Radev, Eduard Hovy, and Kathleen McKeown. Introduction to the special issue on summarization. *Comput. Linguist.*, 28(4):399–408, December 2002. ISSN 0891-2017. doi: 10.1162/089120102762671927. URL http://dx.doi.org/10.1162/089120102762671927.

Delip Rao and Deepak Ravichandran. Semi-supervised polarity lexicon induction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 675–682. Association for Computational Linguistics, 2009.

Sebastian Raschka and Vahid Mirjalili. *Python Machine Learning, 2nd Ed.* Packt Publishing, Birmingham, UK, 2 edition, 2017. ISBN 978-1787125933.

GJ Rath, A Resnick, and TR Savage. The formation of abstracts by the selection of sentences. part i. sentence selection by men and machines. *Journal of the Association for Information Science and Technology*, 12 (2):139–141, 1961.

Korbinian Riedhammer, Dan Gillick, Benoit Favre, and Dilek Hakkani-Tür. Packing the meeting summarization knapsack. In *Ninth Annual Conference of the International Speech Communication Association*, 2008.

Jodi Schneider, Brian Davis, and Adam Wyner. Dimensions of argumentation in social media. *Knowledge Engineering and Knowledge Management*, pages 21–25, 2012.

Parinaz Sobhani. *Stance Detection and Analysis in Social Media*. PhD thesis, Université d'Ottawa/University of Ottawa, 2017.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

Swapna Somasundaran and Janyce Wiebe. Recognizing stances in online debates. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 226–234. Association for Computational Linguistics, 2009.

Swapna Somasundaran and Janyce Wiebe. Recognizing Stances in Ideological On-Line Debates. pages 116–124, 2010.

Swapna Somasundaran, Josef Ruppenhofer, and Janyce Wiebe. Detecting arguing and sentiment in meetings. In *Proceedings of the SIGdial Workshop on Discourse and Dialogue*, volume 6, 2007.

Dhanya Sridhar, Lise Getoor, and Marilyn Walker. Collective stance classification of posts in online debate forums. In *Proceedings of the Joint Workshop on Social Dynamics and Personal Attributes in Social Media*, pages 109–117, 2014.

Josef Steinberger. Multilingual summarisation and sentiment analysis. 2013.

Josef Steinberger and Karel Jezek. Using latent semantic analysis in text summarization and summary evaluation. In *Proc. ISIM?04*, pages 93–100, 2004.

Josef Steinberger, Massimo Poesio, Mijail A Kabadjov, and Karel Ježek. Two uses of anaphora resolution in summarization. *Information Processing & Management*, 43(6):1663–1680, 2007.

Josef Steinberger, Mohamed Ebrahim, Maud Ehrmann, Ali Hurriyetoglu, Mijail Kabadjov, Polina Lenkova, Ralf Steinberger, Hristo Tanev, Silvia Vázquez, and Vanni Zavarella. Creating sentiment dictionaries via triangulation. *Decision Support Systems*, 53(4):689–694, 2012.

Josef Steinberger, Peter Krejzl, and Tomáš Brychcín. Pyramid-based summary evaluation using abstract meaning representation. In *Proceedings of the International Conference RANLP 2017*, pages 701–706, 2017.

Ilya Sutskever, Geoffrey E Hinton, and Graham W Taylor. The recurrent temporal restricted boltzmann machine. In *Advances in Neural Information Processing Systems*, pages 1601–1608, 2009.

Ilya Sutskever, James Martens, and Geoffrey E Hinton. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1017–1024, 2011.

Matt Thomas, Bo Pang, and Lillian Lee. Get out the vote: Determining support or opposition from congressional floor-debate transcripts. In *Proceedings of the 2006 conference on empirical methods in natural language processing*, pages 327–335. Association for Computational Linguistics, 2006.

Ivan Titov and Ryan McDonald. Modeling online reviews with multi-grain topic models. In *Proceedings of the 17th international conference on World Wide Web*, pages 111–120. ACM, 2008.

Stephen Toulmin. The uses of argument, 1958.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 384–394. Association for Computational Linguistics, 2010.

Peter D Turney. Thumbs up or thumbs down? Semantic Orientation applied to Unsupervised Classification of Reviews. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 417–424, 2002. ISSN 0738467X.

Peter D Turney and Michael L Littman. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems (TOIS)*, 21(4):315–346, 2003.

Martin Tutek, Ivan Sekulic, Paula Gombar, Ivan Paljak, Filip Culinovic, Filip Boltuzic, Mladen Karan, Domagoj Alagić, and Jan Šnajder. Takelab at semeval-2016 task 6: stance classification in tweets using a genetic algorithm based ensemble. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 464–468, 2016.

Jeffrey David Ullman. *Mining of massive datasets.* Cambridge University Press, 2011.

Marilyn A Walker, Pranav Anand, Robert Abbott, and Ricky Grant. Stance classification using dialogic properties of persuasion. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 592–596. Association for Computational Linguistics, 2012a.

Marilyn A Walker, Jean E Fox Tree, Pranav Anand, Rob Abbott, and Joseph King. A corpus for research on deliberation and debate. In *LREC*, pages 812–817, 2012b.

Wan Wei, Xiao Zhang, Xuqin Liu, Wei Chen, and Tengjiao Wang. pkudblab at semeval-2016 task 6: A specific convolutional neural network system for effective stance detection. In *SemEval@ NAACL-HLT*, pages 384–388, 2016.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2):165–210, 2005.

T Wilson, J Wiebe, and P Hoffman. Recognizing contextual polarity in phrase level sentiment analysis. *Acl*, 7(5):12–21, 2005a. ISSN 0891-2017.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*, pages 347–354. Association for Computational Linguistics, 2005b.

Kam-Fai Wong, Mingli Wu, and Wenjie Li. Extractive summarization using supervised and semi-supervised learning. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 985–992. Association for Computational Linguistics, 2008.

Fu Xianghua, Liu Guo, Guo Yanyan, and Wang Zhiqiang. Multi-aspect sentiment analysis for chinese online social reviews based on topic modeling and hownet lexicon. *Knowledge-Based Systems*, 37:186–195, 2013.

Wen-tau Yih, Joshua T Goodman, Lucretia H Vanderwende, and Hisami Suzuki. Document summarization by maximizing informative content words, April 20 2010. US Patent 7,702,680.

Hong Yu and Vasileios Hatzivassiloglou. Towards answering opinion questions: separating facts from opinions and identifying the polarity of opinion sentences. *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 129–136, 2003. ISSN 02779536. URL http://portal.acm.org/citation.cfm?doid=1119355.1119372.

Guido Zarrella and Amy Marsh. Mitre at semeval-2016 task 6: Transfer learning for stance detection. *arXiv preprint arXiv:1606.03784*, 2016.

Liang Zhou and Eduard Hovy. A web-trained extraction summarization system. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 205–211. Association for Computational Linguistics, 2003.