Martin-Luther-Universität Halle-Wittenberg
Naturwissenschaftliche Fakultät III
Institut für Informatik
Studiengang Informatik

Stance Classification for Answering Comparative Questions

Master's Thesis

Niklas Homann
Born September 19, 1995 in Braunschweig
Matriculation Number 218242283

1. Referee: Prof. Dr. Matthias Hagen
2. Referee: Alexander Bondarenko, M.Sc.

Submission date: December 21, 2020

Declaration

Unless otherwise indicated in the text or references, this thesis is entirely the product of my own scholarly work.

Halle, December 21, 2020

. . . .
Niklas Homann

Abstract

Comparative questions, i.e., questions that ask for a comparison of different alternatives, are everywhere in our daily life. They express the comparative information need which is underlying decision making. Often, answers to these questions contain pro and con arguments for the compared options. In this thesis, we deal with the identification of the polarity of such argumentative answers, i.e., the detection of the stance towards the comparison.

We present a new dataset for stance detection of answers to comparative questions consisting of 1,034 questions and answers annotated with the stance targets and the answer's stance. Besides analyzing methods to identify those targets, we focus on exploring approaches to detect the stance of the answers. We evaluate different ways to employ pre-trained transformer-based models for this task. One of these ways is the application of the idea of sentiment flow for stance detection and breaking the classification down to sentence-level. We propose a classifier based on RoBERTa that manages to correctly predict the stance given by four labels (*none*, *neutral*, *first*, *second*) with an accuracy of 0.62 on our dataset.

Contents

Acknowledgments

First, I would like to thank professor Matthias Hagen for giving me the chance to write my thesis in his research group about this interesting topic. My biggest thanks go to Alexander Bondarenko for the fruitful collaboration. His supervision, constructive feedback, and suggestions were essential for the success of this thesis. I would also like to thank Yamen Ajjour for the regular exchange and his helpful thoughts and ideas. Furthermore, I would like to thank Ekaterina Schirschakova for helping Alexander and me annotate the questions.

Thanks to my brother Daniel for proofreading this thesis and the idea to use the Stack Exchange dumps. My last thanks are due to Oleksandra for her constant support, especially in these last weeks.

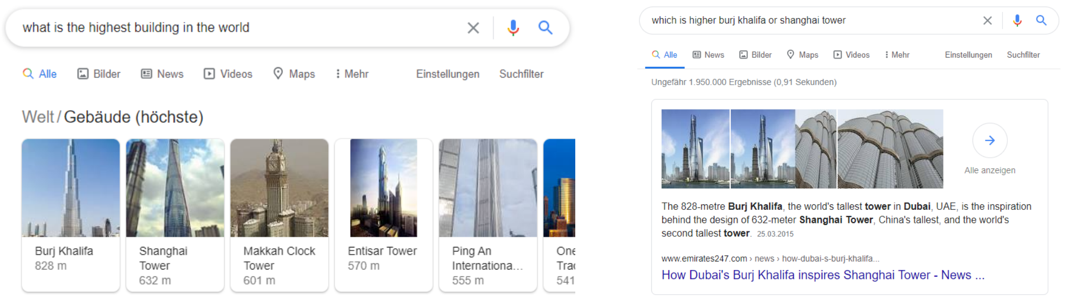
Chapter 1

Introduction

In daily life, people constantly make decisions and informed choices between different options. Nowadays, a vast amount of knowledge, both facts and opinions, is accumulated on the web, which can be accessed with ease using modern technologies. The abundance of human-generated opinions on the web might in fact complicate decision making let alone that opinions can be skewed or one-sided. Hence, identifying a side of opinion (pro or con) towards one or another option might help web technologies, e.g., search engines, to present users a more comprehensive and diverse overview of alternatives.

This thesis deals with the task of identifying the stance of answers to comparative questions, i.e., questions that request for a comparison of different options. We, thus, focus on the following research question: How well can we reliably identify the position of an answer's author towards alternatives provided in a comparative question?

To make informed choices people look into the alternatives, collect arguments for and against the possible options, compare them, and finally decide for one side. Comparative questions asking for a comparison of at least two options often stand in the center of such a decision-making process. These questions can be about product choices (e.g., "Which smartphone should I buy?"), about important life choices (e.g., "Is university A or university B better for Computer Science?"), and much more. Although search engines have advanced from plain information retrieval systems to sophisticated systems that attempt to present the most valuable information to a searcher directly, they still struggle to do this for some comparative questions. For factual questions, including comparative ones, they will often directly present the relevant (and often correct) answer in front of any other search results. In Figure 1.1 featured snippets (i.e., direct answers) are shown exemplarily for the Google search engine.



what is the highest building in the world

Welt / Gebäude (höchste)

Burj Khalifa 828 m
Shanghai Tower 632 m
Makkah Clock Tower 601 m
Entisar Tower 570 m
Ping An Internationa... 555 m
One Tra... 541

which is higher burj khalifa or shanghai tower

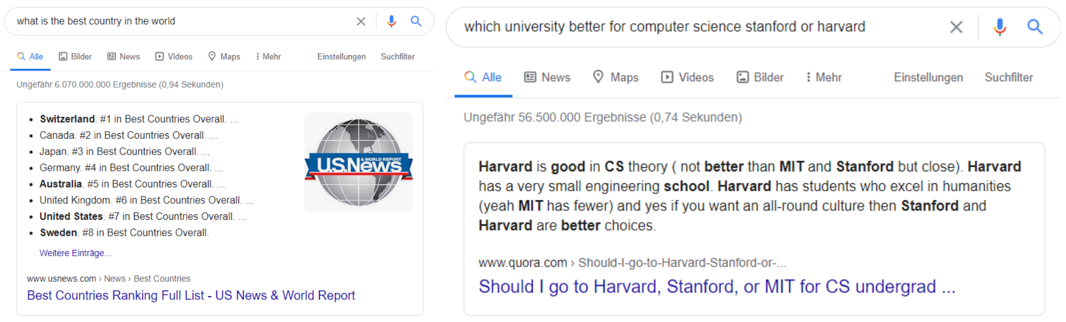
Ungefähr 1.950.000 Ergebnisse (0,91 Sekunden)

The 828-metre **Burj Khalifa**, the world's tallest **tower** in **Dubai**, UAE, is the inspiration behind the design of 632-meter **Shanghai Tower**, China's tallest, and the world's second tallest **tower**. 25.03.2015

www.emirates247.com › news › how-dubai-s-burj-khalifa...
How Dubai's Burj Khalifa inspires Shanghai Tower - News ...

