NTNU
Norwegian University of
Science and Technology

Interactive Exploration of Consensus in Climate Science

Petter Fagerlund Asla
Henrik Bøhler

Master of Science in Computer Science
Submission date: June 2016
Supervisor: Rune Sætre, IDI
Co-supervisor: Erwin Marsi, IDI

Norwegian University of Science and Technology
Department of Computer and Information Science

Petter F. Asla & Henrik Bøhler

Interactive Exploration of Consensus in Climate Science

Computer Science, Master's Thesis (TDT 4900), Spring 2016
Supervisors: Erwin Marsi & Rune Sætre

Artificial Intelligence Group
Department of Computer and Information Science
Faculty of Information Technology, Mathematics and Electrical Engineering

Abstract

Global warming has been a controversial topic of discussion in environmental studies in recent years. This thesis describes an approach to automatically classify stance on anthropogenic (human-induced) global warming in climate science with the aim to investigate the development of the consensus after 2011. In addition, we explore how stance is related to external factors.

The thesis is based on the work of Cook et al. (2013). They manually examined and labeled 11,944 abstracts related to climate change by their stance on anthropogenic global warming (*favor*, *against*, *none*). Of those taking a position, Cook et al. reported an observed consensus of 97 % endorsing the fact. We have explored approaches to automate Cook et al.'s work by classifying stance using machine learning.

Our system is divided into three major components: Search & Information Retrieval, Stance Detection and Visualization. The Search & Information Retrieval component make up the foundation of the system. It provides the classification component with the raw data obtained from literature databases such as Web of Science. It shows solid results for collection of meta-data, successfully retrieving data for 95.70 % of the publications. Evaluation of the strategy for obtaining recent climate literature indicates that approximately 18 % of records can be proved relevant. However, analysis of the data suggests that most of the collected literature is relevant.

Using the data collected by Cook et al. and The Consensus Project, we conducted a large number of experiments to determine the best stance detection model. The final system combines a Logistic Regression classifier trained on GloVe word embeddings and an optimized SVM in a voting scheme, representing the best of both classifiers. The approach achieves a substantial improvement over the baseline, achieving a macro F-score of 60.67 % on the test set. The predictions of new data, along with collected meta-data, serves as input to the visualization component. The resulting visual representation suggests no major change in the consensus. The number of publications per country plotted on a geographical map, inferred by meta-data regarding author affiliations, suggests a distinction between developed and developing countries.

Keywords

text categorization, stance detection, classification, visual analytics, opinion mining, visualization, scientific consensus, anthropogenic global warming, climate change

Sammendrag

Klimaforskning har i nyere tid hatt fokus på global oppvarming. Temaet er kontroversielt og er svært dagsaktuelt. Denne avhandlingen beskriver en løsning for automatisk klassifisering av dokumenter basert på deres holdning til klodens klimaendringer. Dokumentene kategoriseres basert på om de er *for*, *imot*, eller *nøytrale* til utsagnet om at global oppvarming er menneskeskapt. Målet med kategoriseringen er å muliggjøre utforskning av konsensus uttrykt i klimaforskning etter 2011. I tillegg undersøker vi hvordan oppfatning er påvirket av eksterne faktorer.

Avhandlingen baserer seg på arbeidet til Cook et al. (2013). De samlet og undersøkte $11,944$ sammendrag fra publikasjoner relatert til klimaendring. Sammendragene ble kategorisert basert på deres holdning til utsagnet om at global oppvarming er menneskeskapt. Basert på innhentingen rapporterte Cook et al. en enighet om menneskeskapt global oppvarming på 97 % blant de publikasjonene som uttrykket en mening (for eller imot). Vi har undersøkt muligheten til å automatisere arbeidet Cook et al. har utført ved hjelp av maskinlæring.

Vårt system består av tre komponenter: Informasjonsinnhenting, Klassifisering og Visualisering. Informasjonsinnhentingen danner basen til systemet. Den supplerer systemet med data hentet fra litteraturdatabaser som Web of Science. Komponenten viser gode resultater for henting av meta-data til publikasjonene, med hele 95.70 %. Ved å evaluere strategien for henting av klimaforskning publisert etter 2011, fastslår vi at sirka 18 % kan bevises å være relevant. En overfladisk analyse av dataene antyder derimot at en langt større andel trolig er relevant.

Data innhentet av Cook et al. og The Consensus Project ble brukt da vi gjennomførte et større antall eksperimenter for å avdekke den beste klassifiseringstrategien. Vårt endelige system kombinerer to tilnærminger til problemet i en avstemningsarkitektur for å oppnå det beste resultatet. Endelige prediksjoner er basert på Logistisk Regresjon trent på GloVe kjennetegn og en optimalisert SVM. Systemet oppnår en test macro F-score på 60.67 %, som er en betydelig forbedring over en standard implementasjon.

Systemets prediksjoner, sammen med innhentet meta-data, danner grunnlaget for visualiseringskomponenten. De produserte grafiske representasjonene antyder ingen betydelig endring i konsensus for menneskeskapt global oppvarming. Ved å plotte forfatter tilhørighet på et geografisk kart, fastslått i meta-data, undersøker vi sammenheng mellom tilhørighet og oppfatning av global oppvarming. Våre funn antyder en forskjell mellom industri- og utviklingsland.

Preface

This project initially started at the Department of Computer and Information Science (IDI) at the Norwegian University of Science and Technology (NTNU) during the fall of 2015. The work carried out in this period is considered as preliminary work and allowed us to explore ideas and construct a research foundation. This master thesis was conducted during spring of 2016 and continues where the preliminary work ended.

The subject was provided by one of our supervisors, Erwin Marsi. The purpose of the project has been to find out if we could, by using state-of-the-art techniques within stance detection, classify documents according to their stance on anthropogenic (human-induced) global warming. If successful, it will enable us to observe if there has been a noticeable change in the general position on anthropogenic global warming in recent years. The topic at hand is human-induced global warming, and the stances we wanted to detect were *in favor*, *against* and *neutral (no stance)*.

Our field of study is Artificial Intelligence and we have worked specifically on machine learning techniques for the past couple of years. Our motivation has been to see if we could apply what we have learned so far to a real problem.

The project consisted of three parts: information retrieval, document classification, and data visualization. We present our findings on the information retrieval and cover how we extracted information from papers using web crawling methods along with how we collected meta-data. We will give a summary of our work and research on state-of-the-art systems within stance detection and how to use visualization to perform exploration of data. We will aslo describe the system we implemented to detect stance in climate scientific literature and how we processed the results for interactive presentation.

During the preliminary work an interesting task, organized by SemEval, was presented to us. Based on our supervisors' suggestion we decided to participate. The goal of the task was to detect stance on different subjects in tweets. One of the topics was *Climate Change is a Real Concern*, which is highly relevant regarding our thesis. Although the task utilizes quite different data, the work done on the shared task is considered an important part of the development of this master thesis. In addition to providing us with more task specific knowledge, the participation gave us the opportunity to experience several aspects of scientific work such as submitting a paper, reviewing and presenting our findings.

Petter F. Asla & Henrik Bøhler
Trondheim, June 6, 2016

Acknowledgements

We would like to thank our supervisors: Erwin Marsi[1] and Rune Sætre[2] for supervising us throughout the course of this thesis. Erwin Marsi has contributed with knowledge across multiple fields, offered guidance to solve problems and imperative, constructive feedback. Rune Sætre made it possible for us to attend the SemEval conference in San Diego, 2016, where we will present our proposed solution to the shared task. In light of this, we want to thank IDI and NTNU for allowing us to travel and for covering traveling expenses to the conference.

At last, we want to thank our families for their support. It has been invaluable.

[1]`https://www.ntnu.edu/employees/emarsi`
[2]`https://www.ntnu.edu/employees/satre`

Contents

List of Figures

List of Tables

Glossary

ADC	Automatic Document Classification
AI	Artificial Intelligence
ANN	Artificial Neural Networks
API	Application Programming Interface
CNN	Convolutional Neural Network
CSS	Cascading Style Sheets
FNN	Feedforward Neural Network
HTML	Hyper Text Markup Language
IE	Information Extraction
ISO	International Organization for Standardization
NLP	Natural Language Processing
OFS	Off-the-Shelf
POS	Part-of-Speech
PDF	Portable Document Format
RNN	Recurrent Neural Network
SGML	Standard General Markup Language
SOAP	Simple Object Access Protocol
SVM	Support Vector Machine
TCP	The Consensus Project
TF-IDF	Term Frequency-Inverse Document Frequency
UI	User Interface
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
WSDL	Web Service Description Language
WWW	World Wide Web
XML	Extensible Markup Language
XPath	Extensible Markup Language Path Language

Chapter 1

Introduction

With a world generating more data than ever, new techniques are required to interpret the hidden information it contains. Because of the extreme quantities produced, we can no longer keep track and manage the data without the help of computer systems. The field of Artificial Intelligence (AI) has proven to be useful operating on large data sets (Lohr, 2012), assisting us in handling the data and uncovering facts. Several tasks can be performed employing AI. We have focused on document classification, which is a task falling under Natural Language Processing (NLP). Arguably, machine learning is a useful tool achieving this. A large variety of document classification tasks exists, such as Information Extraction (IE) and Sentiment Analysis.

Up until the mid-90s, IE focused on extracting factual information from texts. In the years following, IE started considering opinionated documents. Wiebe and Bruce (2001) worked on separating paragraphs into blocks that would contain sentences with subjective views of the same author. Hatzivassiloglou and McKeown (1997) showed that it was possible to predict and classify the positive and negative semantic orientation of adjectives in texts by constructing a machine learning approach based on a large corpus. Subasic and Huettner (2001) studied affect analysis on free texts for decision making on news stories and movie reviews. They show good correlation with a human judgment of affect content. Further tasks related to detecting opinions have been researched into the 21st century. Examples of such are opinion mining and sentiment analysis.

Sentiment analysis is a well-known task within the field of NLP. Sentiment analysis aims to identify the sentiment or attitude of the author in documents. The attitude of the author may be a judgment or evaluation expressed in texts, like a movie review, the emotional state of a sentence, or an intended emotional communication to the reader. Hu and Liu (2004) used techniques to discover features in customer reviews related to a product and decide whether the features were positive or negative. Sentiment analysis has developed into a popular tool for analyzing data collected from social media. Two natural sources to practise sentiment analysis is the microblogging platform Twitter (Neri et al., 2012; Mohammad et al., 2016) and online debate forums (Shi et al., 2009) to study people's attitude toward certain topics. Sentiment analysis can be accomplished by classifying phrases, sentences, small parts of a document, or whole documents. An example of data to analyze is the following sentence:

> "I love the summer in Oslo, but I hate the winter."

The sentence above should be divided into two segments. "I love the summer in Oslo" would be classified as positive, while the last phrase: "but I hate the winter" would be classified as negative.

In this project we wanted to detect the stance of papers concerning the specific question "Is global warming human-induced?". Stance detection can be described as polarity analysis with an additional dimension. Instead of inspecting positive or negative features in a text, stance detection aims to find the document's position held toward an object, idea, or cause. The topic in question may not be explicitly mentioned in the text. Earlier research in detecting stance has focused on ideological debates that discuss matters like abortion, gay marriage, and atheism. Somasundaran and Wiebe (2010) investigated whether arguing expressions of opinions and sentiment are useful for stance classification. Somasundaran and Wiebe used a technique to create an arguing lexicon with characteristics of opinion-target pairs together with sentiment. Their approach proved to be better than baseline systems based on bag-of-words models. Another paper that researched online debates analyzed the dialogue context of a post to detect the correct stance (Walker et al., 2012). Their approach also proved to be better than the bag-of-words baseline system. Mohammad et al. (2016) implemented a system to identify position held in Twitter texts with a linear kernel SVM classifier and features such as word and character n-gram features, sentiment features from lexicons and word-embeddings.

1.1 Motivation

Some important issues in society, like climate change, appear to be controversial in both science and media. In particular, many lay persons think that the scientific community is divided on the question whether or not climate change is due to human activities. However, this is in fact false. Cook et al. (2013) showed, by manually classifying a large and representative sample of the scientific literature on climate change, that the majority of literature actually supports the stance that global warming is caused by humans.

Cook et al. (2013) collected peer-reviewed publications and manually labeled them. In total $11,942$ papers were gathered in order to analyze the consensus on human-induced climate change. We found Cook et al.'s work inspiring and their result exemplifies discoveries that can be achieved when analyzing vast amounts of data. Their approach on resolving an apparently controversial issue such as climate change is interesting. However, time and money is an indisputable concern. It took Cook and his coworkers more than a year to complete the process, making it both time consuming and costly. As their process stopped in 2011, we want to develop methods for automatic stance detection in text to explore the development of consensus after their work ended. An automated solution to their approach would be economical, competitive and have the ability to give online representations of findings.

Automatic stance detection has the potential to make a significant contribution to global socio-economic and political debates on issues such as climate change and refugees by quantifying the public or scientific opinion on these matters. Such examples makes stance detection appealing for further exploring. We also believe that a system to classify documents in this manner can uncover engaging facts such as a connection between science institutions and their position held toward climate change.

1.2 Goals and Research Questions

Goal 1: *Automatically classify publications from TCP according to their stance held towards human-induced climate change.*

Our primary goal in this project is to determine if the position of a paper is *in favor*, *against*, or *neutral (no stance)* regarding the question "Is global warming human-induced?".

Goal 2: *Extrapolate TCP data to publications after 2011.*

As stated in the section 1.1, we want to examine the possibility of automating the work performed by Cook et al. (2013). We want to utilize the system to extend their work and evaluate the development of the consensus after 2011 through visualization.

Goal 3: *Discover how stance is related to external factors.*

Besides collecting and classifying paper abstracts, we hope to expose connections in the research between stance and external factors such as the nationality of the author(s). The results should be visually presented, so correlations in the data can be explored.

In order to guide our research goals, the following more specific research questions were formulated.

Research question 1 *What is the state-of-the-art concerning stance detection?*

This requires a literature study in order to establish the state-of-the-art in automatic stance detection.

Research question 2 *How can relevant scientific articles be retrieved?*

To observe a change in the consensus we need data from recent years (after 2011) opposed to what was collected in TCP. The information needed should be extracted from the web.

Research question 3 *What type of machine learning model is well suited for training a (stance) classifier?*

There is a wide range of machine learning algorithms appropriate for various tasks, and we need to select the learners which are best for our purpose.

Research question 4 *Is it possible to detect relations between a paper's stance and external factors?*

This question investigates connections between a paper and its stance regarding external factors such as the nationality of the author(s).

1.3 Task Description

This project is based on one fundamental question, *"Is global warming human-induced?"*. The question is in part answered by Cook et al. (2013). They started research on this topic to determine the consensus in the scientific community. However, their work is static and ended in 2011. As we are interested in the current situation in scientific literature, we formulated a set of goals and research questions (presented in section 1.2) that will guide us in our attempt to automate their work. These goals and research questions give rise to the following task description:

A thorough study has to be conducted to map state-of-the-art systems and obtain background knowledge in stance related disciplines. The insight should provide the knowledge required to design a system that automatically performs text classification based on a text's (abstract) position on human-induced global warming in climate science. The system should detect the stances *in favor*, *against* and *no stance (neutral)* .

Creation of the system can be divided into three sub-tasks. The first task consists in retrieving essential scientific data including additional attributes (metadata). Multiple literature databases are accessible via the web and should be investigated. This task along with the background study is fundamental to the success of later stages. The second sub-task encompasses the utilization of information obtained by Cook et al. to engineer a stance classification system. Evaluating multiple machine learning techniques and classifiers, using this data, will determine the topmost approach to perform stance classification of climate literature. Further, the data obtained in task one along with predictions from task two will form the foundation for exploration of the consensus on anthropogenic global warming through visualization. The final sub-task rely on the data just described to visually represent the development of the consensus after 2011 and expose relations between stance and attributes such as author affiliations.

1.4 Research Methodology

The research methods are outlined in this section. These can be divided into the three types:

Literature Review

A literature review has been conducted on the topics of information retrieval, stance detection, sentiment analysis, and visualization. This resulted in an overview of state-of-the-art techniques applied mainly to stance detection, but also to general application of sentiment analysis and gave us the theoretical background required for this project.

These are the literature search engines we have used to search and access relevant information: Google Scholar[1], IEEE Xplore[2] , Scopus[3] and Web of Science[4]. Some of the keywords used to search for literature were:

- Stance detection
- Stance classification
- Automatic stance classification systems
- Document classification
- Opinion mining
- Sentiment analysis
- Natural language processing

The resulting papers and articles were examined for information relevant to this project. By writing short summaries from each paper we were able to gain overview in the field of stance detection. This report contains the information we found most important.

Data-driven AI

Empirical data is a term used when knowledge is acquired through experimentation and observation. One computational discipline that has the capability to perform empirical modeling is machine learning. The field that encompasses such methods is called data-driven modeling. Data-driven modeling analyzes data in a system and finds connections between input and output. The model find relations without explicit knowledge of the system. To achieve this, machine learning algorithms use gold standard data which represents the system's intended behavior. Gold standard data is manually annotated by humans. It is divided into development data, validation data and test data. The development data is used for tuning and selecting models. During the development phase verification is needed to establish progress which is achieved by using the validation data. The test data is held out until the very end of the project and is used to measure performance of the best model. The evaluation is an estimate of the system's future performance on any unlabeled data. The annotations from TCP was used as gold standard for development, validation and testing. Our model builds on an experiment-and-test procedure by coding computer programs using machine learning techniques. Experiment-and-test procedure means to rewrite the code as long as it improves our model. The model can be tested by using the gold standard data or by adding new independent data.

[1]`https://scholar.google.no/`, as of 2016-06-06
[2]`http://ieeexplore.ieee.org/Xplore/home.jsp`, as of 2016-06-06
[3]`http://www.scopus.com/`, as of 2016-06-06
[4]`http://wokinfo.com/`, as of 2016-06-06

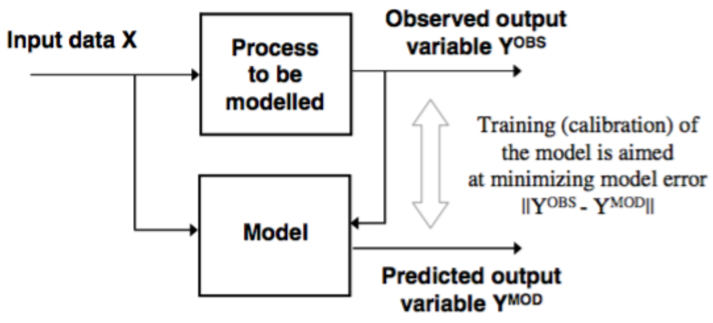

Input data X
Process to be modelled
Observed output variable Y^{OBS}
Training (calibration) of the model is aimed at minimizing model error $||Y^{OBS} - Y^{MOD}||$
Model
Predicted output variable Y^{MOD}

Figure 1.1: Model for learning proposed by Solomatine and Ostfeld (2008).

Applying new information to the system determine how the model predict unseen data. Figure 1.1 illustrate the data-driven learning model that will be used in our system.

Interactive Visualization

Interactive visualization was employed to reveal commonalities in data obtained in this project. Presentation of dynamical charts enabled us to perform visual exploration and observations. Interactive views with capabilities such as zooming expose details not easily grasped by computers analyzing the data. Two components must be dealt with to suitably renew the charts: the type representation and how the elements adjust according to the user's input. After a visualization has taken place users can analyze data utilizing their visual system.

1.5 Thesis Structure

Chapter two provides fundamental background theory for the project, the experiments conducted and the results. Chapter three includes architecture, models and implementation of the system. Chapter four, five and six describes the experiments and evaluates the outcome of these. The last chapter (chapter seven) summarizes results, provides a general discussion and suggestions for future work.

Chapter 2

Background

This chapter provides the fundamental theory that is required to understand the foundation of this thesis. It explores the background theory needed to understand the existing systems along with tools needed to reach the goal. The chapter includes subjects such as earlier work on information retrieval, state-of-the-art systems within stance detection, and important concepts in visualization and machine learning.

2.1 The Consensus Project

This thesis is based on the work of Cook et al. (2013) and The Consensus Project[1] (TCP). TCP was a citizen-science project that started in 2011 and lasted for over a year. The TCP team consisted of 24 volunteers that contributed to the rating of 11,944 paper abstracts. Research literature collected as part of the TCP project was on the subject of climate change. The terms used to find the scientific papers were ”global climate change” and ”global warming” and were restricted to the type ”article”.

The collected abstracts were categorized into six different research classes (based on research types). These are presented in Table 2.1. In addition to the research class, each abstract was given an endorsement level. The endorsement level is a score of the author’s stance towards human-induced global warming. A description of the different levels are found in Table 2.2. The labeling and rating were performed by the TCP team in two rounds, where the second one was performed after a consultation stage. In addition to the ratings by the team, all authors of literature that was included in the study were asked to provide their

[1]`http://theconsensusproject.com/` as of 2016-06-06

Category	Description
1 - Impacts	Effects and impacts of climate change on the environment, ecosystems or humanity.
2 - Methods	Focus on measurements and modeling methods, or basic climate science not included in the other categories.
3 - Mitigation	Research into lowering CO2 emissions or atmospheric CO2 levels.
4 - Not climate related	Social science, education, research about people's views on climate.
5 - Opinion	Not peer-reviewed articles.
6 - Paleoclimate	Examining climate during pre-industrial times.

Table 2.1: Categories used in TCP.

own self-endorsement. This opens the opportunity to see if any misinterpretations had occurred during the labeling process. Most of the authors responded to this request, but not all. We have used the ratings performed by the TCP in this project.

To provide a better insight to how the categorization and labeling of endorsement level for the data was performed, we will provide a short summary of the process Cook et al. (2013) used. After the collection of literature the abstracts were distributed (randomly) to the team of raters. The raters had only access to the title and the abstract and were given a specific criteria when rating the abstracts (shown in Table 2.1 and Table 2.2). The abstracts were categorized two times. Each time by an independent rater. This means that each abstract did receive two independent ratings.

After the first round, 27 % of the category ratings and 33 % of endorsement level ratings disagreed. They proceeded by allowing the raters to compare, justify, and update their ratings. After this step, only 11 % of category ratings and 16 % of endorsement ratings disagreed. The remaining disagreements were resolved by introducing a third party rater. For 2, 142 of the abstracts, a rating performed by the author was retrieved (not used in this project). The comparison between the TCP team rating and self-rating can be reviewed in Table 2.3.

TCP wanted to raise awareness of the scientific consensus on climate change. Cook et al. (2013) refers to that the general public considered approximately

Endorsement level	**Description**
1 - Explicit endorsement with quantification	Explicitly states that humans are the primary cause of recent global warming.
2 - Explicit endorsement without quantification	Explicitly states humans are causing global warming or refers to anthropogenic global warming/climate change as a known fact.
3 - Implicit endorsement	Implies humans are causing global warming. E.g., research assumes greenhouse gas emissions cause warming without explicitly stating humans are the cause.
4 - No position	Does not address or mention the cause of global warming.
5 - Implicit rejection	Implies humans have had a minimal impact on global warming without saying so explicitly E.g., proposing a natural mechanism is the main cause of global warming.
6 - Explicit rejection without quantification	Explicitly minimizes or rejects that humans are causing global warming.
7 - Explicit rejection with quantification	Explicitly states that humans are causing less than half of global warming.

Table 2.2: Endorsement levels used in TCP.

Position	**TCP team rating**		**Self-rating**	
Endorse	761	(36.9 %)	1,342	(62.7 %)
No position	1,339	(62.5 %)	761	(35.5 %)
Reject	12	(0.6 %)	39	(1.8 %)

Table 2.3: Comparison of rating from Cook et al. (2013).

50 % of climate scientists to believe that the climate change was anthropogenic (human-induced). However, results from the experiment shows that the reality is 97 %.

TCP measured the level of consensus in published, peer-reviewed climate research on global warming by determining whether the author believed global warming is a result of human interference. TCP analyzed 21 years worth of peer-reviewed papers on ”global warming” or ”global climate change”. Among the 11, 944 papers, the project identified 4, 014 abstracts authored by a total of 10, 188 scientists that stated a position on man-made global warming. Amidst the 4, 014 abstracts, 97.1 % endorsed anthropogenic global warming. Other papers such as Anderegg et al. (2010) came up with results backing up TCP’s findings.

We have taken advantage of the data collection and labeling that Cook et al. (2013) performed. See Appendix C: Data from TCP for a detailed description of the data.

2.2 Natural Language Processing

NLP explores how computers can interpret natural language in texts and speech, as well as manipulate content (Chowdhury, 2003). Researchers in this field aim to make computers able to fully understand human language. Some NLP applications are speech recognition, text processing and machine translation. People working in the field of natural language range from linguistics and computer scientist to psychologists and mathematicians.

At the core of NLP (besides understanding the natural language) there are three major problems: first is the thought process, second is the representation and linguistic input and last is the real world knowledge. An example: NLP applications may begin at the word level to determine, for instance, meaning. From here it can move on to sentence level to determine semantics and meaning of the composed words and then later the overall document context. A single word or sentence may propose special meaning given a certain context, which relates to other words or sentences. An example of this dual meaning is presented below. Both the sentences contain the word *bow*, but the have different meaning.

The ropes are at the upper bow deck.

Should we take a bow?

Liddy (1998) proposes seven interdependent levels people use to extract meaning from text and spoken language:

- **Phonological:** Deals with pronunciation

- **Morphological:** Smallest part of words that carry meaning
- **Lexical:** Lexical meaning of words and Part-of-Speech (POS) analyses
- **Syntactic:** Grammar and structure
- **Semantic:** Possible meaning of words and sentences
- **Discourse:** Interpret structure of larger texts
- **Pragmatic:** Knowledge from outside the document

When analyzing and applying NLP to systems, some or all of the levels may be involved. This depend on to what extent NLP is needed in the system to reach the goal.

Tokenization

Application of natural language often begins with text processing. This is very typical for a system that takes a sentence or a collection of sentences as input. The processing module decomposes the text into individual words or sentences. It identifies for instance words by looking for a white space or a tab separator. These words are interpreted by the processor as tokens. Therefore the first step of text processing is called tokenization and the processing module doing the work is called the tokenizer (Carus, 1999).

Stop Words

When applying text processing on a collection of documents, the returned vector of words may be massive. Often, it is desirable to filter out words that (often) do not carry any significance in further processing, to reduce the word vector. For instance, words such as *a, and, are,* or *the* in English are often removed. The list of stop words are usually referred to the most common words in a language, but there is no single unique list acquired for natural language processing. Any group of words can be used as a stop word list, it depends on the context (Rajaraman et al., 2012).

Stemming and Lemmatization

According to Manning et al. (2008), the goal of both stemming and lemmatization is to reduce the inflected form to a common dictionary base form. Stemming usually refers to the crude form of chopping of ends of words with the hope of retrieving the base word. Lemmatization is more context dependant in the way that the process use vocabulary and morphological analysis of words to find the base. For instance, the verb to *play* can be inflected in several ways such as *plays, played,* and *playing.* Both stemming and lemmatization will find the base *play.* But for the adjective *worse* that has *bad* as its lemma, stemming miss the link while lemmatization finds the base form.

Term Frequency x Inverse Document Frequency

Term Frequency times Inverse Document Drequency (TF-IDF) is a statistic representation of words in a document intended to reflect the importance of a word. This is often applied when the documents are part of a larger collection. The TF-IDF scores are higher for the terms that best characterize the topic of a document (Rajaraman et al., 2012). For instance, let us suppose we have a collection of N documents. F_{ij} is the frequency of term (word) i in document j. The term frequency is defined in Equation 2.1.

$$TF_{ij} = \frac{F_{ij}}{max_k F_{kj}} \tag{2.1}$$

The number of occurrences of word i in document j is normalized by dividing F_{ij} by the maximum number of occurrence of any term in the same document, as seen in equation Equation 2.1. In other words, the most frequent word in document j receives a term frequency of 1 and all other terms receive a value between 0 and 1. Let us say word i appears in n_i of the N documents. The inverse document frequency is defined in Equation 2.2.

$$IDF_i = log_2(\frac{N}{n_i}) \tag{2.2}$$

The TF-IDF is defined by multiplying term frequency (Equation 2.1) with the inverse document frequency (Equation 2.2). This is shown in Equation 2.3.

$$TF - IDF = TF \times IDF \tag{2.3}$$

If we have 2^{20} documents in our collection and the word w appears in 2^{10} of these documents, then the IDF for word w is 10. For the particular document j, we know that the word w occurs 15 times along with being the most frequent word in document j. This gives word w the term frequency $TF_{wj} = \frac{15}{15} = 1$, while the IDF_{wj} for word w is 10 resulting in a $TF - IDF$ score of 10 (Equation 2.4).

$$TF_{wj} \times IDF_{wj} = \frac{15}{15} \times log_2(\frac{2^{20}}{2^{10}}) = 1 \times 10 = 10 \tag{2.4}$$

In another document k, the word w appear *once* while the maximum frequency of any other word is 15. The term frequency is therefore $TF_{wk} = \frac{1}{15}$ resulting in a TF-IDF score for word w in document k equal to $\frac{1}{15} \times 10 = \frac{2}{3}$

2.3 Sentiment Analysis (Opinion Mining)

Liu (2012) refers to sentiment analysis (also called opinion mining) as the discipline that analyzes people's opinions, sentiments, evaluations, attitudes, and emotions from written language. According to him it is one of the most active research areas in NLP and is also widely studied in data mining, web mining, and text mining. There are many slightly different tasks with different names that all fall under sentiment analysis (Liu, 2012). Examples of such tasks are subjectivity analysis, affect analysis, and emotion analysis. NLP techniques like sentiment analysis are capable of identifying and extracting important information from text. Using sentiment analysis to decide the attitude of the author can assist us regarding the goal of automatically classify text documents.

A common application of sentiment analysis is to detect polarity in a document, whether the document is a sentence or a larger text such as an essay. The polarity of a text is classified into positive, negative or neutral. One strategy of finding the polarity of a larger text is to summarize the predicted class of each sentence and proceed by picking the majority of the predicted class as polarity of the document. For instance, consider the following review of a camera (numbers in brackets are only included for reference later on).

> [1] The camera is great for action shots. [2] It is light and easy to use. [3] However, I should mention that the camera takes poor images when not used in daylight. [4] Overall I'm happy with purchase.

In the review above, sentences [1], [2], and [4] express a positive sentiment towards the camera while sentence [3] express negativity. In this case, the strategy would predict that the document was positive towards the camera. Hu and Liu (2004) focused on the application of classifying polarity in reviews using techniques such as part-of-speech (POS) tagging. POS tagging gives information about the class of each word, whether it is a verb, noun or adjective. Their results shows that these techniques are highly effective.

An early study, conducted by Pang et al. (2002) on movie reviews, shows that creating a lexicon of keywords could help finding sentiment in documents. By enquiring two humans to pick out keywords that would indicate sentiment and apply machine learning techniques together with the keywords, Pang et al. achieved over 80 % accuracy in sentiment classification. Detecting sentiment can be both context and domain dependent. The following example used in Pang and Lee (2008) is domain dependant:

> ”Go read the book”

In a book review, this sentence is likely to indicate positive sentiment. But for a movie review, this indicate a negative sentiment. Niu et al. (2005) used a more fine-grained classification model for polarity by introducing classes that represents degrees of positive and negative sentiment.

An advanced version of polarity detection is classification of emotional states such as happy, angry and sad (Liu, 2012). Another task in sentiment analysis is to classify whether a sentence is subjective or objective (Liu, 2012).

When working with polarity classification, the data sets used for modelling are usually opinionated, that is the data (usually text) contains opinions. For several applications, it is required to detect opinionated text before commencing with sentiment detection. Early work of Hatzivassiloglou and Wiebe (2000) used adjective orientation in order to detect subjectivity. The goal was to classify whether the text was subjective based on the adjectives that appeared in a text. Once subjectivity was found in the document, prediction of the sentiment could take place.

The subjectivity exercise can be more difficult than polarity classification for the reason that finding subjectivity in documents may impose understanding the context in specific sentences. The definition of subjectivity can be unclear and objectivity might have subjective words in the sentence. Wiebe et al. (2004) did a comprehensive study on learning subjective language. The features used in the paper were clues such as low-frequency words, collocations, adjectives, and verbs. Their goal was to learn subjective language from corpora. They showed that unique words are valuable clues to subjectivity. A corpus used in computational linguistics consists of a large and structured texts. POS tagging is a familiar process of making a corpus more useful.

A popular strategy when working with opinion mining tasks, is to take advantage of previously created lexical resources. For instance there is a lexicon consisting of words in relation to subjectivity, which is great for detecting subjectivity. Some known lexicons are: Miller (1995) WordNet[2], Esuli and Sebastiani (2006) SentiWordNet[3], Wilson et al. (2005) Subjectivity lexicon[4], and Somasundaran et al. (2007) Arguing lexicon[5].

There exists multiple sub-tasks of sentiment analysis, that have not been introduced here, such as comparison tasks like ”The picture quality of Camera X is as good as that of Camera Y” and ”The Intel chip is the fastest” (Liu, 2010).

[2]`https://wordnet.princeton.edu`, as of 2016-06-06
[3]`http://sentiwordnet.isti.cnr.it`, as of 2016-06-06
[4]`http://mpqa.cs.pitt.edu/lexicons/subj_lexicon/`, as of 2016-06-06
[5]`http://mpqa.cs.pitt.edu/lexicons/arg_lexicon/`, as of 2016-06-06

2.4 Stance Detection

There is no agreement on a definition of stance in linguistics. During our research, we found multiple definitions of the term *stance*. Some of them are found in Chindamo et al. (2012).

Stance is often presented with a distinction between epistemic and affective stance. Ochs (1996) stated that affective stance is related to the emotional feelings of the object in question. An example of affective stance is the happiness uttered in gestures or verbally toward a statement. Ochs (1996) expressed that an epistemic stance is related to the degree of certainty regarding the object of discussion. For instance, it can be the certainty shown while answering a question.

The expression of affect is connected with affective stance (Scherer et al., 1985). Important things like gestural features and vocal tone when analyzing the process of stance taking can be illustrated in the following example.

> ”I really love football!”

In Chindamo et al. (2012), they ask if the statement above would have the same effect (and meaning) if spoken with high pitch and a smile opposed to low pitch and serious face. An answer to the enquiry above can be characterized by Biber and Finegan (1989) when they describe that stance is ”marked by tone of voice, duration, loudness, and other paralinguistic features”.

Douglas Biber, a Professor of Applied Linguistics Northern Arizona University defined stance in Biber (2004) as the expression of one’s personal viewpoint concerning proposed information. Biber (2004) and Biber and Finegan (1989) provide examples of lexico-grammatical features connected to epistemic stance. The features consisted of adverbs, verbs, adjectives and nouns, described in Table 2.4. For the rest of this thesis, when terms such as stance, stance detection, or stance classification, are used we refer to the definition given by Biber (2004); stance is the expression of one’s personal viewpoint concerning the proposed information.

In computational linguistic stance detection or stance classification refers to the task of determining the position held by an author towards an object, idea or cause in a document. Stance classification can be applied to a phrase, sentence or larger text document. Like polarity classification, stance classification has three classes. The classes are: *in favor, against* and *neutral. Neutral* meaning no stance held by the author. Similar to subjectivity detection and sentiment detection explained above, a stance has to be detected. Once it is detected, prediction of the stance can be performed. An example of earlier work on stance classification is found in Somasundaran and Wiebe (2010). Their focus was to

POS	**Expressing certainty**	**Expressing doubt**
Adverb	Actually, certainly, obviously	Apparently, perhaps, possibly
Verb	Conclude, determine, know	Believe, doubt, think, seem
Adjectives	Certain, clear, obvious	Possible, probable, uncertain
Nouns	Conclusion, fact, observation	Assumption, claim, opinion

Table 2.4: Features used in epistemic stance described in Biber (2004) and Biber and Finegan (1989).

detect stance on ideological online debates on topics like abortion, gay marriage, and weapon rights. It was stated in the paper that arguing opinions are important when people defend their stand, and opinions themselves are not as informative as the combination of opinions and targets. The following example is taken from their paper.

> *Government* is a **disease** pretending to be its own cure. [side: against healthcare]

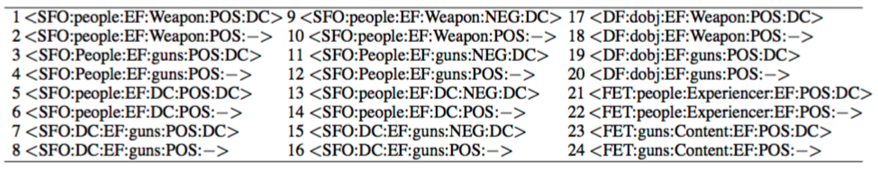
The writer is expressing negative sentiment regarding the government and healthcare. The opinion spans are highlighted in bold text and the target is highlighted in italic. The strategy used in Somasundaran and Wiebe was to create an arguing lexicon based on opinion-target pairs, where target is what the opinion was about. Their experiments implemented an SVM learner scoring an average accuracy of 63.93 %.

Another paper that conducted research on finding stance in ideological online debates was Hasan and Ng (2013). Their strategy was to induce a lexico-syntactic pattern, based on syntactic dependencies and semantic frames. The strategy aimed to understand the meaning in a sentence, generate a semantic generalization and improve classification in a novel way. The following two sentences are syntactically unequal, but semantically equivalent.

> Some people hate guns

> Some people do not like guns

Figure 2.1 display a sample pattern created from two sentences above, using the semantic frame strategy. Utilizing SVM, the method scored an average of 73.25



1 <SFO:people:EF:Weapon:POS:DC>	9 <SFO:people:EF:Weapon:NEG:DC>	17 <DF:dobj:EF:Weapon:POS:DC>
2 <SFO:people:EF:Weapon:POS:−>	10 <SFO:people:EF:Weapon:POS:−>	18 <DF:dobj:EF:Weapon:POS:−>
3 <SFO:People:EF:guns:POS:DC>	11 <SFO:People:EF:guns:NEG:DC>	19 <DF:dobj:EF:guns:POS:DC>
4 <SFO:People:EF:guns:POS:−>	12 <SFO:People:EF:guns:POS:−>	20 <DF:dobj:EF:guns:POS:−>
5 <SFO:people:EF:DC:POS:DC>	13 <SFO:people:EF:DC:NEG:DC>	21 <FET:people:Experiencer:EF:POS:DC>
6 <SFO:people:EF:DC:POS:−>	14 <SFO:people:EF:DC:POS:−>	22 <FET:people:Experiencer:EF:POS:−>
7 <SFO:DC:EF:guns:POS:DC>	15 <SFO:DC:EF:guns:NEG:DC>	23 <FET:guns:Content:EF:POS:DC>
8 <SFO:DC:EF:guns:POS:−>	16 <SFO:DC:EF:guns:POS:−>	24 <FET:guns:Content:EF:POS:−>

Figure 2.1: A sample of Hasan and Ng (2013)'s semantic frame strategy for two semantically equivalent sentences.

% in terms of accuracy. Hasan and Ng also explored methods like same-author information and similar-minded authors. They claimed it was likely that two posts written by the same author had the same stance.

Faulkner (2014) used annotated corpus consisting of student essays to detect stance. He presented a paper with an approach designed to capture stance, stance targets, and topical relationships between an expressed opinion and the student's essay. The paper differentiates between opinion-bearing language and stancetaking language based on the semantic targets. The distinction is illustrated in the following sentences.

> ”Snake Eyes” is the most **aggravating** kind of [movie]: the kind that shows so much potential and then becomes **unbelievably disappointing**. (opinion=positive)
>
> This **indicates** that [our prisons are higher institutions for criminals]. (stance=for)

While opinions take entities as targets (first sentence), stance take propositions as targets (last sentence). The targeted entity and proposition are bracketed, while the opinion-bearing language and stancetaking language is bold. Faulkner (2014) proposed a strategy toward the goal by replicating the arguing lexicon from Somasundaran and Wiebe (2010) and generate another lexicon consisting of: stance word - main verb - target patterns to create a generalized dependency sub-tree. The following sentence demonstrates how the second lexicon was created.

> I **can** only say that this statement is completely **true** $--$ > pattern: *can-say-true*

The sentence above illustrates a sentence in favor with the pattern can-say-true and the main verb: say. Figure 2.2 shows how they created the generalized dependency sub-tree. Their highest scoring version measured 82.0 % accuracy, implementing the SVM learner with RBF kernel.

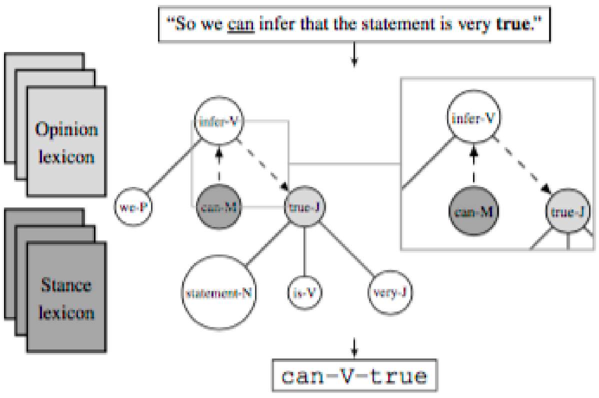


Figure 2.2: Faulkner (2014)'s generalized dependency sub-tree.

Recent papers have focused on online debates and Twitter feeds attempting to detect stance of each post (Mohammad et al., 2016). Mohammad et al. show that sentiment features expressed in tweets is beneficial toward sentiment detection, but not that effective toward stance detection. Their system takes a simpler approach than previously mentioned research and is based on detecting target in sentence (by "target" they mean which topic stance is expressed toward), sentiment expressed in the text, various n-grams of words and characters, hashtags and encodings such as the presence and absence of exclamation marks. Their final system, based on a linear SVM, scored an average of 70.3 % using the macro F metric. With the subject of this thesis in mind, their score on the topic *Climate Change is a Real Concern* was 42.3 %.

An important aspect to take into consideration when doing research on document classification is the written language that is practised. In Faulkner (2014), it is stated that language used in online debates often involves causal relations, using words like: *therefore, if..then, because.* Text written in similar applications tend to have an informal character. Words and sentences in informal text can often be sarcastic, ironic, use unknown abbreviations and in the case of online forums; questioning earlier posts. This type of language can make it difficult to infer and detect the correct stance as there is unknown or hidden information. Twitter is a an application where each post only has 140 characters which severely limits the amount per document. 140 characters equals one or two sentences. The length of a document will affect the decision if there exists a stance. With a compact document such as a tweet, there is not much room for discussion or reflections. A climate concerned tweet goes as follows.

> ”SO EXCITING! Meaningful climate change action is on the way! #abpoli #GHG”[6]

For humans, it is easy to deduce that this tweet is in favor of the fact that climate change is a concern, however this is not always the case.

> ”I'm in hell. I feel the fiery tips of insidious flames swiftly sauntering into my room. Or maybe I don't have AC”[7]

The tweet above may not be so easy to deduce as the previous tweet. The latter tweet was manually labeled with no stance according to the training set provided by SemEval 2016. It is difficult for computers to interpret meaning of the texts like the ones above, but then again computer can take advantage of clues like exclamation marks and capital letter together with statistical analysis.

In this thesis we review the language used by scientists. Scientists carrying out research and present their findings in papers under the topic of climate change are analyzed. The articles are written by scientists who have extensive knowledge in their research discipline and have worked for years to develop experience and results. Scientific papers are written for a targeted audience, which enables them to include particular terms and expressions that may be unfamiliar to the average person. In contrast to online debates and tweets, scientific papers are formal with proper grammar and few spelling errors. The authors often provide a discussion around their findings, but it is not usually (and should not be) emotional. Rather, the arguments are based on findings from experiments and research.

The data used in this thesis are the abstracts from scientific papers related to climate change. An abstract is a summary of a paper with an intention to convey the most important information in the article. Following are two sentences taken from the abstract of the paper on global warming by Fawcett and Barron (1991):

> ”The possibility of global warming resulting from the anthropogenic addition of carbon dioxide and other industrial gases into the atmosphere is a topic of much recent concern. Global climate models are used to make predictions for possible future climatic changes, but their ability to simulate climatic states other than the present day is not well constrained”

[6]`http://alt.qcri.org/semeval2016/task6/data/uploads/semeval2016-task6-trainingdata.txt`, as of 2016-06-06

[7]`http://alt.qcri.org/semeval2016/task6/data/uploads/semeval2016-task6-trainingdata.txt`, as of 2016-06-06

The language is formal and well structured. Detecting subjectivity or opinions in research papers may increase the chance of finding stance. The chance of finding stance may also increase if arguments are detected. To the best of our knowledge, there is no prior work on stance detection in scientific papers in general and no work using the TCP data set in particular.

2.5 Word Embeddings

Word embedding[8] is the task of learning vector space representations of words used in NLP (Pennington et al., 2014). The process maps words or phrases from the vocabulary to vectors of real numbers. This section presents two ways of achieving so. We will not present the tools in great detail, as this is beyond the scope of this thesis, and we refer the readers to the cited papers for a more detailed description.

2.5.1 GloVe

GloVe (Pennington et al., 2014) is an unsupervised learning algorithm for obtaining vector representations of words. It creates word vectors based on the distributional statistics of words, in particular how frequently words co-occur within a certain window in a large text corpus such as the Gigaword corpus from Parker et al. (2011). The resulting word vectors can be used to measure semantic similarity between word pairs, following the hypothesis that similar words tend to have similar distributions. The Euclidean or Cosine distance between two word vectors can thus be used as a measure of their semantic similarity. For the word *frog*, for example, we can find related words such as *frogs, toad, litoria, leptodactylidae, rana, lizard, eleutherodactylu.*

2.5.2 word2vec

word2vec (Mikolov et al., 2013a; Mikolov et al., 2013b) is a software[9] created by Mikolov et al. (2013b) and his colleagues (Goldberg and Levy, 2014). word2vec provides an efficient implementation of the continuous bag-of-words and skip-gram architectures, both used for computing vector representations of words. Like Glove, word2vec creates word vectors that can be used to measure semantic similarity between words.

[8]`https://en.wikipedia.org/wiki/Word_embedding`, as of 2016-06-06
[9]`https://code.google.com/p/word2vec/`, as of 2016-06-06

2.6 Machine Learning

According to Russell and Norvig (1995), machine learning is the sub-field of AI concerned with programs that learn from experience. Algorithms are created that enables computers to learn from a set of data and make predictions on similar data sets. Rather than programming static algorithms and follow strict rules, a machine learning algorithm is able to detect patterns data. For the learning step to be successful, the given data should cover all cases of interest. This process is achieved by constructing a model based on a set of data called training data.

Russell and Norvig (1995) partition machine learning into three main categories: supervised learning, semi-supervised learning, and unsupervised learning. Supervised learning is the focus of this dissertation. In supervised learning, the desired output is provided. The correct output is provided for each input, which enables machine learning algorithms to train and build models. The goal of the model is to correctly map a given input to the desired output.

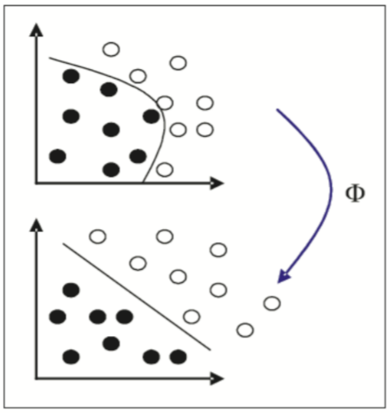
One of the main applications of machine learning is classification problems. A classifier is defined by Domingos (2012) as a system that inputs a vector of discrete or continuous feature values and outputs a single discrete value, the class. Based on a set of inputs, the goal is to predict the correct class. This is called automatic document classification (ADC).

ADC is generally defined as content-based assignment of one or more predefined categories to documents (Goller et al., 2000). Music, images, and texts are examples of documents which can be classified. Automatic classification is concerned with the systems that can make comparison between the terms used, according to Jaiswal (1999).

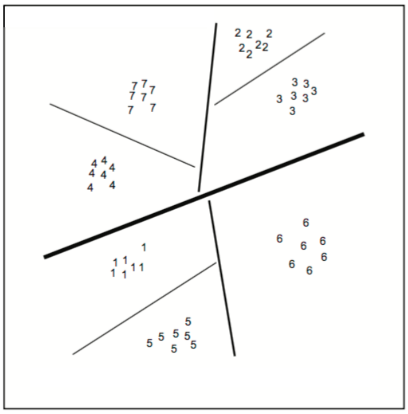
AlShaari (2014) stated that document classification can be split into two types. The first type is request-oriented classification. This type of classification is a task in which requests from users are influencing the document classification. The second type, which is relevant for this thesis, is the content-based classification. The content-based approach is based on weighting particular occurrences in a document and use the weight information when classifying.

2.6.1 Classifiers

When performing supervised classification there exists a lot of tools to take advantage of when solving problems. Classifiers are divided in two types: discriminative and generative. While generative classifiers learn from the joint probability of the input and output, discriminative classifiers learn a direct map from input to the class labels (Jordan, 2002).



(a) Binary SVM



(b) Multi-class SVM

Figure 2.3: Madzarov and Gjorgjevikj (2009)'s illustration of the difference between a binary and a multi-class SVM.

Support Vector Machine

One popular tool for classification is the support vector machine (SVM) (Madzarov and Gjorgjevikj, 2009; Steidl, 2015). SVM is a learning model that uses algorithms to analyze data and recognize patterns. The SVM algorithm takes an input set for training and learning. A model is created and becomes capable of placing new examples into separate classes. The difference between binary and multi-class SVM can be seen in Figure 2.3.

SVM are distinguished by two types: linear classification and non-linear classification. For a binary classification where the training data is linearly separable, it is possible to draw a line that isolate the two classes. When drawing the line, the SVM strive to find the maximum distance to the class boundary for each data point in the class. The maximum distance to the class boundary is the region that separates classes the most and is called the margin. Campbell (2002) illustrates the margin in Figure 2.4

When the data is not linearly separable, the problem becomes a non-linear classification problem. A solution to the problem is to apply a kernel substitution. The kernel's task is to map each data point into a higher dimensional space. The mapping from a non-linear problem to a linear problem with help from the kernel can be seen in Figure 2.3a. The application of a kernel works because data that is not linearly separable in low dimensions becomes linearly separable when mapped into higher dimensions (Campbell, 2002).

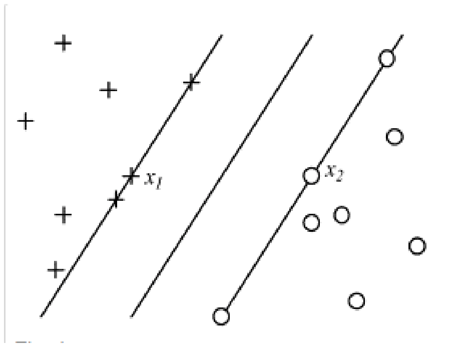


Figure 2.4: Campbell (2002) states that the margin is the perpendicular distance between the separating hyperplane and the hyperplane through the closest points.

Naive Bayes

The Naive Bayes approach to classification, for instance text classification, is by using a probabilistic model that make strong assumptions about how the data is generated. Then the model use a collection of labeled training samples to estimate parameters of the generative model. The classification of unseen data is performed using Bayes' rule (2.5) by selecting the class with highest probability (2.8) (McCallum et al., 1998).

$$P(A \mid B) = \frac{P(B \mid A)\,P(A)}{P(B)} \tag{2.5}$$

The Naive Bayes classifier assumes all attributes of the training data are independent of each other given the context of the class, making it a very simple model. This assumption is called the Naive Bayes assumption (2.6, 2.7) (Zhang and Li, 2007) .

$$P(A_i|C, A_j) = P(A_i|C) \tag{2.6}$$

The joint probability can be rewritten as

$$P(C, A_1, \cdots, A_n) = P(C)\prod_{i=1}^{n} P(A_i|C) \tag{2.7}$$

This independent assumption greatly simplifies learning when the attribute number is high. For text classification, the number attribute reflect the number of different words, which can be quite large. To determine which class a test sample belongs to, the Naive Bayesian algorithm can use the argmax (2.8) which yields simply choosing the class with highest probability.

$$v_{NB} = \underset{C_j \in C}{\arg\max}\, P(C_j) \prod_i P(A_i|C_j) \tag{2.8}$$

When using Naive Bayes for text classification there are two different generative models which both use the Naive Bayes assumption (McCallum et al., 1998). The first model represent a document by a binary vector indicating whether or not words appear in the document. In this model, the word frequency is not captured. The probability of a document is calculated by multiplying all the attribute values with the binary vector. If we say the document is an "event", the words absence or presence represent the attributes of this event. This distribution is based on a multi-variate Bernoulli event model.

The second model is based on the frequency of the occurrence of words in the document (McCallum et al., 1998). The probability is calculated by multiplying the probabilities of the words that occur in the document. For this model we can say that the document is a collection of events while the word occurrence is the "event". This model is called the multinomial event model. None of the two models capture context, meaning that the order of words are lost, but this model does at least capture word occurrences.

According to McCallum et al. (1998) the multi-variate Bernoulli model is traditionally applied in Bayesian networks where the tasks have a fixed number of attributes, while the multinomial model is more traditional in speech recognition creating statistical language models.

Logistic Regression

Logistic Regression is a model predicting the outcome based on measures between the relationship of the outcome variable (class label) and input variables. Logistic regression can take on binary or multi-label problems. Binary logistic regression deals with two possible outcomes (i.e 1/0 or "success"/"failure"). In multi-label situations, the outcome is often interpreted as 1 or 0 where 1 represent correct label and 0 represent some other label (Hosmer Jr and Lemeshow, 2004). The difference between a linear regression and logistic regression is the assumptions and in the choice of the parametric model. Logistic regression follows the same principles used in linear regression when the difference is accounted for (Jordan, 2002). One of the reasons why logistic regression is applied over linear regression is the logistic function (2.9), deciding the class label. The logistic function depicted in Figure 2.6 (often referred to as a sigmoid curve), is calculated in a way that despite the range of input, the output is always returned between zero and one. In (2.9), the Z parameter may range from -∞ to ∞.

$$f(Z) = \frac{1}{1 + e^{-Z}} \tag{2.9}$$

Stochastic Gradient Descent

In supervised learning problems, we are given a data set with training samples consisting of input and output. For each sample, the classifier try to estimate the correct label. This is done by finding the function $f(x)$ parametrized by a weight w vector that minimize the loss $l(f_w(x), y)$ averaged on the input. The expected loss is estimated in (2.10).

$$E_l(f) = \frac{1}{n} \sum_{i=1}^{n} l(f(x_i), y_i) \tag{2.10}$$

To minimize the loss, it is possible to apply the Gradient Descent (GD) algorithm. In GD, all samples are calculated (batch learning) as the true gradient before the weights are updated (2.11).

$$w_{t+1} = w_t - \gamma \frac{1}{n} \sum_{i=1}^{n} \nabla_w l(f_{wt}(x_i), y_i) \tag{2.11}$$

In (2.11), w_{t+1} represent the next weight, w_t represent current weight in iteration t, γ is the learning rate constant and ∇ is the gradient of the weight vector. Stochastic Gradient Descent (SGD) is a simplification of GD (Bottou, 2010; Zhang, 2004). The gradient is calculated for each (randomly picked) sample in an iterative approach (online learning) resulting in a faster convergence (2.12).

$$w_{t+1} = w_t - \gamma \nabla_w l(f_{wt}(x_i), y_i) \tag{2.12}$$

This makes SGD scalable and efficient for large data sets. The convergence of GD and SGD has been studied extensively (i.e. Vapnik and Chervonenkis, 2015; Bordes et al., 2009). Convergence results usually have to satisfy a condition such as $errorrate < threshold$. Note that the SGD algorithm can be applied on classifiers like ANNs.

Artificial Neural Networks

Artificial Neural Networks (ANNs) are described by O'Shea and Nash (2015) as computational processing systems that are heavily inspired by biological nervous systems. It is beyond the scope of this thesis to explain all of the details regarding ANNs. More details and ANNs relation to biology can be found in Rojas's (2013) book *Neural Networks: a Systematic Introduction.*

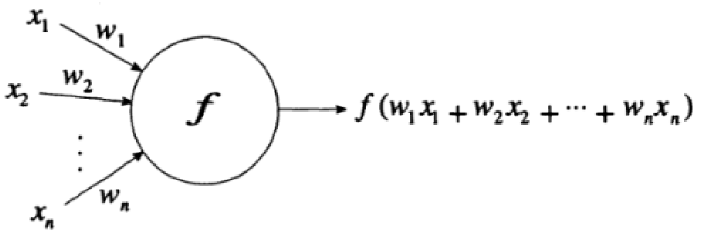


Figure 2.5: An abstract neuron (Rojas, 2013).

ANNs are built using a vast number of interconnected computational nodes. These are referred to as neurons. Neurons can also be found in the brain and can be studied as input-output devices (Hopfield, 1988; Rojas, 2013). Rojas presents the abstract neuron, seen in Figure 2.5. A neuron has n input channels x_i, a primitive function f (also known as the activation function (Specht, 1990)), and one output channel. The neuron's input channels usually have an associated weight which is multiplied with the input value to obtain the final input value. Once the input values has been determined, they are evaluated using the activation function, which determines the output signal. One way to achieve functionality similar to the brain, is to use a sigmoid function (Figure 2.6) as the activation function.

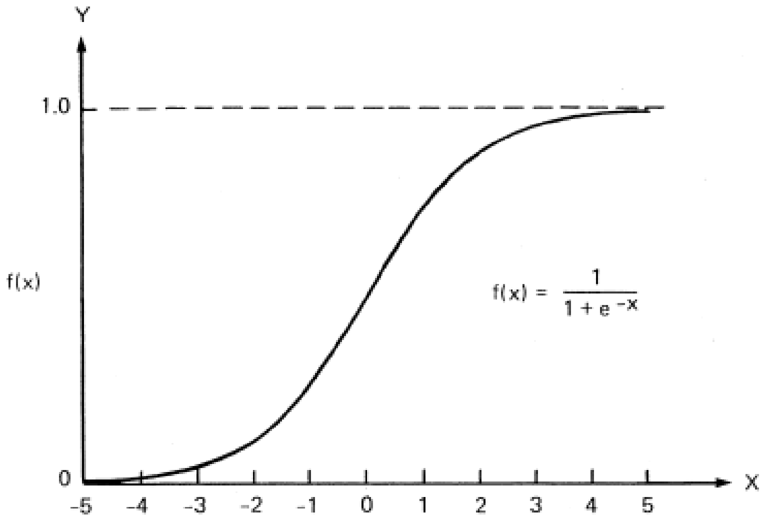


Figure 2.6: Sigmoid function as presented by Dayhoff and DeLeo (2001).

The neurons in the network work intertwined in a distributed fashion to be able to (collectively) learn from the input to optimize its final output. ANN can be structured in numerous ways, but a basic approach is illustrated in Figure 2.7. The network works by loading the input data through the input layer which in turn distribute its result to the hidden layers. The hidden layers will then make decisions based on the output of the previous layer and weigh up how a stochastic change within itself detriments or improves the final output. This is the learning process (O'Shea and Nash, 2015; Hopfield, 1988). By stacking multiple hidden layers on top of each-other, we retain what is known as deep learning.

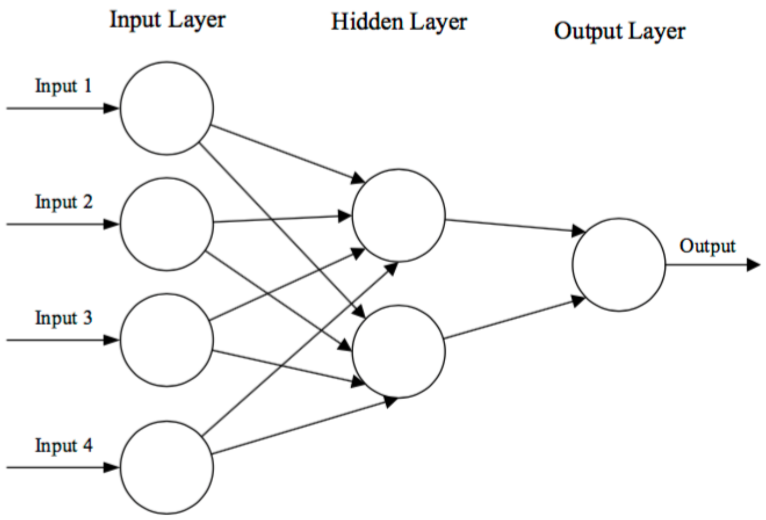


Figure 2.7: Basic structure of an Artificial Neural Network (O'Shea and Nash, 2015). The figure demonstrate a simple three layered neural network, consisting of a input layer, a hidden layer and an output layer.

A typical way of training ANNs is backpropagation (Hecht-Nielsen, 1989), the input is first propagated through the network, and if a misclassification occurs, the error is "backpropagated" to update the weights and minimize the error. The last layer will contain loss functions associated with the classes, which is the value the network tries to minimize.

Sebastiani (2002) describes an ANN text classifier as a network where the input units represent terms, the output units represent the categories of interest, and the weights on the edges connecting units represent dependence relations. For classifying a test document d_j, its term weights w_{kj} are loaded into the input units. The activation of these units is propagated forward through the network, and the value of the output units determine the categorization decision.

There exists several architectures for ANNs, the one described above is known as a Feedforward Neural Network (FNN) as feed the input values trough layers of non-linear hidden units between its input and output channels (Hinton et al., 2012). Other architectures are Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs).

Convolutional Neural Networks
A CNN is similar to the FNN described above as they both consist of neurons that self-optimize through learning. Each neuron will, as in a FNN, receive an input and perform some operations to optimize the output (O'Shea and Nash, 2015). However the CNN was created for working especially with images as input as regular ANNs do not scale well to images (O'Shea and Nash, 2015).

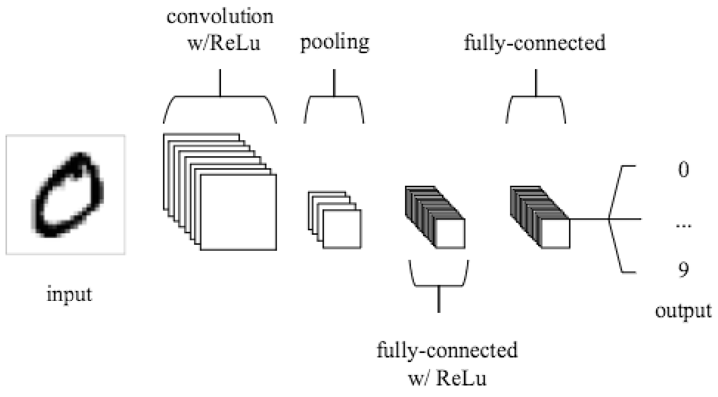


Figure 2.8: A basic architecture of a CNN as presented by O'Shea and Nash (2015).

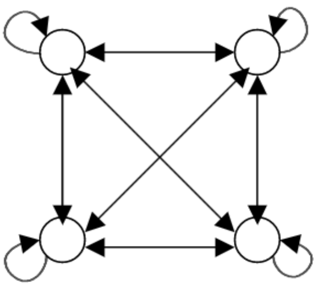
Figure 2.8 illustrates a basic setup for a CNN used for classification on the MNIST[10] data set. According to O'Shea and Nash (2015), CNNs are comprised of three types of layers, convolutional layers, pooling layers and fully-connected layers. Combining and stacking these elements gives the most usual, and basic CNN structures. As a further description is beyond the scope here we refer the reader to O'Shea and Nash (2015) for a more detailed description of architecture and layer functionality.

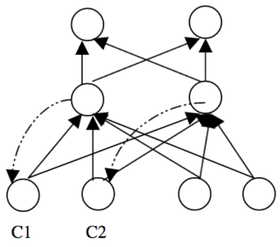
Most of the techniques used for text classification are based on words, such as n-grams. Many researchers have found that CNNs are useful in extracting

[10]`http://yann.lecun.com/exdb/mnist/`, as of 2016-06-06

information from raw signals (LeCun et al., 1989; LeCun et al., 1998). CNNs are primarily used in the field of pattern recognition within images, but Zhang et al. (2015) explored the possibility of using CNNs for classification of texts by using characters as the raw signals.

Recurrent Neural Networks
According to Fausett (1994) a RNN is a neural network with feedback (closed loop) connections. RNNs are designed to learn sequential or time-varying patterns (Medsker and Jain, 2001). RNNs have been applied to a wide variety of problems.


(a) A fully connected RNN.



(b) A simple RNN.

Figure 2.9: RNN architectures as presented by Medsker and Jain (2001).

The architecture of RNNs ranges between the two examples presented in Figure 2.9, that is from fully interconnected to partially connected nets. The partially connected nets include multilayer feedforward networks with distinct input and output layers. The fully connected networks do not have distinct input layers of nodes, and each node has input from all other nodes. It is also possible with feedback to the node itself (Medsker and Jain, 2001). For further details on RNNs we refer the reader to seek more information in Medsker and Jain (2001) and Fausett (1994).

2.6.2 Skewed Class Distributions

Skewed data distributions in machine learning are a common problem. A data set is considered skewed if the classes are not approximately equally represented. In Provost (2000), two assumptions are stated regarding machine learning algorithms:

1. The goal is often to maximize accuracy

2. The classifier will operate on data drawn from the same distribution as the training data

Considering these assumptions when applying machine learning algorithms on imbalanced data sets might produce unsatisfactory classifiers. The reason is illustrated with an example from Provost (2000): given a situation where 99 % of the data is from one class, which is a realistic situation. Then a learning algorithm will find it hard to perform better than the 99 % accuracy achievable by a simplistic classifier that labels everything with the majority class[11]. Based on the underlying assumption, it is the intelligent thing to do. It is more striking when one of the learning algorithms behaves otherwise.

The distribution from the TCP project is illustrated in Figure 2.10. The histogram displays a highly imbalanced distribution of 7,976 paper abstracts taking no position (32.6 %), 3,898 endorses (66.7 %), and 77 rejecting (0.7 %).

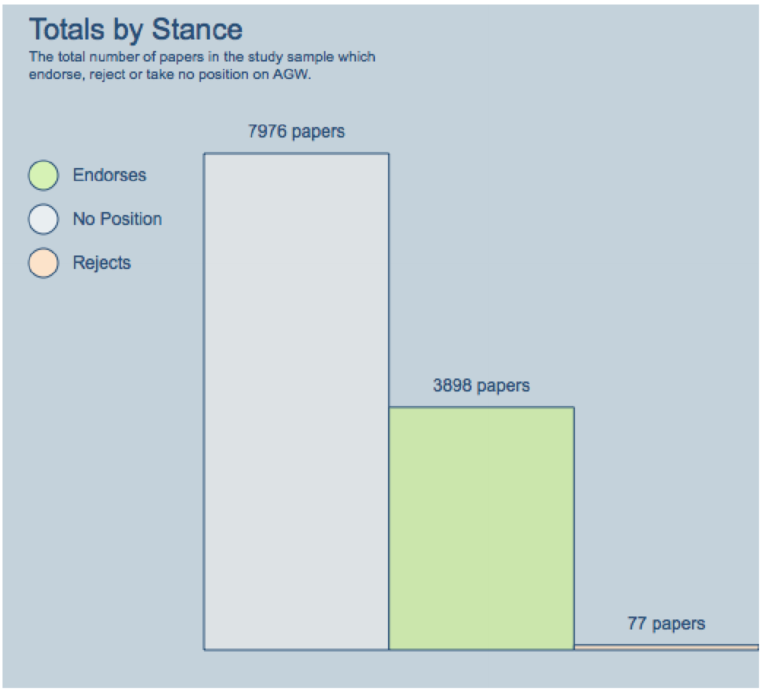


Figure 2.10: Distribution of 11,951[12]abstracts by stance on human-induced global warming (Cook et al., 2013).

[11]Class that occurs the most in the data set.

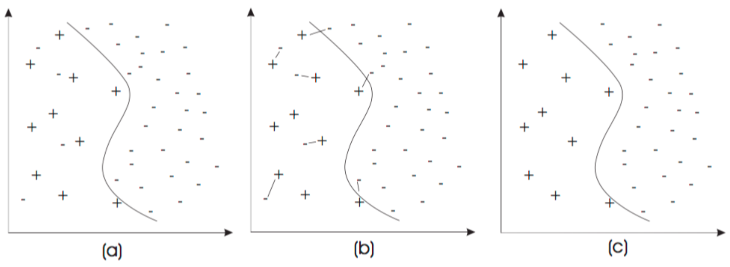


Figure 2.11: Detecting bordeline and noisy data by using Tomek links. (a) shows the original data, (b) Tomek links identified, and (c) which shows only the safe samples (Tomek links removed).

Many attempts to find a solution to highly skewed data distributions have been proposed and perhaps the most common is under- and over-sampling (Monard and Batista, 2002; Tang et al., 2009; Provost, 2000). Under- and over-sampling handles the problem by artificially re-balance the skewed data. Monard and Batista (2002) describes under-sampling as a method to balance the data set by eliminating samples from the majority class. Over-sampling is described as a method to replicate samples from the minority class to achieve a more balanced distribution. Unfortunately, both of these method have known drawbacks. The drawbacks mention in Monard and Batista (2002) states that under-sampling can throw away potentially useful data, while over-sampling can increase the probability of overfitting, as most methods of over-sampling replicate exact copies from the minority class. Research has been conducted to overcome the drawbacks of under- and over-sampling (Chawla et al., 2002). Chawla et al. combines both methods, but instead of replicating exact copies when over-sampling, the new samples are interpolated between the ones that lie close together in the minority class. They try to avoid overfitting by spreading the minority class into the majority space. Batista et al. (2000) and Kubat et al. (1997) analyzed techniques to minimize the potential loss of useful data when applying under-sampling. Both techniques address samples in the training data as "safe", "noise" and "borderline" of the majority class. The borderline and noisy data are detected using the Tomek links and removed (Tomek, 1976). Only safe majority class labels will be used as training data. Figure 2.11 illustrate an understanding of Tomek links.

Besides the techniques suggested above, the use of appropriate evaluation metrics can be helpful toward training a classifier on a skewed class distribution.

[12]The official TCP collected 11,944 paper abstracts. The figure was fetched from `http://www.skepticalscience.com/tcp.php?t=home`, 2016-06-06

		Assignment: as Positive	as Negative
Label:	**Positive**	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Table 2.5: Confusion matrix for binary classification.

Jeni et al. (2013) experimented with threshold metrics like accuracy and F1-score, and rank metrics like area under the Receiver Operating Characteristic (ROC) curve and area under the Precision-Recall curve. While the latter metrics depends only on the ordering of the cases (not the actual predictions), the former metrics have a threshold value which depends on actual predictions. The predictions are taken from a confusion matrix which consist of correctly and incorrectly recognized examples of each class (Goutte and Gaussier, 2005). Table 2.5 presents a binary confusion matrix.

In Jeni et al. (2013) they stated that while accuracy is widely used for empirical measuring performance, it can be misleading when the data distribution is imbalanced. The accuracy is measured in equation Equation 2.13 (Sokolova et al., 2006).

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.13}$$

Further, Jeni et al. (2013) implied that F1-score is a better choice of empirical measure the system and can be interpreted as the harmonic mean of precision and recall. The precision, recall and F1-score can be seen in equation Equation 2.14, Equation 2.15, and Equation 2.16:

$$Precision = \frac{TP}{TP + FP} \tag{2.14}$$

$$Recall = \frac{TP}{TP + FN} \tag{2.15}$$

$$F_1 = \frac{2 \times (P \times R)}{P + R} \tag{2.16}$$

In Equation 2.16, the capital P represent precision and the capital R represent recall. While an appropriate evaluation metric is helpful, it does not really solve the problem of skewed data distribution. However, it does help to estimate the system's performance more accurately. We will use the average F_1 score between the three classes (favor, against and none) which we call macro F-score.

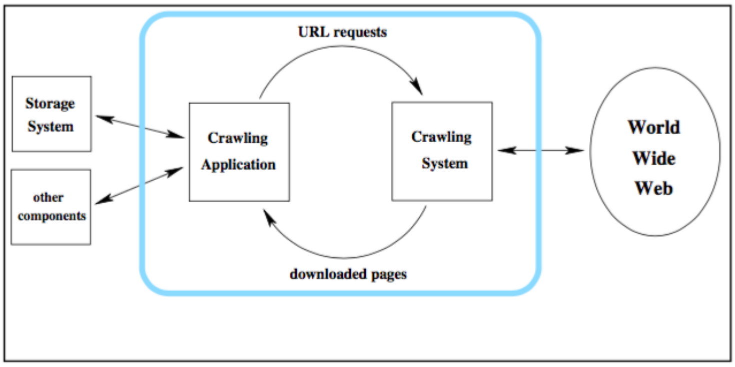


Figure 2.12: Shkapenyuk and Suel (2002)'s suggested architecture for web crawling.

2.7 Web Crawling & Scraping

A web crawler (also known as is a web robot) is a program that systematically scans web pages on the World Wide Web (WWW) (Najork and Heydon, 2002). The web crawler starts off with a collection of Uniform Resource Locators (URLs) that the bot wish to explore. These URLs are called seeds. When the crawler explores the seeds, it identifies links in the web page and store them for further exploration at a later time. The new URLs are called the crawl frontier. The old URLs are at this point discarded. The web crawler loops through all the URLs, and by following a set of policies, it decides on what to do next. Search engines rely on massive collections of web pages which are acquired by web crawlers (Shkapenyuk and Suel, 2002). Researchers have looked at crawling strategies, storage, and indexing to analyze the most effective outcome for different crawlers. An example of web crawler architecture is displayed in figure 2.12. If the crawler is documenting information on the web pages, the document information is stored as a snapshot. This information can be accessed and processed, or studied.

One of the problems that needs handling when crawling the web is retrieval of duplicate content from web pages. Large numbers of versions of the URLs for a web page may exist, making only a tiny part of the content unique. Content like an image can be stored in various sizes and formats, each resulting in an URL. The crawler must therefore take this into consideration when deciding which URLs to explore (Shkapenyuk and Suel, 2002; Najork and Heydon, 2002). Another problem is scaling. All of the popular search engines (like the Google search engine (Sehgal et al., 2009) use crawlers that must scale to substantial portion of the web. Due to the competitive search engine business, the design of these crawlers are not publicly available (Najork and Heydon, 2002).

How the crawler behaves is a result of a set of policies. There exists various policies (Najork and Heydon, 2002)

1. A selection policy decides which web pages to download. The crawler can be restricted to only find HyperText Markup Language (HTML) web pages and leave out other types like PDFs.

2. A re-visit policy is created to check for updates. Some search engines use web crawling in order to update indexes and content. This policy is used to avoid outdated resources.

3. A politeness policy is constructed to avoid overloading of the server. A web crawler can retrieve data quicker than humans. The crawler can perform multiple requests per second, and a policy is useful to avoid crashing servers.

While web crawlers visit web pages through URLs and stores the whole page, web scraping or web data extraction is a technique used to extract certain information from web pages. Web scraping is used on the WWW along with web crawling. The focus of web scraping is transforming unstructured data into a format that is understandable and organized. Any content that can be viewed on a Web page can be scraped. Solutions of web scraping range from software that requires human effort, to fully automated systems that scrapes entire web pages.

Despite the fact that web pages can distribute Application Programming Interfaces (APIs), the developers of a particular web page may change or delete such accesses. A web scraper will not be affected by the changes, however it is highly sensitive to changes in the format of the web page. The first step of web scraping is the same as web crawling. The web scraper need a list of URLs to return data. The problematic part is to fetch the actual data from the page's markup. Web pages are usually formatted in a way so that the markup is wrapped in classes and identities. When the format is changed, the only way to fix the problem is if the developer adapt the web scraper to the new format (Salerno and Boulware, 2006). It is often not beneficial to scrape the entire page, as this causes a vast amount of unnecessary data to be stored. The result is that one can scrape only selected classes or identities (potentially) reducing the amount of data collected significantly.

2.8 Visualization

Visual representation of data can provide key insights when performing analysis of data. The way the data is represented is essential to reveal characteristics that earlier were undiscovered. Unsuitable representations prevent users from observing the data in a productive manner. This subject falls under field of *visual analytics*[13]

In the book Miller and Han (2009) data mining is described as the process of distilling data into information about the data set. Friedman (1998) discuss the relationship between data mining and statistics. Data mining is in a way comparable to a statistical analysis, but in the case of statistical analysis the analyst is required to specify a model *a priori* and then later test these hypnotises using tools such as probability theory and decision theory. On the other hand, when performing data mining, the hidden patterns or information we seek are rather difficult to specify *a priori*. And as a result the information being sought can only be vaguely described in advance (Miller and Han, 2009). An example is the second goal of this thesis; *Discover how stance is related to external factors*. Since we do not know exactly what we are looking for it, is hard to provide a specific description. This results in a hypotheses that is formulated and changed as we go.

According to Idreos et al. (2015), data exploration is about efficiently extracting knowledge from data even if we do not know exactly what we are looking for. Keim et al. (2002) argues that for data mining to be effective it is important to include the human in the data exploration process. Humans will provide creativity and general knowledge that a computer is not capable of. Humans have the ability to recognize patterns, similarities, and dissimilarities through their visual system. By presenting findings in a graphical way, one will be able to explore data analysis and hopefully observe interesting correlations or patterns. Visual data exploration, in general, aims to present the data in a visual form that allow humans to get insight into the data. Keim et al. (2002) states that data mining techniques have proven to be quite useful when conducting exploratory data analysis. In Figure 2.13, an example of how to visualize the development of consensus on anthropogenic global warming over time.

Keim (2001) states that while it is important to include a human in the process, it will not be sufficient to achieve great results. Keim says that we have to combine the human skills with the vast storage capacity and computational power of today's computers. If we successfully manage to integrate humans in the data exploration process, and as a result be able to apply their perceptual abilities to the data at hand, we might gain new insights that would have been unable to perceive using the two resources individually.

According to Keim (2001), visual data exploration is especially useful when little is known about the data and the exploration goals are vague. As the user is directly involved in the exploration process, adjusting the exploration goals might be done along the way using interactive visualization.

[13]`https://en.wikipedia.org/wiki/Visual_analytics`, as of 2016-06-06

In our case, the visual representations opens up the possibility for others to explore the data. This might lead to observations we were not able to make and is beneficial to the cause of exploring the consensus in climate science.

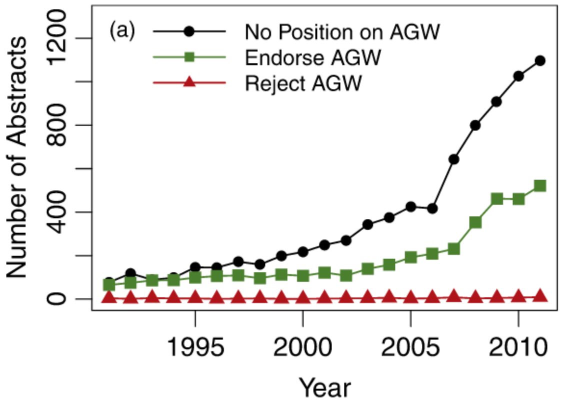


Figure 2.13: Cook et al. (2013): Visualization of the development of the consensus on anthropogenic global warming. Shows total number of abstracts categorized into *in favor*, *against* and *neutral (no stance)*.

When creating an interactive visualization it is essential to understand the data. What do you have at hand? How many dimensions? Do you care for all dimensions? If working with multidimensional data, a simple visualization is not possible and one might have to filter out some of the dimensions to create a useful representation. Common visual representations for two-dimensional data include quite basic, but powerful statistical charts such as bar, and pie charts, and simple X vs Y plots.

There exists numerous useful visual representations, all depending on what is to be presented and the data available. A specific example of a visual representation from this collection is a geographic plot on a map, shown in Figure 2.14. This can be useful when the goal is to illustrate a geographical distribution.

[14] `http://bl.ocks.org/mbostock/4060606`, as of 2016-06-06

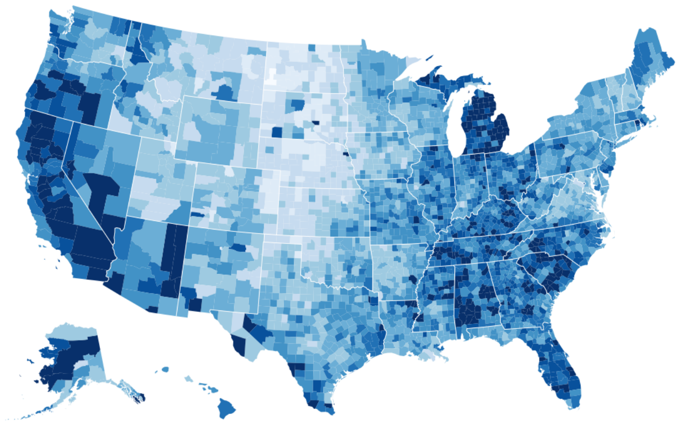


Figure 2.14: Choropleth visulization of geographic data[14].

Chapter 3

Architecture and Implementation

This chapter presents the architecture of the system, its data models, and the implementation. The architecture provides insight to how the components interact while the implementation digs into the more technical details such as frameworks and libraries. The first section explains the overall architecture of the system along with the programming languages we have used. It should give the reader a broad understanding of the system before each of the individual components and their implementations are described. Note that we do not include any code in this thesis.

3.1 Architecture

The overall architecture introduces the three major components of the system: Search & Information Retrieval, Stance Classification and Visualization. Figure 3.1 provides insight to how these interact as well as a basic understanding of how the data flow through the system, from information retrieval to visualization.

The information retrieval component is the foundation of the system. It provides the classification component with the raw data scraped from the web (section 3.2). After receiving the raw data from information retrieval, the classification component processes the data and utilizes the information to construct a model which in turn is used to classify data (section 3.3). The predictions of new data along with additional information serves as input to the visualization component. This last component analyzes the incoming data from the former two components through visual representations to expose patterns of stance in

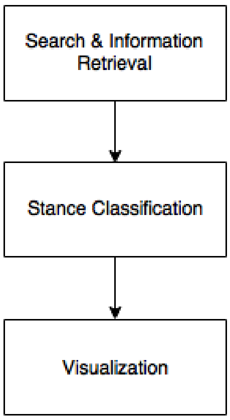


Figure 3.1: Overall Architecture of the System.

climate science (section 3.4).

3.1.1 Programming Languages

Python

We have chosen Python[1] as the main programming language when implementing the system. Python provides high-level data structures such as lists and associative arrays, dynamic typing, modules and classes. It has simple and elegant syntax that makes it easy to work with. Additionally there exists a lot of good libraries including the scikit-learn (Pedregosa et al., 2011) library for machine learning. Python is utilized in the search and information retrieval component and the stance classification component.