Figure 1.1: Example answers from Google Search for factual questions: The highest building in the world (left) and the higher building among the Burj Khalifa and the Shanghai Tower (right).

However, Bondarenko et al. [7] showed that at least for the queries submitted to the Russian search engine Yandex, the majority of the comparative questions were argumentative or opinionated.[1] For this type of question, Google Search will not go beyond extracting a relevant snippet and providing the links to the websites as shown in Figure 1.2. Such a result presentation only shows a single opinion, does not provide a summary or different arguments, and gives no overall stance of arguments in the answer.



what is the best country in the world

Ungefähr 6.070.000.000 Ergebnisse (0,94 Sekunden)

- **Switzerland**. #1 in Best Countries Overall. ...
- Canada. #2 in Best Countries Overall. ...
- Japan. #3 in Best Countries Overall. ...
- Germany. #4 in Best Countries Overall. ...
- **Australia**. #5 in Best Countries Overall. ...
- United Kingdom. #6 in Best Countries Overall. ...
- **United States**. #7 in Best Countries Overall. ...
- **Sweden**. #8 in Best Countries Overall.

Weitere Einträge...

www.usnews.com › News › Best Countries
Best Countries Ranking Full List - US News & World Report

which university better for computer science stanford or harvard

Ungefähr 56.500.000 Ergebnisse (0,74 Sekunden)

Harvard is **good** in **CS** theory (not **better** than **MIT** and **Stanford** but close). **Harvard** has a very small engineering **school**. **Harvard** has students who excel in humanities (yeah **MIT** has fewer) and yes if you want an all-round culture then **Stanford** and **Harvard** are **better** choices.

www.quora.com › Should-I-go-to-Harvard-Stanford-or-...
Should I go to Harvard, Stanford, or MIT for CS undergrad ...

Figure 1.2: Example answers from Google Search for argumentative questions: The best country in the world (left) and the better university for computer science among Stanford University and Harvard University (right).

The web is filled with human-written text comparing different options such as answers on community question answering (CQA) platforms like Quora, online articles, or discussions in debate forums. For particular products, there are even special online platforms comparing different alternatives.[2] These websites, however, mostly address a specific narrow domain, can be unknown to

[1] Due to the strong overlap and similarity, we will be following Bondarenko et al. [7] and merge these into one question type. For simplicity, we will call the answer and question argumentative, if the question asks for an opinionated and/or argumentative answer.

[2] `www.check24.de`, `www.pcmag.com`, etc.

a searcher, and can provide a biased opinion of review authors. Web search engines have access to all this textual information on the web, but they do not account for the argumentative and comparative nature of queries and extracted information. Direct answers often lack diversity, and a searcher still has to follow different links and read through multiple sources.

Several argument search engines have been developed to tackle argumentative information need and to improve the search experience for such queries. However, they often retrieve documents from the limited and outdated web crawls like Common Crawl and ClueWeb.[3] For instance, args.me [69] retrieves relevant argumentative text passages to a given (controversial) query and sorts them into pro and con columns based on the explicit labels provided by the argument authors. However, its retrieval coverage is limited to a crawl from several online debate portals. ArgumenText [60] retrieves only single sentences from the Common Crawl and classifies their pro/con stance. In contrast, TARGER [10] retrieves text document passages from the Common Crawl, tags argumentative claims and premises in text, but does not identify their stance. The Comparative Argumentation Machine (CAM) [54] has been specifically developed to address a comparative information need and is based on retrieval of comparative sentences from a dependency-parsed fraction of the Common Crawl. Additionally, CAM classifies the compared objects' preference in the retrieved sentences, i.e., whether one or another compared options should be preferred. Our idea is to extend the stance classification of answers to comparative questions beyond just sentences for systems like CAM.

To simplify the challenging research task, we only analyze comparative questions with exactly two compared objects. This makes it much easier to identify the compared objects and use them as the targets for stance detection. Given a comparative question with two compared objects and a relevant answer, our task is to identify the answer's stance toward the objects, i.e., whether the answer argues for one of the target objects (and against the other), the answer is neutral (argues for or against both objects), or it does not express any stance. The classifier's output is then sought in the form of a category label from the set: {None, Neutral, First, Second}. As the targets for the stance detection task are not explicitly given by the question, one important subtask is to identify these objects in the question before detecting the stance towards them. This thesis investigates this stance detection task and different approaches to automatically identify the targets and the stance.

Our first contribution is the creation of a stance dataset consisting of 1,034 comparative questions with answers. We sample questions from CQA plat-

[3] `https://commoncrawl.org/`, `https://lemurproject.org/clueweb12.php/`

forms, select the comparative ones with a classifier, and annotate their answers with a label describing the stance towards the comparison. The second contribution is the design and evaluation of methods to identify the targets in the question and the answer. Our main contribution is the extensive evaluation of different approaches to detect the stance in the answer towards the comparison. We focus in our analysis on transformer-based models, as they deliver the best results for the task and conduct a range of experiments to further improve their performance. In the process, we manage to improve on the previous best results on the CompArg [47] and VAST [2] dataset. For our dataset, we propose a stance classifier based on RoBERTa using answers with masked targets as the input. This model achieves a micro-averaged accuracy of 0.62 for detecting the stance.

To present these results, this thesis is structured as follows. The next chapter gives an overview of related research. In Chapter 3, we describe the process of creating the dataset, which will be the basis for our work. Chapter 4 studies the task of extracting the targets out of the question and identifying related targets in the answer. Chapter 5 describes the main contribution of this thesis. Different approaches to identify the stance are evaluated and compared. Lastly, Chapter 6 summarizes the main finding of this thesis and identifies areas for further research.

Chapter 2

Related Work

Stance detection is a well-established task in natural language processing (NLP). Based on a text, the aim is to identify the author's stance towards a target, which may be for instance a topic, an entity, or a claim. [43]. There are many different variations of stance detection differing in the analyzed text, the kind of target, the evaluation setup, and the sought category labels representing the stance. Wang et al. [74] categorize the research on stance detection based on the debate setting such as congressional floor debates, company internal discussions, online forum ideological debates, and hot-event oriented debates on social media. Küçük and Can [32] divide stance detection problems based on the targets for the stance analysis and the labels which were used to categorize the stance. They observed, that the research task most frequently studied is stance detection in a piece of text towards one given target with stance labels describing a favoring, neutral, opposing, or not given stance. Other common forms of stance detection analyze the stance towards multiple related targets (Multi-Target Stance Detection) [57], are trained and tested on distinct targets (Cross-Target Stance Detection) [78], or analyze the stance of a text towards a headline to facilitate fake news detection (Fake News Stance Detection) [14]. A task similar to stance detection is sentiment detection, where a text is analyzed for its polarity of emotion. Sometimes the sentiment of a text is used to detect the stance of a text [5, 37].

Stance detection can be seen as one of the main methods in computational argumentation. Argumentation itself has already been studied by Aristotle [31] in ancient times and has been a topic of research since then. Computational argumentation builds on this long history and tries to automatically analyze and synthesize natural language argumentation. The research covers tasks such as argument mining [21], argumentation quality assessment [68], and argument generation [6]. Some of the envisioned applications of computational argumentation methods are argument search [69], personal assistants [51], argument summarization [73], or writing support [59].

In recent years the task of stance detection has drawn a lot of research attention. Earlier work has mostly focused on online debates [32] using rule-based [3, 44, 70, 71] or feature-based approaches using classifier such as SVM [23, 58, 62, 71] or naive Bayes [3, 23, 49, 71]. In 2016 there have been several stance detection competitions [43, 61, 79] and microblogs foremost twitter became a popular data source for the text to be analyzed [4, 43, 52, 72]. Within the same context, neural networks such as CNN [26, 65, 76] or LSTM [66, 82, 83] based on word embedding became popular as classifiers. With the recent rise of transfer-learning and transformer-based models for many tasks in NLP, they have also been proven to be state-of-the-art for stance detection [14, 19, 55, 64]. These types of models excel other neural network architectures especially when trained on small task-specific datasets. Another common trend is the use of ensemble classifiers. The individual classifiers can be combined by majority voting [56], random forest algorithms [22, 38, 63], or some custom scheme [75, 84].

While there has been much work on stance detection recently, most of it centered around classifying the stance in very short texts (often single sentences) [5, 32, 43] and using the few same targets for training and evaluating the classifiers for stance detection [1, 5, 64]. This manifests in two significant differences to our task.

In contrast, Allaway and McKeown [2] propose the task of zero-shot stance detection on online comments containing multiple sentences. They use the term zero-shot stance detection to describe the classification of stance towards targets not used in training the classifiers. The authors analyze comments from online debates for their stance towards controversial topics. They propose a neural network classifier, which utilizes BERT embeddings and generalized topic representations to capture relationships between the different targets. On their dataset, this classifier manages to beat a fine-tuned model based on BERT. However, their classifier is not applicable to our task, because it is based on clustering the targets, as these can all be assigned to one of a few overarching topics. This is an essential difference to our task, as we aim to analyze answers on a variety of different topics. Additionally, they use one target for stance detection, and we research how to detect the stance towards a comparison.

Faulkner [16] as well deals with detecting the stance in longer texts, more precisely student essays. His classifier identifies stance words in certain syntactic structures and links these to one of a few given essay topics. However, the classifier strongly relies on the uniform structure and language of essays, making his approach not usable for the linguistic variation found in online answers to comparative questions.

For sentiment analysis, Wachsmuth et al. [67] take a different approach to get document-level predictions for online reviews. They predict the local sentiment for each sentence and use the idea of sentiment flow [41] to deduce the overall sentiment. Sentiment flow is the concept that the global sentiment can be represented by the sequence of local sentiments. Wachsmuth et al. [67] propose a classifier based on this notion and show in their experiments that the idea of sentiment flow can be used domain-independent to predict the global sentiment based on the local sentiment. We adapt this idea to stance detection, to combine sentence stance predictions into an overall stance classification.

While the approach developed by Bar-Haim et al. [5] was designed for a very different stance detection task, it proves to a valuable base for one of our baselines. The classifier was created for analyzing single sentence claims towards controversial topic targets. The actual stance detection step included in the approach is based on lexicons and does not need training. Therefore, in contrast to many other recent approaches, it does not depend on recurrent targets, even though it was used in such a setting. We adapt the stance detection method to be able to analyze longer texts towards two compared objects.

Most of the previously mentioned work focuses on stance detection towards single targets. Our task demands a different approach because we aim to analyze the stance towards a comparison, i.e., two targets given by a comparative question. Schildwächter et al. [54] account for this comparative information need and built a comparative argumentation machine (CAM), an information retrieval system to compare objects based on information extracted from the web. They showed the potential of this kind of system in a user study where the participants were able to answer (manually selected) comparative questions faster and more accurately with the help of the system. The CAM allows the user to input two objects which should be compared in general or based on optional aspects. The system shows supporting arguments (sentences) for each option, generates superior aspects for each option, and calculates a total score. The system is internally comprised of four different steps. First, relevant sentences are retrieved from the Common Crawl corpus, then their stance is classified. A score based on relevance and stance is calculated to determine the order of the sentences and calculate the total score. Finally, the application looks for relevant aspects for each option. The second step is similar to our research task, but it differs due to some simplifications. Their system operates on a sentence-level, which can only contain simple arguments. Also, they always compare exactly two objects, which are given explicitly by the user and not as part of some free-text query.