Extensible Markup Language

Extensible Markup Language (XML) describes a class of data objects known as XML documents (Bray et al., 1998). XML is a strict form of the Standard Generalized Markup Language (SGML) (ISO, 1986). The documents are made up of units called entities. These entities contains parsed or unparsed data. Documents that are parsed contains characters that form either character data or markup. The markup encodes to a description of the document's layout and structure. The XML can put constraints on the layout and structure of a document through the

[1] `https://www.python.org/`, version 2.7.10 as of 2016-06-06

XML Schema (Fallside and Walmsley, 2004; Thompson et al., 2004; Biron et al., 2004). XML is utilized in the information retrieval component.

XML Path Language

XML Path Language (*XPath*) is an expression language that allows extraction of values from XML documents (Berglund et al., 2003). Each XML document provides a tree representation which *XPath* can take advantage of. The values in XML documents can be boolean values, characters, integers and such. The *XPath* provides a hierarchical addressing of nodes in the document. *XPath* is utilized in the information retrieval component.

Web Service Description Language

Web Service Description Language (WSDL) is a language that follow the XML format (Christensen et al., 2001). WSDL describes the network services as a set of endpoints operating on messages. The operations are bound to a network protocol with a message format defining an endpoint. A collection of abstract endpoints in a WSDL document are called services. The WSDL document has a template it follows for defining a network service. The template includes elements such as types, messages, operations, port types, bindings, ports and services. WSDL is utilized in the information retrieval component.

JavaScript

JavaScript[2] is one of the most common languages for developing content for the WWW. We have used JavaScript, alongside HTML and CSS, to create the visual representation component of this system.

3.2 Search & Information Retrieval

The Search & Information Retrieval component is the base component of the system. The component provides the crucial data needed for further work. This section describes the component in detail. Required background theory is found in section 2.7.

TCP[3] is the main inspiration for this thesis and they gathered data that is essential for this project. The data is available as a text file containing the 11, 942 abstracts they used in their work. For each paper they have included article

[2]`http://www.w3schools.com/js/`, as of 2016-06-06

[3]Introduced in section 2.1.

id, the year of publication, category, endorsement level, title, and the paper abstract[4]. The understanding of what TCP has provided is essential to understand decisions made during the work with this thesis.

3.2.1 Model

The data collection task is divided into two sub-tasks; retrieval of climate science literature published after 2011 (abstracts and meta-data), and retrieval of meta-data for publications found in TCP data. The meta-data is intended to enrich the TCP data with additional information such as author affiliation and paper length. The additional data will be utilized to create features for classification and used to find new dimensions to explore through visualization. As the information retrieval component is divided in two tasks, we use a generalized model. Figure 3.2 presents this model. For some of the approaches the scraping procedure is excluded as they utilize a Python module to retrieve data instead of scraping with a web crawler.

Figure 3.2 shows three steps: pre-processing, execution, and post-processing. Pre-processing consists of text processing (such as removal of punctuations) and the creation of customized search queries (these are slightly different for each literature database). Once a query is created, the search is executed in the environment of the targeted database. The result is stored and passed to the post-processing step. In post-processing the system compares the retrieved data against the search criteria assuring that the collected data match the desired data. The final results are stored as JSON files.

3.2.2 Implementation

In the search for scientific papers, we explored three different literature databases: Crossref, Scopus and Web of Science (WoS). We have made an effort to find the best database, or a way to combine results from all of them in order to achieve the highest possible success rate for retrieval of papers and their meta data. As a result, five slightly different approaches are described. In common for all of them is that the *Pandas* python library will be used to read input files and manipulate data structures.

One approach retrieve paper abstracts only, three approaches retrieve meta-data (two of them often includes abstracts) and one retrieve relevant papers (abstracts and meta-data). The combination of methods that make up the final version is described in section 4.4-

[4]See Appendix C: Data from TCP for a detailed description of the data.

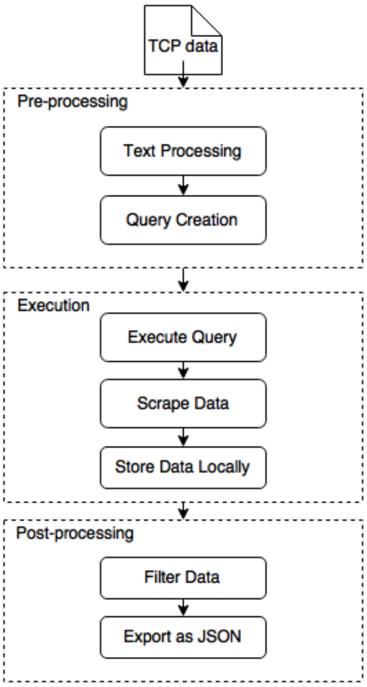


Figure 3.2: Information Retrieval Model.

Three of the implementations are based on web crawlers using *Scrapy*[5]. Web crawlers are widely used but not all are welcomed by the hosting servers. To make our crawlers less intrusive and destructive we implemented a politeness policy to avoid that the databases blocked our crawlers. A few experiments of less relevance to our research were carried to create a well-behaved crawler. These are listed in Appendix E: Additional Experiments and Results.

Scopus: Abstract Retrieval

This strategy was implemented as part of the preliminary work as an initial approach to obtain data related to the TCP data. The implementation was intended to be part of a strategy where we first would obtain titles of related papers, then retrieve their abstracts. Later work shifted our focus to meta data and we discovered more efficient and successful ways of obtaining both meta data and abstracts.

[5]See section 3.2.2

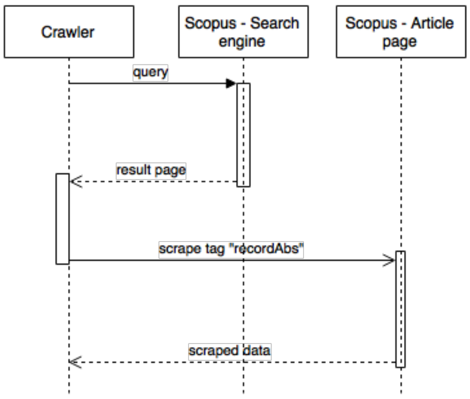


Figure 3.3: Sequence diagram showing the web crawlers strategy for retrieving an article abstract.

This approach is based on a web crawler that search for publications based on titles. The abstracts are scraped if they are embedded in the article web page. As described in section 2.7, web crawlers relies on URL seeds as starting points. This means that we would have to come up with a way allowing the crawler to perform searches in the Scopus database through the use of URLs. With this in mind a set of manual searches were carried out to analyze the structure of the Scopus database search URLs. We found that a search URL contains 33 query parameters, but only three of these are required (*searchterm1*, *st1*, *s*) to search for the papers based on the title. To initialize a customized URL for the Scopus web page, the first two terms are set equal to the title of the article, while the last parameter, *s*, is set equal to the title with the keyword *TITLE* inserted in front. Values in the query parameters were based on the TCP data. For more details on how the URLs were generated, a complete list of the available query parameters, an example URL and figures of the procedure, see Appendix D: Search & Information Retrieval.

After conducting a search, the crawler obtains a page containing a list of matching papers (if any). The crawler stores the URL of the top ranked hit. This URL points to the location of the top ranked paper's article page. In a new thread the crawler visit the new URL and locates the specific HTML tag id *recordAbs* to extracts the abstract. If the abstract exists, it is stored locally. These steps are illustrated in Figure 3.3.

This approach is prone to errors as there is no guarantee the results have matching titles. To filter out potential errors we implemented a comparison check between the title searched for and the title retrieved. The filtering check uses the module *difflib*[6] in the Python standard library. The module includes a class named *SequenceMatcher* which has the method *ratio*. The method performs a character comparison and returns a floating number between 0 and 1. 1 indicates a 100 % match and 0 indicates that they have nothing in common. If the similarity is above 95 %, the check accepts the paper. The reason we did not use exact similarity is that we found minor differences in paper titles such as missing colons. Two examples can be seen below with the original source in brackets:

- "Atmospheric lifetime, its application and its determination: cfc-substitutes as a case study" [**Scopus title**]
- "Atmospheric lifetime, its application and its determination - cfc-substitutes as a case-study" [**TCP title**]

The first example display a subtle difference of the symbol ":" and "-", but they represent the same paper.

- "Signals of anthropogenic influence on European warming as seen the trend patterns of daily temperature variance" [**Scopus title**]
- "Signals of anthropogenic influence on European warming as seen in the trend patterns of daily temperature variance" [**TCP title**]

The second example show a difference of one word missing (the word "in") in the Scopus title. But they too represent the exact same paper.

Scopus: Meta-Data Retrieval

We disregarded meta-data retrieval from Scopus in the preliminary work as Scopus did not provide an easy-to-use RESTful API like CrossRef did (see next section). However, after spending time browsing the Scopus web page we found a way to export meta-data as CSV-files (Appendix D: Search & Information Retrieval provides illustrations from the web interface on how to do this). The implementation is similar to the one just described, so we will focus on the differences.

The URLs generated in this implementation introduce two additional query parameters: *yearFrom* and *yearTo*. These restricts the returned documents based on publication year. As we observed that the publication years listed on Scopus did not always match the ones found in TCP data[7], a tolerance of ± one year was included.

[6] `https://docs.python.org/2/library/difflib.html` as of 2016-06-06

[7] Observed that some samples in Scopus was off by one year compared to the year registered in TCP data.

The next step introduces the biggest difference between this and the previous implementation. Rather than to crawl onto the article page and scrape information, we extract an unique Scopus identifier (*eid*) from the article page URL. This URL is located in the HTML document inside a div tag with id *resultDataRow0*. The Scopus identifier is further used to generate a new URL[8] enabling *scrapy* to directly download the CSV-file with meta-data (often including abstract). The sequence of events is similar to the one shown in Figure 3.3, but instead of scraping HTML tags the second step is replaced by the CSV download. Post-processing is done according to the details explained in Scopus: Abstract Retrieval.

CrossRef: Meta-Data Retrieval

During the preliminary work, we came across CrossRef[9] and their RESTful API[10] for retrieval of meta-data. The API returns results as JSON records wrapped in HTML markup. Meta-data collection using Crossref is achieved by generating a URL and scraping the resulting page[11].

The process of creating URL seeds for Crossref were more or less equal to ones described for Scopus. However, the parameters were different. The URL seeds required only the three query parameters *query*, *rows* and *sort*. We set *query* to the paper title as it is the main search criteria. *Rows* restricts the number of returned results. We only want one and set the value accordingly (1). *Sort* determines how the results are arranged. We set it to 'score' which orders the records based on the highest similarity score. More details on the use of the API and how we created the URLs can be found in Appendix D: Search & Information Retrieval.

The web page containing the results in JSON format is stored as an XML tree structure object by *scrapy*. By using *XPath* we were able to easily extract the information. The values are returned as strings, and after using regular expression (*re* python module) to remove HTML markup they were stored in a JSON-file. Post-processing is done according to the details explained in Scopus: Abstract Retrieval. No abstracts were found in the meta-data we collected from Crossref.

[8]See Figure D.8 in Appendix D: Search & Information Retrieval for the procedure.

[9]`http://www.crossref.org/`, as of 2016-06-06

[10]`https://github.com/CrossRef/rest-api-doc/blob/master/rest_api.md`, as of 2016-06-06

[11]We are aware that using python to simply request the data directly without having to scrape it of the web page is possible. At the time we discovered the API web crawling was our main focus, so we decided to continue in this direction.

Web of Science: Meta-Data Retrieval

During the work on this thesis, our faculty gained access to the WoS database. Out of curiosity, ten randomly selected titles (that Scopus did not find) were used to test the content coverage of the WoS database. The searches returned 10 out of 10 records. Based on this intriguing result a strategy employing WoS to collect meta-data was implemented.

In contrast to the other databases (Scopus & Crossref), there exists a python module[12] that provides easy access to the WoS database. Valid credentials are required to access the WoS database. By using the python module we avoided the work of analyzing URL links and configuring *Scrapy*.

The WoS database use the *SOAP* protocol for exchanging structured information with XML as its message format. The *SOAP* protocol was therefore used when we used the web services *search* and *related record*. The *search* web service provides custom search functionality for records. Within the *search* service several methods exists. We used the *query* method which provides an easy way to perform searches for data using terms such as title, year, or WoS ID. Our queries consisted of title and year (like in Scopus: Meta-Data Retrieval, ± one year were used). A query to the web service looks like this:

> ”TI=CLIMATIC EFFECTS ON THE PHENOLOGY OF GEOPHYTES AND PY=(1990 OR 1991 OR 1992)”

”TI” is short for title and ”PY” is short for publication year. Each title is stripped for the WoS search operators: ”AND”, ”OR”, ”NEAR”, ”NOT” and ”SAME”. If any these words were detected, they were replaced with a blank space. The results were returned as XML tree structures, where we utilized *XPath* to extract information. An example showing how to extract references is provided below.

> ”*../static_data/fullrecord_metadata/refs*”.

A complete collection of *XPaths* for retrieving values from the returned XML documents are included in Appendix D: Search & Information Retrieval. The implemented strategy retrieved the following data:

[12]See Table 3.1

- Abstract
- Author name(s)
- Language
- Number of references
- Keywords
- Headers
- Sub headers
- Subjects
- Document type
- WoS identifier
- Publication info (Month, volume, type, issue, length)
- Organization info (country, city, street, name)

Post-process filtering is performed in accordance with previous approaches.

Web of Science: Relevant Paper Retrieval

Earlier we observed that the Scopus database (often) provides related publications on their web page. Therefore one solution to this task could be to use *scrapy* and look for static HTML markup ids. However, this is not a very clean approach and no clever way was discovered to perform this. Luckily we came across an easier alternative.

The previously described method based on WoS led us to discover the *related records* web service. This service performs a search for records that are related to a specified WoS identifier. When we collected meta-data in the previous sub section (Web of Science: Meta-Data Retrieval) these identifiers were obtained for the TCP data. This means that we could use these to find publications related to data found in TCP. The relation between the records searched for and retrieved using the *related records* service is defined in the *Web of Science Web Services (version 3.0) User Guide*[13] (from here on referred to as the WoS user guide) as articles with one or more citations in common. From the user guide:

> ”Searches for articles that have one or more cited references in common with the article specified by the unique identifier (UID).”

The python module previously mentioned did not provide functionality for the *related records* service. So we implemented the functionality by extending the module containing the *search* service according to specifications found in the *WoS user guide.*

For every record with meta-data[14] we asked for up to 8 related records. An additional restriction is added based on the publication year. As Cook et al. (2013) and TCP ended their work in 2011, we only searched for publications after 2011. The results are formatted as XML document objects. Using *XPath* we extracted the values of each record like described in Web of Science: Meta-Data Retrieval.

[13]July 7, 2015

[14]Found using the implementation from Web of Science: Meta-Data Retrieval.

The post-filtering implementation for this step is quite different from previous implementations as we do not have any data to cross reference with. Post-filtering is performed by constructing a pipeline of four filtering stages. The first step removes all records not containing abstracts. The next step removes duplicates and the third step removes irrelevant subjects. The irrelevant subjects were uncovered during an inspection of the obtained data. We noticed several records with subjects that were not related to climate science at all. To determine which subjects to include, a list of unique subjects found in the TCP data (202) were extracted and compared to subjects found in the new data (258). This resulted in a list containing 56 subjects that were not present in the TCP data. In addition to the 56 subjects, we manually selected 11 subjects observed in the TCP data that seemed to, in general, introduce irrelevant data. The final list contained 67 subjects that were removed from the newly obtained data. The complete list of the subjects we removed are included in Retrieval of Meta-Data, while a subset is listed below.

- **Artificial Intelligence**
- **Hardware & Architecture**
- **Public Administration**
- **Robotics**
- **Sport & Tourism**
- **Women's Studies**

Below we introduce the first part of abstracts from three of the removed records. The two first seem to be highly irrelevant while the last one possibly is relevant.

> ”This study investigates the rhetoric and practice of two actors (Public and Regime) involved in the foreign affairs of Iran. The literature section...”

> ”An integrative framework is proposed to advance management research on technological platforms, bridging two theoretical perspectives: economics, which...”

> ”The implementation of climate change adaptation polices has barely occurred in developed countries. This paper examines...”

The observation that some, seemingly relevant, publications were removed gave rise to the final post-processing step; from articles previously removed, we include the papers containing the word ”*climate*” in its abstract.

Tools and Frameworks

An overview the non-standard frameworks utilized in this part of the system is provided in Table 3.1.

Framework	Description
Scrapy[15]	Is an open source and collaborative library which provides a set of tools for crawling websites and extracting data.
Pandas[16]	Is an open source python library providing high-performance, easy-to-use data structures.
Crossref[17]	RESTful API.
JSON[18] module	Lightweight standard python module for encoding and decoding json files.
Regular Expression[19]	Module provide regular expression matching.
Python Module for WoS[20]	Web of Science python module. Provides access to the database using web services. The module is wrapping, for instance, the suds[21] module to create a lightweight SOAP client.

Table 3.1: Frameworks and tools utilized for information retrieval.

3.3 Stance Classification

This section presents the models and describes the implementations used to create the stance classification component. The content of this section is based on background knowledge presented in chapter 2, especially section 2.2 and section 2.6.

[15]`http://scrapy.org/`, version 1.0 as of 2016-06-06
[16]`http://pandas.pydata.org`, version 0.17.1 as of 2016-06-06
[17]`https://github.com/CrossRef/rest-api-doc/`, as of 2016-06-06
[18]`https://docs.python.org/2/library/json.html`
[19]`https://docs.python.org/2/library/re.html`, as of 2016-06-06
[20]`https://pypi.python.org/pypi/wos/0.1.1`, as of 2016-06-06
[21]`https://bitbucket.org/jurko/suds`, as of 2016-06-06

3.3.1 Model

The task of detecting stance is divided into two steps. The first step is pre-processing of data using techniques from the field of NLP. The second consists of training a stance classifier and performing classifications on unlabeled data. The model for this component is illustrated in Figure 3.4.

The input to this component is first and foremost the data from TCP (described in Appendix C: Data from TCP). This data is used for model creation and is separated in three individual sets: training, validation and test. The model illustrated in Figure 1.1 shows how the classifier is trained and later tested. The performance is validated on the validation set while the test set is held out until the very end and used to estimate how our final system performs on unlabeled data. In addition, the Search & Information Retrieval component supply recent climate literature that will be labeled using the created model.

When the input (in our case the abstracts) is text it is common to perform several steps of pre-processing. Pre-processing data before passing it to the system may reduce processing time, reduce the vector space, and increase performance of the classifier. The main focus will be on tokenization of the input. But in addition operations such as removal stop words and application of lemmatization will be tested. This can boost performance. An example of pre-processing of text is shown in Figure 3.5.

3.3.2 Implementation

Classification can be implemented using numerous different strategies. This subsection describes the implementation details of the approaches we have used. Some aspects of implementation are the same for multiple approaches and will be presented first. After this our approaches will be described.

The splitting of the TCP data into the mentioned data sets were performed using *scikit learn*'s *train_test_split* method. All splits were stratified, meaning that all sets have an equal class distribution. The resulting sets were stored in CSV-files. To efficiently load data and handle the data structures in the system we used the *Pandas* python library. Where optimization is used it is performed using *scikit-learn*'s *GridSearchCV* (see Appendix F: Grid Search). *GridSearchCV* enables us to perform cross-validated grid searches to detect optimal parameter values for the classifiers.

Unless otherwise stated the classifiers work with multi-class classification. The class labels are *in favor, against*, and *none.*

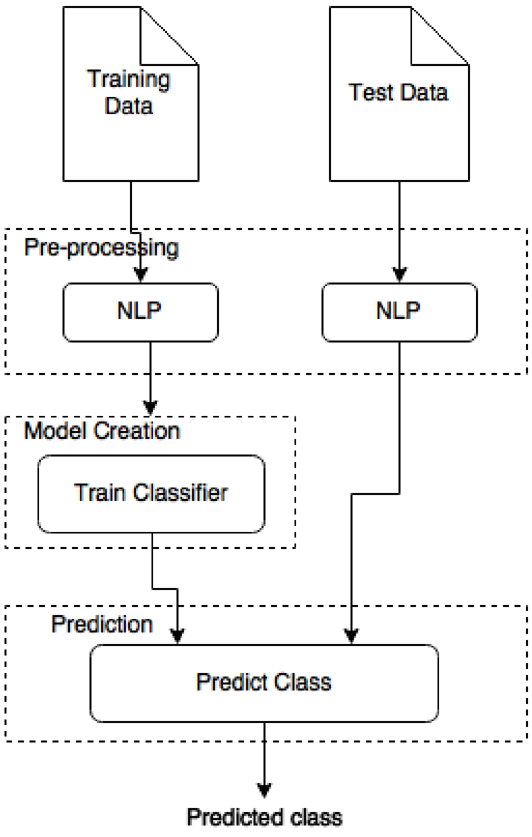


Figure 3.4: Classification model.

Baseline

The baseline system is an important first step in developing a classification system. It will be used to evaluate all other alternate implementations to measure the improvement/deterioration of new strategies. The baseline is simplistic and will in our case be implemented to show how an *easy-to-implement/out-of-the-box* system performs.

We implemented the baseline using the *scikit-learn* (Pedregosa et al., 2011) Python library. We utilize the *pipeline* to set up the training procedure. It does exactly what it sounds like: connects a series of steps in training to one object and use that object to make predictions. An example of a pipeline is provided in Figure 3.6.

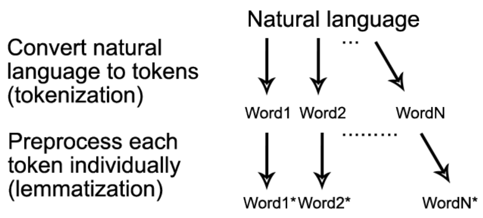

Natural language
Convert natural language to tokens (tokenization)
Word1 Word2 WordN
Preprocess each token individually (lemmatization)
Word1* Word2* WordN*

Figure 3.5: Example of one natural language processing approach.

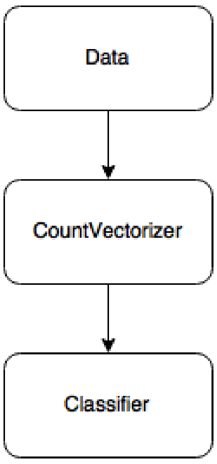


Figure 3.6: Example of a simple pipeline.

The implementation is best described step by step:

1. First we select and prepare the correct data. In our case the abstracts collected by TCP that we have stored in CSV-files.

2. In the second step we create the pipeline consisting of feature extraction methods and a classifier. The feature extraction for the baseline is implemented using *scikit-learn*'s *CountVectorizer* class. The *CountVectorizer* executes tokenization and other NLP procedures set by the input parameters and then count the frequency of instances (i.e. words). *CountVectirizer* offers options such as removal of stop words, convert input text to lower-

case, set (maximum) token limit, analyzer type[22] and different shapes of n-grams. The last part of the pipeline is the classifier. The classifiers used in this implementation are listed in Table 3.2.

3. The final step is training and evaluation. *Scikit-learn*'s cross-validation is used to implement the training procedure. Once the pipeline is trained, the unseen data can be predicted. The pipeline class provides a built-in method called *predict* which does exactly this. For evaluation of performance *scikit-learn* provides a class named *metrics* which enables us to implement standardized metrics to evaluate the classifiers. It also allows us to create a confusion matrix displaying precision, recall and F1-score for each class.

We chose to implement the baseline using a set of different classifiers. These are listed in Table 3.2. All of the classifiers can be found in the *scikit-learn* library. As mentioned earlier, we wanted the baseline to be simplistic. As a result all classifiers are implemented using the default settings and the *CountVectorizer* only creates simple unigram features[23].

Classifier	**Description**
Most frequent	Dummy model: always predicts the class that is most represented in the training data
Stratified	Dummy model: generates predictions by respecting the training set's class distribution
Multinomial Naive Bayes	Generative probabilistic model based on frequency (i.e of words)
Bernoulli Naive Bayes	Generative probabilistic model based on the appearance (i.e of words)
Support Vector Machine	Discriminative model that assign samples to separate classes
Linear Support Vector Machine	Linear implementation of Support Vector Machine
Logistic Regression	Regression model based on regression analysis
Stochastic Gradient Descent	Stochastic approximation model based on minimized loss

Table 3.2: Baseline classifiers.

[22] Word, character, or characters within word boundaries

[23] *analyzer='word'*, *ngram_range=(1,1)*, *stop_words=None*, *max_features=None*

Classification Using Additional Shallow Features

This implementation is quite similar to the one previously described. However, the pipeline is expanded to include more features than just word count. The new pipeline takes advantage of the *FeatureUnion* class found in *scikit-learn*. *FeatureUnion* applies several feature extractions in parallel before it concatenates the results, giving one set of features. An example of such a pipeline is depicted in Figure 3.7, illustrating input data being processed in parallel with features like *TF-IDF*[24], abstract length and language before being put together ready for training.

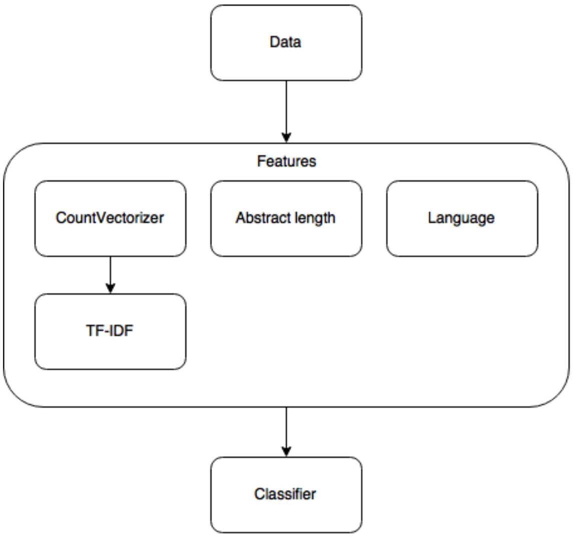


Figure 3.7: A pipeline processing features in parallel.

All features in *scikit-learn* have to be transformed into numerical vectors for the classifier. Fortunately, *scikit-learn* provides tools for these conversions such as the *OneHotEncoder* and *DictVectorizer*. The *OneHotEncoder* encodes input categories (integers) to vectors while *DictVectorizer* map feature-values to vectors. The complete list of features that we have implemented is provided in Table 3.3. This strategy is implemented with several classifiers, and since the approach is not part of the baseline we optimized their parameters.

[24]See section 2.2

Feature	**Description**
Count-vect	Counting words
Year	Mapping year to value
Language	Mapping language to value
Reference	Mapping reference count to value
Orgs-feat	Mapping country to value
Keyword	Length of keywords
Month	Mapping month to value
Volume	Mapping volume to value
Type	Mapping type to value
Issue	Mapping issue to vale
Publication length	Mapping page length to value
Author	Mapping author to value
Document type	Mapping document type to value
Subject	Mapping subject to value
sub-header	Mapping sub headers to value
Header	Mapping header to value
Title length	Length of title
Abstract length	Length of abstract
LDA	Counting words from LDA abstract topics

Table 3.3: Shallow features.

Stance vs No-stance & In Favor vs Against

To take a different approach to the classification problem we implemented a classifier consisting of two classification steps. The implementation uses the same libraries and functions described for the baseline implementation. The difference is that it utilizes two binary classifiers. The strategy is illustrated in Figure 3.8.

Instead of simply classifying an abstract directly into one of the three classes (*in favor, against*, or *none*), we wanted to investigate if it would be beneficial to perform two binary classifications. First classify whether an abstract contains a stance (*in favor* or *against*) or not (*none*). Second perform a classification to predict whether the abstract (that were labeled as containing a stance) is *in favor* or *against.*

Before training the first classifier, the two classes *in favor* and *against* are merged into the class *stance* and leaving the class *none* unchanged. While the first classifier use this data for training, the second classifier is trained on all samples from the classes *in favor* and *against* found in the training set (disregarding *none*).

When training is complete predictions are made. First the abstracts are presented to classifier number one. The outcome of this prediction is propagated to the second classifier. Classifier number two performs predictions on the data labeled as *stance* by the first classifier. As a result there might be three classes present in the set, but predictions were restricted to *in favor* and *against*.

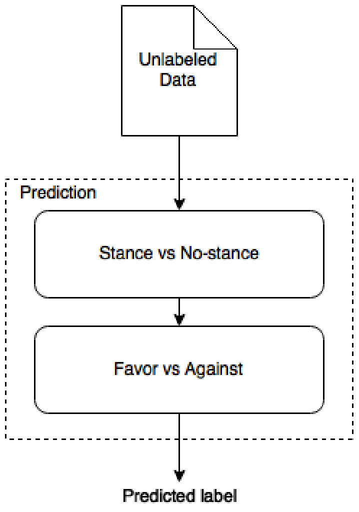


Figure 3.8: The stance vs no-stance + in favor vs against classification strategy.

The first classifier is responsible for final predictions of the label *against* while the second classifier produces the final labels for the classes *in favor* and *against*.

Word Embeddings

We have created features based on word embeddings. The word embeddings are obtained using GloVe and word2vec.

GloVe

Section 2.5 introduced GloVe, an unsupervised learning algorithm used to create word vectors. In this strategy we utilize multiple types of word vectors created by Pennington et al. (2014). The word vectors comes in a several dimensions (25, 50, 100, 200, 300) that supposedly capture different granularities of meaning. We used pre-trained word vectors trained on Wikipedia 2014 + Gigaword 5. Details on these can be found in Appendix G: Pre-trained Word Vectors.

Word embeddings enables us to measure semantic similarity of words. However, in order for us to measure the semantic similarity between abstracts, rather than the isolated words, we needed a way to obtain vector representations of documents (collections of words). Mitchell and Lapata (2010) looked at the possibility to use word vectors to represent the meaning of word combinations in a vector space. They suggest, among other things, to use vector composition, operationalized in terms of additive (or multiplicative) functions. Accordingly, vector representations were created of abstracts by combining the vectors of their words. In total we adapted 10 different sets of word vectors created using GloVe. The resulting features (from here on called GloVe features) were obtained by summing the GloVe vectors, per dimension, for all unique terms in an abstract.

word2vec

word2vec can be used to create word embeddings that, just like GloVe, enables us to measure the semantic similarity of words. By following the exact same steps as when processing the GloVe word vectors we created vector representations of abstracts by sum the vectors words.

The word2vec word vectors we used were pre-trained on the Google News data set. Details about the pre-trained word vectors are provided in Appendix G: Pre-trained Word Vectors. The features generated from these word vectors (from here on called word2vec features) were obtained by summing the word2vec vectors, per dimension, for all unique terms in an abstract.

Classification Using Word Embedding Features

To test the effectiveness of GloVe and word2vec features three approaches were implemented. The first approach is implemented using a basic classifier. The pipeline consists of the word embedding feature (either GloVe or word2vec features) and a Logistic Regression classifier[25].

Based on the experience from SemEval 2016 (see Appendix A: SemEval 2016 - IDI@NTNU) we also implemented a voting classifier, taking input from multiple classifiers. The voting scheme decides the class label by taking the maximum argument of the probabilities produced from the included classifiers. Two versions of the voting scheme were implemented: one employing voting between the word embedding classifier from approach one and the best tuned classifier, the other approach using the two best tuned classifiers plus the word embedding classifier.

Both approaches were implemented using GloVe features and word2vec features (separately and combined) resulting in 9 combinations.

[25] C=0.83

Convolutional Neural Network Trained on Characters

The implementation we have used for a CNN trained on characters is similar to the one described by Zhang et al. (2015). The implementation is adapted from the example provided by TensorFlow[26]. Changes were made regarding importation of data, number of classes, metric type (changed from accuracy to macro F-score) and early stopping criteria. The CNN only use the first 100 bytes of the abstract as data due to lack of processing power and time.

Early stopping is achieved by implementing a *ValidationMonitor*[27] using *skflow*[28]. Training is performed continuously for up to 100 steps times 10 (if not stopped early).

We refer the reader to referenced paper and link to implementation for a detailed description of the code.

Convolutional Neural Network Trained on Words

In addition to the CNN trained on characters, we tried a CNN trained on words. The implementation was provided by TensorFlow[29] and the same changes were made to adapt the code as for the previously described CNN. The main difference is that we use the 100 first words of each abstract instead of the 100 first characters.

Reccurent Neural Network Trained on Words

The last neural network is a RNN trained on words. The implementation was adapted by TensorFlow[30]. Same changes as mentioned above were made.

[26]`https://github.com/tensorflow/tensorflow/blob/master/tensorflow/examples/skflow/text_classification_character_cnn.py`, as of 2016-06-06

[27]*early_stopping_rounds*=200, *batch_size*=10

[28]See Table 3.4

[29]`https://github.com/tensorflow/tensorflow/blob/master/tensorflow/examples/skflow/text_classification_cnn.py`, as of 2016-06-06

[30]`https://github.com/tensorflow/tensorflow/blob/master/tensorflow/examples/skflow/text_classification.py`, as of 2016-06-06

Tools and Frameworks

The most essential tools used to implement the classifiers are listed in Table 3.4.

Framework	**Description**
Pandas[31]	Is an open source python library providing high-performance, easy-to-use data structures.
Scikit-Learn[32]	Provides simple and efficient tools for data mining and data analysis.
Scikit Flow[33]	A simplified interface for TensorFlow, to get people started on predictive analytics and data mining.
TensorFlow[34]	TensorFlow is an open source software library for numerical computation using data flow graphs. Developed for the purposes of conducting machine learning and deep neural networks research.
word2vec[35]	A Python interface to Google word2vec[36].

Table 3.4: Frameworks and tools utilized for the stance classification system.

3.4 Visualization

This section presents the model and implementation of the visualization component.

3.4.1 Model

Figure 3.9 illustrate the visualization process. Visualizations are based on data from the two previous components. The data is merged and stored in a database

[31] `http://pandas.pydata.org`, as of 2016-06-06
[32] `http://scikit-learn.org/`, as of 2016-06-06
[33] `https://github.com/tensorflow/tensorflow/tree/master/tensorflow/contrib/learn/python/learn`, as of 2016-06-06
[34] `https://www.tensorflow.org`, as of 2016-06-06
[35] `https://github.com/danielfrg/word2vec`, as of 2016-06-06
[36] `https://code.google.com/archive/p/word2vec/`, as of 2016-06-06

for easy access. The resulting database provides all the necessary information for the visualization component. After extracting data from the database the component generate a (predefined) set of graphical representations. The produced graphical representations are displayed in a single page web application. The produced graphics are interactive and will be used to conduct visual exploration.

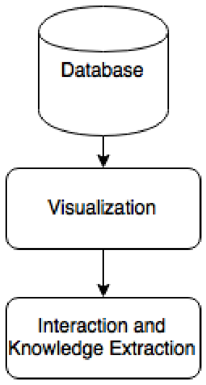


Figure 3.9: The visualization model.

3.4.2 Implementation

Back-End

We have chosen to use *MongoDB*[37] for data storage. This is not a database following the strict regime of column structure like a *SQL* database. Their web page describe *MongoDB* as: ”... an open-source, document database designed for ease of development and scaling”. A record in *MongoDB* is called a document, which is a structure composed of field and value pairs similar to what we find in a JSON object. Documents are stored in collections (not tables like in *SQL*). Our database consist of one collection containing TCP data and one collection with the scientific literature published after 2011.

We use *mLab*[38] to realize our database. *mLab* is a fully managed cloud database service that hosts *MongoDB* databases. The free of charge database option allows for up to $500MB$ of stored data. The server is set up using the the free application *Heroku*[39]. We use *Express*[40] as the server language along with

[37]`https://docs.mongodb.org/`, as of 2016-06-06
[38]`https://mlab.com/`, as of 2016-06-06
[39]`https://www.heroku.com/`, as of 2016-06-06
[40]`http://expressjs.com/`, as of 2016-06-06

Node.js[41].

The easy access of data is achieved by implementing an API. The API offered a predefined set of queries to the database. An overview of the available API calls is included in Table 3.5.

API Path	Description
/api/data	get all data
/api/data/:id	get data from a specific id
/api/data/stance	get all stances and corresponding ids
/api/stance/year/:stance	get count of a stance grouped by year for TCP data
/api/stance/year/new/:stance	get count of a stance grouped by year for new data
/api/visual/old/organization/	get TCP author affiliation data
/api/visual/new/organization/	get related data author affiliation

Table 3.5: API description.

Front-End

The visualization web application is based on a template[42] created by Cristian Trifan. The template provides a basic setup for a single web page application using tools such as *jQuery*, *RequireJS*, *Knockout*, *Bootstrap*, *Font Awesome*, and *Crossroads.js*. In addition to the tools from the template, other frameworks and libraries are utilized to create the visualizations. See Table 3.6 for an extensive list of these tools.

When it comes to visualization we have selected three visual representations of the data. The first is a bar chart showing the stance distribution per year. The second representation aims to explore subject-stance relations in climate science presented by a relational graph. The last graphic illustrates a world map with aggregated affiliation counts per country according to stance. The implementations are described below.

Bar Chart

The bar chart is the main visualization chosen for this project. The implementation is based on loading the data using HTTP requests to our API, then preprocess the obtained data to fit the *c3* charting library. *c3* uses *d3* to make con-

[41]`https://nodejs.org/en/`, as of 2016-06-06
[42]`http://spa-ko.crissdev.com/`, as of 2016-06-06

venient visualization templates. They are created for fast rendering and enables interactive capabilities by default (see Table 3.6 for description of the libraries). The bar chart illustrate how the class distribution of the climate articles has developed from 1991, when the first paper in TCP is dated, to 2016. The bar chart offers two representations of the data. One shows the number of articles per class per year and the other shows percentage distribution per year. In addition, different views separate TCP data and recent climate data thus making it easier explore details of each of the data sets.

Graph Relations

The visual exploration of the relationship between stance and common subjects in climate science is achieved by following three steps. First we extract the unique subjects from the meta-data using Python and the *Pandas* library. The subjects and articles are stored in CSV-files after extraction. The second step utilizes *Neo4j*. The unique subjects and the publications serve as nodes. Relationships between nodes are then composed by a query written in *Cypher*, *Neo4j*'s own query language. For every relationship, an edge is created. When edges are in place we export two data files: one node file and one edge file stored in CSV format. This is done as Graph Common utilize CSV-files for importing data in the final step. The final step is to import the node and edge files to the Graph Common platform. This three-step procedure is carried out for both data sets.

World Map

This visualization requires some pre-processing of data. It is performed using Python and *Pandas*. In this process country affiliations are extracted for each paper. The affiliations are then counted and stored in the database as country-stance-count triples.

Based on this data we implement a world map visualization using *amCharts*, a library for JavaScript. *amCharts* enables us to create a bubble map where the bubbles symbolize the number of author affiliations per country ordered by stance. This means that it is possible to observe, for example, how many authors affiliated with Norway that have published an article in favor of human-induced global warming. The countries were mapped by a predefined set of coordinates defined by *amCharts*.

Tools and Frameworks

In Table 3.6, we have listed all the libraries and frameworks used to create the single page web application and the visualizations.

Framework	Description
JQuery[43]	A very useful JavaScript library. It provides features like event handling, animation, and Ajax in a simple and easy-to-use API that works in several browsers.
RequireJS[44]	A JavaScript file and module loader. By using a modular script loader like RequireJS, we improve the speed and quality of your code.
Knockout[45]	Knockout offers some neat features when if comes to data bindings, UI refreshing, and dependency tracking.
Bootstrap[46]	A framework that simplifies front-end web development. It comes with a set of features and graphical components, and can be used to create solutions for all kinds of devices.
Font Awesome[47]	Font Awesome provides scalable vector icons that can be customized by the use of CSS.
Crossroads.js[48]	Crossroads.js is a routing library. It can be used to parse strings and to decide the actions to be taken given the recognized pattern. If used properly it can reduce code complexity by decoupling objects and also by abstracting navigation paths.
Gulp[49]	Gulp is a build system.
Jasmine[50]	Jamsine is a behavior-driven framework for testing in JavaScript. It makes it easy and intuitive to write tests.

[43] `http://jquery.com/`, as of 2016-06-06
[44] `http://requirejs.org/`, as of 2016-06-06
[45] `http://knockoutjs.com/`, as of 2016-06-06
[46] `http://getbootstrap.com/`, as of 2016-06-06
[47] `http://fontawesome.io/`, as of 2016-06-06
[48] `http://millermedeiros.github.io/crossroads.js/`, as of 2016-06-06
[49] `http://gulpjs.com/`, as of 2016-06-06
[50] `http://jasmine.github.io/`, as of 2016-06-06

D3[51]	D3.js is a JavaScript library for manipulating documents based on data. D3 helps you bring data to life using HTML, SVG, and CSS.
C3[52]	C3.js is a D3-based reusable chart library.
amCharts[53]	amCharts is an advanced charting library that offers plenty of data visualization options such as pie chart and maps.
Graph Commons[54]	Graph Commons is a collaborative 'network mapping' platform and a knowledge base of relationships.
Neo4j[55]	Neo4j is a highly scalable native graph database that leverages data relationships as first-class entities, helping enterprises build intelligent applications to meet today's evolving data challenges.

Table 3.6: List of front-end frameworks used.

[51]`https://d3js.org/`, as of 2016-06-06
[52]`http://c3js.org/`, as of 2016-06-06
[53]`https://www.amcharts.com`, as of 2016-06-06
[54]`https://graphcommons.com`, as of 2016-06-06
[55]`http://neo4j.com`, as of 2016-06-06

Chapter 4

Evaluation of Search & Information Retrieval

This chapter investigates our Goals and Research Questions presented in section 1.2 regarding Search & Information Retrieval. The experiments aim to measure the performance of different strategies and have been used to track improvement between implementations in order to determine the *best* implementation.