Panchenko et al. [47] analyze different approaches including the one used by Schildwächter et al. [54] to classify the preference in comparative sentences. They evaluate it on sentences retrieved from the Common Crawl corpus by looking for certain pairs of objects and comparative cue phrases. Their best classifier, a gradient boosting model based on pre-trained sentence embeddings, manages to reach an F1-score of 85%. However, as previously stated, this task is different because it only classifies sentences, which have to be comparative itself. Furthermore, their classifier also detects non-comparative sentences as one of their classes, which makes the results even less comparable. Even though their sampling method makes their CompArg dataset not very diverse, we use it for our classifiers on sentence-level to be able to analyze the influence of local stance on the global stance.

Ma et al. [40] propose a novel model that leverages dependency graph features to detect the stance in comparative sentences. When tested on the CompArg dataset by Panchenko et al. [47], their approach outperformed the previous state-of-the-art classifier. As their classifier is specialized for this dataset and utilizes the dependency structure in the sentences, it is not applicable for longer texts. Even for the CompArg dataset, we show how to achieve better results using a transformer-based model.

Other research conducted on comparative questions has focused on identifying and analyzing comparative structures [20, 28–30, 36]. Especially relevant work on comparative questions was done by Bondarenko et al. [7] and Dittmar [13]. Bondarenko et al. [7] analyzed real question queries on the Russian search engine Yandex. They identified that approximately 3% of these queries are comparative and more than 65% of these are not factoid and require argumentation. Furthermore, they designed a classifier to identify comparative questions, categorized them into several subclasses, and analyzed different classifiers for these subclasses. Dittmar [13] analyses the structure of comparative questions and researches different approaches to recognize comparative questions and their structural components. We build on results from both to design our classifiers to select the relevant question for annotation from the source datasets.

Chapter 3

Dataset

As there is no publicly available dataset for stance detection on answers to comparative questions, we create one by annotating answers to comparative questions with their stance. We use dumps from the CQA sites Yahoo! Answers and Stack Exchange to sample comparative questions and human-written answers. In our dataset, we aim to minimize domain dependency by sampling questions on diverse topics from these two sources.

Bondarenko et al. [7] observed that comparative questions can be further divided into two distinct categories, superlative and non-superlative questions. These two different question types differ significantly regarding their question structure and the answers. Superlative questions usually ask for an object out of some group that fulfills some criteria to the greatest degree. The following example question asks for the best alternative out of the group of all smartphones.

Example 3.1 *Which smartphone is the best?*

Non-superlative questions ask for a comparison of several (mostly two) concrete objects. They clearly name the objects which are of interest. For instance, the following non-superlative questions ask for a comparison of exactly two specified models of smartphones.

Example 3.2 *Should I buy the Samsung Galaxy S20 or the iPhone 11?*

Additionally, an answer to a superlative question will only discuss a selection of options. It will not evaluate all possible objects which belong to the specified group. In contrast, an answer to a non-superlative question will usually compare all objects specified by the question. Therefore, it is much easier to identify the targets of comparison for non-superlative questions. As we need a target for stance detection, we focus on these non-superlative question and leave the superlative questions for further research. Furthermore, as we research how to detect the stance of answers, we only include argumentative questions and exclude any factoid ones.

How we choose the source for our dataset is explained in Section 3.1. The process of selecting just the relevant questions is described in Section 3.2. Section 3.3 focuses on our annotation with the stance labels. This chapter ends with a short analysis of the final dataset in Section 3.4.

3.1 Data Sources

As we aim to build a realistic dataset for comparative questions asked by humans, we sample comparative questions and answers from the two CQA sites Yahoo! Answers[1] and Stack Exchange[2]. We choose these datasets because they contain argumentative questions, have a diverse selection of topics, and have an answer that was selected by humans to be a sensible and relevant answer to the question.

The Yahoo! Answers corpus was built by collecting all questions and answers from the CQA site Yahoo! Answer as of October 2007. It contains approximately 4.5 million questions and their corresponding answers. The metadata includes which answer was selected as the best answer, either by the asker or by the other users by voting, if the asker did not select a best answer.

Stack Exchange is a collection of different community-driven websites, with each site covering an individual topic.[3] The first site was Stack Overflow, a site for programming questions. Most of the other big sites are also related to computer science. The dumps from Stack Exchange are separated by site and each dataset contains all the questions, answers, and additional metadata for one site. One essential part of the metadata is the information which answer was selected by the questioner as the best answer.

We choose the sites from Stack Exchange such that we get a selection of diverse topics, even though this is not entirely possible because most (and the biggest) sites are related to computer science. We exclude those sites that have not enough argumentative and comparative questions after filtering (see

[1]L6 - Yahoo! Answers Comprehensive Questions and Answers version 1.0 (multi part) retrieved from `http://webscope.sandbox.yahoo.com`

[2]Stack Exchange Data Dump (2nd June 2020) retrieved from `https://archive.org/details/stackexchange`

[3]An overview of the different sites is available under `https://stackexchange.com/sites`

Section 3.2) and those where a manual exploration indicates that the question and answers are not suitable for our task because they contain a lot of text that is not natural language. For instance, the sites Mathematics and Stack Overflow are excluded, because they are filled with mathematical formulas and code snippets, respectively. We choose the sites shown in Table A.2 as a compromise between diversity and an unmanageable number of distinct sources.

For each question, we select the best answer as specified by the metadata in the dataset. On Yahoo! Answers as well as Stack Exchange the asker has the possibility to write a more detailed description to add to the actual question. However, we only use the title for all further steps, since we want to keep the questions as short as possible to enable the automatic target identification. By ignoring potential details about the questions we might create incomprehensible question-answer pairs. However, during the manual annotation (see Section 3.3) inappropriate question and answer pairs can still be excluded. Additionally, we exclude all pairs with an answer that has less than 10 words, as manually exploring the datasets shows that answer with fewer words generally do not contain sensible arguments. Lastly, we clean the question and answer of any HTML-tags and replace any hyperlinks with placeholders.

3.2 Comparative Question Selection

To select questions and answers from the source datasets for labeling, we classify whether the questions are comparative and argumentative. For that, we train a classifier to recognize comparative argumentative questions.

Bondarenko et al. [7] reported that for classifying Russian questions as comparative and argumentative it worked best to classify the questions in two steps. They first use a classifier to recognize comparative questions and afterward classify the questions in the reduced corpus as (not) argumentative. The alternative to this two-step approach is to do this all in one step and employ a classifier to directly recognize comparative argumentative questions. To compare these two approaches we train a model for each of these three individual classification tasks.[4]

For text classification tasks, pre-trained transformer-based neural network architectures such as BERT [12], XLNet [81], and RoBERTA [39] produce in general state-of-the-art results. Additionally, Dittmar [13] showed that approaches based on BERT and XLNet have very competitive effectiveness for

[4]Classifying questions which are a) comparative, b) argumentative and c) argumentative & comparative

detecting comparative questions. These models were only outperformed by an ensemble classifier. As we are still able to manually exclude any questions during annotation to ensure the quality of the dataset, we decide against implementing a more effective, but also more complex, ensemble classifier.

To train the models, we use a dataset of 30,000 questions annotated as (not) comparative by Dittmar [13], where 15,000 of these were additionally annotated as (not) argumentative by Bondarenko et al. [7]. These questions were sampled in equal parts from Google's Natural Questions [34], MS MARCO [45], and Quora Question Pairs [27]. Additionally, we translate around 50,000 Yandex queries and 15,000 Otvety questions annotated as (not) comparative and (not) argumentative from Russian to English. Yandex is the most popular Russian search engine and Otvety is a Russian CQA website.

The questions are lowercased and cleaned. This includes removing duplicate punctuation marks and emoticons. This both increases the effectiveness significantly. To validate that including the translated Russian queries improves the prediction, the precision on an originally English test set is compared when adding the Yandex and Otvety queries. The precision and recall can be improved and therefore all further experiments are conducted with the complete corpus.

The resulting dataset consists of 80,000 questions annotated as (not) comparative and (not) argumentative and additional 15,000 questions only annotated as (not) comparative. The dataset is divided with an 80-20 train-test split into sets for training and evaluation. The 15,000 more coarsely annotated questions are left out and only used as additional training examples for the comparative classifier. To implement the classifiers, we use the Python library Simple Transformers.[5] The training set is used in a 5-fold cross-validation strategy to optimize the models' hyperparameters and find the best model architecture. We test the base models of BERT (uncased), RoBERTa, and XLNet. We choose a 5-fold setup because the equally common 10-fold strategy results in a higher standard deviation for the metrics between the individual test run. The test set is used after fine-tuning to compare the two approaches (one-step vs. two-step) and report the results. We use the F1 score to evaluate the models, as recall and precision are equally important. A high recall guarantees a wide diversity in the selected questions and a high precision reduces the number of questions we have to manually remove during annotation. For each of the three classifying tasks, the best model is based on the uncased BERT model. The hyperparameters are shown in Table A.1.

[5]`https://github.com/ThilinaRajapakse/simpletransformers`

Our classifier for comparative questions, when trained and evaluated on the dataset from Dittmar [13], outperforms his best individual classifier significantly. He reports a recall of 0.82 and 0.38 for a precision of 0.8 and 1.0, respectively. By optimizing the hyperparameters and choosing the uncased variant of BERT, we manage to increase the recall to 0.86 and 0.45 outperforming his classifier by 0.04 and 0.07.

As previously motivated, we want to compare the model detecting comparative argumentative questions (one-step) with the combination of comparative classifier and argumentative classifier (two-step). For the two-step approach to classify a question as comparative and argumentative, both individual models have to predict the positive class. The total results on the test set are similar for both approaches, even though the training corpus for the two-step classifier is larger in total (because 15,000 queries were only coarsely annotated as comparative or not). The two-step approach achieves a precision of 0.72 and a recall of 0.69 for the positive class resulting in an F1 score of 0.7. The one-step classifier has a precision of 0.76 and a recall of 0.63 resulting in an F1 score of 0.69. As the difference in results for the two different approaches is minimal, we decide to use the one-step classifier as it is trained faster and easier to employ.

As we limit our research on non-superlative questions, we train another classifier on the same dataset we used for the previous classifiers. However, this time we use the annotations to remove all non-comparative and non-argumentative questions before the training. The annotation of whether a question is superlative is used as the label to train the classifier. As previously, a 5-fold cross-validation strategy is used to find the best model and hyperparameters. However, this time we use accuracy as the metric to evaluate the models, as both classes are equally sized. The best model achieves an accuracy of 0.92. It is based on the uncased variant of BERT. The hyperparameters are shown in Table A.1.

We use these two classifiers to select the relevant questions from Yahoo! Answers and Stack Exchange. First, we classify the questions as (not) comparative argumentative, then, we classify the comparative argumentative ones as (not) superlative. This results in a collection of 39,545 questions and answers from Yahoo! Answers and 3,833 from the selected Stack Exchange sites.