The chapter is divided in four sections. We will first introduce the specific goals and research questions explored. Following sections present the experiments conducted to assess the component performance for the the task in question along with the outcome. Discussion of our findings is provided in chapter 7.

4.1 Introduction

Experiments conducted on the topic of Search & Information Retrieval address multiple approaches of retrieving meta-data and abstracts for climate scientific literature.

We investigate Research question 2 of how relevant climate articles published after 2011 can be retrieved. Additionally, we describe experiments that aims to quantitatively measure the performance of our strategies for abstract and meta-data extraction. The meta-data is important as it could contribute to automatically classify papers. It is also essential to reach our third goal of uncovering how stance is related to external factors (if possible). This is achieved by answering Research question 4. The external factors can be anything from publication year to author affiliations. We define this kind of information as meta-data for published papers. We refer the reader to The Consensus Project (section 2.1) when references about TCP data are made.

4.2 Abstract and Meta-data Retrieval

Initially, our focus was on obtaining only the paper abstracts, however the focus shifted to retrieving meta-data. There are two reasons for this. Firstly, we observed that abstracts often were included in meta-data. Secondly, we got hand of all abstracts associated with TCP. This section evaluates the strategies used to collect abstracts and meta-data to supplement TCP data. Three methods to obtain meta-data have been tested and one for abstract retrieval. Experiments/results from this section that are significant, but not considered important enough to include here are placed in section E.1.

4.2.1 Setup

When searching for publication meta-data and abstracts, we used the data set provided by TCP[1]. This enabled us to verify that the retrieved abstracts or meta-data were in fact correct. Verification is achieved by comparing the title found in original data and from the retrieved data. The experiments are designed to measure the performance using *precision* and *recall*. Recall is, in information retrieval, the fraction of the documents that are relevant to the query that are successfully retrieved, while precision is the fraction of retrieved documents that are relevant to the query. In this case, the relevant documents are the documents found in the TCP data. The only information we have used to search for records are title and publication year.

Only one approach is created created specifically to collect abstracts. However, the methods for retrieving meta-data (from WoS and Scopus) often included abstracts and are therefore introduced as alternative approaches. The approaches used in these experiments are introduced below.

Approach 1: Crossref

The Crossref approach is based on a web crawler taking advantage of the Crossref API to retrieve meta-data from their database. The implementation is described in CrossRef: Meta-Data Retrieval (section 3.2.2) and the reader is encouraged to read up on the strategy to fully understand the experiment.

[1]More details of the data is in Appendix C: Data from TCP.

Approach 2: Scopus

Meta-data retrieval:
This approach explored the use of a web crawler to extract meta-data from the Scopus. The strategy exploit Scopus' functionality to download meta-data as CSV-files. The CSV-files often contained abstracts and is therefore an alternative to the abstract retrieval approach mentioned below. Implementation details for this strategy is provided in Scopus: Meta-Data Retrieval (section 3.2.2).

Abstract retrieval:
Abstracts were collected using the strategy described in Scopus: Abstract Retrieval (section 3.2.2). In short, the strategy is based on a web crawler scraping abstracts directly from Scopus' web site. The experiment was supposed to be extended to retrieve new (relevant) papers, but later findings revealed a more convenient way.

Approach 3: WoS

Meta-data Retrieval:
Using a python module, we extracted meta-data from the WoS database. Further details of the approach is found in Web of Science: Meta-Data Retrieval (section 3.2.2). Knowledge of the approach is of importance to understand the experiment.

Abstract Retrieval:
When collecting meta-data from WoS we noticed that most of the retrieved data also contained the abstract. Based on this observation, we decided to conduct a simple experiment to see how the recall for abstract retrieval using WoS compared to the one using Scopus.

4.2.2 Results

Meta-data Results

Table 4.1 shows the scores from retrieving meta-data. The results are evaluated in terms of precisions and recall. The approach based on Crossref shows decent results with 70.31 % recall and 76.81 % precision, but when compared to the other approaches, its performance proves to be poor. The Crossref search engine seems to return data regardless of search terms giving a low precision. The strategy based on the Scopus database outperforms Crossref. It shows an impressive precision of 95.98 %. However, the recall is not as impressive, only 76.96 %. This is still better than Crossref. The last approach, retrieving meta-data from WoS, outperformed both of the previous strategies by a large margin. The precision

Database	Precision		Recall	
Crossref	76.48 %	(8,397/10,980)	70.31 %	(8,397/11,942)
Scopus	95.98 %	(9,191/9,576)	76.96 %	(9,191/11,942)
Web of Science	95.97 %	(11,353/11,830)	95.07 %	(11,353/11,942)

Table 4.1: Results from three different scientific databases. The second column shows precision, while the third shows recall.

was approximately equal to Scopus ($\approx$ 0.01 % difference), but it achieved a significantly higher recall (95.67 %) than any of the other approaches. The good precision and recall makes this the best strategy for obtaining meta-data. We believe it is a good result, leaving only 589 records without additional meta-data. The records without meta data were distributed over all classes (11 *against*, 252 *in favor* and 326 *none*).

Abstracts Results

Table 4.2 shows the the number of abstracts retrieved along with recall for the approaches tested in this experiment. When scraping abstracts from Scopus, we managed to obtain just over two thirds (68.75 %) of the 11,942 found in the TCP data. This was the first approach and set the standard of what we at least could expect from later implementations for abstract retrieval. As we suspected, the abstracts obtained when retrieving meta-data from Scopus by downloading CSV-files matched perfectly with the abstracts retrieved when scraping directly from the web. As a result only one score for Scopus is included in Table 4.2. Choosing among the strategies using Scopus, we believe the one collecting CSV-files is best as it retrieves additionally meta-data with abstracts and is more robust than the one scraping abstracts directly from the web.

Database	Recall	
Scopus	68.75 %	(8,210/11,942)
WoS	92.06 %	(10,994/11,942)

Table 4.2: The success rate of retrieving abstracts. Abstracts were scraped from Scopus.

However, the most successful approach (by far) is to utilize WoS. From this database, we were able to obtain 92.06 % of the abstracts searched for. This is a strong result and superior to the approaches based on Scopus. It tells us that WoS contains more of the information needed and is the preferred option.

4.3 Retrieving Relevant Data

4.3.1 Setup

Using a python module we retrieve relevant, more recent, climate literature from the WoS database. This experiment aims to measure the recall for retrieval of new data following the strategy described in Web of Science: Relevant Paper Retrieval. This strategy relies on the results from approach three, described in section 4.2.1, as we retrieve relevant climate science literature published after 2011 by taking advantage of the unique WoS identifier collected as part of the related experiment.

Evaluating the performance of retrieval of relevant data is problematic as we do not have any data to compare with. However, we came up with an evaluation metric that allows us to measure the recall. By using the TCP publications dated 1991 to 2010 we searched for related papers published in 2011. For each of the papers $(1991-2010)$, we queried WoS for two related papers dated between *2011-01-01* and *2012-01-01*. The performance is measured by calculating the recall based on the number of papers retrieved out of the available publications from 2011 in the TCP data.

4.3.2 Results

The total amount of papers retrieved was $20,636$. After removing duplicates, $17,035$ papers remained. We examined how many of these papers we could find in TCP data published in 2011. TCP contained $1,624$ papers dated 2011. Our comparison found that the newly obtained relevant data contained 296 out of the $1,624$ resulting in a recall of just 18.23 %.

The related data was retrieved based on citations from the Web of Science literature database. Even though the achieved score was rather low, we do not believe the quality of the rest of the collected data was poor. The number of papers on climate change published in 2011 supersedes the number of articles included in TCP data, meaning only recall was measured. Returning almost a fifth of the records from TCP dated 2011 is, in our opinion, an indication that the strategy works.

	Relevant data after 2011			
	2 relevant		**8 relevant**	
Filtering property	**Result**	**Change**	**Result**	**Change**
Returned documents	21,073		83,507	
Abstract filtering	20,791	- 282	82,540	- 967
Duplicate filtering	2,835	- 17,956	10,141	- 72,399
Irrelevant subject filtering	2,619	- 216	9,359	- 782
Word "climate" filtering	**2,633**	+ 14	**9,418**	+ 59

Table 4.3: Statistics for relevant documents returned from Web of Science. The last row marked in bold show the final result.

4.4 Final Component Strategy

Although several approaches were implemented with decent results, the final system is based on the outstanding performance of the WoS database.

From Table 4.1 we know that we obtained meta-data for in total $11,353$ documents. Following the proposed strategy we queried for two records for every document out of the $11,353$. This could potentially return $22,706$ relevant documents. The actual number of retrieved documents and how many that we considered relevant can be seen in Table 4.3. The post-filtering process removed the majority of the records. Based on this we decided to collect another batch. In the new run we queried for eight related records for every one of the $11,353$ we have WoS identifiers for. This could potentially return $90,824$ documents.

The first batch from the WoS database consisted of $21,073$ documents of which $20,791$ contained abstracts. Second filtering step removed $17,956$ duplicates leaving $2,835$ articles. Third step eliminated 216 records based on irrelevant subjects[2] while the final processing step added 14 records which contained the term "climate" in its abstract, giving a final of $2,633$ relevant climate science records distributed over five years (2012-2016).

The second batch consisted of $83,507$ records of which $82,548$ contained abstracts. The duplicate filtering removed $72,399$ records reducing the number of papers to $10,149$. After removing 782 records with irrelevant subjects and adding 59 records containing the term "climate" in the abstract, the final result was $9,426$ relevant records from the years 2012 to 2016.

We believe the component performs as expected and retrieves a suitable

[2]See implementation details in section 3.2.2 regarding irrelevant subjects.

amount of data. We will evaluate the quality of the data as we go in the remaining chapters of this thesis[3].

[3]See section 5.10 and chapter 6.

Chapter 5

Evaluation of Stance Detection

This chapter investigates the Goals and Research Questions presented in section 1.2 regarding Stance Detection. By conducting a set of experiments we have collected quantitative data that aim to measure the performance of different classification strategies. The results have been used to track improvement between the selected implementations as a measure to determine the *best* implementation. The topmost strategy is used as our model to predict stance in recent literature in climate science.

The first section introduce the specific goals and research questions explored in this chapter. Following sections presents the conducted experiments in detail and their outcome. The last section describes the final system and the presents the predictions on new data. Further discussion is provided in chapter 7.

5.1 Introduction

This chapter explores the main goal to automatically classify stance in publications from TCP. More specifically, it investigates Research question 3; finding the most suitable machine learning model(s) for stance detection. The experiments investigate a variety of classifiers that explores different approaches to the problem of automatically detecting stance in scientific literature.

General Setup

The metric used to evaluate the classifiers is the harmonic average macro F-score. The average is based on all classes (*in favor*, *against*, and *none*). This metric is used in every experiment we have conducted if not explicitly stated otherwise.

When experiments are conducted, the classifiers will be trained, validated, and tested using data provided by[1]. The data set relies on the endorsement levels mentioned in Table 2.2 in The Consensus Project. The endorsement rating indicates to what degree the paper in question agrees with global warming being human-induced. The classes are obtained by dividing the endorsement levels into three categories:

1. *In favor* - Endorsement 1-3
2. *None* - Endorsement 4
3. *Against* - Endorsement 5-7

After the class conversion we divided the data into a training, validation, and test set. Each classifier will be trained on a segment of 52.5 % of the original data (the training set). The trained system is then validated on the validation set consisting of 22.5 % of the original data. Once we have obtained the validation scores from all experiments, we select the best performing one as our model. This model is in turn trained on both training and validation data. To determine the final performance of our system, the model will predict the labels of the test set, containing the remaining 25 % of the available data. Note that the splits were stratified, meaning that the class distribution of each data set (training, validation, and test) were equal. Table 5.1 provides a statistical overview of the data sets.

Property	**Percentage**	**Samples**	**Against**	**Favor**	**None**
Training set	52.5 %	6,270	41	2,046	4,183
Validation set	22.5 %	2,687	17	877	1,793
Training + Validation	75.0 %	8,957	58	2,923	5,966
Test set	25.0 %	2,986	19	973	1,991

Table 5.1: Summary of the data sets.

Development scores are obtained by using a 10-fold cross validation. Validation scores are achieved by training on training set and predicting on the validation set. Each of the following sections describe an experiment followed by the result.

To evaluate the results in greater detail, we included precision, recall and F1-score per class alongside the average macro F-score. All scores are expressed in percentage.

[1]See details on TCP data in Appendix C: Data from TCP.

We chose not to include space and time complexity as all experiments have been conducted using an off-the-shelf personal computer[2]. Experimental results not considered an improvement are listed in Additional Experiments For Chapter 5.

5.2 Baseline Using Dummy Classifiers

5.2.1 Setup

As stated in section 3.3, the dummy classifiers are supposed to provide us with the bear minimum performance for the system. We chose to use the *majority guess* classifier and *stratified* classifier for this purpose.

The majority guess classifier simply guesses the class label that occurs most frequently. The stratified classifier inspects the class distribution of the training data and performs a qualified guess where it maintains the class distribution. This is different as the former assumes all future data belongs to the same class.

5.2.2 Results

The outcome of classification with the dummy classifiers can be seen in Table 5.2. Majority guess achieved a validation score of 26.68 %, which is the same as the development score. This is not surprising as the distribution of both sets were the same and given the strategy it was quite straightforward to foresee. The table shows that all predictions were made toward the *none* class (the majority class). For this class the recall was 100 % and all scores for *against* and *favor* were zero.

The *stratified* dummy classifier returned a higher score than *majority guess.* With a score of 32.82 %, the increase is of over 6 %. The precision, recall and F1-scores reflect the strategy of the classifier given the data distribution. *Against* received zero as the class is barely present in the data, while *favor* is close to one third and *none* close to two thirds.

We consider the score of 32.82 % as an indication of the lower boundary of what we can expect from the system. Results below this line will not reported.

5.3 Baseline Using Learning Classifiers

5.3.1 Setup

Because of the naive nature of the dummy classifiers, we wanted to establish a stronger baseline using ”real” classifiers. The experiment was conducted for the following classifiers: Multinomial Naive Bayes (MNB), Bernoulli Naive Bayes

[2]Some of the optimizations took place on a server.

Dummy: Majority Guess

	Development			Validation		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	00.00	00.00	00.00	00.00	00.00	00.00
Favor	00.00	00.00	00.00	00.00	00.00	00.00
None	66.71	100.0	80.03	66.73	100.0	80.04
Macro F:			26.68			26.68

Dummy: Stratified

	Development			Validation		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	00.00	00.00	00.00	00.00	00.00	00.00
Favor	31.50	30.99	31.24	32.24	31.24	31.73
None	66.21	66.75	66.48	66.39	67.32	66.85
Macro F:			32.57			32.86

Table 5.2: The results of classification using the dummy classifier strategies majority guess and stratified. Macro F-score is based on *Favor*, *None* and *Against* stance.

(BNB), Support Vector Machine (SVM - scikit-learn's *SVC*), Linear SVM (scikit-learn's *LinearSVC)*, Logistic Regression, and Stochastic Gradient Descent. No substantial pre-processing steps were tested, meaning the features were only obtained using simple word unigrams of the abstracts.

5.3.2 Results

Table 5.3 and Table 5.4 present the results from the six classifiers. The scores indicate a significant boost in performance by utilizing *"real"* classifiers instead of the dummy versions. Stochastic Gradient Descent returned the lowest score with a macro F-score of 48.66 % (based on validation). Despite being the poorest classifier, it is an increase of 15 % compared to the best dummy classifier from Table 5.2. The highest score was obtained by Linear SVM with 54.23 % on validation.

The classifiers performed quite even around $48 - 49$ %. A common denominator seems to be the poor performance on the class *against*. This is expected and illustrates the problem caused by the skewed class distribution. MNB, BNB and

Multinomial NB

	Development			Validation		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	00.00	00.00	00.00	00.00	00.00	00.00
Favor	57.63	75.90	65.51	58.82	77.19	66.77
None	85.45	73.03	78.76	86.65	74.23	79.96
Macro F:			48.09			48.91

Bernoulli NB

	Development			Validation		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	00.00	00.00	00.00	00.00	00.00	00.00
Favor	60.28	68.23	64.01	62.84	71.15	66.74
None	82.70	78.17	80.37	84.47	79.81	82.08
Macro F:			48.13			49.60

Linear SVM

	Development			Validation		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	14.29	02.44	04.17	40.00	11.76	18.18
Favor	58.34	58.31	58.32	62.26	63.40	62.82
None	79.21	79.87	79.54	81.78	81.60	81.69
Macro F:			47.34			54.23

Table 5.3: Baseline results from out-of-the-box versions of the classifiers: Multinomial NB, Bernoulli NB, and Linear SVM.

SVM

	Development			Validation		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	11.54	07.32	08.96	11.11	05.88	07.69
Favor	58.44	58.36	58.40	61.86	63.63	62.73
None	79.27	79.61	79.44	81.87	81.09	81.48
Macro F:			48.93			50.63

Logistic Regression

	Development			Validation		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	25.00	02.44	04.44	00.00	00.00	00.00
Favor	64.14	58.31	61.09	66.23	62.83	64.48
None	80.07	84.34	82.15	82.09	84.89	83.47
Macro F:			49.23			49.32

Stochastic Gradient Descent

	Development			Validation		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	11.11	02.44	04.00	16.67	05.88	08.70
Favor	65.38	54.84	59.65	51.53	88.37	65.10
None	79.10	85.94	82.38	91.08	59.79	72.19
Macro F:			48.68			48.66

Table 5.4: Baseline results from out-of-the-box versions of the classifiers: SVM, Logistic Regression, and Stochastic Gradient Descent.

LR completely fail to predict any samples labeled against. The reason Linear SVM seems to come out on top is due to its high recall and precision toward *against*. Although the Linear SVM scored highest on validation with 54.23 %, we can observe a large gap between validation and development (47.34 %) and the score might not be 100 % representative.

5.4 Optimized Learning Classifiers

5.4.1 Setup

From previous experience (Appendix B: Official SemEval 2016 Paper), we know that machine learning classifiers are rather sensitive to parameter values. Based on this, we wanted to explore the effects of pre-processing and parameter tuning to optimize the baseline classifiers (not dummy).

A cross-validated grid search on the training data was executed, detecting the optimal pre-processing steps and parameters. Table 5.5 shows the best values for each of the six classifiers. Note that lemmatizing abstracts as pre-processing did not enhance performance for any of the classifiers and was therefore left out for further testing.

Classifier			**Pre-processing**			**Parameter Tuning**
	Token	**Analyzer**	**Type**	**Stop Words**	**Limit**	
Multinomial NB	Unigram Bigram Trigram	Word	Counts	'english'	50,000	Alpha=0.1 Fit Prior=True
Bernoulli NB	Bigram Trigram	Character	Term Frequency IDF=True	False	None	Alpha=0.1 Fit Prior=True
Linear SVM	Unigram Bigram	Word	Term Frequency IDF=False	False	None	C=1.178 Multi class=ovr
SVM	Unigram	Word	Term Frequency IDF=False	'english'	None	C=6.918 Kernel=Linear Decision Shape=ovo
Logistic Regression	Unigram Bigram Trigram	Word	Term Frequency IDF=False	False	None	C=22.759 Penalty=l2 Solver=lbfgs
SGD	Unigram Bigram Trigram	Character	Counts	'english'	None	Alpha=0.0001 Loss=Squared Hinge

Table 5.5: Results of parameter tuning for pre-processing steps and classifiers.

5.4.2 Results

The development and validation results from classification are presented in Table 5.6. Linear SVM and Stochastic Gradient Descent are not listed in detail because they did not stand out in particular. Linear SVM improved in development by over 7 %, but the validation score was lower than the non-optimized. SGD did improve both on development (+1.68 %) and validation (+3.64 %), but not compared to the competing classifiers. More details are included in Additional Experiments For Chapter 5.

The lowest performance in this experiment (based on validation score) was obtained by the Bernoulli NB classifier. It achieved an average macro F-score of 51.20 %, which is lower than the best non-optimized baseline score (54.23 %). However, it shows an improvement of 3.89 % from its non-optimized version.

At the other end of the ranking is the SVM classifier, which obtained an average macro F-score of 54.78 %, an increase of only 0.55 % from the best baseline. Although the improvement is quite low on validation score, we can observe a significant increase in development score. No classifier performed below 50 % on development, nor validation. Multinomial NB scored highest in development (58.78 %) and 54.54 % on validation, which can be a sign of overfitting.

Predicting *against* seems be the most varying aspect between the classifiers. Five out of six classifiers were able to classify *against*. The only classifier not being able was LR. The learners seem to classify *favor* and *none* fairly equal.

5.5 Up- and Down-Sampling

5.5.1 Setup

Section 2.6.2 introduced our highly skewed data set. One of the proposed solutions to handle imbalanced data sets was to utilize up- and down-sampling. The experiment aims to investigate the effects of up- and down-sampling applied with the best performed classifier so far.

Up- and down-sampling was tested using the training set in an attempt to even out class distribution. The experiment was conducted by changing up- and down-sampling ratios. According to Table 5.1, the *against* class was heavily underrepresented and therefore increased by duplicating the samples, while *favor* and *none* classes removed instances with individual rates to find the optimal solution.

It is important to note that when using up-sampling, the development score may be affected by the fact that identical samples are present in both training and testing folds. Therefore, the focus on the experimental outcome was to the validation set.

Multinomial NB

	Development			**Validation**		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	32.35	26.83	29.33	16.67	11.76	13.79
Favor	60.05	75.07	66.72	61.50	76.85	68.32
None	85.78	75.42	80.27	87.02	76.63	81.49
Macro F:			58.78			54.54

Bernoulli NB

	Development			**Validation**		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	13.85	21.95	16.98	05.00	05.88	05.41
Favor	59.91	74.58	66.45	62.77	70.35	66.34
None	85.46	74.73	79.73	87.31	71.78	78.79
Macro F:			54.39			51.20

SVM

	Development			**Validation**		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	41.67	12.20	18.87	33.33	11.76	17.39
Favor	62.59	60.02	61.28	64.14	64.65	64.40
None	80.49	82.67	81.57	82.47	82.65	82.56
Macro F:			53.90			54.78

Logistic Regression

	Development			**Validation**		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	00.00	00.00	00.00	00.00	00.00	00.00
Favor	68.19	69.26	68.72	70.13	72.29	71.20
None	84.30	84.46	84.38	85.98	85.50	85.74
Macro F:			51.03			52.31

Table 5.6: Tuned baseline results with classifiers: Multinomial NB, Bernoulli NB, SVM and Logistic Regression

SVM - Down-sampling

Validation

Stance	Prec	Recall	F1	Support
Against	35.71	29.41	32.26	17
Favor	69.16	45.27	54.72	877
None	77.23	90.41	83.30	1793
Macro F:			56.76	

Table 5.7: Down-sampling results with the best classifiers in tuned parameter experiment: SVM.

5.5.2 Results

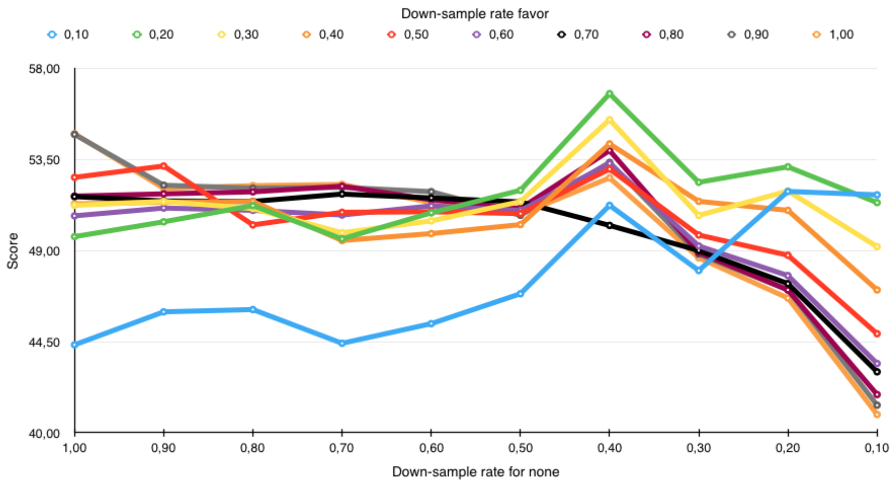
Training scores for this experiment are misleading as we used the data labeled *against* twice when performing up-sampling. Additionally, employing cross validation combined with down-sampling means that we alter the class distribution in test folds. This changes class distribution of test data, which in turn results in test folds that not are representative for the real task. The results can therefore not be used. However, the validation scores are valid and are being used for evaluation.

Figure 5.1 shows average macro F-scores when classifying validation data against down-sampling rate with and without up-sampling. The colored lines each represent a ratio for *favor* down-sampling. The X-axis shows the down-sample rate for the class *none* and the Y-axis display macro F-score.

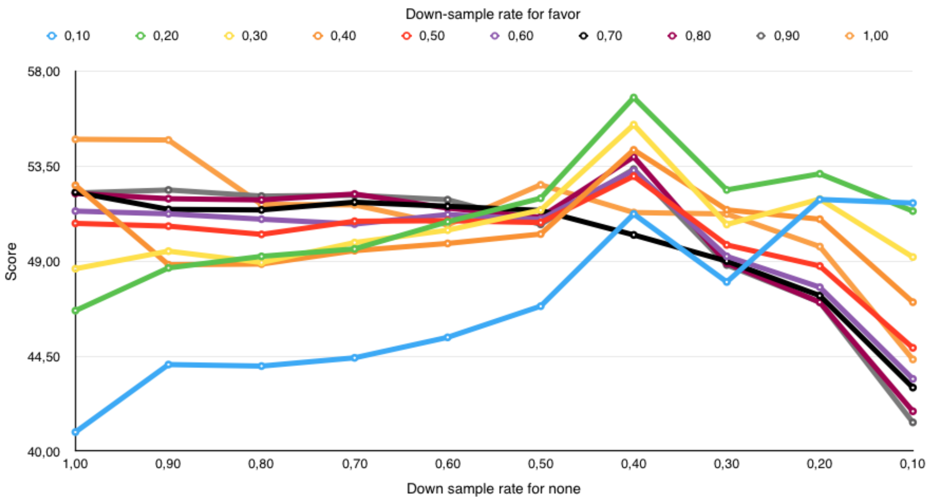
Examining Figure 5.1, we notice two things: first, the best model significantly enhance the performance of the base classifier. Second, there seems to be a sweet spot when 20 % labeled *in favor* and 40 % of the samples labeled *none* were used. Up-sampling has close to no impact on the classifiers performance.

Details on the best classifiers performance can be seen in Table 5.7. Only detailed results of the best down-sampling setup is included. This strategy has only experimented with the best classifier from Table 5.6, SVM. The table shows strong F1-scores for *against* and *none* labels, while *favor* drops almost 10 %. The final macro F-score of 56.76 % shows an increase of 2.53 % from the baseline.

Up-sampling itself only deteriorated the performance of the classifier.



(a) Down-sampling (only)



(b) Up- and down-sampling

Figure 5.1: Graphs showing average macro F-scores on validation data for various down-sampling rates with and without the use of up-sampling. Up-sampling (including samples for *against* twice) is only used in (b). Each color indicate the down-sample rate for the stance *favor*, while the X-axis show the down-sample rate for the stance *none*. Example: best validation score is obtained using 20 % of samples from the class *favor* and 40 % of samples from the class *none*.

5.6 Shallow Features

The meta-data collected in chapter 4 is exploited as features in this experiment. The 'ablation-strategy' is used to test the effect of the new shallow features. The strategy involves leaving out one (or more) features and see how it affects the performance. The features used are listed in the implementation Table 3.3. During the experiment, no particular feature gave any improvements. Two features were observed to significantly drop the performance. These were the author names and the organization countries. Because no combinations showed any improvement the results were listed in the Additional Experiments For Chapter 5.

5.7 Binary Classification

5.7.1 Setup

So far, we have only been working with multi-class classification. This experiment explores an alternative strategy using two-step binary classification.

The strategy divides classification into two steps; binary classification of *stance* versus *no stance (none)* followed by a classification of *favor* versus *against*. There are two benefits of using a two step binary classification strategy. In this case, it makes the class distribution (for step one) more even as we merge the two smallest classes[3]. The second advantage is that two classifiers can be optimized, potentially giving more accurate predictions. The disadvantage is that errors are made in in step one is propagated and affects the performance in step two.

As a result of the data flow from step one, there might be three classes (misclassified *none* samples) present in step two. However, the classification is still restricted to *in favor* and *against*. We encourage the reader to section 3.3.2 for more details.

The results are evaluated using the average macro F-score. The score for *none* is provided by the first classifier, while the scores for *in favor* and *against* are provided by the second step.

5.7.2 Results

Table 5.8 presents the results using SVM and Linear SVM. In the first step, Linear SVM outperforms SVM significantly. Linear SVM scores highest (on validation) with 78.88 %, while SVM only scores 65.99 %, as its recall for *stance* is terrible (only 24.39 %).

[3]Approximately ratio: 1 : 2.

SVM

Development

Step 1: Stance vs No Stance					**Step 2: Favor vs Against**					**Final**
Stance	Prec	Recall	F1	Support	Stance	Prec	Recall	F1	Support	F1
None	71.98	96.92	82.61	4183	None	00.00	00.00	00.00	129	82.61
Stance	79.78	24.39	37.36	2087	Favor	79.00	100.0	88.27	504	88.27
					Against	00.00	00.00	00.00	5	00.00
Macro F:			59.98	6270	Macro F:			44.14	638	Macro F: **56.96**

Validation

Step 1: Stance vs No Stance					**Step 2: Favor vs Against**					**Final**
Stance	Prec	Recall	F1	Support	Stance	Prec	Recall	F1	Support	F1
None	74.39	97.99	84.57	1793	None	00.00	00.00	00.00	36	84.57
Stance	88.92	32.33	47.42	894	Favor	87.08	100.0	93.09	504	93.09
					Against	00.00	00.00	00.00	6	00.00
Macro F:			65.99	2687	Macro F:			46.55	546	Macro F: **59.22**

Linear SVM

Development

Step 1: Stance vs No Stance					**Step 2: Favor vs Against**					**Final**
Stance	Prec	Recall	F1	Support	Stance	Prec	Recall	F1	Support	F1
None	84.92	83.86	84.39	4183	None	00.00	00.00	00.00	675	84.39
Stance	68.44	70.15	69.29	2087	Favor	68.30	100.0	81.17	1446	81.17
					Against	81.82	100.0	90.00	18	90.00
Macro F:			76.84	6270	Macro F:			85.59	2139	Macro F: **85.19**

Validation

Step 1: Stance vs No Stance					**Step 2: Favor vs Against**					**Final**
Stance	Prec	Recall	F1	Support	Stance	Prec	Recall	F1	Support	F1
None	86.68	84.55	85.60	1793	None	00.00	00.00	00.00	277	84.57
Stance	70.47	73.94	72.16	894	Favor	69.34	99.85	81.84	650	93.09
					Against	00.00	00.00	00.00	11	00.00
Macro F:			78.88	2687	Macro F:			40.92	938	Macro F: **55.81**

Table 5.8: Binary classification results in a two-step procedure: First detect stance vs no stance then classify favor vs against. Carried out using SVM and Linear SVM.

When we evaluate the second stage it is important to notice the variation of support between SVM and Linear SVM. In development, for the second step SVM only classified 638 samples while Linear SVM 2139.

Even though SVM had the highest performance on validation, scoring 59.22 %, it only classified 546 instances where 504 of them were *favor*. For this experiment, the SVM performed a majority guess of *favor*. The high recall for *none* leaves the second step with a substantially lower number of samples to predict. The macro F-score is, however, 4.99 % higher than the baseline.

Linear SVM shows the topmost performance for the first step with macro F-scores of 76.84 % and 78.88 % in development and validation respectively. The development score in step two is remarkable with a macro F of 85.59 %. The validation score however drops to 40.92 %, resulting in the final average macro F score on validation of 55.81 %.

It is also interesting that none of the classifiers were able to predict *against* on validation data.

We also investigated up- and down-sampling with the MNB classifier, but it did not give any substantial improvement and have not been included here. The MNB results are included in Additional Experiments For Chapter 5.

5.8 Word Embeddings

5.8.1 Setup

This experiment investigates the effects of using word2vec and GloVe features. word2vec features are created from word vectors trained on the Google News data. The GloVe features are created from GloVe word vectors trained on different corpora. We encourage the reader to look at the implementation details presented in section 3.3 and the overview of word vectors given in Appendix G: Pre-trained Word Vectors. We have tested three approaches for both the GloVe and word2vec features and three where both features were used. Training and validation is performed according to the procedure described in General Setup.

Approach 1: LR trained on word embeddings The first approach experimented with a classifier trained on the word embeddings. Logistic Regression was applied based on previous experience[4].

Approach 2 and 3: Soft Voting The second approach investigated a voting strategy. The voting was between the LR classifier and the best performing (based on validation scores) optimized baseline classifier which was SVM. The third

[4]See Appendix B: Official SemEval 2016 Paper.

approach explored the voting strategy between LR and the two best performing classifiers. These were SVM and MNB.

A final experiment is conducted with a combination of word2vec and GloVe features utilizing the approaches mentioned above.

5.8.2 Results

Only the best results from the word embeddings and the combination is included in Table 5.9. Results that we considered redundant are listed in Additional Experiments For Chapter 5.

GloVe features alone with a Logistic Regression classifier performed well, with a validation score of 57.38 % (appendix). However, combining the use of GloVe features (using Logistic Regression) and SVM significantly outperformed all models evaluated this far. As seen in Table 5.9, the model achieve a development score of 57.23 % and a validation score of 62.25 %. The high performance is partly because the trained model manage to obtain high F1-score for *against.* Together with strong performance on the predictions of *favor* and *none*, the result is 8.03 % higher than the best baseline.

word2vec alone performed worse (48.65 %) than GloVe and most of the other results are not included here. However, introducing SVM in a voting scheme boosted the score to 56.22 % (listed number two in Table 5.9). Including all three classifiers in a voting procedure reduced the performance slightly (56.08 %). These results are still better than the baseline performance, but significantly lower than result of utilizing GloVe and SVM.

Combining word2vec and GloVe in a voting strategy gave a decent results with validation score of 57.23 % (appendix). Adding the SVM in the voting scheme boosted the performance further up to 60.02 %, which is the second highest score to this point.

Based on these results, it is safe to say that utilizing word embeddings when detecting stance in climate science significantly boosts performance. Using Logistic Regression trained on GloVe features in a voting scheme with the optimized SVM have resulted in the best performing classifier so far. As a side note, it is interesting to see that all three versions (GloVe, word2vec, GloVe+word2vec) performed the best when in a voting scheme with the optimized classifier (SVM).

5.9 Artificial Neural Networks

We were curious of how ANNs would perform in the text classification setting. We therefore selected three implementations from Google's Tenserflow library for neural networks. We have tested CNN trained on either words or characters, and RNN trained on words.

Voting: GloVe(840B.300d) and SVM

	Development			Validation		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	14.46	29.27	19.35	23.33	41.18	29.79
Favor	68.33	68.23	68.28	71.25	71.49	71.37
None	84.46	83.67	84.06	85.99	85.22	85.60
Macro F:			57.23			62.25

Voting: Word2vec and SVM

	Development			Validation		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	13.70	24.39	17.54	16.67	23.53	19.51
Favor	66.04	60.26	63.02	67.59	63.74	65.61
None	81.09	83.93	82.49	82.57	84.55	83.55
Macro F:			54.35			56.22

GloVe, Word2vec and SVM

	Development			Validation		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	21.74	24.39	22.99	26.67	23.53	25.00
Favor	65.69	68.77	67.19	68.74	71.95	70.31
None	84.08	82.05	83.05	85.69	83.83	84.75
Macro F:			57.74			60.02

Table 5.9: Best performances from word embeddings.

Final System

Voting: Logistic Regression (glove.840B.300d) and SVM

	Development			**Validation**			**Test**		
Stance	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1
Against	14.46	29.27	19.35	23.33	41.18	29.79	16.98	47.37	25.00
Favor	68.33	68.23	68.28	71.25	71.49	71.37	72.25	70.09	71.15
None	84.46	83.67	84.06	85.99	85.22	85.60	85.95	85.74	85.84
Macro F:			57.23			62.25			60.67

Table 5.10: Test results of final system using a voting strategy between a Logistic Regression classifier trained on GloVe features and the optimized SVM.

The models are trained using training data, but rather than to track the development on development scores, we focused on loss. The training is considered complete when the minimized loss was consistent across multiple steps, based on validation data. As a result, only scores based on validation data is reported. If training is not stopped early, the ANNs will at some point adapt to the development data and start to overfit.

This experiment is included more as a curiosity than an approach we focused on. As the results we obtained were rather poor they are included on Additional Experiments For Chapter 5. We suspect that the poor performance is due to lack of data as ANNs normally require large data sets to be efficient. Poor performance may also be caused by an error in the setup.

5.10 Final System Strategy

5.10.1 Setup

The last experiment with stance detection is carried out by recreating the best performing model (determined by validation score). The experimental results include development, validation and test scores. The experiment aims to estimate how well the system performs on unseen data. The model consists of Logistic Regression trained on GloVe features along with the optimized SVM.

5.10.2 Results

The final system, seen in Table 5.10, scored 57.23 % on development, 62.25 % on validation and the final test score of 60.67 % which is in our opinion a strong result, especially given the data distribution. The final score is an improvement of 6.44 % over the best baseline of 54.23 % (validation score for Linear SVM). The final system classified *against* with F1-score of 25.00 %, which is close to the validation score of 29.79 %. The other two labels show strong performance with good F-scores. 71.15 % for *favor* and 85.84 % for *none*, which is on par with both development and validation values. The system shows almost no sign of overfitting (lower test score than validation), which is good.

Previous results showed (section 5.5) that down-sampling of the most represented classes in training data improved the performance of the classifier. However, this was not the case for the model used in the final system.

5.10.3 Predictions for Related data

The information retrieval component retrieved relevant climate papers published after 2011. As part of the evaluation of our final system, we performed predictions on this new, unlabeled data. Predictions were made for the two data sets presented in section 4.4. The final predictions are listed in Table 5.11.

Final System Predictions

	Small: 2633 records			**Large: 9418 records**		
Property	**Favor**	**Against**	**None**	**Favor**	**Against**	**None**
2012	48	1	571	175	4	2084
2013	41	3	541	174	11	2134
2014	45	4	557	144	9	2012
2015	59	1	594	156	7	1883
2016	14	1	153	64	1	560
Total	207	10	2416	713	32	8673

Table 5.11: Overview of predictions on the unlabeled data. Results are shown for both the small and large set collected by the information retrieval component.

If we compare the class distribution of the newly obtained data and TCP depicted in Figure 5.2, we can clearly see that the *none* predictions represent the majority class in all sets. However, the partition of *none* has increased significantly in new data and stands for 91.8 % of predictions in the small set and 92.1 % in the large. Given the classifiers performance on this label we believe the predictions are mostly correct. However, this could be a result of training on a skewed data

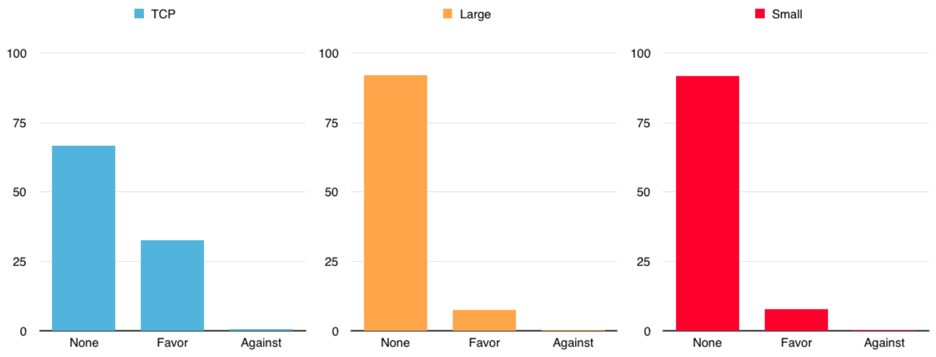


Figure 5.2: Class distribution for TCP and the new data sets (small/large).

set or it could tell us something about the quality of the retrieved data. Putting aside *none*, the predictions show 95.4 % and 95.7 % endorsement of the small and large data set, respectively. This is close to what Cook et al. reported. For the remainder of this thesis, we have chosen to use the largest data set.

We have extracted three abstracts and included them in Table E.29. These abstracts were analyzed to evaluate the relevance of publications and the prediction quality. We searched for evidence that could indicate the relevance and cues reasoning with the system's predicted label. As we are not domain experts, these evaluations were highly speculative.

The first abstract, labeled *against*, is in short summarized as a discussion of the sea level changes. It argue that sea levels have been stable over the past 50 years, if not slightly falling. In our opinion, this is highly relevant to climate change and well predicted. However, the second abstract (labeled *against*) discusses Einstein's gravitational theories. This does not seem too relevant.

The third abstract is labeled *in favor* and elaborates on meta-modeling approaches to reach EU's climate agreements in 2050. Based on this, the favor prediction seems reasonable.

Chapter 6

Evaluation of Visualization

This chapter investigates the Goals and Research Questions presented in section 1.2 regarding Visualization. The purpose of this chapter is to explore and evaluate the visual representations we have created. The first section gives a brief introduction while the following sections describe the experiments along with their results. A further discussion of our findings is provided in chapter 7.