3.3 Data Labeling

Finally, we annotate the questions and answers for our dataset with the question targets, the answer targets, and the stance towards the comparison between the question targets. The question targets are those phrases that are used in the question to describe the different options in the comparison. The author of the answer does not have to use the same words to describe these alternatives. We, therefore, seek also all phrases in the answer that refer to one of the question targets. Lastly, we need the stance in the answer towards the comparison in the question. This means we want to know which of the options the author of the answer prefers. If there is no clear preference for one of the alternatives we want to know if the answer has a neutral stance or does not state any stance towards the comparison.

To simplify the collection of labeled data at scale, we also attempt to automatically extract labels. The publicly available dataset MS MARCO [45] contains in addition to the questions and answers a summary of some answers. The dataset consists of approximately one million queries submitted to the search engine Bing and around 9 million website passages relevant to these queries. Due to the very inconsistent structure of the summaries, the target extraction is challenging. Nevertheless, we can reliably extract the preferred target for around 70% of answers using a rule-based approach (see Algorithm B.1 for the pseudocode). The next step would be to extract the targets from the question and combine them with the previous targets into an overall stance for the answer. However, we have not found any method to achieve the reliability needed for a ground truth dataset. Additionally, the quality of the dataset suffers from irrelevant answers, wrong summaries, and poorly parsed text. Due to these reasons, we decide against further pursuing this approach and to not use the MS MARCO dataset at all.

For the manual annotation, we randomly sample half of the questions and answers from Yahoo! Answers and half from Stack Exchange. We internally recruited 3 annotators and instructed them to annotate the spans for the question and corresponding answer targets and to label the stance of the answer. We use the stance labels *none*, *neutral*, *first*, and *second* to describe a stance that is non-existent, neutral, preferring the first or the second object of the question, respectively. We also added one label to mark those questions that were misclassified, i.e., not comparative, not argumentative, or superlative questions, and one label for those questions that ask for a comparison of more than two objects.

We used the tool WebAnno by Eckart de Castilho et al. [15] for the annotation. After one pilot annotation with around 120 questions and answers, we refined the annotation guidelines to remove uncertainties. Afterward, a pilot run with another 120 questions is annotated by each of the annotators. We use Fleiss' kappa [18] to evaluate the annotator agreement, as we have more than two annotators but no incomplete annotations. The annotators reach an inter-annotator agreement of a Fleiss' $\kappa = 0.61$ (substantial agreement) for the stance labels and an agreement of a Fleiss' $\kappa = 0.69$ (substantial agreement) for the target annotations. Even though we gave clear instructions after the first pilot annotation to include the articles in front of the target phrases, the agreement rises to a Fleiss' $\kappa = 0.72$ (substantial agreement) if we ignore articles. Based on this high agreement, we decide to let each annotator annotate an individual set of questions to be able to produce a larger dataset in less time. To this end, exactly 1,441 questions have been annotated.

3.4 Dataset Analysis

The resulting dataset consists of 1,441 questions and answers, approximately half from Yahoo (711) and half from Stack Exchange (730). The questions from Stack Exchange originate from nine different domains, each contributing between 50 and 130 questions. The exact number for each site can be found in Table A.2. Some examples of the questions and answers with their annotations are shown in Table A.3. In the process of creating our corpus, we filtered the original source datasets to less than 1% of their size and annotated less than 0.03% of the original questions. The detailed numbers for the two sources for each step in the process are shown in Table 3.1.

Table 3.1: The number of the total, selected, and annotated questions extracted from the two sources. The selected questions are those that are classified as comparative, argumentative, and non-superlative. From this set, the annotated questions were randomly chosen.

	Number of questions		
	Total	**Selected**	**Annotated**
Yahoo! Answers	4,483,032	39,545 (0.88%)	711
Stack Exchange	686,886	3,833 (0.56%)	730

Manually analyzing the questions and answers shows that the questions from Yahoo! Answers are mostly everyday questions with short and simple answers (average length is 80 words), whereas the questions from Stack Exchange are quite complicated and specific, but also have longer and more detailed answers (average length is 194 words). The questions can be categorized based on the annotation into three different groups: questions comparing two objects, those with more than two objects, and misclassified ones (i.e., not comparative, not argumentative, or not superlative). The numbers for each category are shown in Table 3.2. The questions with two objects amount for the largest part, whereas those with more than two objects only form a small part of the dataset. These numbers additionally justify the limitation of our research on questions with exactly two objects. Nearly a third of the questions is misclassified. This seems consistent with the result of the classifier on the test set. Interestingly, the classification for the Yahoo! Answers dataset seems to work much better than for the questions from Stack Exchange.

Table 3.2: The number of annotated questions per type.

	Number of questions per type		
	Two objects	Three or more objects	Misclassified
Total	969 (67%)	65 (4%)	411 (28%)
Stack Exchange	451 (61%)	26 (4%)	257 (35%)
Yahoo! Answers	518 (73%)	39 (5%)	154 (22%)

To conclude this short analysis of the dataset, we examine the annotated stance for the answers. The number of questions for each stance label is shown in Table 3.3. We assumed that the stance of the answer would correlate with the order of the objects, as, for instance, seen for the preference on comparative sentences in the CompArg dataset [47]. We supposed that the asker mentions his preference (if he already has one) first and rather choose an answer as the best one that approves his stance. Additionally, the asker might be more likely to mention a more popular option first. However, the minimal difference in the number of answers preferring the first object compared with those preferring the second object cannot confirm this hypothesis.

Table 3.3: The number of answers for each stance.

Stance	**First**	**Neutral**	**Second**	**None**
Answers	329 (34%)	290 (30%)	280 (29%)	70 (7%)

Chapter 4

Target Identification

Our main research task is to investigate how well we can identify the stance of an answer towards a comparison given by a question. The target towards which we aim to determine the stance is the comparison between two objects stated in the question. As we can dissect this comparison into the two compared objects, we can also use the two objects to represent the comparison. To be able to explicitly use these targets of comparison for the stance detection methods, we analyze in this chapter how we can identify the targets given the question. In the following example question and answer the targets are in bold.