6.1 Introduction

Visualizing of our findings is important. Our thesis title, *Interactive Exploration of Consensus in Climate Science*, emphasizes this. In section 2.8, we described how visual exploration can aid us in obtaining key insights to the data at hand. As humans tend to overlook facts hidden in large quantities of data, represented as numbers or texts, visualization is used as a tool to explore relations. It is challenging to conduct experiments by virtue of visualization rather than performing quantitative experiments. By utilizing the meta-data information obtained in chapter 4 along with the predictions of recent climate papers from chapter 5, we hope to uncover relations not previously known or back up already existing claims. The experiments described here delve into two of our main goals: discovering how stance is related to external factors and extrapolate TCP data to publications dated after 2011. This can be achieved by answering Research question 4 which investigate whether there exist relations between external factors, such as the author affiliation or geographical location, and the paper's stance. The second goal relies on the findings in Research question 2 and 3, as these provide us with the data needed to visualize and analyze change in recent climate papers.

Common for the experiments are the data sets. The experiments provide visualization for TCP and climate papers published after 2011. We have explored three different visual representations: a histogram illustrating the development of stance over the years, a graph displaying relationships between publication subjects and stance, and a bubble world map where author affiliations are mapped based on the paper's stance. The representations discriminate labels by dissimilar colors. *Favor* is colored green, *against* is colored red and *none* colored blue.

6.2 Histogram: Stance Devolopment Over Time

6.2.1 Setup

The Consensus Project presents the development of consensus in climate papers from 1991 to 2011. The agreement issuing that global warning is human-induced is overwhelming. In this experiment, we aim to examine the development in the years after 2011. We chose to use a stacked bar chart for this purpose, as it allows to examine the distribution per year. Because of the interactive representation, we can easily filter on the individual stances.

6.2.2 Result

Figure 6.1 includes two bar charts showing development of the consensus in climate science from 1991 up until today (2016). Figure 6.1a show the number of papers found in the data set divided into the three stances. The same data is used in Figure 6.1b, but instead percentages are shown. The percentages indicate the stance distribution (for each year).

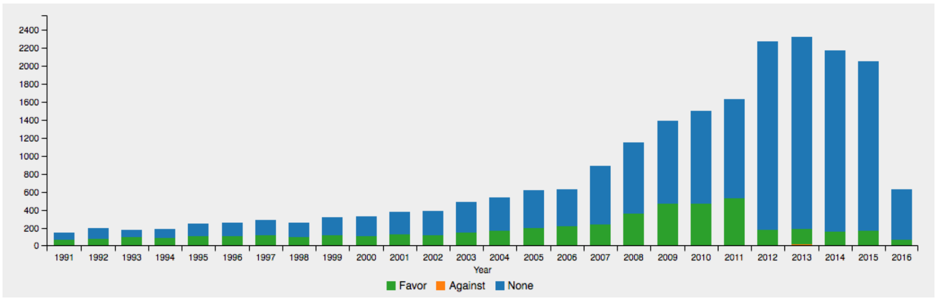
Figure 6.1a provides an overview of the data we are operating with. We can observe that the number of papers increase from 1991 to 2015. This can most likely be explained by the following scenarios:

1. The yearly production of scientific paper has increased in general since 1991. Papers are published digitally and are easily accessible opposed to before.

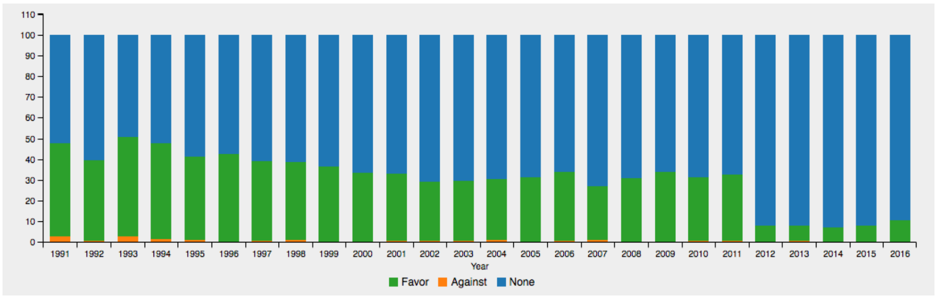
2. Global warming is an increasing controversial concern for humans, thus there is an acceleration of papers produced in recent years within this topic.

3. A combination (1) and (2).

We consider option (3) as the most likely scenario. In addition to the general growth in number of papers, we notice a significant increase of papers after 2011. This might be a result of the query strategy used and the amount of data we obtained in chapter 4. The low number of publications from 2016 is due to the fact that data collection took place in May that year, which severely restricts the



(a) Histogram illustrating stance development from 1991 to 2016 (counts).



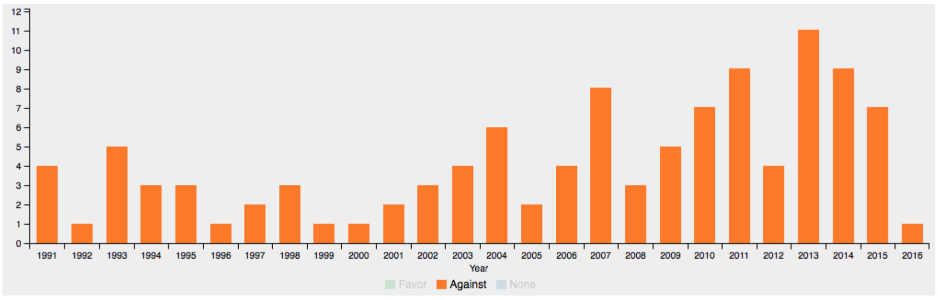
(b) Histogram illustrating stance development from 1991 to 2016 (percentage).

Figure 6.1: Histogram visualisations of stance distribution per year.

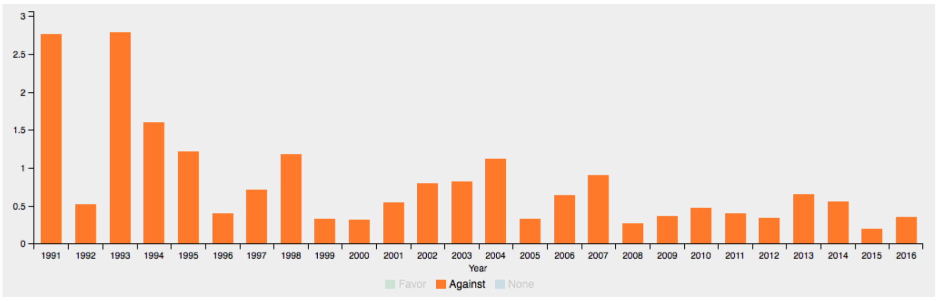
number of publications.

When investigating the development of the general consensus on global warming, it makes sense to look at percentage distributions per year. Figure 6.1b illustrates a decreasing trend of papers containing stance. Publications after 2011 support this observation. If we look at papers *in favor*, we notice a decrease of almost 30 %[1] from 1991 to 2011, dropping even further in the following years. For *against* (Figure 6.2) we can observe a decrease of 80 %[2] for the same period, also here it is dropping even further in more recent years. From 1991 to 2016 the change has been over 94.20 %[3]. If we combine this with the observed increase of papers with no stance, it suggests that there is a higher uncertainty of whether or not global warming is caused by humans. It could also indicate that human-induced global warming decrease in popularity as a research subject, but this seems unlikely.

[1]Drop from 44.83 % to 32.06 %.
[2]From 2.76 % to 0.55 %
[3]From 2.76 % to 0.16 %



(a) Histogram showing the development from 1991 to 2016 (counts).



(b) Histogram showing the development from 1991 to 2016 (percentage).

Figure 6.2: Histogram visualisations of stance *against* per year.

However, a drastic change in distribution from 2011 to 2012, as observed in Figure 6.1, is unlikely and could tell us something about the information retrieval results. More specifically, it could indicate that a rather high share of irrelevant publications are being retrieved. Although the change in distribution (all stances taken into consideration) is big, we can observe from Figure 6.3 that the relation between *favor* and *against* stays withing reasonable bounds. By only focusing on the publications taking a stance, we can observe that the majority still endorse anthropogenic global warming. In total 97.37 % of all publications from 1991 to 2016 (taking stance) endorse human-induced global warming. This is in line with what Cook et al. (2013) reported. This is an indication that it is more likely the quality of the retrieved data that causes the drastic change in distribution rather than the performance of the classifier (as we weakly suggested in section 5.10).

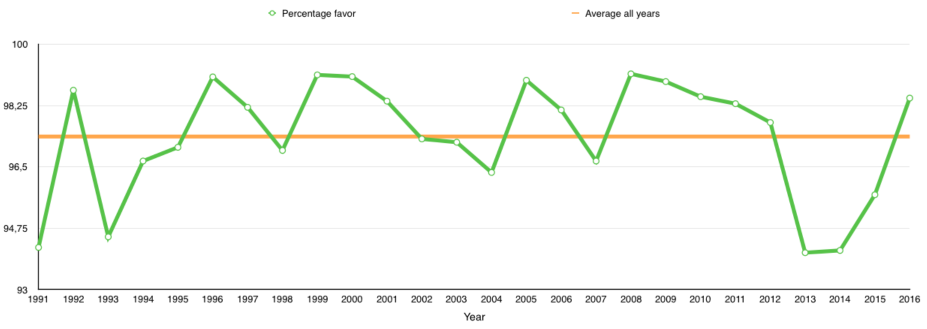


Figure 6.3: Percentage of publications endorsing human-induced global warming out of papers taking a stance (*none* disregarded) per year. Average is shown in orange.

6.3 Relational Graph: Exploring Stance vs Subjects

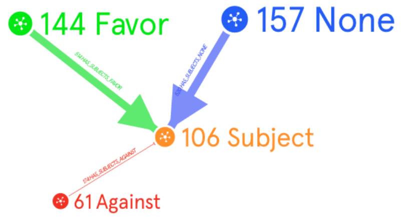
6.3.1 Setup

In this experiment, we investigate relations between paper subjects and stances. In particular, we were interested in *favor* versus *against*. Subjects were extracted from the data sets to create relationships between subjects and a paper's stance using *Neo4j*. Subjects and papers are represented as nodes and the edges are the paper's relation to subjects. The graph was exported to the *Graph Common* platform. Due to (we believe) a node restriction in Graph Common, not all papers were included. For this reason we extracted all *against* papers together with a random sample selection of *favor* and *none* labels across all years. Although we just utilized a sub-sample of the data, the graphs should still provide interesting insight to the subject-stance relations. The nodes illustrating paper stance (colored red, blue, or green) have fixed size while the size of the subject nodes (colored orange) are determined by the number of relations.

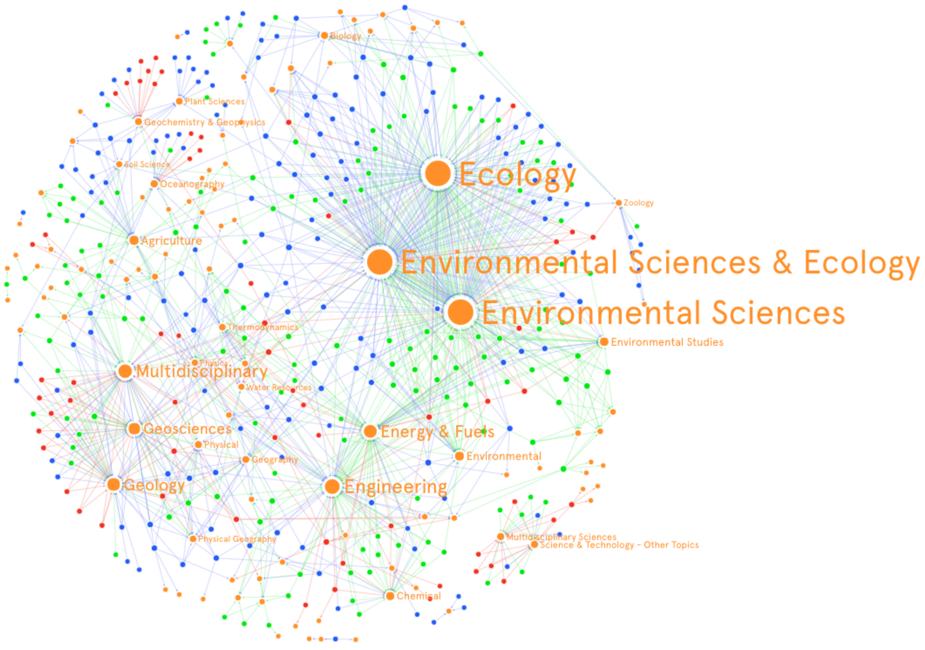
6.3.2 Result

A graph overview of TCP nodes is presented in Figure 6.4a, while the full graph is displayed in Figure 6.4b. The same graph for publications after 2011 is shown in Figure 6.5. The graphs contain a lot of information, making it troublesome to detect commonalities at first sight. We think of the graphs as controlled chaos.

The first thing we noticed in Figure 6.4 and Figure 6.5 was the size of the orange nodes for *Environmental Sciences*, *Environmental Sciences & Ecology* and *Ecology*. This imply that these subjects make up the major subjects. A selection



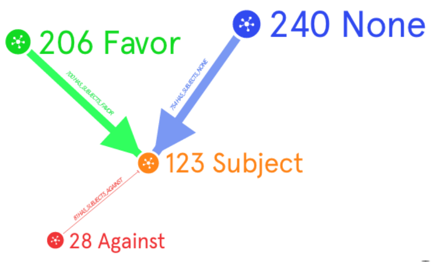
(a) Overview of nodes in the graph (362 paper nodes + 106 subject nodes).



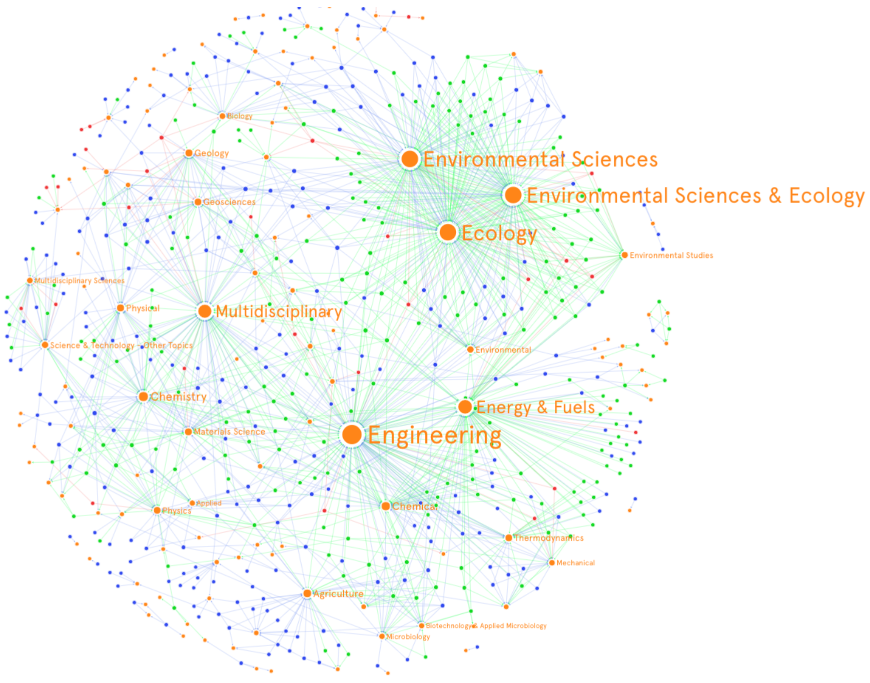
(b) Relational graph display common subjects from TCP data.

Figure 6.4: Relational graph providing an overview of subjects and paper stances in TCP data.

of the middle sized subjects were *Geology*, *Energy & Fuels*, *Multidisciplinary* and *Engineering*. All of the mentioned subjects seem (to us) like natural subjects to discuss climate science. The fact that both data sets seem to contain the same subjects (and with similar distributions) indicate that we have been successful in



(a) Overview of nodes in the graph (474 paper nodes + 123 subject nodes).

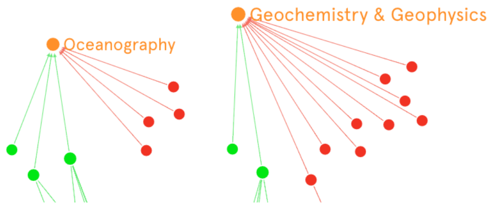


(b) Relational graph displaying common subjects in climate science after 2011.

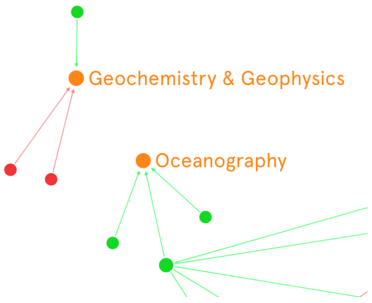
Figure 6.5: Relational graph providing an overview of subjects and paper stance in data after 2011.

retrieving representative data.

The green and blue nodes dominate in the graphs. This is in accordance with the general data distribution observed in previous experiments. The red nodes in the TCP graph seem to connect to the minor subjects in the outer circle of the graph (especially for Geochemistry & Geophysics), while it is ambitious to state any tendencies in the graph for recent climate publications as the distribution seem more random.



(a) Oceanography and Geochemistry & Geophysics subject from TCP data



(b) Oceanography and Geochemistry & Geophysics subject after 2011

Figure 6.6: Difference in class distribution for Oceanography and Geochemistry & Geophysics subjects before and after 2011.

Because of the challenging task of making observations with all data present, we dug deeper into the labels *favor* and *against*. In this procedure, we filtered out all *none*-nodes and isolated two subjects where *against*-nodes were the majority. Figure 6.6 display the two subjects with incoming edges. The selected subjects were Oceanography and Geochemistry & Geophysics. With these subjects, we explored the change between TCP data and data after 2011. In TCP Oceanography had 4 rejection (*against*) and 3 endorsement (*favor*) relations. Geochemistry & Geophysics had 9 rejections and 2 endorsements. Figure 6.6 illustrates a change for the Oceanography subject, where TCP data contained a majority of rejections while the new data showed 100 % endorsements. The second subject, Geochemistry & Geophysics, remained unchanged with a majority of rejections for both sets. (9 and 2 *against* while *favor* had 2 and 1, accordingly).

6.4 Bubble Map: Author Affiliation And Stance

6.4.1 Setup

Collection of geographical information, such as author affiliations for each paper, gave rise to this experiment. The setup is simply mapping geographical location of affiliations on a world map, ordered by stance. The bubble sizes represent the number of authors affiliated with the country.

6.4.2 Results

Figure 6.7 contains two visualizations of the geographic distribution of papers around the world, based on author affiliations. Our first observation is the distinction of developed and developing countries. North-America, Western Europe, Japan, China and Australia contribute with the majority of papers in climate science. The focus on global warming suggest higher concern in countries with advanced economics. We can also observe that the distribution (geographically) is very similar in both TCP and new data, which may indicate representative data was successfully obtained.

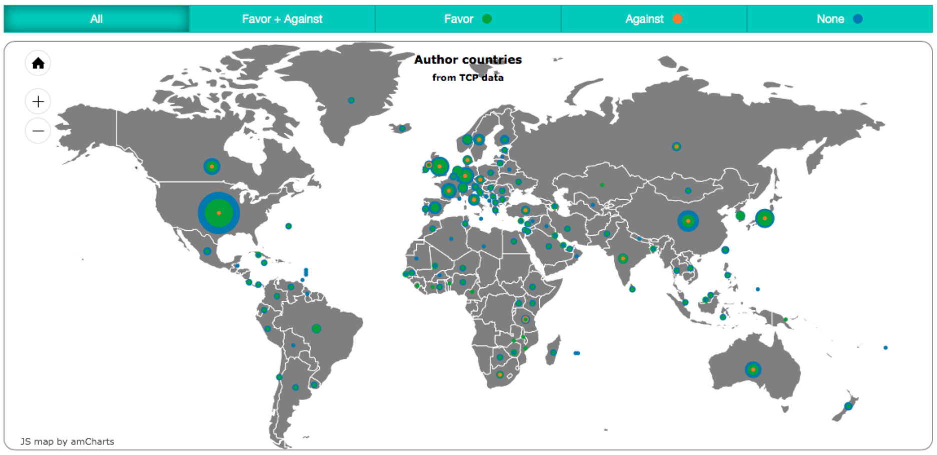
Eastern and Western Europe show a clear distinction in the amount of climate publications, however it is interesting to observe that none of the eastern countries reject human-induced global warming (except Czech Republic).

Another natural comparison to make is America against Europe. America had far more publications opposing human-induced global warming up until 2011, but in recent years the situation is opposite.

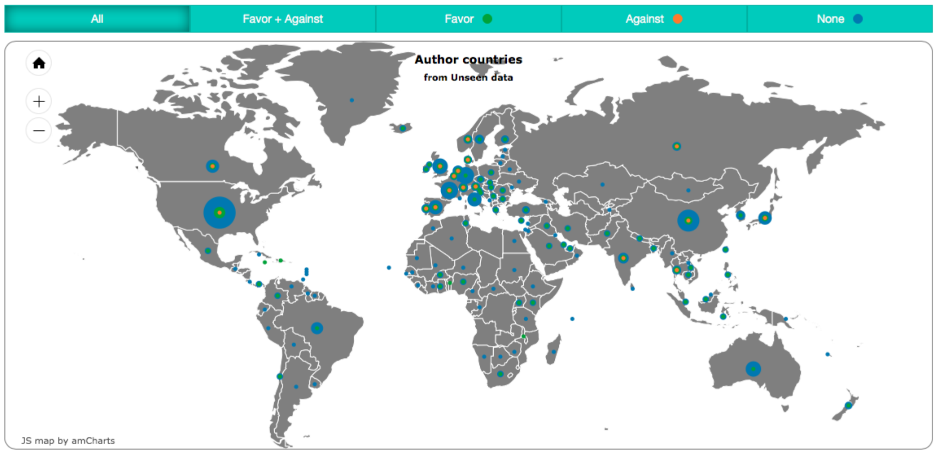
Stance in TCP data divide the Scandinavian countries (including Finland) in two. Norway and Finland with endorsing publications only, while Sweden and Denmark have both positions. For publications after 2011, Norway and Sweden switch sides.

The following analysis disregard the stance *none*. This is shown in Figure 6.8. In TCP data, we observe that nationalities like Norway, the Netherlands, Spain and Switzerland only have publications endorsing man-made climate change. However, after 2011 these countries have publications on both stances. The opposite can be found for Turkey and Italy.

An interesting observation is that North Ireland is the only country holding with more publications rejecting anthropogenic global warming than endorsing (in TCP). This changed in recent years with only endorsing papers.



(a) Mapping of author affiliations from TCP data.

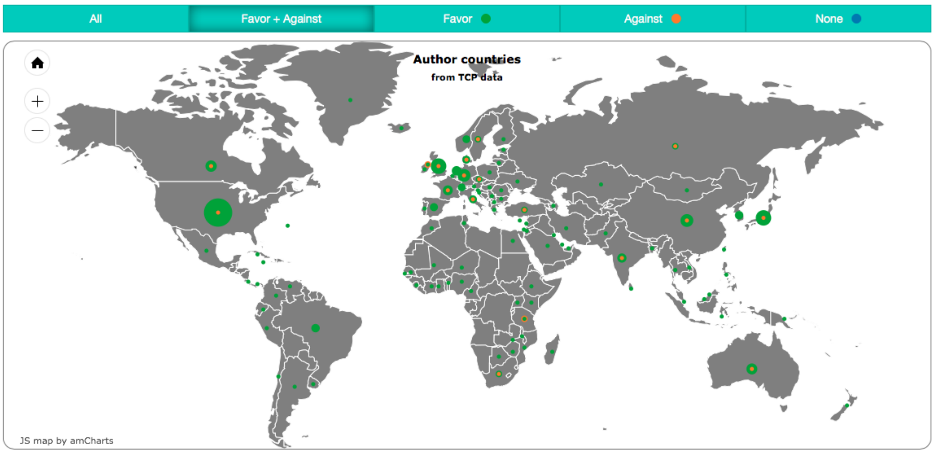


(b) Mapping of author affiliations from new, related, data.

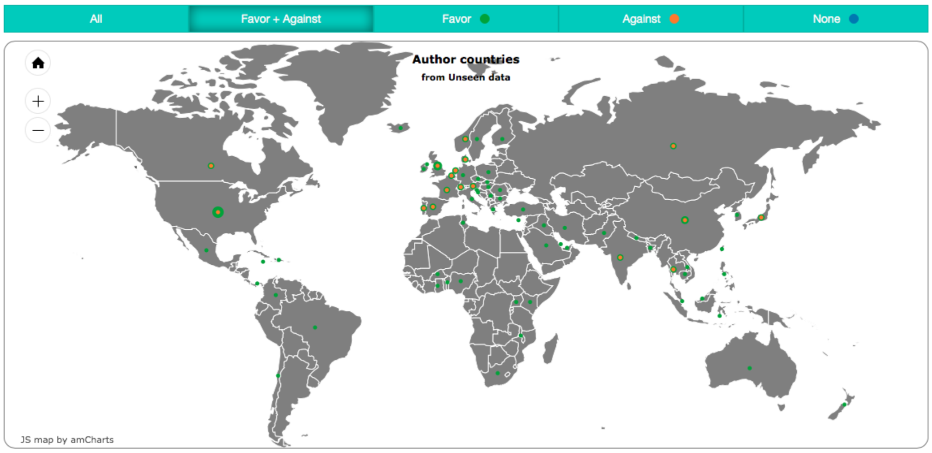
Figure 6.7: Mapping of author affiliations for all stances.

Tanzania, which currently is on the list of least developed countries[4] (LDC), is the only LDC in Africa opposing human-induced global warming (one opposing and one endorsing) in TCP (in newer data no publications take a position).

[4]`https://en.wikipedia.org/wiki/Least_developed_country`, as of 2016-06-06



(a) Mapping of author affiliations from TCP data.



(b) Mapping of author affiliations from new, related, data.

Figure 6.8: Mapping of author affiliations for stances *favor* and *against*.

Chapter 7

Discussion and Conclusion

This chapter provides discussion around the decisions made, the strategies used, the experiments conducted and the results obtained during the work on this thesis. It also provides a summary of main findings together with mistakes and failures. In the end, we present our thoughts on future work regarding the three components of our system: Search & Information Retrieval, Stance Detection and Visualization.

7.1 Summary of Results

Goal 1: *Automatically classify publications from TCP according to their stance held towards human-induced climate change.*

With the use of machine learning and state-of-the-art stance detection techniques, we have created a system that automatically classifies stance found in climate science papers. The system achieved a 60.67 % macro F-score on the test set.

Goal 2: *Extrapolate TCP data to publications after 2011.*

The system allows users to interactively explore the consensus on human-induced global warming from 1991 to 2016. In our opinion, there are no substantial changes in stance visible. Our system shows that 95.70 % of the publications after 2011 that holds a position endorse human-induced global warming. Which is close to what Cook et al. (2013) reported for the years 1991 to 2011.

Goal 3: *Discover how stance is related to external factors.*

The geographic map that visualizes stance distribution per country, as inferred through meta-data regarding author affiliations, suggest a distinction between developed and developing countries. The relational graphs weakly imply changes in pattern concerning subject-stance correlation before and after 2011.

RQ 1: *What is the state-of-the-art concerning stance detection?*

A literature study reveals that stance detection has become a popular topic in recent years. Multiple findings indicate that stance is difficult to predict. For instance, tasks like sentiment analysis in natural language processing is by itself not sufficient to predict stance properly. Some of the best performing methods create features on the syntactic and semantic structure of the documents. The systems often implement SVM learners and their performance range within 65 – 75 % measured in accuracy.

RQ 2: *How can relevant scientific articles be retrieved?*

Exploring alternatives to retrieve meta-data for the already known TCP data made us aware of possible approaches to fetch relevant papers related to climate science. The Web of Science database offers a method to collect relevant papers, defined in terms of common citations between publications. We established an evaluation strategy showing that approximately one out of five retrieved papers are relevant. However, a brief analysis of the obtained publications suggests that most are relevant.

RQ 3: *What type of machine learning model is well suited for training a (stance) classifier?*

Our experiments show that the discriminative SVM classifier works best for detecting stance. Combining the SVM classifier with a Logistic Regression classifier trained on GloVe features (word embeddings) in a voting scheme gave an even higher performance.

RQ 4: *Is it possible to detect relations between a paper's stance and external factors?*

Mapping of author affiliation by stance showed differences for continents and individual countries. The relational graph illustrated changes in subject-stance correlations before and after 2011. Such observations indicate that (weak) patterns are visible between stance and external factors.

7.1.1 Code and Online Resources

All code from our work is available on github[1]. Code is provided for the three major components. In addition, the single page web application we created is available online[2]. The application contains all visualizations, data sets and information about the team.

7.2 Discussion

7.2.1 Search & Information Retrieval

In the Search & Information Retrieval part of this thesis, we investigated multiple approaches to enrich the TCP data by retrieving meta-data. Additionally, we explored ways to obtain abstracts and meta-data for more recent literature within climate science.

Selecting a search strategy is an important, but challenging task. We wanted to match publications in TCP data to records located in literature databases. Our proposed solution is based on matching titles (within a similarity threshold) and year of publication[3]. The performance of this strategy relies heavily on the ranking algorithm used by the search engines. If a search engine ranks the desired paper second, it will not be obtained. Another problem arises if two records published in the same year have highly similar titles and therefore pass the threshold. As a result, the naive approach using only title and year may lead to wrong records being returned. Such errors could potentially have great impact on the system as mislabeled or missing records influence the learners performance negatively, but also give a misleading presentation of data in following visualizations. To further improve the search strategy, matching of author(s) could be included[4]. Another improvement could be to utilize multiple databases to cross-check the obtained results.

Based on our results it was not difficult to decide on the strategy for retrieving recent scientific publications on climate change. Web of Science delivered high precision and recall which significantly outperformed the other approaches. However, measuring the relevance of papers proved to be problematic. We tested an

[1]`https://github.com/petterasla/IECCS`, as of 2016-06-06
[2]`http://ieccs.herokuapp.com/`, 2016-06-06
[3]See implementation details in subsection 3.2.2.
[4]See Appendix C: Data from TCP for details on available data.

evaluation technique where publications in TCP prior to 2011 were used to search for applicable papers published in 2011[5]. The scoring is based on how many of the retrieved publications were identical to those found in TCP data from 2011. The achieved recall was approximately 18 % which means that roughly one out of five relevant papers were actually retrieved. The evaluation is quite restrictive and considered only papers found in TCP as relevant. Although TCP's work was extensive, they did not cover all published papers. The resulting data set is therefore not exhaustive (it contains only relevant, but not ALL relevant publications). However, eyeballing the retrieved publications suggests that most are relevant and therefore the precision is probably high. This is further backed by the similar distribution amongst the observed subjects in the data sets. As a result, we should consider 18 % as an lower boundary meaning that at least 18 % of the obtained recent publications are relevant. There is no way to determine the exact number of relevant publications retrieved. Results (presented in section 6.2) suggest that we have obtained a rather large percentage of publications with no stance. This might be an indication that a share of irrelevant papers was retrieved. Irrelevant publications will cause misleading results shown in later stages such as visualization. One way to improve the quality of recent climate papers could be to search for literature databases that offer similar services and find related papers from both. The results could be analyzed and only publications suggested by both databases could be included. Other options could be to redefine what we consider as a relevant publication (not basing this only on common citations).

7.2.2 Stance Detection

The main goal of this thesis was to develop a system that could detect stance in climate science literature. In chapter 5 we conducted extensive experiments to reach this goal.

Data sets have great influence on the performance of machine learning techniques. Multiple properties of the data set will affect the learners performance, such as the data distribution, noisiness and the complexity of the model relative to the data size. The data set we experimented with came from The Consensus Project. TCP manually labeled climate papers based on stance. In total, the data set contained $11,942$ records. In the context of machine learning, it is not an enormous data set. But it has a highly skewed class distribution: only 0.7 % (77 records) were labeled *against*[6]. A possible result of a such distribution is weak perfor-

[5]See experiment details in section 4.3.
[6]See illustration of the distribution in Figure 2.10.

mance on the minority class. We explored the use of up- and down-sampling to even out the distribution. The technique showed promising results but did not improve the performance when applied to the final system. Further testing could reveal that a different set of down-sampling rates would be beneficial for the final system. An alternative approach is to search for additional data to increase the size of the minority class, as this could help improve performance. New problems presumably arise as there are substantially fewer papers rejecting man-made global warming. In Appendix A: SemEval 2016 - IDI@NTNU, we describe experiments conducted to test the effect of additional training samples using label propagation. The results were mixed and we could not conclude that the extra data did, in fact, improve performance. However, it is possible that this technique could be more beneficial for our system.

Our literature study showed that performance of stance classification systems range approximately from 65 to 75 %. These systems are mainly trained on data sets from social medias. In comparison, our system predict stance in scientific literature on climate change with a macro F-score of 60.67 %. Analyzing whether or not this performance should be considered good is difficult as we did not discover any systems suitable for comparison. In addition, earlier works often employ the accuracy metric instead of macro F-score we have used. The approach we consider most similar to our system is implemented by Mohammad et al., (2016)[7]. Their data was gathered from Twitter (with informal language) on multiple topics. However, one topic considers climate change. The class distribution is almost equal, containing few rejections and larger amounts containing endorsements and no stance. Their system achieved a 43.8 % macro F-score on the climate topic[8]. Compared to this performance our system appears significantly better.

Our system was developed with the intention of detecting the position held toward human-induced climate change. Global warming is however not the only controversial topic discussed in news and on social media platforms like Twitter. Could our system be generalized to other domains? As our prediction model is built using supervised methods, it is highly dependent on labeled data (see section 5.1). To generalize to new fields, a data set similar to the one provided by TCP is required. In addition, the models we have introduced are optimized based on the TCP data. New optimizations are required and still there is no way to know how the performance will be, it can be better or worse. All this aside we believe that with appropriately annotated data to the particular field along with optimizations and creation of new GloVe/word2vec features it is possible

[7]This paper presents the data set utilized in Appendix A: SemEval 2016 - IDI@NTNU.

[8]Our system, proposed in Appendix B: Official SemEval 2016 Paper, scores significantly better on this topic.

to apply our system to new domains. Previous experience with comparable subjects and a similar solution supports this claim (see Appendix A: SemEval 2016 - IDI@NTNU and Appendix B: Official SemEval 2016 Paper for further details).

7.2.3 Visualization

The Visualization component investigated three different visual representations provided in an attempt to observe the change in consensus. It also provides tools to study the relation between stance and author affiliations, and common subjects relative to stance before and after 2011.

Evaluating the visual representations is an observational task compared to evaluating the experiment-and-test scores obtained for the stance detection component. While scores are definite, observations may differ from person to person. Our findings are based on subjective observations. We have tried to discover trends and patterns in the representations. This approach might have missed some aspects because we simply overlooked them. The evaluation of results could disagree with other readers or even misinterpreted. We therefore consider it important that what we have created is available for others to explore and interpret. In addition to the observational differences there is a risk of misleading data being presented because of errors from information retrieval and stance detection. Due to time restrictions, we have only included three visual representations. This could leave important patterns undiscovered. These may have been observed using other visual tools. Deeper analysis of the properties obtained from the meta-data could provide additional insight into the relation between external factors and stance. We leave it to future work (described in section 7.3) to provide ideas for new charts and graphs that could aid further research.

7.3 Future Work

We are pleased with what we have accomplished. However, there is always something to improve. In this section, we provide some ideas and suggestions for further work. Our thoughts include everything from new methods to additional features that can be utilized for system improvements.

Search & Information Retrieval

The Search & Information Retrieval component achieved excellent results when enriching TCP data. However, further validation of the retrieved data could be beneficial to decrease potential errors.

- We suggest extra post-processing steps to assure that new papers are suitable. Alternatively, the current filtering strategy could be further optimized.

- In addition to post-filtering, other strategies to find more recent climate publications should be explored. This could include using a set of databases, only retrieving papers suggested by more than one.

7.3.1 Stance Detection

- Our system performs classifications based on the paper abstracts. We suggest to explore the effects of using the full text of articles instead. This would include retrieving complete publications and possibly extracting text from PDFs, but could enrich the model with more information that has the potential to improve results.

- Our experiments covered a wide range of machine learning classifiers and different strategies, but we did not dig deep into the syntactic, semantic and rhetorical structure of the abstracts. Work for the future could be to create a system that extracts features on the basis of POS tags, syntactic trees, semantic roles or the rhetorical structure.

- The experiments conducted with neural networks for detecting stance were limited and results rather poor. It would be interesting to see if, by optimizing existing implementations or by customizing new networks, classification could be improved.

- The data was highly skewed. Searching for extra data (in multiple literature databases) to even the distribution could lead to a more balanced data set and most likely higher performance. Labeling new data using for example label propagation is also an approach that could be explored in an attempt to even out the data distribution.

- Investigate the effects of a feature based on the title explicitly mentioning global warming (or other related terms).

- Perform more extensive grid searches for further optimizations of the classifiers used.

7.3.2 Visualization

Interactive exploration of papers in climate science opens numerous options when the proper data is accessible.

– We chose three representations exploring different dimensions. For future work, we suggest other dimensions and new visual representations to extend the available tools for analysis.

– In the current implementation, only a subset of the data is used to investigate relations between subjects and stance using *Graph Common*. We would like to extend this to include all publications. We are also excited by their time line feature. It would be interesting to apply this to enable easy exploration of changes year by year.

7.3.3 Web Application

The web application we created is available for the public[9]. It has a lot of potential for visualization. Some features that could be considered for future work include:

– Provide a service for online classification of abstracts: an abstract is uploaded and a prediction is shown in the browser.

– Allow for uploading papers that have been manually labeled.

– Make data sets available for search, showing abstract, meta data and label.

[9]`http://ieccs.herokuapp.com/`, as of 2016-06-06

Bibliography

Alilla, R., Epifani, C., and Dal Monte, G. (2013). The blacker elder plant as an indicator of climate change: analysis of time series and verification of the iphen phenological model. *ITALIAN JOURNAL OF AGROMETEOROLOGY-RIVISTA ITALIANA DI AGROMETEOROLOGIA*, 18(2):56–64.

AlShaari, M. A. (2014). Text documents classification using word intersections. *IACSIT International Journal of Engineering and Technology*, 6(2):119.

Anderegg, W. R., Prall, J. W., Harold, J., and Schneider, S. H. (2010). Expert credibility in climate change. *Proceedings of the National Academy of Sciences*, 107(27):12107–12109.

Babonneau, F., Haurie, A., and Vielle, M. (2015). Assessment of balanced burden-sharing in the 2050 eu climate/energy roadmap: a metamodeling approach. *Climatic Change*, pages 1–15.

Batista, G. E., Carvalho, A. C., and Monard, M. C. (2000). Applying one-sided selection to unbalanced datasets. In *MICAI 2000: Advances in Artificial Intelligence*, pages 315–325. Springer.

Berglund, A., Boag, S., Chamberlin, D., Fernández, M. F., Kay, M., Robie, J., and Siméon, J. (2003). Xml path language (xpath). *World Wide Web Consortium (W3C)*.

Biber, D. (2004). Historical patterns for the grammatical marking of stance: A cross-register comparison. *Journal of Historical Pragmatics*, 5(1):107–136.

Biber, D. and Finegan, E. (1989). Styles of stance in english: Lexical and grammatical marking of evidentiality and affect. *Text - Interdisciplinary journal for the study of discourse*, 9(1):93–124.

Biron, P., Malhotra, A., Consortium, W. W. W., et al. (2004). Xml schema part 2: Datatypes. *World Wide Web Consortium Recommendation REC-xmlschema-2-20041028*.

Bordes, A., Bottou, L., and Gallinari, P. (2009). Sgd-qn: Careful quasi-newton stochastic gradient descent. *The Journal of Machine Learning Research*, 10:1737–1754.

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer.

Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., and Yergeau, F. (1998). Extensible markup language (xml). *World Wide Web Consortium Recommendation REC-xml-19980210.*, 16.

Campbell, C. (2002). Kernel methods: A survey of current techniques. *Neurocomputing*, 48(1):63–84.

Carus, A. B. (1999). Method and apparatus for improved tokenization of natural language text. US Patent 5,890,103.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. *Journal of artificial intelligence research*, pages 321–357.

Chindamo, M., Allwood, J., and Ahlsen, E. (2012). Some suggestions for the study of stance in communication. In *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)*, pages 617–622. IEEE.

Chowdhury, G. G. (2003). Natural language processing. *Annual review of information science and technology*, 37(1):51–89.

Christensen, E., Curbera, F., Meredith, G., Weerawarana, S., et al. (2001). Web services description language (wsdl) 1.1.

Cook, J., Nuccitelli, D., Green, S. A., Richardson, M., Winkler, B., Painting, R., Way, R., Jacobs, P., and Skuce, A. (2013). Quantifying the consensus on anthropogenic global warming in the scientific literature. *Environmental Research Letters*, 8(2):024024.

Dayhoff, J. E. and DeLeo, J. M. (2001). Artificial neural networks. *Cancer*, 91(S8):1615–1635.

Domingos, P. (2012). A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87.

Esuli, A. and Sebastiani, F. (2006). Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of LREC*, volume 6, pages 417–422. Citeseer.

Fallside, D. C. and Walmsley, P. (2004). Xml schema part 0: Primer second edition. *W3C recommendation*, 16.

Faulkner, A. (2014). Automated classification of stance in student essays: An approach using stance target information and the wikipedia link-based measure. *Proceedings of the Twenty-Seventh International Florida Artificial Intelligence Research Society Conference.*

Fausett, L. (1994). Fundamentals of neural networks: architectures, algorithms, and applications.

Fawcett, P. J. and Barron, E. J. (1991). A geological perspective on climatic change: Computer simulation of ancient climates. *Geoscience Canada*, 18(3).

Friedman, J. H. (1998). Data mining and statistics: What's the connection? *Computing Science and Statistics*, 29(1):3–9.

Gallo, J. M., Bueno, J., and Schuchardt, U. (2014). Catalytic transformations of ethanol for biorefineries. *Journal of the Brazilian Chemical Society*, 25(12):2229–2243.

Goldberg, Y. and Levy, O. (2014). word2vec explained: Deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722.*

Goller, C., Löning, J., Will, T., and Wolff, W. (2000). Automatic document classification-a thorough evaluation of various methods. *ISI*, 2000:145–162.

Goutte, C. and Gaussier, E. (2005). A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In *Advances in information retrieval*, pages 345–359. Springer.

Hagen, M., Potthast, M., Büchner, M., and Stein, B. (2015). Twitter sentiment detection via ensemble classification using averaged confidence scores. In *Advances in Information Retrieval*, pages 741–754. Springer.

Hasan, K. S. and Ng, V. (2013). Frame semantics for stance classification. *CoNLL-2013*, 124.

Hatzivassiloglou, V. and McKeown, K. R. (1997). Predicting the semantic orientation of adjectives. In *Proceedings of the 35th annual meeting of the association for computational linguistics and eighth conference of the european chapter of the association for computational linguistics*, pages 174–181. Association for Computational Linguistics.

Hatzivassiloglou, V. and Wiebe, J. M. (2000). Effects of adjective orientation and gradability on sentence subjectivity. In *Proceedings of the 18th conference on Computational linguistics-Volume 1*, pages 299–305. Association for Computational Linguistics.

Hecht-Nielsen, R. (1989). Theory of the backpropagation neural network. In *Neural Networks, 1989. IJCNN., International Joint Conference on*, pages 593–605. IEEE.

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

Hopfield, J. J. (1988). Artificial neural networks. *Circuits and Devices Magazine, IEEE*, 4(5):3–10.

Hosmer Jr, D. W. and Lemeshow, S. (2004). *Applied Logistic Regression*. John Wiley & Sons.

Hu, M. and Liu, B. (2004). Mining opinion features in customer reviews. In *AAAI*, volume 4, pages 755–760.

Hutto, C. J. and Gilbert, E. (2014). Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth International Conference on Weblogs and Social Media (ICWSM-14)*.

Idreos, S., Papaemmanouil, O., and Chaudhuri, S. (2015). Overview of data exploration techniques. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 277–281. ACM.

ISO, R. R. (1986). Information processing-text and office systems-standard generalized markup language (sgml).

Jaiswal, B. (1999). Automatic document classification. *DESIDOC Bulletin of Information Technology*, 19(3):23–28.

Jeni, L. A., Cohn, J. F., and De La Torre, F. (2013). Facing imbalanced data–recommendations for the use of performance metrics. In *Affective Computing and Intelligent Interaction (ACII), 2013 Humaine Association Conference on*, pages 245–251. IEEE.

Jordan, A. (2002). On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *Advances in neural information processing systems*, 14:841.

Keim, D. et al. (2002). Information visualization and visual data mining. *Visualization and Computer Graphics, IEEE Transactions on*, 8(1):1–8.

Keim, D. A. (2001). Visual exploration of large data sets. *Communications of the ACM*, 44(8):38–44.

Kubat, M., Matwin, S., et al. (1997). Addressing the curse of imbalanced training sets: One-sided selection. In *ICML*, volume 97, pages 179–186. Nashville, USA.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551.

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

Liddy, E. D. (1998). Enhanced text retrieval using natural language processing. *Bulletin of the American Society for Information Science and Technology*, 24(4):14–16.

Liu, B. (2010). Sentiment analysis and subjectivity. *Handbook of natural language processing*, 2:627–666.

Liu, B. (2012). Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, 5(1):1–167.

Lohr, S. (2012). The age of big data. *The New York Times*, 11.

Madzarov, G. and Gjorgjevikj, D. (2009). Multi-class classification using support vector machines in decision tree architecture. In *EUROCON 2009, EUROCON'09. IEEE*, pages 288–295. IEEE.

Manning, C. D., Raghavan, P., Schütze, H., et al. (2008). *Introduction to Information Retrieval*, volume 1. Cambridge university press Cambridge.

McCallum, A., Nigam, K., et al. (1998). A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 752, pages 41–48. Citeseer.

Medsker, L. and Jain, L. (2001). Recurrent neural networks. *Design and Applications.*

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781.*

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Miller, G. A. (1995). Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41.

Miller, H. J. and Han, J. (2009). *Geographic Data Mining and Knowledge Discovery*. CRC Press.

Mitchell, J. and Lapata, M. (2010). Composition in distributional models of semantics. *Cognitive science*, 34(8):1388–1429.

Mohammad, S. M., Sobhani, P., and Kiritchenko, S. (2016). Stance and sentiment in tweets. *arXiv preprint arXiv:1605.01655*.

Monard, M. C. and Batista, G. E. (2002). Learning with skewed class distributions. *Advances in Logic, Artificial Intelligence, and Robotics: LAPTEC 2002*, 85:173.

Najork, M. and Heydon, A. (2002). *High-performance Web Crawling*. Springer.

Neri, F., Aliprandi, C., Capeci, F., Cuadros, M., and By, T. (2012). Sentiment analysis on social media. In *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012)*, pages 919–926. IEEE Computer Society.

Niu, Y., Zhu, X., Li, J., and Hirst, G. (2005). Analysis of polarity information in medical text. In *AMIA Annual Symposium Proceedings*, volume 2005, page 570. American Medical Informatics Association.

Ochs, E. (1996). Linguistic resources for socializing humanity. *Rethinking Linguistic Relativity*, pages 407–437.

O'Shea, K. and Nash, R. (2015). An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*.

Østman, B., Lin, R., and Adami, C. (2014). Trade-offs drive resource specialization and the gradual establishment of ecotypes. *BMC evolutionary biology*, 14(1):1.

Pang, B. and Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, 2(1-2):1–135.

Pang, B., Lee, L., and Vaithyanathan, S. (2002). 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.

Parker, R., Graff, D., Kong, J., Chen, K., and Maeda, K. (2011). English gigaword fifth edition ldc2011t07. dvd. *Philadelphia: Linguistic Data Consortium.*

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Peron, R. (2014). Testing general relativistic predictions with the lageos satellites. *Advances in High Energy Physics*, 2014.

Provost, F. (2000). Machine learning from imbalanced data sets 101. In *Proceedings of the AAAI'2000 workshop on imbalanced data sets*, pages 1–3.

Rajaraman, A., Ullman, J. D., Ullman, J. D., and Ullman, J. D. (2012). *Mining of massive datasets*, volume 1. Cambridge University Press Cambridge.

Rojas, R. (2013). *Neural Networks: a Systematic Introduction.* Springer Science & Business Media.

Rörsch, A., Ziegler, P., and Mörner, N.-A. (2013). Sea level changes past records and future expectations. *Energy & Environment*, 24(3-4):509–536.

Russell, S. and Norvig, P. (1995). *Artificial Intelligence: A Modern Approach, (2nd ed.).* Pearson.

Salerno, J. J. and Boulware, D. M. (2006). Method and apparatus for improved web scraping. US Patent 7,072,890.

Scherer, K. R., Feldstein, S., Bond, R. N., and Rosenthal, R. (1985). Vocal cues to deception: A comparative channel approach. *Journal of psycholinguistic research*, 14(4):409–425.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, 34(1):1–47.

Sehgal, U., Kaur, K., and Kumar, P. (2009). The anatomy of a large-scale hyper textual web search engine. In *2009 Second International Conference on Computer and Electrical Engineering*, pages 491–495. IEEE.

Shi, L., Sun, B., Kong, L., and Zhang, Y. (2009). Web forum sentiment analysis based on topics. In *Computer and Information Technology, 2009. CIT'09. Ninth IEEE International Conference on*, volume 2, pages 148–153. IEEE.

Shkapenyuk, V. and Suel, T. (2002). Design and implementation of a high-performance distributed web crawler. In *Data Engineering, 2002. Proceedings. 18th International Conference on*, pages 357–368. IEEE.

Sokolova, M., Japkowicz, N., and Szpakowicz, S. (2006). Beyond accuracy, f-score and roc: a family of discriminant measures for performance evaluation. In *AI 2006: Advances in Artificial Intelligence*, pages 1015–1021. Springer.

Solomatine, D. P. and Ostfeld, A. (2008). Data-driven modelling: Some past experiences and new approaches. *Journal of hydroinformatics*, 10(1):3–22.

Somasundaran, S., Ruppenhofer, J., and Wiebe, J. (2007). Detecting arguing and sentiment in meetings. In *Proceedings of the SIGdial Workshop on Discourse and Dialogue*, volume 6.

Somasundaran, S. and Wiebe, J. (2010). Recognizing stances in ideological on-line debates. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, pages 116–124. Association for Computational Linguistics.

Specht, D. F. (1990). Probabilistic neural networks. *Neural networks*, 3(1):109–118.

Steidl, G. (2015). Supervised learning by support vector machines. *Handbook of Mathematical Methods in Imaging*, pages 1393–1453.

Subasic, P. and Huettner, A. (2001). Affect analysis of text using fuzzy semantic typing. *Fuzzy Systems, IEEE Transactions on*, 9(4):483–496.

Tang, Y., Zhang, Y.-Q., Chawla, N. V., and Krasser, S. (2009). Svms modeling for highly imbalanced classification. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 39(1):281–288.

Thompson, H. S., Beech, D., Maloney, M., and Mendelsohn, N. (2004). Xml schema part 1: Structures second edition. *W3C Recommendation (28 October 2004).*

Tomek, I. (1976). Two modifications of cnn. *IEEE Trans. Syst. Man Cybern.*, 6:769–772.

Vapnik, V. N. and Chervonenkis, A. Y. (2015). On the uniform convergence of relative frequencies of events to their probabilities. In *Measures of Complexity*, pages 11–30. Springer.

Walker, M. A., Anand, P., Abbott, R., Tree, J. E. F., Martell, C., and King, J. (2012). That is your evidence?: Classifying stance in online political debate. *Decision Support Systems*, 53(4):719–729.

Wiebe, J., Wilson, T., Bruce, R., Bell, M., and Martin, M. (2004). Learning subjective language. *Computational linguistics*, 30(3):277–308.

Wiebe, J. M. and Bruce, R. F. (2001). Probabilistic classifiers for tracking point of view. *Progress In Communication Sciences*, pages 125–142.

Wilson, T., Wiebe, J., and Hoffmann, P. (2005). 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.

Zhang, H. and Li, D. (2007). Naive bayes text classifier. In *Granular Computing, 2007. GRC 2007. IEEE International Conference on*, pages 708–708. IEEE.

Zhang, T. (2004). Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *Proceedings of the twenty-first international conference on Machine learning*, page 116. ACM.

Zhang, X., Zhao, J., and LeCun, Y. (2015). Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, pages 649–657.

Zhou, D., Bousquet, O., Lal, T. N., Weston, J., and Schölkopf, B. (2004). Learning with local and global consistency. In *Advances in Neural Information Processing Systems 16*, pages 321–328. MIT Press.

Zhu, X. and Ghahramani, Z. (2002). Learning from labeled and unlabeled data with label propagation. Technical report, Citeseer.

Appendix A: SemEval 2016 - IDI@NTNU

Semantic Evaluation, or SemEval[10] [11], is an ongoing series of evaluations where participants create computational semantic analysis systems. The evaluations are intended to explore meaning in natural language. Alongside the work on our master thesis we participated in task 6 A[12] in SemEval 2016. The purpose of the task was to explore approaches to detect stance in tweets. Our approach takes advantage of supervised machine learning algorithms in order to create an automated system for predicting stance. The official publication paper can be found in Appendix B: Official SemEval 2016 Paper. However, this appendix describes the task at hand, the system, and the experiments carried out in detail.

The first section describes the task and the data used. This should provide the reader with a sufficient understanding of the challenge. Later sections introduce approaches explored while striving to build a robust and successful system. We describe our experiments, how they were conducted and the results of these. The end of this appendix presents a post-analysis describing our thoughts and findings from the time after the gold labels were released. Additionally, a discussion of the results is provided.

A.1 Task 6 A: Detect Stance in Tweets

This section describes (in depth) the task and the data set provided for SemEval task 6 A.

[10]`https://en.wikipedia.org/wiki/SemEval`, as of 2016-06-06
[11]`http://alt.qcri.org/semeval2016/`, as of 2016-06-06
[12]`http://alt.qcri.org/semeval2016/task6/`, as of 2016-06-06

A.1.1 Task Description

The goal of task 6 A in SemEval 2016 was to detect stances in Twitter messages. Twitter and other microblogging platforms are popular places for people to express their opinions on everything from elections to the color of soda cans. With this in mind it is safe to say that there is a vast amount of information on peoples stance towards a huge variety of topics on Twitter.

There is no agreement on a single definition of the term stance in linguistics. Stance can be expressed in many ways. However, for this task the organizers of SemEval 2016 task 6 defined stance detection as: "Automatically determining from text whether the author is in favor of the given target, against the given target, or whether neither inference is likely". The training data provided was labeled which made this a supervised task.

The most efficient and precise approach to detect stance is to see if the text explicitly states whether or not it is in favor or opposing the target in question. The task gets hard when the text does not explicitly mention the target, but echoing somebody or something related to a target. This scenario lays the foundation for the challenge ahead.

A.1.2 Data

2814 tweets were collected and provided as training instances for the participants of SemEval 2016 task 6 A. The data set was made available for download in August 2015. Annotators of the training data was restricted to those living in the USA and they followed a questionnaire[13] when manually labeling the data. The annotators would classify each tweet with one of the following three labels: *favor, against* or *none*. Below we have extracted three tweets from the *Climate Change is a Real Concern* target. Each tweet exemplifies a class (the stance is listed in brackets at the end of each tweet).

> "SO EXCITING! Meaningful climate change action is on the way! #abpoli #GHG" [**Favor**]
>
> "The Climate Change people are disgusting assholes. Money transfer scheme for elite. May you rot." [**Against**]
>
> "#Netherlands just taught the rest of the world a very important lesson" [**None**]

[13] `http://alt.qcri.org/semeval2016/task6/data/uploads/stance-question.pdf`, as of 2016-06-06

The training data (and test data) was stored in a file with tab separated values. The files contained four columns; *id, topic, tweet*, and *stance*. The 2814 tweets were categorized into five targets: *Atheism, Climate Change is a Real Concern, Feminist Movement, Hillary Clinton,* and *Legalization of Abortion*. The class distribution per target can be seen in Table A.1 and in Figure A.1. A pie chart representing an overview of the target distribution can be seen in Figure A.2.

Target	**Favor**	**Against**	**None**	**Total**
Atheism	92	304	117	513
Climate Change is a Real Concern	212	15	168	395
Feminist Movement	210	328	126	664
Hillary Clinton	112	361	166	639
Legalization of Abortion	105	334	164	603
All	731	1342	741	2814

Table A.1: Statistics of the training data provided by the organizers of SemEval 2016 for task 6A.

As the target *Climate Change is a Real Concern* was related to our master thesis in the way that both tasks wanted to automatically detect stance for climate related topics, we initially wanted to direct our focus towards this target. However, the focus shifted and we ended up with a general system. The data distribution for the climate change target was highly imbalanced, which can be seen in Table A.1. The *against* class represented only 3.8 % (15 samples) of all the labels for this particular target.

A.2 The System

To classify stance in tweets, a supervised machine learning system was implemented using the Python programming language and the scikit-learn machine learning library (Pedregosa et al., 2011). Our system consists of a soft voting classifier that predicted label on the basis of the best results out of three individual classifiers. This section defines the features and the models that were trained and used for our final delivery.

A.2.1 Features

All features described below were generated from the raw data supplied in the training set. In the feature extraction process for the word bigrams, character

Properties	**Climate Change is a Real Concern**
Top 5 hashtags: FAVOR	13: #climate 12: #tip 12: #mission:climate 8: #environment 7: #cop21
Top 5 hashtags: AGAINST	3: #carbontaxscam 3: #chemtrails 1: #fraud 1: #anthropogene 1: #liberty
Top 5 hashtags: NONE	11: #peace 5: #lovewins 5: #gop 4: #valkilmer 4: #democracy
Top 10 term frequency	49: climate 25: change 17: need 15: like 14: global 13: future 12: attenborough 12: right 12: water 12: say
# of hashtags	669
# of unique words in hashtags	578

Table A.2: Overview of hashtag properties for the target Climate Change is a Real Concern.

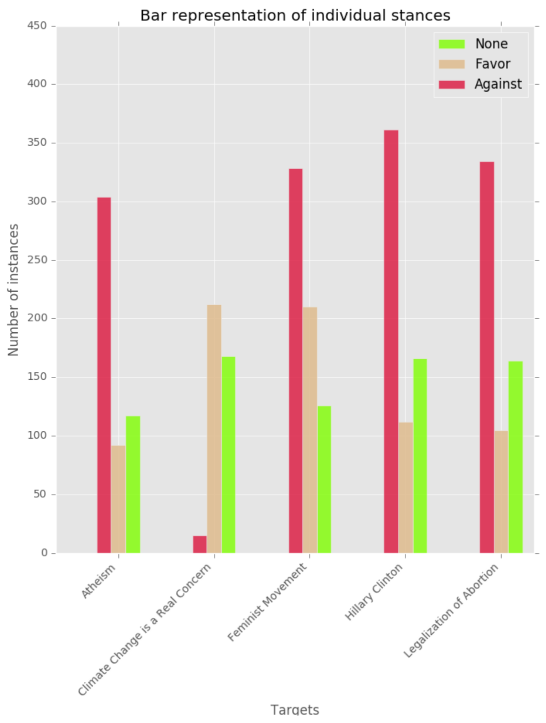


Figure A.1: Histogram representations of the training data in SemEval 2016 per target.

trigrams, and the GloVe vectors, the input was processed by performing simple tokenization of the text to obtain a vector representation of token counts. The submitted system used the following features:

1. **Word bigrams:** All pairs of consecutive words
   - Punctuation ignored
2. **Character trigram:** All triples of consecutive characters
   - Punctuation ignored
   - Converted to lowercase
   - Ignored terms that had a document frequency strictly lower than 5 (cut-off)
3. **GloVe vectors:** Word embeddings for all words in a tweet
   - Punctuation ignored

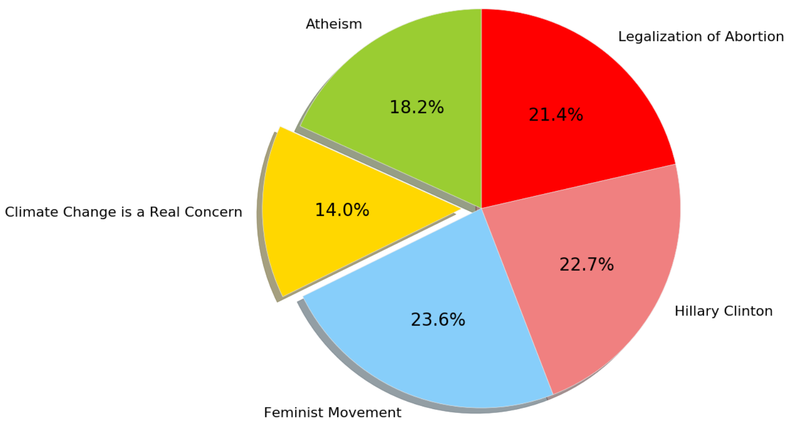


Figure A.2: Pie chart representation of the distribution of targets in the SemEval task 6A training data (in %).

- Converted to lowercase
- Removed stop words

In addition to the features used in the submission, we experimented with a set of additional shallow features

- **Negation:** Presence of negation in the sentence
- **Length of tweets:** Number of characters divided by the maximum length (140 characters)
- **Capital words:** Number of capital words in the tweet
- **Repeated punctuation:** Number of occurrences of non-single punctuation (e.g. !?)
- **Exclamation mark last:** Exclamation mark found last in non-single punctuation (e.g. ?!)
- **Lengthening of words:** Number of lengthened words (e.g. smoooth)
- **Sentiment:** Detecting sentiment in tweet using the Vader system (Hutto and Gilbert, 2014)

– **Number of tokens:** Count of total number of tokens in the tweet

These features were not included in the final system. They were left out as they did not improve the systems performance (section A.3.4 provides more details).

GloVe

GloVe (Pennington et al., 2014) is an unsupervised learning algorithm for obtaining vector representations of words. It creates word vectors based on the distributional statistics of words, in particular how frequently words co-occur within a certain window in a large text corpus such as the Gigaword corpus (Parker et al., 2011). The resulting word vectors can be used to measure semantic similarity between word pairs, following the hypothesis that similar words tend to have similar distributions. The Euclidean or Cosine distance between two word vectors can thus be used as a measure of their semantic similarity. For the word *frog*, for example, we can find related words such as *frogs, toad, litoria, leptodactylidae, rana, lizard, eleutherodactylu.*

To measure the semantic similarity between tweets, rather than isolated words, we needed a way to obtain vector representations of documents. Mitchell and Lapata (2010) looked at the possibility to use word vectors to represent the meaning of word combinations in a vector space. They suggest, among other things, to use vector composition, operationalized in terms of additive (or multiplicative) functions. Accordingly we created vector representations of tweets by combining the vectors of their words. We used word vectors created by Pennington et al. (2014) trained on Wikipedia 2014 + Gigaword 5[14] and Twitter data[15]. The word vectors come in several versions with a different number of dimensions $(25, 50, 100, 200, 300)$ that supposedly capture different granularities of meaning. The resulting features (from here on called GloVe features) were obtained by summing the GloVe vectors, per dimension, for all unique terms in a tweet.

The glove vectors are represented by a matrix where each row represents a word, and each column a dimension, this is illustrated in (A.1).

$$gv_{l,n} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{l,1} & a_{l,2} & \cdots & a_{l,n} \end{bmatrix} \tag{A.1}$$

We created the features by following these four (simple) steps:

[14]`http://nlp.stanford.edu/data/glove.6B.zip`, as of 2016-06-06
[15]`http://nlp.stanford.edu/data/glove.twitter.27B.zip`, as of 2016-06-06

1. Create a vocabulary, v, containing all words found in the tweets (remove stop words) and extract the vocabulary, gv, represented in the GloVe vectors.

2. Compare the contents of v and gv to obtain a shared vocabulary, $v\prime$, and the GloVe vectors, $gv\prime$, that represent the shared vocabulary.

3. Use $v\prime$ to perform a new token vectorization for all the original tweets resulting in u.

4. Obtain the final features by taking the dot product of the two matrices u and $gv\prime$, which amounts to summing the Glove vectors for all terms in a tweet per dimmension. See (A.2) and (A.3).

(A.2) shows the formula for acquiring the features. i is the number of tweets, j is the number of words in the shared vocabulary $v\prime$, m is the number of glove vectors in $gv\prime$, and n is the number of dimensions.

$$features_{i,n} = u_{i,j} \cdot gv\prime_{m,n} \tag{A.2}$$

$$\begin{aligned} features_{i,n} = &\begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,j} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,j} \\ \vdots & \vdots & \ddots & \vdots \\ a_{i,1} & a_{i,2} & \cdots & a_{i,j} \end{bmatrix} \\ \cdot &\begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{bmatrix} \end{aligned} \tag{A.3}$$

A.2.2 Models

To detect stance we constructed separate models for each of the five topics, each of the five models were in the form of a soft voting classifier. We used the voting classifier from scikit-learn found in Pedregosa et al. (2011). The voting classifiers took input from the following three classifiers:

1. **Multinomial Naive Bayes** trained on word bigrams

2. **Multinomial Naive Bayes** trained on character trigrams

3. **Logistic Regression** trained on Glove features

The soft voting classifier is – in contrast to a hard voting classifier – able to exploit prediction probabilities from the separate classifiers. For each sample, the soft voting classifier predicts the class based on the argmax of the sums of the predicted probabilities from the input classifiers.

In the task description it was stated that it was not necessary to predict stance for every tweet in the test set, leaving the uncertain ones with an *unknown* label. We decided to use a threshold value, using the extracted probabilities, to prevent predictions with low confidence. Labels predicted with a probability below the threshold were thus changed into *unknown*. Details of the selection of the threshold value are presented at the end of section A.4.

Due to the imbalanced distribution of labels in climate change data, our system had a low prediction rate of *against* stances on this target. For that reason we included a second slightly different model for the climate change target. The difference between the first and second model was that the second used a hard (majority rule) voting classifier, which performed slightly better on the against labels in the climate data. The combination of the two models was implemented in a way such that for each of the *against* predictions in the hard voting model, we overwrote the soft model's prediction, labeling the tweet *against*. Our submitted system thus consisted of two models for predicting the climate class, giving a total of six models.

To summarize, the system contained six models, where five of them was a soft voting classifier with input from the three different classifiers introduced above. The sixth was a hard voting model that supplementing the soft voting model for the climate change target.

A.3 Experiments and Results

To measure the system performance we conducted multiple experiments using the training data to examine the effects of various shallow features and the use of GloVe features with a varying number of dimensions. Other approaches such as label propagation where also evaluated through experiments.

This section describes the experiments conducted through the development phase, how they were set up and the outcome. Some of the experiments described here were not included in the original paper as these were not part of the final submission.

All experiments were conducted on the training set using stratified five-fold cross-validation and the results were evaluated using macro F-score based on precision and recall on the class labels *favor* and *against*. Our system used supervised machine learning algorithms supplied by the scikit-learn library Pedregosa et al. (2011).

A.3.1 Dummy Classifiers

As an initial approach we explored how the most simplistic strategies for classification performed. These approaches was considered as the baseline for further development and would represent the minimum performance expected from the system. When implementing these approaches the scikit-learn came in handy with a classifier called *Dummy Classifier*. This classifier comes with a few adjustable parameters to determine the classifiers strategy. We chose two of the possible strategies; the *most frequent* and *stratified*. The *most frequent* strategy simply takes the majority of the class representation in the training set and predict this class for every sample in the test set. The *stratified* option generates predictions based on the class distribution in the training set.

A histogram showing the results for this simple baseline can be seen in Figure A.3. The bars represent the macro F-score for each classifier and the tiny black line at the top of every bar represent the standard deviation. A table containing the performance results measured in macro F-scores for each individual targets, as well as all targets combined, can be seen in Table A.3

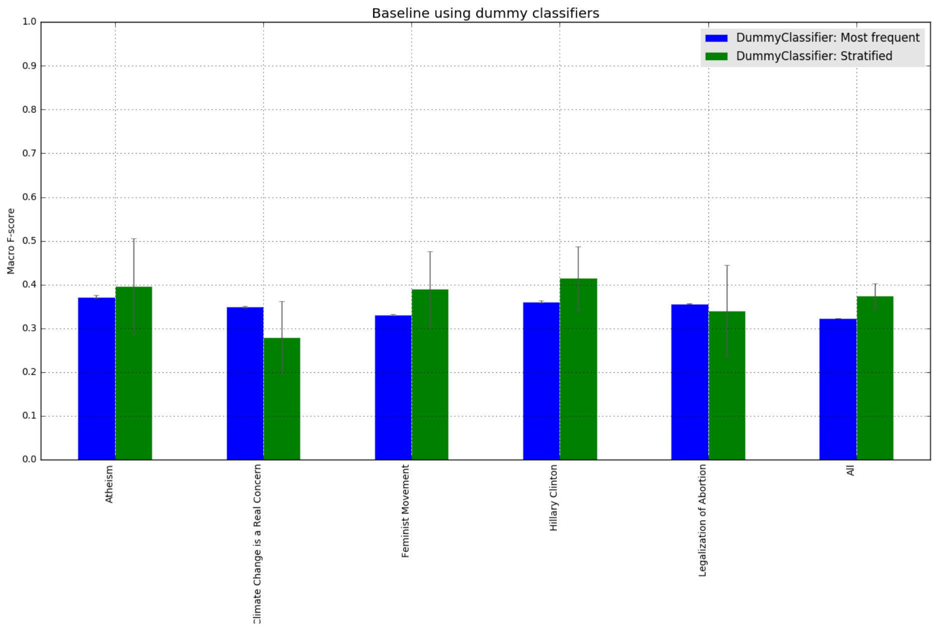


Figure A.3: Bar chart of the baseline using dummy classifiers representing macro F-score with standard deviation for individual targets and all targets combined.

Targets	Most Frequent	Stratified
Atheism	0.3721 (0.0041)	0.4116 (0.1204)
Climate Change is a Real Concern	0.3493 (0.0022)	0.2525 (0.0509)
Feminist Movement	0.3306 (0.0017)	0.3781 (0.0447)
Hillary Clinton	0.3610 (0.0020)	0.3767 (0.1056)
Legalization of Abortion	0.3565 (0.0003)	0.3549 (0.1381)
All	0.3229 (0.0005)	**0.3754** (0.0279)

Table A.3: Results of the dummy classifiers using the strategies *most frequent* and *stratified*. Table is showing the macro F-score for each target. Standard deviation is included in parenthesis.

In Table A.3, the *most frequent* baseline shows a reasonable steady macro F-score around 0.3500. This means the strategy classified about one out of three correct. Looking at the result when all the targets were combined, it is interesting to see that the overall score is 0.3229 even though the most frequent class (*against*) was represented by almost 48 % (1342/2814) of the tweets. A reasonable explanation is the use of macro F-score which evens out some of the effects of skewed data distributions in test results (at least compared to accuracy).

The *stratified* strategy performed better than *most frequent*. The F-score increased by 0.0525 resulting in 0.3754 (all targets combined).

A.3.2 Multiple Classifiers

To improve the systems performance multiple classifiers and their parameters were investigated. We chose a set of classifiers that would represent some different approaches to the problem. The following classifiers were included: Support Vector Machine (SVM), Linear SVM and Multinomial Naive Bayes (MNB) throughout the development phase. In work done after submission we also included two other classifiers; Stochastic Gradient Descent (SGD) and Logistic Regression (LR). A broad selection of classifiers were presented to see how they performed on classification of tweets.

Unigram word and grid search on parameters was utilized to optimize settings. The result of the experiment is presented in Figure A.4 where the bars are aligned beside each other for each target. The bars represent the classifier's achieved macro F-score and the tiny black line at the top of every bar represent the standard deviation. The macro F-scores and the standard deviations are

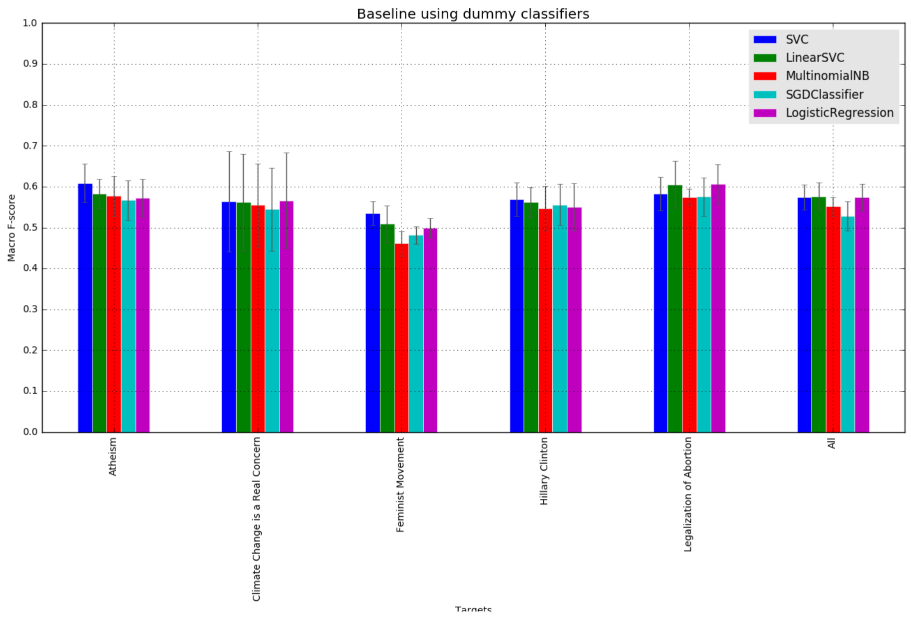


Figure A.4: Bar histogram showing macro F-score with standard deviation for individual targets and all targets combined per classifier.

listed in Table A.4.

The results show that every classifier performed significantly better than the dummy baseline. The scores are all at least 0.10 better. We can observe that the top two classifiers where Logistic Regression with an macro F-score of 0.5736 and Linear SVM with the score 0.5819.

A.3.3 Bootstrapping Attempts

As our master thesis focused on climate related texts, we concentrated a bit more on the target *Climate Change is a Real Concern* throughout the development phase. The challenge with this target was the skewed class distribution. Ideas for retrieving additional Twitter data for the *against* class was explored to even out the distribution. However, the task description explicitly stated that systems using additional data that was manually labeled would be competing in a separated class. We therefore explored automated labeling, more specifically label propagation (Zhu and Ghahramani, 2002; Zhou et al., 2004). Additionally, we investigated the possibility of using skeptical myths from TCP to supplement

Targets	**SVM**	**Linear SVM**	**MNB**	**SGD**	**LR**
Atheism	0.6093 (0.0467)	0.5830 (0.0363)	0.5772 (0.0481)	0.5672 (0.0485)	0.5728 (0.0453)
Climate Change is a Real Concern	0.5642 (0.1222)	0.5621 (0.1185)	0.5551 (0.1018)	0.5447 (0.1012)	0.5664 (0.1170)
Feminist Movement	0.5353 (0.0287)	0.5089 (0.0449)	0.4612 (0.0302)	0.4816 (0.0210)	0.4996 (0.0240)
Hillary Clinton	0.5698 (0.0407)	0.5619 (0.0362)	0.5468 (0.0548)	0.5564 (0.0502)	0.5504 (0.0575)
Legalization of Abortion	0.5827 (0.0413)	0.6019 (0.0578)	0.5749 (0.0191)	0.5755 (0.0464)	0.6063 (0.0490)
All	0.5701 (0.0311)	**0.5819** (0.0333)	0.5514 (0.0232)	0.5290 (0.0358)	0.5736 (0.0325)

Table A.4: Performance of classifiers measures by macro F-score. Standard deviation in parenthesis.

the minority class (*against*) for the climate change target.

A problem was discovered when experimenting with bootstrapping attempts. Scikit-learn did not have any cross validation methods suited for label propagated data. This is because the standard implementation takes the training data and divides it into n folds of equal size, then trains on $n-1$ and leaving the last fold for testing. This is carried out n times so that in the end all of the n folds have been used for test once. The problem with this approach combined with label propagation is that we only want to use the label propagated data for training, not for testing. This is because the label propagated data may contain errors which could lead to a misleading evaluation of the classifier. We eventually solved this by implementing a custom train-test fold generator and made some minor changes to the scikit-learn cross validation implementation allowing train and test folds to be of different sizes. This job took longer than expected and caused some minor delays in the development.

Label Propagation

Label propagation is a semi-supervised learning algorithm. A semi-supervised learning situation is whenever your training data have some unlabeled samples. The semi-supervised algorithm has the ability to make use of the unlabeled data to capture the underlying distribution and generalize better to new test samples

(Zhu and Ghahramani, 2002; Zhou et al., 2004). The label propagation algorithm usually performs better when the distribution represent small amounts of labeled data and rather large size of unlabeled data. Below is one tweet for each stance labeled by using label propagation. The tweet's stance is added in brackets at the end.

> ”Our children need to be protected from #climate pollution! Tell your Governor #CleanPowerPlan” – [FAVOR]
>
> ”First coffee, now this.... truly, tis the end of days. #climatechange #beer #coffee #Whybotherliving” – [NONE]
>
> ”How #climate change disproportionately affects women #social #sustainability” – [AGAINST]

As the original data was retrieved from Twitter, we wanted to do the same to get similar data. The Twitter API[16] was employed to download large quantities of tweets related to climate change. The related content was found by searching for hashtags. Statistics shown in Table A.2 shows the the most common hashtags that were searched for. Amongst these were #climate, #chemtrails and #cop21. #cop21 was referring to the climate conference in Paris, December 2015. #chemtrails was just a conception used in, for instance, conspiracy theories. From all the tweets collected we manually handpicked a small amount that seemed within limits of the climate change target and somewhat similar to the tweets in the training data. Most of the selected tweets contained #climate.

Classifier	**Unlabeled tweets**	**Original score**	**Label Propagation score**
MNB	77 samples	0.5551	0.5470
MNB	144 samples	0.5551	0.5460
LinearSVM	77 samples	0.5621	0.5162
LinearSVM	144 samples	0.5621	**0.5803**
SVM	77 samples	0.5642	0.5780
SVM	144 samples	0.5642	0.5762

Table A.5: Experimental results from applying label propagation to add extra data for the Climate Change is a Real Concern target.

The label propagation was executed on a (small) representative sample of the labeled training data along with the collected, hand picked, unlabeled tweets.

[16]`https://dev.twitter.com/rest/public`, as of 2016-06-06.

Table A.5 shows that with MNB, label propagation did not improve the results, the score slightly deteriorated. When using Linear SVM, label propagation show a significant drop when propagating 77 samples, while an improvement (and the highest score of 0.5803) when doubling the number of tweets labeled using label propagation. The SVM yield a small improvement for both sample tests. We found that adding more data to our system did not result in substantial improvement. An explanation could be that the gathered tweets were not meaningful enough to be effective. As Table A.5 shows the results were inconclusive. Adding new data did not (solely) improve the performance. The additional data was therefore not used in subsequent experiments.

Skeptical Myths

As mentioned previously we focused more on the target: *Climate Change is a Real Concern* early in the development phase of our system. In the preliminary work for our master thesis, we had browsed the web page[17] providing The Consensus Project (TCP) and other facts on global warming. The web page provided close to 200 climate change myths[18]. The myths were short sentences claiming climate change is not human-induced. Additionally, the web page provided scientific answer to each myth. Table A.6 display three of the most used climate myths as well as their scientific counter argument (minimized).

When using the myths we tested several batch sizes, but only two are included (40, 100) in Table A.7 as the other sizes did not stand. Our intention was to use these myths as supplement to data labeled *against* for the *Climate Change is a Real Concern* target. This approach, if included in our system, would have required us to get a note from the SemEval organizers as to whether this would be considered manually labeling data or not. Table A.7 contains results from the experiment.

As mentioned the scores of this bootstrapping attempt did not show enough support for us to move forward with further testing. All but the second row adding 100 myths with MNB show a lower F-score when the myths were added to the original data. MNB including the 100 myths increased the score from 0.5551 to 0.5825. This increase would only concern one out of the five targets.

Neither of the bootstrapping attempts provided a significant boost in the system, and were not used in any later experiments.

[17]`http://www.skepticalscience.com`, as of 2016-06-06
[18]`http://www.skepticalscience.com/argument.php`, as of 2016-06-06
[19]`http://skepticalscience.com`, as of 2016-06-06

Climate myths	vs	**Scientific counter argument**
”Climate’s changed before”	-	”Climate reacts to whatever forces it to change at the time; humans are now the dominant forcing.”
”It’s the sun”	-	”In the last 35 years of global warming, sun and climate have been going in opposite directions”
”It’s not bad”	-	”Negative impacts of global warming on agriculture, health & environment far outweigh any positives.”

Table A.6: Top three of most used climate myths collected from Skeptical Science[19]as well as their scientific counter argument (minimized)

Classifier	**Added myths**	**Original**	**Myths**
MNB	40 myths	0.5551	0.5484
MNB	100 myths	0.5551	**0.5825**
LinearSVM	40 myths	0.5621	0.5367
LinearSVM	100 myths	0.5621	0.5378
SVM	40 myths	0.5642	0.5366
SVM	100 myths	0.5642	0.5366

Table A.7: Experimental results of adding skeptical myths labeled **against** for the Climate Change is a Real Concern target.

A.3.4 Shallow Features

In the experiments regarding additional shallow feature we took advantage of the scikit (Pedregosa et al., 2011) library which easily lets you add new features using a feature union. Instead of pipelining feature by feature, the system combines all features at once. The outcome of these experiments can be seen in Table A.8
In the development phase, the data set was divided by the individual targets creating five respective data sets. The experiments on shallow features began by including more and more shallow features. We started off by applying various

forms of n-grams (uni-, bi- and trigram of words and characters). The classifier that achieved the highest cross-validated macro F-score from these experiments was MNB using character trigram. The achieved score was 0.6290. This would serve as the base for testing the additional shallow features.

Among the additional features were detection of negation, number of capital words, and sentiment in a sentence (Hagen et al., 2015; Walker et al., 2012). By this time we had explored other parameters of n-gram, applying term-frequency inverse document frequency (tf-idf), *C* and *alpha* parameters and different loss functions.

Shallow Features	**Macro F**	**Change**
Trigram characters	0.6290	
.+negation	0.6308	(+ 0.0018)
.+length of tweets	0.6311	(+ 0.0003)
.+capital words	0.6313	(+ 0.0002)
.+non-single punctuation	0.6356	(+ 0.0043)
.+exclamation mark last	0.6358	(+ 0.0002)
.+lengthening words	**0.6360**	(+ 0.0002)
.+sentiment	0.6352	(- 0.0008)
.+number of tokens	0.6264	(- 0.0088)

Table A.8: Average macro F-scores for different sets of shallow features from five-fold CV experiments with MNB classifier on the entire training set.