Example 4.1 *When it is better* ***to offload work to the RDBMS*** *rather than* ***to do it in code****?*

You want to do all set-based operations ***in the database*** *for performance reasons. [...] The only reason I might ever do something like this* ***in a database query*** *is if it required lots of columns that I wouldn't otherwise select that could actually amount to enough data to meaningfully slow down my query.*

As you can see in this example, the author of the answer does not have to use the same words used by the question to describe the compared objects. To refer to the question targets the answer's author can use synonyms, abbreviate, or paraphrase them. As there are multiple sentences in one answer, there can even be multiple answer targets for each question target. If we can identify those answer targets we can give these as additional information to the stance classifiers to simplify their task. Depending on the stance detection approach, they either use the question targets, the answer targets, or both.

The next section will investigate how to identify the question targets in the question. The second section deals with the identification of the answer targets based on the question targets. We end this chapter with a short discussion of the results.

4.1 Target Identification in Questions

We evaluate two different approaches to identify the targets in the question. As the questions are short and often have a certain structure, we assume that it is possible to design a rule-based algorithm to identify those targets. The second approach is based on a transformer model, the current state of the art for most NLP tasks. We compare these two approaches based on the accuracy they achieve for classifying the right phrases as targets. As both methods classify the tokens, we also evaluate the precision and recall on token level to see how well they are able to construct the phrases based on their token predictions.

For the rule-based approach, we use spacy [24] to tokenize the question and tag the tokens with part-of-speech (POS) and dependency tags. While rules based on finding cue words such as "vs" or "difference ... between" work very accurately, their application is limited to a very small number of questions. With the POS and dependency information we can design less specific but at the same time less accurate rules. The algorithm that works best is shown in algorithm B.2. It manages to classify the tokens with a precision of 0.60 and a recall of 0.72. This method correctly classifies the targets with an accuracy of 0.37.

The other approach to identify the question targets is based on transformer models configured for token classification. We tokenize the questions with spacy [24] and label whether the tokens belong to a target phrase. We supplement our training set with 2,372 comparative questions from search engine queries and CQA sites annotated by Dittmar [13]. The author evaluated different approaches to identify the components of comparative questions. Among those components, one category corresponds to the compared objects, which we are looking for. We transform the five different labels into the two we seek to classify (*None*, *Target*). As the domain of this dataset strongly resembles ours, the additional data is very helpful to improve our results. We use a 5-fold cross-validation strategy to optimize the hyperparameters and to evaluate different strategies to construct the targets out of the token classifications. We concatenate all successive tokens that were classified as part of a target into one target phrase. However, this does not necessarily generate exactly two target phrases. If we get more than two target phrases, we keep only those with the highest average token confidence. If we get just one target phrase, we split it at the token that has the lowest confidence. We get the best results for a model based on BERT with the hyperparameters shown in Table A.1. We get a precision of 0.91 and a recall of 0.92 on token level and the targets are correctly identified with an accuracy of 0.76. A manual analysis shows that a significant part of the misclassified targets is either including slightly too many or too few tokens. However, using dependency information to extend or shorten the targets to match whole phrases is not successful.

4.2 Target Identification in Answers

For identifying the targets in the answer, we first sample phrases from the answer as target candidates and then classify them as a target for one of the question targets using similarity measures. To evaluate our approaches we use the answer and the annotated question targets as input and calculate the token-based precision and recall using the annotated answer targets as ground truth.

POS-tagging and dependency parsing the answers showed that around 90% of the annotated answer targets are noun phrases and a further 6% are verb phrases. The remaining 4% arise do not coincide with the phrases detected by the dependency parser or have been tagged with one of the other POS tags. To generate the target candidates we, therefore, sample all noun and verb phrases from the answer as potential targets. We limit the length of these phrases to a maximum of 10 tokens as more than 99% of the answer targets are shorter than this length. Neither generating all possible phrases nor generating only the more relevant noun phrases can improve the overall results.

We employ different similarity measures to identify whether the candidates relate to one of the question targets. We calculate the similarity between the target candidate and each of the question targets and if it is beyond some threshold we classify it as an answer target. For potentially overlapping target candidates only the one with the highest similarity is kept. The first similarity measure is the cosine similarity of the word2vec [42] representations of the phrases. As word2vec gives vector representations for single tokens, the vector representation for the phrases are calculated by averaging the token vectors. A threshold of 0.7 is identified as the best boundary on the train set with a precision of 0.56 and a recall of 0.38 on the test set. The second measure is the Wu-Palmer Similarity [77] on WordNet [17]. A threshold of 1.0 yields the best F1 score with a precision of 0.57 and a recall of 0.30 on the test set.

We can improve these results by combining both similarity measures. The highest F1 score of all combinations is achieved by classifying the candidates as target phrases if one of the similarity measures outputs a value higher than the previously mentioned thresholds (word2vec: 0.7, Wu-Palmer: 1.0). This classifier can increase the precision to 0.52 and the recall to 0.46.

We evaluate a transformer-based model using BERT as an alternative. The model is trained on the pairs of question targets and answer target candidates, which are labeled belonging to the positive class if the target candidate is indeed an answer target for the given question target. The recall of this model is extremely low with 0.19 and the precision of 0.64 was not much higher than the precision of the previous approach.

We would like to compare the achieved performance with human performance on the same task. For the part of the dataset that has been annotated by all three annotators, the ground truth is the majority decision on token-level. We compare the individual precision and recall of each annotator with this ground truth and take the average over all three. This results in a precision of 0.89 and a recall of 0.94. An analysis of the differences indicates that the position of the targets is undisputed but rather the length of the phrase is controversial. While this effect occurs also during the automatic target identification, it has minor effects on the performance, as 96% of the answer targets are included in the target candidates and these phrases are classified as a whole.