Looking at Table A.8, several features like detection of negation and number of capital words increase the performance slightly. Combining the features negation detection, length of tweets, number of capital words, punctuation marks, exclamation marks and lengthening words proved to be the best F-score of 0.6360. The last two features, adding sentiment and number of tokens, decreased the performance. Summarized, Table A.8 shows that adding more shallow features yielded only a slight increase in macro F-score from 0.6290 to 0.6360. Based on this, relatively small, improvement it is difficult to imply that the addition of features gave any substantial performance boost of the system. These features were therefore not included in the final submission.

A.3.5 GloVe - Word Embedding

Experiments regarding GloVe were tested with a Logistic Regression classifier using GloVe features (explained the previous chapter). To create the GloVe features we used pre-trained word vectors from different corpora with a various number of dimensions ($corpus\ sizes = [(6Btokens, 400Kvocab), (27Btokens, 1.2Mvocab)]$ and $dimensions = [25, 50, 100, 200, 300]$). The various dimensions supposedly capture different granularities of meaning obtained from the corpora they were extracted from[20].

Table A.9 and Table A.10 the baseline score of 0.5819 increased to 0.6360 when applying the best shallow features. It also shows that using only the Logistic Regression classifier with GloVe vectors did not perform well. A combination was tested for multiple classifiers. Initially, we tried wrapping the Logistic Regression classifier and the MNB classifier from Table A.8 in a voting classifier. However, this new voting classifier did not improve the performance, instead a further drop in performance occurred. Later, reducing the feature set of the MNB classifier down to only applying versions of n-grams was more successful. Our best result was achieved using the Logistic Regression classifier using GloVe features, MNB classifier using bigram words, and a MNB classifier with trigram characters wrapped inside a soft voting classifier. The final submission therefore included only n-gram features, the rest of the features were discarded. As seen in Table A.9 this scored 0.6751, which was a significant improvement over the performance baseline.

Features	**Overall**	**std (σ)**
Baseline (LinearSVM)	0.5819	0.0494
Best shallow features	0.6360	0.0891
Glove features	0.6067	0.0722
Glove + best shallow	0.6048	0.0659
Glove + n-gram	**0.6751**	0.0704

Table A.9: Average overall macro F-scores for different combinations of feature sets from five-fold CV experiments on the entire training set.

[20]The final submission used the following word vectors: Atheism (*size*=$6B$, *dimension*=200), Climate Change (*size*=$27B$, *dimension*=200), Feminist Movement (*size*=$27B$, *dimension*=100), Hillary Clinton (*size*=$27B$, *dimension*=200), Legalization of Abortion (*size*=$27B$, *dimension*=100).

Features	**Atheism**	**Climate**	**Feminism**	**Hillary**	**Abortion**
Baseline	-	-	-	-	-
Best shallow features	0.6601	0.5923	0.6246	0.6022	0.7006
Glove features	0.6516	0.6256	0.5553	0.5898	0.6102
Glove + best shallow	0.5775	0.5604/ **0.6754**	0.6291	0.5479	0.7088
Glove + n-gram	**0.7055**	0.6540/ 0.6404	**0.6537**	**0.6427**	**0.7204**

Table A.10: Average macro F-scores per target, for different combinations of feature sets from five-fold CV experiments on the entire training set. Baseline model was not trained per target, therefore no individual scores are available. Where two scores are listed, there were two models used (soft/hard voting).

A.4 Post-Processing

Our submitted approach achieved a macro F-score of **0.6247** on the test data, while the best system on task 6A achieved a score of 0.6782. After the gold labels were released, we ran the test ourselves in order to see how well we did on precision, recall, and F-score. Table A.11 shows our final results. The high precision on the class *against* shows that predictions for this label were mostly correct, albeit with a relatively low recall.

Stance	**Precision**	**Recall**	**F-score**
Favor	0.5750	0.6053	0.5897
Against	0.8770	0.5287	0.6597
Overall macro F-score			0.6247

Table A.11: The overall result of *Favor* and *Against* stance after the release of gold labels showing precision, recall and f-score

At the end of A.2.2, we mentioned a threshold that was established in our system. The threshold value was set at the last minute using a rule of thumb as we did not have time to perform experiments to determine the optimal setting, or even whether it was beneficial at all. Our intention was to use this approach only for the category *Climate Change is a Real Concern*, as this was the most skewed topic. However, by accident, it was applied to *all* topics. Comparing our best result in the development phase with the test result, we can observe a substantial drop in performance. This is a result of the threshold that was – by mistake – applied to all predictions. To measure how much this affected our system, we performed an overall test run where the threshold as used in the original submission was disregarded. This resulted in a macro F-score of 0.6660 - an increase of 0.0413 relative to our submitted score. The threshold proved to have lowered the recall for both *favor* and *against* and explains the low recall in the submitted system predictions (Table A.12).

Stance	**Precision**	**Recall**	**F-score**
Favor	0.5432	0.7237	0.6206
Against	0.8042	0.6378	0.7114
Overall macro F-score			0.6660

Table A.12: Precision, recall and F-score of the submission without the applied threshold per class as well as overall macro F-score.

As the target *Climate Change is a Real Concern* has a highly skewed class distribution which made learning particularly hard, we found it interesting to include the score of the climate change target, which can be seen in Table A.13 with the final macro F-score of 0.5486.

Stance	**Precision**	**Recall**	**F-score**
Favor	0.7967	0.8673	0.8305
Against	0.1818	0.5000	0.2667
Overall macro F-score			0.5486

Table A.13: Results of *Favor* and *Against* after the release of gold labels from the target: *Climate Change is a Real Concern* showing precision, recall and F-score

It is interesting to see how the macro F-score for the climate target compare to overall macro F-score. It is Almost 0.0800 lower. Not surprisingly, we can see that the F-score of the label *against* is very poor (0.2667) compared to a strong *favor* result, which has F-score of 0.8305. These results matched what we expected given the data distribution. We can imply that the system is confident when predicting *favor* stances toward climate targets, but have problems detecting *against* stances.

It is worth mentioning that even though the addition of all shallow features gave poor results during training, it performed a lot better on the test data, scoring 0.6939 from Table A.14. This is most likely due to differences in the training and the test data.

Stance	Precision	Recall	F-score
Favor	0.5617	0.7039	0.6248
Against	0.7722	0.7538	0.7629
Overall macro F-score			0.6939

Table A.14: Result of all shallow features after the release of gold labels showing precision, recall and F-score.

Mohammad et al. (2016) (organizers of SemEval) released their findings toward sentiment and stance detection post deadline of the shared task of detecting stance in tweets. Their system used a linear SVM classifier with sentiment features, multiple n-grams, word embeddings and encodings (such as presence/absence of hashtags and exclamation marks). The system performed better than the first place in the official competition. While their final method scored 70.3 %, the climate change target only obtained 43.8 %.

A.5 Conclusion

This appendix summarizes in depth the system created for SemEval 2016 task 6A - *Detecting Stance in Tweets*. Using shallow features alone performed well, but combining shallow features and word embeddings created from GloVe word vectors increased the score substantially. With the combination we were able to detect stance in tweets with a macro F-score of 0.6247, giving us the 10th place.

Post-analysis revealed that the application of an ad-hoc threshold to prevent low-confidence predictions was a mistake, resulting in a 0.0413 loss in overall macro F-score. The threshold should have been set using cross-validation, or even better, not at all. Not using the threshold we would have achieved 4th place.

During the experiments we found the parameters to be of great importance to the classifier's performance. A small change can affect results significantly.

Our main focus was drawn from the target *Climate Change is a Real Concern* to the overall performance during the development. But we experienced the difficulty of working with training data that has a highly skewed class distribution.

In retrospect other approaches to re-balancing the distribution such as up- and down-sampling should have been investigated. Our attempts to add data turned out to be difficult to make effective and were discarded.

Participating in this shared task has increased our knowledge on machine learning techniques, especially for text classification. In addition we have gained more insight to which tools to use and which procedures to follow when conducting experiments. We both agree that with more development time we could have improved the system further and performed better in the competition.

Appendix B: Official SemEval 2016 Paper

The official paper, as published in the SemEval 2016 proceeding, is included on the next page.

IDI@NTNU at SemEval-2016 Task 6: Detecting Stance in Tweets Using Shallow Features and GloVe Vectors for Word Representation

Henrik Bøhler and **Petter Fagerlund Asla** and **Erwin Marsi** and **Rune Sætre**
Norwegian University of Science and Technology
Sem Sælands vei 9
Trondheim, 7491, NORWAY
{henriboh,pettefas}@stud.ntnu.no,{emarsi,satre}@idi.ntnu.no

Abstract

This paper describes an approach to automatically detect stance in tweets by building a supervised system combining shallow features and pre-trained word vectors as word representation. The word vectors were obtained from several collections of large corpora using GloVe, an unsupervised learning algorithm. We created feature vectors by selecting the word vectors relevant to the data and summing them for each unique word. Combining multiple classifiers into a voting classifier, representing the best of both approaches, shows a significant improvement over the baseline system.

1 Introduction

This paper describes our submission to the SemEval 2016 competition Task 6A - *Detecting Stance in Tweets*. The goal of the task is to classify a tweet into one of the three classes – *against*, *favor* or *none* in regard to a certain topic. These classes represents the tweet's stance towards the given target.

Twitter, and other microblogging platforms, have in recent years become popular arenas to apply natural language processing tasks. One of the most popular tasks has been sentiment analysis. Stance detection differs from sentiment analysis because the sentiment of a text – generally positive or negative – does not necessarily agree with its stance regarding a certain topic of debate. For example, a tweet like ”all those climate-deniers are morons” is negative in its overall sentiment, but positive with regard to the stance that climate change is a real concern. We refer the reader to the official SemEval 2016[1] website for a detailed task description.

Our approach to detect stance is based on shallow features (Kohlschütter et al., 2010; Hagen et al., 2015; Walker et al., 2012) and the use of GloVe word vectors (Pennington et al., 2014). During the development phase we explored several approaches by implementing features such as sentiment detection (Hutto and Gilbert, 2014), number of tokens and number of capital words. The experiments later in this paper show that not all features enhanced the performance of the implemented system.

The feature that turned out to boost performance the most, in combination with basic shallow features, was the use of pre-trained GloVe vectors (Pennington et al., 2014). The vector representations of tweets were created by summing the pre-trained word vectors for each unique word. No additional data was added to the training set used for our final submission, although we explored the possibility of gathering and automatically labelling additional tweets by using label propagation (Zhu and Ghahramani, 2002; Zhou et al., 2004). This did enhance our baseline system performance slightly, but not in combination with other features.

2 System description

To predict the stance in tweets we built a supervised machine learning system using the scikit-learn machine learning library (Pedregosa et al., 2011). Our system consists of a soft voting classifier that predicts the class label on the basis of the best re-

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

sults out of the three classifiers described in subsection 2.3.

2.1 Resources

Our system used a limited number of resources. It relies on the annotated training data consisting of 2814 tweets divided into five different topics: *Atheism*, *Climate Change is a Real Concern*, *Feminist Movement*, *Hillary Clinton*, and *Legalization of Abortion*. In addition, it uses pre-trained word vectors[2] created by Pennington et al. (2014).

2.1.1 Bootstrapping attempts

The labels for the climate change target showed a highly skewed distribution where only 3.8% were labelled *against*. Skewed data distributions in machine learning are a common problem. Monard and Batista (2002), Provost (2000) and Tang et al. (2009) discuss this problem and suggests several solutions, such as data under- and over-sampling. We did not have time to investigate the effects of these methods, but Elkan (2001) suggest that changing the balance of negative and positive training samples has little effect on learned classifiers.

In an attempt to even out the distribution of the climate change data, we searched for ways to add additional tweets. The most promising approach explored was label propagation (Zhu and Ghahramani, 2002; Zhou et al., 2004), a semi-supervised learning algorithm. Thousands of tweets were fetched based on the most common hashtags found in the climate topic data. We hand-picked a small portion of tweets that seemed relevant to the climate topic (e.g. same language and containing a statement). These tweets were then automatically labelled using label propagation. The label propagation was performed with a (small) representative sample of the labelled training data together with the collected, hand picked, unlabelled tweets. We found that adding more data to our system did not result in substantial improvement. An explanation could be that the gathered tweets were not meaningful enough to be effective. The additional data was therefore not used in subsequent experiments.

[2]http://nlp.stanford.edu/projects/glove/

2.2 Features

The submitted system used the following features, generated from the raw data supplied in the training set.

1. **Word bigrams:** All pairs of consecutive words
   - Punctuation ignored
2. **Character trigram:** All triples of consecutive characters
   - Punctuation ignored
   - Converted to lowercase
   - Ignored terms that had a document frequency strictly lower than 5 (cut-off)
3. **GloVe vectors:** Word embeddings for all words in a tweet
   - Punctuation ignored
   - Converted to lowercase
   - Removed stop words

In addition, we experimented with the following features, which were not included in the final system. They were left out as they did not improve the systems performance (section 3 will provide more details on this).

- **Negation:** Presence of negation in the sentence
- **Length of tweets:** Number of characters divided by the maximum length (140 characters)
- **Capital words:** Number of capital words in the tweet
- **Repeated punctuation:** Number of occurrences of non-single punctuation (e.g. !?)
- **Exclamation mark last:** Exclamation mark found last in non-single punctuation (e.g. ?!)
- **Lengthening of words:** Number of lengthened words (e.g. smoooth)
- **Sentiment:** Detecting sentiment in tweet using the Vader system (Hutto and Gilbert, 2014)
- **Number of tokens:** Count of total number of tokens in the tweet

2.2.1 GloVe

GloVe (Pennington et al., 2014) is an unsupervised learning algorithm for obtaining vector representations of words. It creates word vectors based on the distributional statistics of words, in particular how frequently words co-occur within a certain window in a large text corpus such as the Gigaword corpus (Parker et al., 2011). The resulting word vectors can be used to measure semantic similarity between word pairs, following the hypothesis that similar words tend to have similar distributions. The Euclidean or Cosine distance between two word vectors can thus be used as a measure of their semantic similarity. For the word *frog*, for example, we can find related words such as *frogs, toad, litoria, leptodactylidae, rana, lizard, eleutherodactylu*.

In order to measure the semantic similarity between tweets, rather than isolated words, we needed a way to obtain vector representations of documents. Mitchell and Lapata (2010) looked at the possibility to use word vectors to represent the meaning of word combinations in a vector space. They suggest, among other things, to use vector composition, operationalized in terms of additive (or multiplicative) functions. Accordingly we created vector representations of tweets by combining the vectors of their words. We used pre-trained word vectors created by Pennington et al. (2014) trained on Wikipedia 2014 + Gigaword 5[3] and Twitter data[4]. The word vectors come in several versions with a different number of dimensions (25, 50, 100, 200, 300) that supposedly capture different granularities of meaning. The resulting features (from here on called GloVe features) were obtained by summing the GloVe vectors, per dimension, for all unique terms in a tweet.

2.3 Models

To detect stance we constructed separate models for each of the five topics, each in the form of a soft voting classifier from scikit-learn (Pedregosa et al., 2011). The voting classifiers took input from the following three classifiers:

1. **Multinomial Naive Bayes** trained on word bigrams
2. **Multinomial Naive Bayes** trained on character trigrams
3. **Logistic Regression** trained on GloVe features

The soft voting classifier is – in contrast to a hard voting classifier – able to exploit prediction probabilities from the separate classifiers. For each sample, the soft voting classifier predicts the class based on the argmax of the sums of the predicted probabilities from the input classifiers.

In the task description it was stated that it was not necessary to predict stance for every tweet in the test set, leaving the uncertain ones with an *unknown* label. We decided to use a threshold value, using the extracted probabilities, to prevent predictions with low confidence. Labels predicted with a probability below the threshold were thus changed into *unknown*. Details of the selection of the threshold value are presented at the end of section 4.

Due to the imbalanced distribution of labels in climate change data, our system had a low prediction rate of *against* stances on this target. For that reason we included a second slightly different model for the climate change target. The difference between the first and second model was that the second used a hard (majority rule) voting classifier, which performed slightly better on the against labels in the climate data. The combination of the two models was implemented in a way such that for each of the *against* predictions in the hard voting model, we overwrote the soft model's prediction, labelling the tweet *against*. Our submitted system thus consisted of two models for predicting the climate class, giving a total of six models.

To summarize, the system contains six models, where five of them consist of a soft voting classifier with input from the three different classifiers introduced above. The sixth is a hard voting model that supplements the soft voting model for the climate change target.

3 Results on Development Data

To measure the system performance we conducted multiple experiments using the training data to examine the effects of various shallow features and the use of GloVe features with a varying number of dimensions. All experiments in this paper were conducted using stratified five-fold cross-validation and

[3] http://nlp.stanford.edu/data/glove.6B.zip

[4] http://nlp.stanford.edu/data/glove.twitter.27B.zip

the results were measured with macro F-score based on precision and recall on the class labels *favor* and *against*. Our system used supervised machine learning algorithms supplied by the scikit-learn library (Pedregosa et al., 2011).

3.1 Baseline

The first experiment was set up to gain insight in the performance of different classifiers and their parameters. We chose a basic approach using only word unigrams (bag of words approach). The best of the resulting models was chosen as the baseline, serving as an indication of the performance of a simplistic system. The models were trained on the entire data set, not divided by individual targets. We chose to perform the experiment with two different Support Vector Machines (SVM) and one Naive Bayes (MNB) classifier with different parameters[5].

One of the hyperparameters we optimized was *C*, which is a regularization term for misclassifications of each sample. Higher values will do a better job correctly labelling the training data during training (smaller hyperplane margin), but are more likely to overfit. Conversely, lower values may have more misclassifications because it will ignore more outliers (larger hyperplane margin), but are less likely to overfit. We also used the *decision function shape* parameter to decide whether to use one-vs-one (ovo) or one-vs-rest (ovr) as decision function. Ovo constructs one classifier per pair of classes. At prediction time, a vote is performed and the class which receives the most votes is selected. The ovr strategy consist of fitting one classifier for each class. The table below displays the results from the experiments.

Classifiers	Parameter specification	Macro F
Multinomial NB	[alpha=0.01]	0.5513
SVM	[kernel='linear', C=0.37]	0.5701
LinearSVM	[kernel='linear', C=0.28]	**0.5819**

Table 1: Average macro F-scores from five-fold CV experiments with different classifiers on the entire training set.

LinearSVM scored highest and established the baseline with the macro F-score of 0.5819. However, the LinearSVM classifier was not beneficial in later experiments when trained individually per target[6] and therefore only used as a performance baseline.

3.2 Improved system

In the development phase, the data set was divided by the individual targets creating five respective data sets. The development experiments began by including more and more shallow features. We started off by applying various forms of n-grams (uni-, bi- and trigram of words and characters). The classifier that achieved the highest cross-validated macro F-score from these experiments was MNB using character trigram. The achieved score was 0.6290. The experiments continued by adding features (listed in section 2.2) to the MNB in addition to the character trigram feature. Results of these experiments can be seen in table 2.

Shallow Features	Macro F	Change
Trigram characters	0.6290	
.+negation	0.6308	(+ 0.0018)
.+length of tweets	0.6311	(+ 0.0003)
.+capital words	0.6313	(+ 0.0002)
.+non-single punctuation	0.6356	(+ 0.0043)
.+exclamation mark last	0.6358	(+ 0.0002)
.+lengthening words	**0.6360**	(+ 0.0002)
.+sentiment	0.6352	(- 0.0008)
.+number of tokens	0.6264	(- 0.0088)

Table 2: Average macro F-scores for different sets of shallow features from five-fold CV experiments with MNB classifier on the entire training set.

Table 2 shows that adding shallow features yielded only a slight increase in macro F-score from 0.6290 to 0.6360. Based on this, relatively small, improvement it is difficult to imply that the addition of features gave any substantial performance boost of the system.

3.3 Final system

Subsequent experiments tested the use of a Logistic Regression classifiers with GloVe feature vectors. We used pre-trained word vectors from

[5]SVM with *kernel*=[$linear, rbf, poly$], *C*=$numpy.logspace(-3, 3, 50)$, *decision_function_shape*=[ovo, ovr] and LinearSVM with *C*=$numpy.logspace(-3, 3, 50)$. MultinomialNB with *alpha*=$numpy.logspace(-1, 1, 10)$).

[6]Average macro F-score over all targets:
LinearSVM with word bigram: 0.4955.
LinearSVM with character trigram: 0.5970.
LinearSVM with shallow features: 0.5974

Features	Overall	std (σ)	Atheism	Climate	Feminism	Hillary	Abortion
Baseline	0.5819	0.0494	-	-	-	-	-
Best shallow features	0.6360	0.0891	0.6601	0.5923	0.6246	0.6022	0.7006
Glove features	0.6067	0.0722	0.6516	0.6256	0.5553	0.5898	0.6102
Glove + best shallow	0.6048	0.0659	0.5775	0.5604/0.6754	0.6291	0.5479	0.7088
Glove + n-gram	**0.6751**	0.0704	0.7055	0.6540/0.6404	0.6537	0.6427	0.7204

Table 3: Average macro F-scores, both overall and per target, for different combinations of feature sets from five-fold CV experiments on the entire training set. Baseline model was not trained per target, therefore no individual scores are available. Where two scores are listed, there were two models used (soft/hard voting).

different corpora with a various number of dimensions ($corpus\ sizes$ = $[(6Btokens, 400Kvocab)$, $(27Btokens, 1.2Mvocab)]$ and *dimensions* = $[25, 50, 100, 200, 300]$). The various dimensions supposedly capture different granularities of meaning obtained from the corpora they were extracted from[7].

From table 3 we can observe that from the baseline score of 0.5819 the result increased to 0.6360 when applying the best shallow features. It also shows that using only the Logistic Regression classifier with GloVe vectors did not perform well. For this reason we decided to combine multiple classifiers. Initially we tried wrapping the Logistic Regression classifier and the MNB classifier from table 2 in a voting classifier. However, this new voting classifier did not improve the performance, instead a further drop in performance occurred. We later inspected the outcome of the combined classifiers when we reduced the feature set of the MNB classifier down to only applying versions of n-grams. This was more successful and our best result was achieved using the Logistic Regression classifier using GloVe features, MNB classifier using bigram words, and a MNB classifier with trigram characters wrapped inside a soft voting classifier. The final submission therefore included only n-gram features and the rest of the features were discarded. As seen in table 3 this scored 0.6751, which was a substantial improvement over the performance baseline.

[7]The final submission used the following word vectors: Atheism (*size*=$6B$, *dimension*=200), Climate Change (*size*=$27B$, *dimension*=200), Feminist Movement (*size*=$27B$, *dimension*=100), Hillary Clinton (*size*=$27B$, *dimension*=200), Legalization of Abortion (*size*=$27B$, *dimension*=100).

4 Results on Test Data

Our submitted approach achieved a macro F-score of **0.6247** on the test data, while the best system on task 6A achieved a score of 0.6782. After the gold labels were released, we ran the test ourselves in order to see how well we did on precision, recall, and F-score. Table 4 shows our final results. The high precision on the class *against* shows that predictions for this label were mostly correct, albeit with a relatively low recall.

Stance	Precision	Recall	F-score
Favor	0.5750	0.6053	0.5897
Against	0.8770	0.5287	0.6597
Overall macro F-score			0.6247

Table 4: Precision, recall and F-score of the official submission per class as well as overall macro F-score.

At the end of section 2.3, we mentioned that we established a threshold in our system. The threshold value was set at the last minute using a rule of thumb as we did not have time to perform experiments to determine the optimal setting, or even whether it was beneficial at all. Our intention was to use this approach only for the category *Climate Change is a Real Concern*, as this was the most skewed topic. However, by accident, it was applied to *all* topics. Comparing our best result in the development phase with the test result, we can observe a substantial drop in performance. This is a result of the threshold that was – by mistake – applied in all predictions. To measure how much this affected our system, we performed an overall test run where the threshold as used in the original submission was disregarded. This resulted in a macro F-score of 0.6660 - an increase of 0.0413 relative to our submission score. The threshold proved to have lowered the recall for both *favor* and *against* and explains the low recall in the submitted system predictions.

Stance	Precision	Recall	F-score
Favor	0.5432	0.7237	0.6206
Against	0.8042	0.6378	0.7114
Overall macro F-score			0.6660

Table 5: Precision, recall and F-score of the submission without the applied threshold per class as well as overall macro F-score.

It is worth mentioning that even though the addition of all shallow features gave poor results during development phase, it performed a lot better on the test data, scoring 0.6939.

5 Conclusion

This paper summarizes our system created for SemEval 2016 task 6A - *Detecting Stance in Tweets*. Using shallow features alone performed well, but combining shallow features and word embeddings created from GloVe word vectors increased the score substantially.

With this system we finished 10th as we were able to detect stance in tweets with a macro F-score of 0.6247 on the test data, whereas the best system in task 6A scored 0.6782. Post-analysis revealed that the application of an ad-hoc threshold to prevent low-confidence predictions was a mistake, resulting in a 0.0413 loss in overall macro F-score. The threshold should have been set using cross-validation, or even better, not at all.

References

Charles Elkan. 2001. The foundations of cost-sensitive learning. In *International joint conference on artificial intelligence*, volume 17, pages 973–978. LAWRENCE ERLBAUM ASSOCIATES LTD.

Matthias Hagen, Martin Potthast, Michel Büchner, and Benno Stein. 2015. Twitter sentiment detection via ensemble classification using averaged confidence scores. In *Advances in Information Retrieval*, pages 741–754. Springer.

Clayton J Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth International Conference on Weblogs and Social Media (ICWSM-14)*, June.

Christian Kohlschütter, Peter Fankhauser, and Wolfgang Nejdl. 2010. Boilerplate detection using shallow text features. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 441–450. ACM.

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive science*, 34(8):1388–1429.

Maria Carolina Monard and Gustavo EAPA Batista. 2002. Learning with skewed class distributions. *Advances in Logic, Artificial Intelligence, and Robotics: LAPTEC 2002*, 85:173.

Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2011. English gigaword fifth edition ldc2011t07. dvd. *Philadelphia: Linguistic Data Consortium.*

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Foster Provost. 2000. Machine learning from imbalanced data sets 101. In *Proceedings of the AAAI2000 workshop on imbalanced data sets*, pages 1–3.

Yuchun Tang, Yan-Qing Zhang, Nitesh V Chawla, and Sven Krasser. 2009. Svms modeling for highly imbalanced classification. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 39(1):281–288.

Marilyn A Walker, Pranav Anand, Rob Abbott, Jean E Fox Tree, Craig Martell, and Joseph King. 2012. That is your evidence?: Classifying stance in online political debate. *Decision Support Systems*, 53(4):719–729.

Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schlkopf. 2004. Learning with local and global consistency. In *Advances in Neural Information Processing Systems 16*, pages 321–328. MIT Press.

Xiaojin Zhu and Zoubin Ghahramani. 2002. Learning from labeled and unlabeled data with label propagation. Technical report, Citeseer.

Appendix C: Data from TCP

The Consensus Project (Cook et al., 2013) is the main inspiration of this thesis and provides us with the foundational data. The data contained $11,944$ abstracts used in their work.

C.1 Data

The data provided by TCP can be downloaded[21] for free. The following data is available in text files:

1. Meta data for each paper and the ratings based on the papers' abstract (Year, Paper Title, Journal, Authors, Category rating (based on abstract), Endorsement level (based on abstract))

2. The authors ratings of their papers (Year, Abstract Endorsement Level, Self-Rated Endorsement Level)

3. The TCP team's first and second ratings. The ratings are ordered sequentially, that is, in the order that original ratings were made (Article Id #, Original endorsement rating, Original category rating, Endorsement rating after consultation stage, Category rating after consultation stage)

4. Data of 1000 "no position" abstracts that were reexamined for expressions of uncertainty about anthropogenic global warming (Article Id #, Expression of uncertainty on anthropogenic global warming. [0 = no position expressed on anthropogenic global warming. 1 = expression of uncertainty])

5. The survey protocol used by the rating team

[21]From `http://www.skepticalscience.com/tcp.php?t=home`, as of 2016-06-06

6. All the papers listed by Id number (Article Id #, Year of Publication and Paper Title)

7. All the paper abstracts (Article Id #, Year of Publication, Category, Endorsement Level, Title, and Abstract)

C.2 Abstracts

The data we have used is the last item in the list: all the paper abstracts. An example of the structure in the comma separated text file is provided below:

> Id,Year,Cat,Endorse,Title,Abstract
> 3623,1991,2,4,A 20-YEAR RECORD OF ALPINE GRASSHOPPER ABUNDANCE— WITH INTERPRETATIONS FOR CLIMATE CHANGE,A 20-year capture-recapture study of alpine grasshoppers spanned three distinct sequences of ...

Appendix D: Search & Information Retrieval

During the work of this thesis there has been a substantial amount of time dedicated to work on Search & Information Retrieval. Some of the work was very place demanding and details not necessarily of great importance for the understanding were included in this appendix. The appendix is divided into Retrieval of Abstracts from Scopus, Retrieval of Meta-Data and Retrieval of Related Data.

D.1 Retrieval of Abstracts from Scopus

When investigating how to retrieve abstracts we performed several searches by hand using the web interface. By doing this we could observe the URL and explore how to manipulate this ”by hand” (adjusting query parameters) and observing the effects. We found that the URL contained a lot of information, especially related to the possible query parameters. The list of parameters is included below. The parameters marked in bold are terms we used in our approach to create custom search URL for obtaining scientific papers in the Scopus database.

- numberOfFields
- src
- clickedLink
- edit
- editSaveSearch
- origin
- authorTab
- affiliationTab
- advancedTab
- scint
- menu
- tablin
- **searchterm1**
- field1
- dateType
- yearFrom
- yearTo
- loadDate
- documenttype
- subjects
- _subjects
- **st1**
- st2
- sot
- sdt
- sl
- **s**
- sid
- searchId
- txGid
- sort
- originationType
- rr

An example of a search URL with title ”climatic effects on the phenology of geophytes” and document type ”article” is presented beneath. The parameters *searchterm*1 and *st*1 were set equal to the title of the article, while *s* was set equal to the title with the keyword ”TITLE” before the actual title.

```
http://www.scopus.com/results/results.uri?numberOfFields=0&
src=s&clickedLink=&edit=&editSaveSearch=&origin=searchbasic&
authorTab=&affiliationTab=&advancedTab=&scint=1&menu=search&
tablin=&searchterm1=CLIMATIC+EFFECTS+ON+THE+PHENOLOGY+
OF+GEOPHYTES&field1=TITLE&dateType=Publication_Date_Type&
yearFrom=Before+1960&yearTo=Present&loadDate=7&documenttype=
Article&subjects=LFSC&_subjects=on&subjects=HLSC&_subjects=
on&subjects=PHSC&_subjects=on&subjects=SOSC&_subjects=on&
st1=CLIMATIC+EFFECTS+ON+THE+PHENOLOGY+OF+GEOPHYTES&st2=&sot=
b&sdt=b&sl=53&s=TITLE%28CLIMATIC+EFFECTS+ON+THE+PHENOLOGY+
OF+GEOPHYTES%29&sid=8239E8C2D486E08F4B873D8664BC6297.
WeLimyRvBMk2ky9SFKc8Q%3A10&searchId=
8239E8C2D486E08F4B873D8664BC6297.WeLimyRvBMk2ky9SFKc8Q%
3A10&txGid=8239E8C2D486E08F4B873D8664BC6297.
WeLimyRvBMk2ky9SFKc8Q%3A1&sort=plf-f&originationType=b&rr=
```

The process of retrieving abstract from Scopus is presented in illustrations Figure D.5 and Figure D.6. The three simple steps are as follow:

1. Search for the publication title. The system does this by generating an URL on the format shown above.

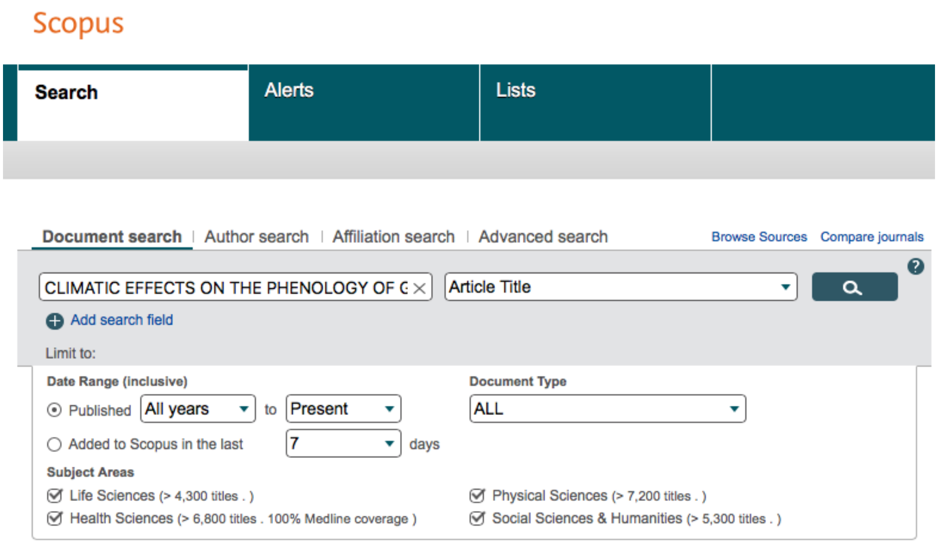
2. From the list of search results, extract the URL for the top ranked result (if any).

3. Crawl the new URL and scrape the abstract from publication page[22].

D.2 Retrieval of Meta-Data

We were not only interested in the abstracts for scientific papers. For classification (training data features) and visualization purposes meta-data was of great importance. This section presents, in greater details, the process of generating the URL used.

[22]A publication page is shown in Figure D.6


Scopus
Search
Alerts
Lists
Document search | Author search | Affiliation search | Advanced search
Browse Sources
Compare journals
Add search field

CLIMATIC EFFECTS ON THE PHENOLOGY OF G | Article Title

Limit to:

Date Range (inclusive)

Published [✓ All years] to [✓ Present]

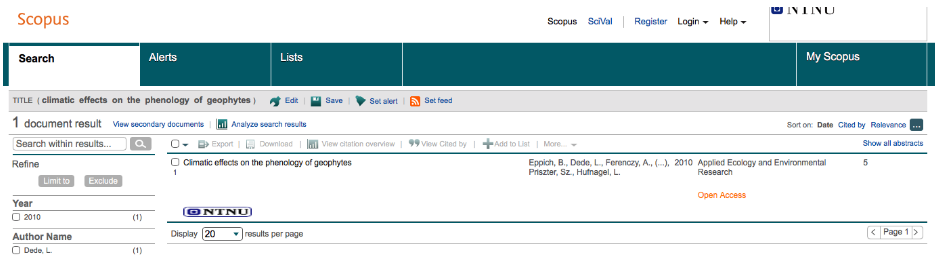
Added to Scopus in the last 7 days

Document Type: ALL

Subject Areas

✓ Life Sciences (> 4,300 titles .)
✓ Health Sciences (> 6,800 titles . 100% Medline coverage)
✓ Physical Sciences (> 7,200 titles .)
✓ Social Sciences & Humanities (> 5,300 titles .)

(a) Scopus web page.



TITLE (climatic effects on the phenology of geophytes)

1 document result

Sort on: Date Cited by Relevance

Refine

Year
2010 (1)

Author Name
Dede, L. (1)

Climatic effects on the phenology of geophytes 1	Eppich, B., Dede, L., Ferenczy, A., (...), Priszter, Sz., Hufnagel, L.	2010	Applied Ecology and Environmental Research Open Access	5

Display 20 results per page

Page 1

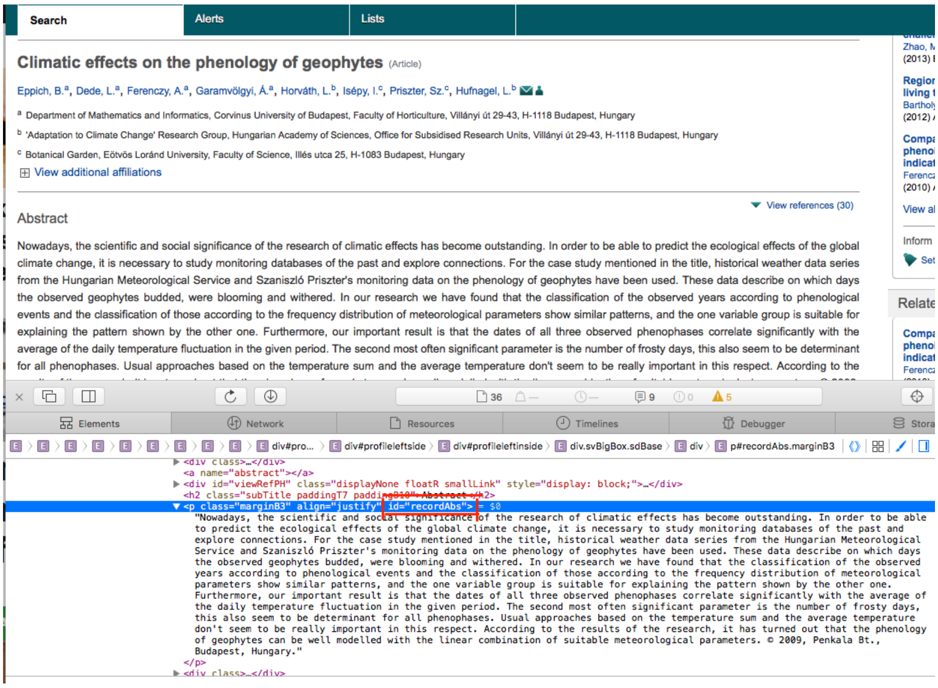
(b) Scopus web page showing returned search result.

Figure D.5: Illustration of a search and returned results on Scopus' web page

D.2.1 Approach 1: Crossref

The first approach of retrieving meta-data used the Crossref database and its API. An example of the URL for returning a JSON in HTML markup is listed below. The title used for this query was "climatic effects on the phenology of geophytes":

```
http://api.crossref.org/works?query=climatic%20effects%20on%
20the%20phenology%20of%20geophytes&rows=1&sort=score
```



Climatic effects on the phenology of geophytes (Article)

Eppich, B.[a], Dede, L.[a], Ferenczy, A.[a], Garamvölgyi, Á.[a], Horváth, L.[b], Isépy, I.[c], Priszter, Sz.[c], Hufnagel, L.[b]

[a] Department of Mathematics and Informatics, Corvinus University of Budapest, Faculty of Horticulture, Villányi út 29-43, H-1118 Budapest, Hungary

[b] 'Adaptation to Climate Change' Research Group, Hungarian Academy of Sciences, Office for Subsidised Research Units, Villányi út 29-43, H-1118 Budapest, Hungary

[c] Botanical Garden, Eötvös Loránd University, Faculty of Science, Illés utca 25, H-1083 Budapest, Hungary

⊞ View additional affiliations

View references (30)

Abstract

Nowadays, the scientific and social significance of the research of climatic effects has become outstanding. In order to be able to predict the ecological effects of the global climate change, it is necessary to study monitoring databases of the past and explore connections. For the case study mentioned in the title, historical weather data series from the Hungarian Meteorological Service and Szaniszló Priszter's monitoring data on the phenology of geophytes have been used. These data describe on which days the observed geophytes budded, were blooming and withered. In our research we have found that the classification of the observed years according to phenological events and the classification of those according to the frequency distribution of meteorological parameters show similar patterns, and the one variable group is suitable for explaining the pattern shown by the other one. Furthermore, our important result is that the dates of all three observed phenophases correlate significantly with the average of the daily temperature fluctuation in the given period. The second most often significant parameter is the number of frosty days, this also seem to be determinant for all phenophases. Usual approaches based on the temperature sum and the average temperature don't seem to be really important in this respect. According to the

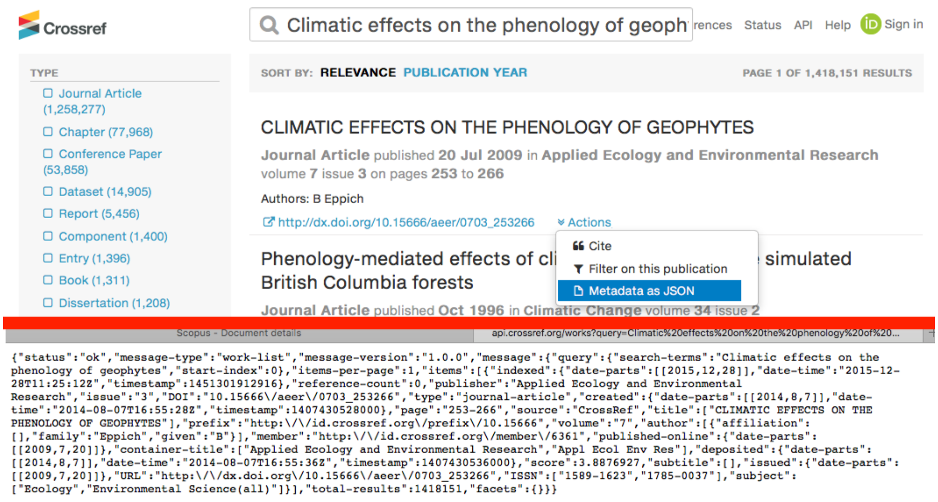
```
▶ <div class>…</div>
  <a name="abstract"></a>
▶ <div id="viewRefPH" class="displayNone floatR smallLink" style="display: block;">…</div>
  <h2 class="subTitle paddingT7 paddingB10">Abstract</h2>
▼ <p class="marginB3" align="justify" id="recordAbs"> = $0
    "Nowadays, the scientific and social significance of the research of climatic effects has become outstanding. In order to be able
    to predict the ecological effects of the global climate change, it is necessary to study monitoring databases of the past and
    explore connections. For the case study mentioned in the title, historical weather data series from the Hungarian Meteorological
    Service and Szaniszló Priszter's monitoring data on the phenology of geophytes have been used. These data describe on which days
    the observed geophytes budded, were blooming and withered. In our research we have found that the classification of the observed
    years according to phenological events and the classification of those according to the frequency distribution of meteorological
    parameters show similar patterns, and the one variable group is suitable for explaining the pattern shown by the other one.
    Furthermore, our important result is that the dates of all three observed phenophases correlate significantly with the average of
    the daily temperature fluctuation in the given period. The second most often significant parameter is the number of frosty days,
    this also seem to be determinant for all phenophases. Usual approaches based on the temperature sum and the average temperature
    don't seem to be really important in this respect. According to the results of the research, it has turned out that the phenology
    of geophytes can be well modelled with the linear combination of suitable meteorological parameters. © 2009, Penkala Bt.,
    Budapest, Hungary."
  </p>
▶ <div class>…</div>
```

Figure D.6: Visualization of how we extracted the abstracts (id="recordAbs") from the article page in Scopus.

Figure D.7 shows how a result from the Crossref API is returned and presented in a browser. The illustration is showing two things, above the red line the Crossref's web page after searching for a paper. Below the red line is the returned JSON containing the meta-data. The meta-data was retrieved by clicking the option for retrieving meta-data as JSON seen in the web interface.

D.2.2 Approach 2: Scopus

The second approach of retrieving meta-data for scientific papers was based on downloading a CSV-file from Scopus for each record. The CSV-file contained all available meta-data for the article in question, however not all articles had the equal data available. The 43 headers of potential meta-data is provided below:


Crossref

Climatic effects on the phenology of geoph

TYPE

- Journal Article (1,258,277)
- Chapter (77,968)
- Conference Paper (53,858)
- Dataset (14,905)
- Report (5,456)
- Component (1,400)
- Entry (1,396)
- Book (1,311)
- Dissertation (1,208)

SORT BY: **RELEVANCE** PUBLICATION YEAR   PAGE 1 OF 1,418,151 RESULTS

CLIMATIC EFFECTS ON THE PHENOLOGY OF GEOPHYTES

Journal Article published **20 Jul 2009** in **Applied Ecology and Environmental Research** volume **7** issue **3** on pages **253** to **266**

Authors: B Eppich

http://dx.doi.org/10.15666/aeer/0703_253266

Phenology-mediated effects of cli simulated British Columbia forests

Journal Article published **Oct 1996** in **Climatic Change** volume **34** issue **2**

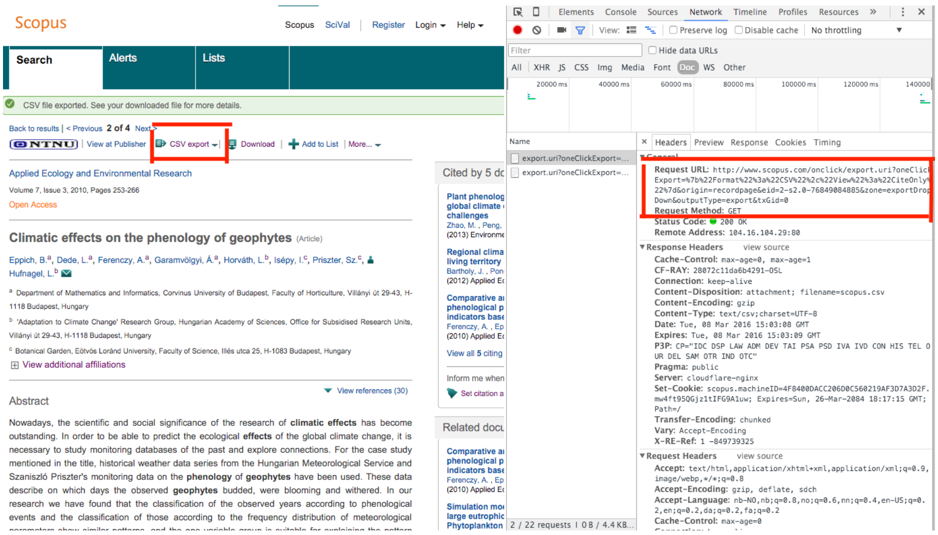
api.crossref.org/works?query=Climatic%20effects%20on%20the%20phenology%20of%20...

```
{"status":"ok","message-type":"work-list","message-version":"1.0.0","message":{"query":{"search-terms":"Climatic effects on the
phenology of geophytes","start-index":0},"items-per-page":1,"items":[{"indexed":{"date-parts":[[2015,12,28]],"date-time":"2015-12-
28T11:25:12Z","timestamp":1451301912916},"reference-count":0,"publisher":"Applied Ecology and Environmental
Research","issue":"3","DOI":"10.15666\/aeer\/0703_253266","type":"journal-article","created":{"date-parts":[[2014,8,7]],"date-
time":"2014-08-07T16:55:28Z","timestamp":1407430528000},"page":"253-266","source":"CrossRef","title":["CLIMATIC EFFECTS ON THE
PHENOLOGY OF GEOPHYTES"],"prefix":"http:\/\/id.crossref.org\/prefix\/10.15666","volume":"7","author":[{"affiliation":
[],"family":"Eppich","given":"B"}],"member":"http:\/\/id.crossref.org\/member\/6361","published-online":{"date-parts":
[[2009,7,20]]},"container-title":["Applied Ecology and Environmental Research","Appl Ecol Env Res"],"deposited":{"date-parts":
[[2014,8,7]],"date-time":"2014-08-07T16:55:36Z","timestamp":1407430536000},"score":3.8876927,"subtitle":[],"issued":{"date-parts":
[[2009,7,20]]},"URL":"http:\/\/dx.doi.org\/10.15666\/aeer\/0703_253266","ISSN":["1589-1623","1785-0037"],"subject":
["Ecology","Environmental Science(all)"]}],"total-results":1418151,"facets":{}}}
```

Figure D.7: Illustration of how meta-data was retrieved in the first approach (Crossref).

1. Authors
2. Title
3. Year
4. Source title
5. Volume
6. Issue
7. Art. No.
8. Page start
9. Page end
10. Page count
11. Cited by,
12. DOI
13. Link,
14. Affiliations
15. Authors with affiliations
16. Abstract
17. Author Keywords
18. Index
19. Keywords
20. Molecular Sequence Numbers
21. Chemicals/CAS
22. Tradenames
23. Manufacturers
24. Funding
25. Details
26. References
27. Correspondence Address
28. Editors
29. Sponsors
30. Publisher
31. Conference name
32. Conference date
33. Conference location
34. Conference code
35. ISSN
36. ISBN
37. CODEN
38. PubMed ID
39. Language of Original Document
40. Abbreviated Source Title
41. Document Type
42. Source
43. EID

The procedure explaining how to retrieve a CSV-file is shown in Figure D.8. The red square to the left indicate where we ”clicked” to retrieve the desired CSV-file, while the red square to the right show the URL used to download the CSV-file. This URL created the foundation of how we later generated an URL for retrieving meta-data. The most important thing to observe here is the use of the Scopus unique identifier (*eid*). This identifier can be obtained from the URL for the desired paper's article page (left side of Figure D.8). Details on obtaining this identifier is presented in chapter 3 in the main thesis.


Scopus
Scopus SciVal | Register Login Help
Search Alerts Lists
Back to results | < Previous 2 of 4 Next >
NTNU | View at Publisher | CSV export | Download | Add to List | More...
Cited by 5 dc
Related docu
Elements Console Sources Network Timeline Profiles Resources
View: Preserve log Disable cache No throttling
Filter Hide data URLs
All XHR JS CSS Img Media Font Doc WS Other
20000 ms 40000 ms 60000 ms 80000 ms 100000 ms 120000 ms 140000
Name
Headers Preview Response Cookies Timing

CSV file exported. See your downloaded file for more details.

Applied Ecology and Environmental Research

Volume 7, Issue 3, 2010, Pages 253-266

Open Access

Climatic effects on the phenology of geophytes (Article)

Eppich, B.[a], Dede, L.[a], Ferenczy, A.[a], Garamvölgyi, Á.[a], Horváth, L.[b], Isépy, I.[c], Priszter, Sz.[c], Hufnagel, L.[b]

[a] Department of Mathematics and Informatics, Corvinus University of Budapest, Faculty of Horticulture, Villányi út 29-43, H-1118 Budapest, Hungary

[b] 'Adaptation to Climate Change' Research Group, Hungarian Academy of Sciences, Office for Subsidised Research Units, Villányi út 29-43, H-1118 Budapest, Hungary

[c] Botanical Garden, Eötvös Loránd University, Faculty of Science, Illés utca 25, H-1083 Budapest, Hungary

View additional affiliations

View references (30)

Abstract

Nowadays, the scientific and social significance of the research of **climatic effects** has become outstanding. In order to be able to predict the ecological **effects** of the global climate change, it is necessary to study monitoring databases of the past and explore connections. For the case study mentioned in the title, historical weather data series from the Hungarian Meteorological Service and Szaniszló Priszter's monitoring data on the **phenology** of **geophytes** have been used. These data describe on which days the observed **geophytes** budded, were blooming and withered. In our research we have found that the classification of the observed years according to phenological events and the classification of those according to the frequency distribution of meteorological

Plant phenolog
global climate
challenges
Zhao, M. , Peng,
(2013) Environme

Regional clima
living territory
Bartholy, J. , Pon
(2012) Applied Ec

Comparative ai
phenological p
indicators base
Ferenczy, A. , Ep
(2010) Applied Ec

View all 5 citing

Inform me when

Set citation a

Comparative ai
phenological p
indicators base
Ferenczy, A. , Ep
(2010) Applied Ec

Simulation mo
large eutrophic
Phytoplankton

```
export.uri?oneClickExport=...
export.uri?oneClickExport=...
2 / 22 requests | 0 B / 4.4 KB...
```

```
▼General
    Request URL: http://www.scopus.com/onclick/export.uri?oneClick
    Export=%7b%22Format%22%3a%22CSV%22%2c%22View%22%3a%22CiteOnly%
    22%7d&origin=recordpage&eid=2-s2.0-76849084885&zone=exportDrop
    Down&outputType=export&txGid=0
    Request Method: GET
    Status Code: ● 200 OK
    Remote Address: 104.16.104.29:80
▼Response Headers      view source
    Cache-Control: max-age=0, max-age=1
    CF-RAY: 28072c11da6b4291-OSL
    Connection: keep-alive
    Content-Disposition: attachment; filename=scopus.csv
    Content-Encoding: gzip
    Content-Type: text/csv;charset=UTF-8
    Date: Tue, 08 Mar 2016 15:03:08 GMT
    Expires: Tue, 08 Mar 2016 15:03:09 GMT
    P3P: CP="IDC DSP LAW ADM DEV TAI PSA PSD IVA IVD CON HIS TEL O
    UR DEL SAM OTR IND OTC"
    Pragma: public
    Server: cloudflare-nginx
    Set-Cookie: scopus.machineID=4F8400DACC206D0C560219AF3D7A3D2F.
    mw4ft95QGjz1tIFG9A1uw; Expires=Sun, 26-Mar-2084 18:17:15 GMT;
    Path=/
    Transfer-Encoding: chunked
    Vary: Accept-Encoding
    X-RE-Ref: 1 -849739325
▼Request Headers      view source
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,
    image/webp,*/*;q=0.8
    Accept-Encoding: gzip, deflate, sdch
    Accept-Language: nb-NO,nb;q=0.8,no;q=0.6,nn;q=0.4,en-US;q=0.
    2,en;q=0.2,da;q=0.2,fa;q=0.2
    Cache-Control: max-age=0
```

Figure D.8: Illustration of how meta-data was retrieved from Scopus. The red square to the left indicate where we "clicked" in order to retrieve the CSV-file, while the red square to the right display the URL we would later use to obtain the CSV-file for each publication.

D.2.3 Approach 3: Web of Science

When executing a query using a python module for WoS access[23] we obtained a result formatted as a XML document. To access the information given in the document, *XPaths* had to be utilized. The XML documents contained all the available meta-data for the article in question. A complete list of possible values to be extracted are listed in Table D.15 and Table D.16.

As not all of the meta-data we retrieved contained the same information we created an overview (shown in Table D.17) that is based on the data we successfully retrieved from the WoS database. The statistic is created based on retrieval of meta-data for articles used in TCP. The statistics is presented by stance and

[23]`https://github.com/enricobacis/wos`, as of 2016-06-06.

Value	**XPath**
Authors	.../static_data/summary/names
Author Full Name	.../static_data/summary/names/name/full_name
Book Authors	.../static_data/summary/names
Group Authors	.../static_data/summary/names
Book Group Authors	.../static_data/summary/names
Document Title	.../static_data/summary/titles
ResearcherID Number	.../static_data/summary/names
Editors	.../static_data/summary/names
Publication Name	.../static_data/summary/titles
Book Series Title	.../static_data/summary/titles
Book Series Subtitle	.../static_data/summary/titles
Language	.../static_data/fullrecord_metadata/languages/language
Document Type	.../static_data/summary/doctypes
Conference Title	.../static_data/summary/conferences/conference/conf_titles/conf_title
Conference Date	.../static_data/summary/conferences/conference/conf_dates/conf_date
Conference Host	.../static_data/summary/conferences/conference/conf_host
Conference Location	.../static_data/summary/conferences/conference/conf_locations
Conference Sponsors	.../static_data/summary/conferences/conference/sponsors
Author Keywords	.../static_data/fullrecord_metadata/keywords
Keywords Plus	.../static_data/item/keywords_plus
Abstract	.../static_data/fullrecord_metadata/abstracts/abstract
Author Address	.../static_data/fullrecord_metadata/addresses
Reprint Address	.../static_data/item/reprint_contact
E-mail Address	.../static_data/summary/names/name/email_addr
Funding Agency	.../static_data/fullrecord_metadata/fund_ack/grants/grant
Funding Text	.../static_data/fullrecord_metadata/fund_ack/fund_text
Cited Reference Count	.../static_data/fullrecord_metadata/refs
Publisher	.../static_data/summary/publishers/publisher/names
Publisher City	.../static_data/summary/publishers/publisher/address_spec
Publisher Address	.../static_data/summary/publishers/publisher/address_spec

Table D.15: XPaths (part 1) for accessing values from a XML document obtained from the WoS database.

combined.

D.3 Retrieval of Related Data

As we decided to filter out a several subject we found to be unrelated to TCP we have included a complete list of which subject that were removed in the post filtering process and was therefore not included in the related data obtained from the WoS database.

Value	**XPath**
Web of Science Category	.../static_data/fullrecord_metadata/category_info
Subject Category	.../static_data/fullrecord_metadata/category_info
ISSN	.../dynamic_data/cluster_related/identifiers
ISBN	.../dynamic_data/cluster_related/identifiers
Book Digital Object Identifier (DOI)	.../dynamic_data/cluster_related/identifiers
29-Character Source Abbreviation	.../static_data/summary/titles
ISO Source Abbreviation	.../static_data/summary/titles
Publication Date	.../static_data/summary/pub_info
Year Published	.../static_data/summary/pub_info
Volume	.../static_data/summary/pub_info
Issue	.../static_data/summary/pub_info
Part Number	.../static_data/summary/pub_info
Supplement	.../static_data/summary/pub_info
Special Issue	.../static_data/summary/pub_info
Beginning Page	.../static_data/summary/pub_info/page
Ending Page	.../static_data/summary/pub_info/page
Article Number	.../dynamic_data/cluster_related/identifiers
Page Count	.../static_data/summary/pub_info/page
Chapter Count in a Book	.../static_data/item/book_chapters
Digital Object Identifier (DOI)	.../dynamic_data/cluster_related/identifiers
Document Delivery Number	.../static_data/item/ids
Accession Number	.../UID

Table D.16: XPaths (part 2) for accessing values from a XML document obtained from the WoS database.

- **Acoustics**
- **Anatomy & Morphology**
- **Andrology**
- **Area Studies**
- **Art**
- **Artificial Intelligence**
- **Asian Studies**
- **Biomedical Social Sciences**
- **Business**
- **Business & Economics**
- **Cardiac & Cardiovascular Systems**
- **Cardiovascular System & Cardiology**
- **Cell & Tissue Engineering**
- **Clinical**
- **Communication**
- **Criminology & Penology**
- **Cultural Studies**
- **Demography**
- **Dentistry**
- **Developmental**
- **Educational**
- **Emergency Medicine**
- **Ergonomics**
- **Experimental**
- **Family studies**
- **Film**
- **Finance**
- **Gastroenterology & Hepatology**
- **Government & Law**

	Stance			
Properties	**None**	**Against**	**Favor**	**All**
Average word length of title	13	10	12	13
Unique words in title	20150	413	10293	24895
Unique publication years	21	21	21	21
Number of languages	15	2	13	18
Unique reference counts	184	39	142	198
Highest count	390	86	370	390
Average reference count	42.39	28.58	36.40	40.39
Unique organization countries	128	18	109	137
Unique organization cities	2480	62	1475	2926
Average publication length	10.95	9.55	11.17	11.01
Max/min publication length	84/1	25/3	51/1	84/1
Number of contributing authors	27884	129	12612	40625
Unique authors	21555	115	9728	28840
Unique headers	5	3	5	5
Unique sub-header	3	2	3	3
Unique Document types	10	2	6	10
Unique Publication types	10	2	6	10

Table D.17: This table provide some statistics of TCP meta-data that was able in the WoS database.

- **Hardware & Architecture**
- **History of Social Sciences**
- **Hospitality**
- **Industrial Relations & Labor**
- **International Relations**
- **Language & Linguistics**
- **Law**
- **Legal**
- **Leisure**
- **Linguistics**
- **Logic**
- **Medical Ethics**
- **Medical Laboratory Technology**
- **Microscopy**
- **Music**
- **Neuroimaging**
- **Oncology**
- **Oral Surgery & Medicine**
- **Orthopedics**
- **Peripheral Vascular Disease**

- **Planning & Development**
- **Political Science**
- **Primary Health Care**
- **Public Administration**
- **Psychoanalysis**
- **Radio**
- **Radio & Television**
- **Religion**
- **Rheumatology**
- **Robotics**
- **Social Work**
- **Special**
- **Sport & Tourism**
- **Substance Abuse**
- **Telecommunications**
- **Television**
- **Textiles**
- **Women's Studies**

Appendix E: Additional Experiments and Results

This appendix aim to provide additional information on experiments that have been conducted as part of our work on this thesis. The experiments described here concern the three experimental chapter of 4, 5 and 6. Where no *Setup* description is provided we refer the reader to the appropriate chapter in the main thesis for details as only additional results are listed for these experiments.

E.1 Additional Experiments For Chapter 4

E.1.1 Determining The Rate Limit

The experiments described here concern the efficiency of web crawlers and whats requires for them to operate efficiently.

Setup

To be able to evaluate the experiments we needed a data set. We therefore selected a random sub-sample of abstracts to extract. This sub-sample will be used in all the experiments regarding determining the rate limit. The data set consisted of 20 titles where 15 of them are available through Scopus.

These experiments aim to determine some parameters that are of importance for web crawlers politeness policy. Our web crawlers interact with servers we do not own or have specific rights to use, therefore we needed to make sure that our crawlers behaved well and did not overload the servers. The key to achieve this is rate limiting. To determine the best rate limit we have conducted a set of trial and error experiments.

The goal was to find a reasonable rate of requests to the target domain (in our case `www.scopus.com`). If we failed to identify an accepted rate or simply neglected to do so, we would behave unethical as this can be considered a denial of service attack. The acceptable rate varies from server to server. We were expecting a rate of between 1 and 5 request per second as a reasonable rate.

The HTML responses we wanted to avoid were 401 (unauthorized), 403 (forbidden), and 503 (service unavailable). Since we never want to be completely blocked (401 & 403) we performed the tests with this in mind. By retrieving as few abstracts as possible, but still being able to see the affects, and stay on the lookout for 503 messages, we hoped to avoid getting blocked. We were reviewing 503 messages as these would indicate whether our requests put pressure on the domain server. There is no guarantee that the 503 messages were related to our crawling. But, as you will see in the results, they were showing a strong correlation with when we pushed the limits of what we expected the server to accept.

There was two particular parameters we wanted to explore the effects of (in regard to the performance of the web crawler) in this experiment. First; the maximum number of concurrent requests, and second; the delay between two request to the same page. To determine the effects of these parameters, we conducted three separate experiments.

1. **Number of concurrent requests**
   Adjust only the maximum number of concurrent requests, leaving the delay setting to default for the Scrapy framework. Strategy: initiate with only one request at the time and leaving the delay between request to default (0 seconds). Proceed by increasing the number of concurrent request until we receive HTML 503 messages from the server.

2. **Delay between requests**
   Adjust only the delay, leaving the maximum number of concurrent request to the default setting (16 concurrent threads). Strategy: start off with a high delay (5 seconds) and decrease the delay until we receive HTML 503 messages from the server.

3. **Combination of settings**
   Determine the best combination of the two settings. Strategy: by using the results from the experiments listed above, find the best possible combination by trial and error. Commence with the best results from the experiments to maximize the number of concurrent requests, and minimize the delay. Adjusting one parameter at the time.

As the test can be affected by factors we are not able to control, such as the load on the servers from other users, we are doing what we can to give each experiment the same conditions. This means that each experiment is conducted at the same time of the day, and they are spread over two days performing 4 each day (with different IP-addresses). All experiments are performed two times per set up to ensure that the results were representative.

Results

Determining The Rate Limit

The results will be presented for each separate sub-experiment.

1. **Number of concurrent requests**

As shown in Table E.18 no problems occurred using 1, 2, or 3 concurrent requests, but as we increased the number to 4, a lot of 503 messages was returned. This shows that the number of concurrent requests do in fact matter significantly for the performance of the web crawler.

Number of concurrent requests	Abstracts searched for	Abstracts retrieved	Request count	Respons count	503 count
6	20	4	62	62	42
4	20	4 / 5	59 / 56	59 / 56	42 / 39
3	20	15	38	38	0
2	20	15	37	37	0
1	20	15	36	36	0

Table E.18: Result of experiment to determine the maximum number of concurrent request. When two numbers are listed it is because the results varied between the first and second test.

2. **Delay between requests**

Table E.19 shows that no problems occurred when the the delay was larger than 0, but when there was no delay we observed a lot of 503 messages. This shows that the delay has an even bigger impact than the number of concurrent requests on the performance of the web crawler.

3. **Combination of settings**

The results are shown in Table E.20. From this table we can observe that the number of concurrent threads do not really matter regarding to the number of 503 messages when a delay is set. However it does seem to result in more requests

Delay (in seconds)	Abstracts searched for	Abstracts retrieved	Request count	Respons count	503 count
5	20	15	51	51	0
4	20	15	51	51	0
3	20	15	51	51	0
2	20	15	51	51	0
1	20	15	51	51	0
0	20	5	71	71	30

Table E.19: Result of experiment to determine the delay between requests to the same page.

being sent which is something we want to avoid. Since we initially wanted to maximize the number of concurrent threads while minimizing the delay between requests, it is reasonable to say that a delay of one second is appropriate while setting the number of concurrent threads to one.

Delay (in seconds)	Number of concurrent threads	Abstracts searched for	Abstracts retrieved	Request count	Respons count	503 count
2	20	20	15	55	55	0
1	10	20	15	45	45	0
1	8	20	15	43	43	0
1	5	20	15	40	40	0
1	4	20	15	39	39	0
1	3	20	15	38	38	0

Table E.20: Result of experiment to determine the combination of delay between requests to the same page and the number of concurrent requests.

Linear SVM

	Development			Validation		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	34.78	19.51	25.00	22.22	11.76	15.38
Favor	57.73	58.02	57.87	59.66	60.55	60.10
None	79.10	79.25	79.17	80.43	80.30	80.31
Macro F:			54.02			51.93

Stochastic Gradient Descent

	Development			Validation		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	11.54	07.32	08.96	08.82	17.65	11.76
Favor	64.69	55.96	60.01	62.19	65.45	63.78
None	79.44	84.96	82.11	82.83	79.92	81.35
Macro F:			50.36			52.30

Table E.21: Tuned baseline results with classifiers: Linear SVM and Stochastic Gradient Descent.

E.2 Additional Experiments For Chapter 5

E.2.1 Optimized Learning Classifiers

Results

Table E.21 show additional results for experiments conducted for optimized learning classifiers.

E.2.2 Shallow Features

Setup

Meta-data resulting from the Search & Information Retrieval section was used in features in hope for a performance boost of the system. The SVM classifier did not show a substantial difference in score by the ablation strategy. The development scored steady around 0.47-0.48 while the validation score was between

0.48-0.49. The Multinomial NB illustrated a trend of decreasing score when increasing features.

It was observed that author information (names) feature and organization address (country) dropped performance significantly. For instance MNB dropped down from 0.43 to 0.18 while LinearSVM dropped from 0.48 to 0.41 on the validation macro F-score. This observation had us ignore similar feature affects in any further experimenting regarding this experiment. However, none of the experiments carried out in this section showed any improvement to the system.

Results

Meta-data in resulting from the Search & Information Retrieval section was used in features in hope for a performance boost of the system. Results are shown in Table E.22. The table illustrate a couple of examples we explored, but it should be representative for this approach.

Classifier	**# of features**	**Development**	**Validation**
SVM	18	47.88	48.58
SVM	17	47.99	48.77
SVM	16	48.03	48.84
⋮	⋮	⋮	⋮
SVM	12	48.52	49.53
SVM	11	48.52	49.56
SVM	10	48.57	49.51
⋮	⋮	⋮	⋮
SVM	6	48.74	50.37
SVM	5	48.31	49.39
SVM	4	48.73	49.87
MNB	18	41.50	42.80
MNB	17	41.29	42.68
MNB	16	41.48	43.13
⋮	⋮	⋮	⋮
MNB	12	45.91	47.39
MNB	11	45.90	47.40
MNB	10	45.91	47.36
⋮	⋮	⋮	⋮
MNB	6	46.32	47.61
MNB	5	46.27	47.54

MNB	4	47.20	49.40

Table E.22: Table of results from applying meta data features as additional data

Table E.23 display the best result found by adding additional meta data. For the SVM classifier, the following list of features were included:
['subject-feat', 'sub-header-feat', 'header-feat', 'type-feat', 'doc-type-feat', 'count-vect']

For the Multinomial NB, the following features were added:

['tokens-title-feat', 'LDA-abstract-feat', 'title-feat', 'count-vect']

Shallow features with SVM

	Development			**Validation**		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	00.00	00.00	00.00	00.00	00.00	00.00
Favor	68.36	57.87	62.68	70.01	62.83	66.23
None	80.32	87.04	83.55	82.52	87.40	84.89
	Macro F:		48.74	Macro F:		50.37

Shallow features with Multinomial NB

	Development			**Validation**		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	00.00	00.00	00.00	00.00	00.00	00.00
Favor	69.05	50.59	58.39	72.33	56.33	63.33
None	78.08	89.05	83.20	80.39	89.85	84.86
	Macro F:		47.20	Macro F:		49.40

Table E.23: Best shallow feature result with two classifiers: Multinomial NB and SVM

E.2.3 Binary Classification

Results

Table E.24 contains additional results for experiments conducted with binary

Multinomial NB

Development

Step 1: Stance vs No Stance					**Step 2: Favor vs Against**					**Final**
Stance	Prec	Recall	F1	Support	Stance	Prec	Recall	F1	Support	F1
None	82.04	82.33	82.19	4183	None	00.00	00.00	00.00	739	82.19
Stance	64.33	63.87	64.10	2087	Favor	63.70	97.95	77.20	1317	77.20
					Against	34.04	100.0	50.79	16	50.79
Macro F:			73.14	6270	Macro F:			64.00	2072	Macro F: **70.06**

Validation

Step 1: Stance vs No Stance					**Step 2: Favor vs Against**					**Final**
Stance	Prec	Recall	F1	Support	Stance	Prec	Recall	F1	Support	F1
None	83.59	84.38	83.99	1793	None	00.00	00.00	00.00	280	83.99
Stance	68.07	66.78	67.42	894	Favor	67.31	100.0	80.46	589	80.46
					Against	00.00	00.00	00.00	8	00.00
Macro F:			75.70	2687	Macro F:			40.23	877	Macro F: **54.82**

Table E.24: Binary classification results in a two-step procedure: First detect stance vs no stance then classify favor vs against. Carried out using SVM and Multinomial

classification.

Inspecting Multinomial NB we notice that the performance of the first step is significantly higher in both development and validation experiment. 73.14 and 75.70 compared to SVM's 59.98 and 65.99. The development score increased due to classification of the *against* label. The second step show a high final score in development with 70.06 but drops down to 54.82 in the validation set. This might be a result of overfitting.

E.2.4 Word embeddings

Resuts

Table E.25 provide details on additional results obtained when conducting experiments with GloVe features. Table E.26 contains results from experiments conducted with word2vec features. And Table E.27 contains reusults from experiments investigating the performance of using both GloVe and word2vec features.

E.2.5 Artificial Neural Network

Results

We wanted to take a look at the problem from a new angle, using neural networks to train data instead of the previous applied classifiers.Table E.28 show results from three different types neural networks: Convolutional Neural Network

Glove(glove.840B.300d)

	Development			**Validation**		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	10.37	34.15	15.91	15.69	47.06	23.53
Favor	58.30	74.00	65.22	60.91	77.65	68.27
None	85.53	72.34	78.38	87.62	74.18	80.34
Macro F:			53.17			57.38

Voting: Glove(840B.300d), SVM and MNB

	Development			**Validation**		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	23.08	07.32	11.11	25.00	05.88	09.52
Favor	65.18	71.26	68.08	67.04	74.46	70.56
None	84.80	81.50	83.12	86.60	82.54	84.52
Macro F:			54.10			54.87

Table E.25: Additional results from GloVe word embedding.

Word2vec

	Development			Validation		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	05.71	24.39	09.26	06.67	29.41	10.87
Favor	55.11	66.72	60.36	54.33	66.59	59.84
None	81.73	70.69	75.81	81.52	69.88	75.26
Macro F:			48.48			48.65

Voting: Word2vec, SVM and Multinomial NB

	Development			Validation		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	31.82	17.07	22.22	25.00	11.76	16.00
Favor	61.61	72.63	66.67	63.93	75.60	69.28
None	84.91	77.86	81.23	86.78	79.48	82.97
Macro F:			56.71			56.08

Table E.26: Additional results from word2vec word embedding.

Logistic Regression: Word2vec and Glove (840B.300d)

	Development			Validation		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	10.64	36.59	16.48	13.79	47.06	21.33
Favor	60.30	74.10	66.49	62.52	77.42	69.18
None	85.86	74.21	79.61	87.75	75.52	81.18
Macro F:			54.19			57.23

Glove, Word2vec, SVM and MNB

	Development			Validation		
Stance	Prec	Recall	F1	Prec	Recall	F1
Against	19.40	31.71	24.07	20.00	35.29	25.53
Favor	63.19	70.97	66.85	66.39	73.20	69.63
None	84.61	78.99	81.70	86.21	81.26	83.66
Macro F:			57.54			59.61

Table E.27: Results from applying both word embeddings: Word2vec and Glove.

	CNN (on chars)			CNN (on words)			RNN (on words)		
	Validation			Validation			Validation		
Stance	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1
Against	00.00	00.00	00.00	00.00	00.00	00.00	04.17	05.88	04.88
Favor	71.66	70.38	71.02	75.02	87.12	80.62	76.60	86.73	81.35
None	42.01	44.36	43.15	60.66	41.85	49.53	63.19	45.61	52.98
Macro F:			38.06			43.38			46.40

Table E.28: Neural network classification results with three different network: Convolutional neural network on characters, on words and recurrent neural network on words

(on characters), Convolutional Neural Network (on words) and Recurrent Neural Networks.

The CNN (on characters) perform worst of the three approaches with 0.3806 as validation score. The second perform a better with a validation score of 0.4338 while the last approach received the best validation score of 0.4640. These scores are significant lower than scores from previous experiments, which imply no boost in system performance. We included the loss graph for which the neural network was stopped. The stopping criteria was depending on the minimized loss between each step.

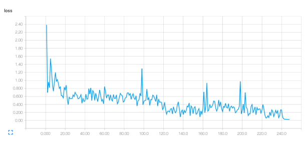


Figure E.9: Training loss CNN words

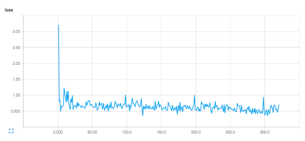


Figure E.10: Training loss CNN chars

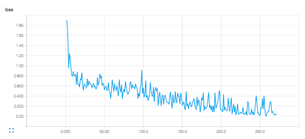


Figure E.11: Training loss RNN words

E.2.6 Abstracts From Recent Climate Data

In a (most likely biased) selection process we extracted 6 abstracts from the retrieved recent climate data along with their predicted labels. The first two abstracts were labeled **in favor**. We evaluate the first as both relevant for climate science and most likely correctly labeled, while the second not. As we are not domain experts we leave it to the reader to evaluate the remaining abstracts.

Title:	Sea Level Changes Past Records and Future Expectations
Reference:	Rörsch et al. (2013)
Predicted label:	Against
Abstract:	The history and development of our understanding of sea level changes is reviewed. Sea level research is multi-fascetted and calls for integrated studies of a large number of parameters. Well established records indicate a post-LIA (1850-1950) sea level rise of 11 cm. During the same period of time, the Earth's rate of rotation experienced a slowing down (deceleration) equivalent to a sea level rise of about 10 cm. Sea level changes during the last 40-50 years are subjected to major controversies. The methodology applied and the views claimed by the IPCC are challenged. For the last 40-50 years strong observational facts indicate virtually stable sea level conditions. The Earth's rate of rotation records a mean acceleration from 1972 to 2012, contradicting all claims of a rapid global sea level rise, and instead suggests stable, to slightly falling, sea levels. Best estimates for future sea level changes up to the year 2100 are in the range of +5 cm +/- 15 cm.

Title:	Testing General Relativistic Predictions with the LAGEOS Satellites
Reference:	Peron (2014)
Predicted label:	Against
Abstract:	The spacetime around Earth is a good environment in order to perform tests of gravitational theories. According to Einstein's view of gravitational phenomena, the Earth mass-energy content curves the surrounding spacetime in a peculiar way. This (relatively) quiet dynamical environment enables a good reconstruction of geodetic satellites (test masses) orbit, provided that high-quality tracking data are available. This is the case of the LAGEOS satellites, built and launched mainly for geodetic and geodynamical purposes, but equally good for fundamental physics studies. A review of these studies is presented, focusing on data, models, and analysis strategies. Some recent and less recent results are presented. All of them indicate general relativity theory as a very good description of gravitational phenomena, at least in the studied environment.

Title:	Assessment of balanced burden-sharing in the 2050 EU climate/energy roadmap: a metamodeling approach
Reference:	Babonneau et al. (2015)
Predicted label:	Favor
Abstract:	In this paper we propose a non-cooperative meta-game approach to designing and assessing climate agreements among 28 European countries that will be compatible with the EU 2050 climate target. Our proposed game model is identified through statistical emulation of a large set of numerical simulations performed with the computable general equilibrium model GEMINI-E3. In this game, the players are the 28 European countries, the payoffs are related to welfare losses due to abatements and the strategies correspond to the supply of emission rights on the European carbon market. We show it is possible to design a fair burden-sharing rule that equalizes welfare losses between countries to approximately 1.2 % of their discounted household consumption. The associated European CO2 price in 2050 reaches $1100, a figure in line with previous studies. Lastly, the paper discusses various implementation issues of these types of negotiations and evaluates the cost of non-cooperation among EU countries.

Title:	Catalytic Transformations of Ethanol for Biorefineries
Reference:	Gallo et al. (2014)
Predicted label:	Favor
Abstract:	Brazil and the USA are the major bioethanol producers in the world, and the main application of this alcohol is as fuel. Since Brazilian ethanol is the cheapest in the world, there is a crescent interest in its use as a building block for biorefineries. Bioethanol can be used for the direct production of drop-in chemicals, such as ethylene, propylene, 1,3-butadiene and larger hydrocarbons, as well as for the production of oxygenated molecules, such as 1-butanol, ethyl acetate, acetaldehyde, and acetic acid. In this critical review, the development of heterogeneous catalysts for the conversion of ethanol into these commodity chemicals will be discussed.

Title:	The blacker elder plant as an indicator of climate change: analysis of time series and verification of the IPHEN phenological model
Reference:	Alilla et al. (2013)
Predicted label:	None

Abstract:	The monitoring of black elder flowering rhythms has provided important information on climate warming occurred in the Northern mid-latitudes. In this regard, the analyses carried out on flowering dates of black elder gathered in the Italian Phenological Database, show a discordant situation. In fact, in the Phenological Garden of S. Pietro Capofiume (BO), which has the longest time series, we observe an advance of flowering of about 1 Idly] from 1990 to 2011, while no signal there is in phenological data registered in the Phenological Garden of Fontanella-S. Apollinare (PG), whose time series is moreover shorter (1997-2011). However, flowering dates observed in S. Apollinare are significantly correlated with the monthly averages (January to April) of daily minimum and maximum temperatures, with an advance of 7.5 [d] for each C of the minimum temperature increase and 5.8 [d/C] of the maximum. Phenological data collected at these sites were used to validate the IPHEN phenological model. Validation has shown that the model, with the current parameters, does not simulate with sufficient accuracy the elder blooming dates in these Gardens; in particular, in the case of S. Pietro Capofiume in the years 2003-06, compared with an advance in flowering observed data, the model estimated a delay of onset of phases. The model was then recalibrated and validated on independent data sets and thus it provided evidently better performance in both Gardens.
Title:	Trade-offs drive resource specialization and the gradual establishment of ecotypes
Reference:	Østman et al. (2014)
Predicted label:	None
Abstract:	Background: Speciation is driven by many different factors. Among those are trade-offs between different ways an organism utilizes resources, and these trade-offs can constrain the manner in which selection can optimize traits. Limited migration among allopatric populations and species interactions can also drive speciation, but here we ask if trade-offs alone are sufficient to drive speciation in the absence of other factors.

Table E.29: A selection of abstracts are included to speculate the relevancy and prediction of the system.

Appendix F: Grid Search

When performing grid search to optimize classifiers we used the parameters described in this appendix. The parameters are divided by pre-processing and classifiers.

F.1 Pre-processing

CountVectorizer

1. *N-gram*: (1, 1), (1, 2), (1, 3) or (2, 3)
2. *Stop words*: none/English
3. *Max # features*: none/50000
4. *Analyzer*: word/char

TfidfTransformer

1. *Use idf*: true/false

F.2 Classifiers

F.2.1 MNB + BNB

Fit prior: true/false

Alpha: [0.1, 0.17782794, 0.31622777, 0.56234133, 1.]

MNB and BNB was tested with different values of the smoothing parameter *alpha* and whether or not to learn class prior probabilities (*fit prior*). If set to false, a uniform prior is applied.

F.2.2 SGD

Loss: ['modified_huber', 'squared_hinge', 'perceptron']

Alpha: [0.0001, 0.001, 0.01]

SGD was tested for two parameters; different values of the smoothing parameter alpha and *loss* functions.

F.2.3 SVM

C: [0.1, 0.16378937, 0.26826958, 0.43939706, 0.71968567, 1.17876863, 1.93069773, 3.16227766, 5.17947468, 8.48342898, 13.89495494, 22.75845926, 37.2759372, 61.05402297, 100]

Kernel: ['rbf', 'linear', 'poly', 'sigmoid']

Decision function shape (multi class for linear SVM): ['ovo', 'ovr']

The *kernel* decides how the classifier maps values from one space to another. *C* parameter is a regularization term for misclassifications of each sample. Higher values will do a better job correctly labeling the training data (low hyperplane margin). Conversely, lower values may have more mis-classifications, but may be more robust toward outliers (larger hyperplane margin). *Decision function shape* parameter decide whether to use one-vs-one (ovo) or one-vs-rest (ovr) as decision function. Ovo constructs one classifier per pair of classes. At prediction time, a vote is performed and the class which receives the most votes is selected. The *ovr* strategy consist of fitting one classifier for each class. Linear SVM were optimized using the same parameters excluding *kernel.*

F.2.4 Logistic Regression

C: np.logspace(-1, 1.3, 6)

Solver: ['liblinear', 'lbfgs']

Multi class: ['ovr', 'multinomial']

The logistic regression was tested with *C* tuning, different types of *solvers* (algorithms used in optimization problem, for instance newton) and types of *penalty.*

Appendix G: Pre-trained Word Vectors

G.1 GloVe

All the pre-trained word vectors we have utilized are available for download through the official GloVe web page[24].

1. Wikipedia 2014 + Gigaword 5 (6B tokens, 400K vocab, uncased, 50d, 100d, 200d, & 300d vectors, 822 MB download): glove.6B.zip
2. Common Crawl (42B tokens, 1.9M vocab, uncased, 300d vectors, 1.75 GB download): glove.42B.300d.zip
3. Common Crawl (840B tokens, 2.2M vocab, cased, 300d vectors, 2.03 GB download): glove.840B.300d.zip
4. Twitter (2B tweets, 27B tokens, 1.2M vocab, uncased, 25d, 50d, 100d, & 200d vectors, 1.42 GB download): glove.twitter.27B.zip

G.2 word2vec

The pre-trained word vector is available for download at the Google Code - word2vec web page[25]

1. Part of Google News dataset (100B tokens, 3M vocab, 300d vectors, 1.65 GB download): GoogleNews-vectors-negative300.bin.gz

[24]`http://nlp.stanford.edu/projects/glove/`, as of 2016-05-16
[25]`https://code.google.com/archive/p/word2vec/`