4.3 Discussion

Our results show that especially the answer target identification is very challenging. Besides synonyms also phrases that only relate in a specific context to the question target have to be identified. While it might be possible to improve our results by combining even more similarity measures or other classifiers, we assume that the improvements would be small.

The focus of our work lies on stance detection and our results enable us to get a more realistic comparison of the approaches that rely on the answer targets with those that solely depend on the question targets. We will analyze the impact these flawed target identifiers have on the stance detection results in the next chapter. Based on the results of this chapter, we will use the transformer-based classifier to identify the question targets and the similarity-based method to identify the answer targets.

Chapter 5

Stance Detection

This chapter investigates how reliable it is possible to detect the stance of answers to comparative questions towards two given targets of comparison. We use the dataset presented in Chapter 3 to train and evaluate our approaches. Given a pair of a question and an answer, the task is to predict one of the four labels in the set: {*None*, *Neutral*, *First*, *Second*}.

We use the micro-averaged accuracy as the main metric to evaluate and compare our approaches, as the important classes (*Neutral*, *First*, *Second*) are equally sized. As our task uses exactly one label per instance, the micro-averaged F1-score is the same as the micro-averaged accuracy and, therefore, would be equally appropriate. With micro-averaging the metric we also follow the common practice for multi-class text classification problems [33, 80]. When we analyze the results in more detail, we use the F1 score for each class, as precision and recall are equally important. For the use-case as part of a search engine, we evaluate the classifiers at the end of the chapter separately, as it makes the task highly precision-oriented.

Since stance detection requires targets to analyze the stance towards, we conduct our experiments in two settings by either using our target classifiers or the annotated targets. As our approaches to identify the targets deliver poor results, we evaluate and compare our methods for stance detection on the manually annotated targets as if we had a perfect target classifier. At the end of this chapter, we use our target identifier to find the targets to evaluate how well our stance detection approaches work in a realistic setting.

The dataset is divided with an 80-20 split into training and test set. For all learned classifiers, the training set is used in a 5-fold cross-validation strategy to optimize the model's hyperparameters using a grid search. The results are always reported on the hold-out test set. To make our results reproducible, we report the hyperparameters for the most relevant models in Table A.1.

To evaluate the effectiveness of our approaches, the first section describes two baseline approaches. The second section deals with classifiers using pre-trained transformer-based models. In the last section, we analyze and discuss the results of these classifiers in a realistic scenario.

5.1 Baselines

We implement the baseline stance detection approach following Bar-Haim et al. [5], as they use a lexicon-based approach that does not need training. This means it works independently of our small training set and does not rely on recurrent targets. The authors reported a highest achieved accuracy of 0.63 on their dataset with two stance labels (*pro*, *con*). We adapt their method, as there are some major differences between the task settings. They detect the stance of single-sentence claims towards one known target. In contrast, we aim to detect the stance of multi-sentence answers towards two compared targets. Their classification is performed in three steps. The first one tries to determine a target in the claim, the second one tries to detect the stance towards this target and the final one will try to detect the contrast between the claim target and the given topic target. The target extraction and contrast detection steps are not applicable in our case, as we identify the targets differently and their method is not usable for our data. Their stance detection algorithm is used to identify the stance towards both targets and we combine these individual stances into the overall stance towards the comparison.

Following Bar-Haim et al. [5], the answer text is scanned for positive and negative sentiment words from the sentiment lexicon created by Hu and Liu [25]. Afterward, a set of shifters ("not", "never", etc.) is applied to swap the sentiment of all sentiment words within a window of 8 tokens to the opposite. Depending on the shifter word, either all sentiments in the preceding or following 8 tokens are inverted. They chose 8 as the size of this scope as it delivered the best results in their experiments. We can confirm this for our dataset as well. Bar-Haim et al. [5] additionally limit the window for applying these shifters by "," and "but". In contrast to their data set, we detect the stance in multi-sentence answers, so we additionally limit the window by any sentence boundaries.

After detecting (and switching) the sentiments, the resulting values are summed up weighted by $d^{-0.5}$, where d is the distance to the target. The final stance score is then calculated as $\frac{p-n}{p+n-1}$, where p and n are the weighted sums of positive and negative sentiments. We calculate this score for both comparison targets. For each of these two, we use the nearest answer target to determine the distance d to the sentiment words. This means we calculate $d = \min(d_1, \ldots, d_n)$, where $d_1, ..., d_n$ are the distances to the answer targets belonging to the question target.

The last step for our task is to combine these two stance scores for each of the comparison targets into one overall stance. We evaluate different features calculated from these scores (e.g., absolute and relative difference, highest absolute values) and different methods to get a prediction based on these features (e.g., decision trees). However, we achieve the most accurate results with very basic rules. We use the following formula to determine the overall stance s_{total}, where s_1 (s_2) is the predicted stance towards the first (second) target:

$$s_{total} = \begin{cases} \textit{none} & s_1 = 0 \land s_2 = 0 \\ \textit{neutral} & (s_1 \neq 0 \lor s_2 \neq 0) \land |s_1 - s_2| \leq 0.1 \\ \textit{first} & s_1 - s_2 > 0.1 \\ \textit{second} & s_2 - s_1 > 0.1 \end{cases}$$

This classifier reaches an accuracy of 0.39. The F1-score for each class is shown in Table 5.1.

In a look for a more competitive baseline, we implement an approach based on an support vector machine (SVM). These have been a common tool for stance and sentiment analysis before neural networks were established. They work reasonably well on small datasets. To train and evaluate the SVM, the text has to be transformed into feature vectors. We evaluate n-gram features as well as embeddings based on InferSent [11], BERT [50], and Universal Sentence Encoder [8]. The best results with an accuracy of 0.40 are achieved by an SVM with a radial basis function kernel (C=10) trained and evaluated on the BERT embeddings. The F1 score per class in comparison to the lexicon-based baseline is shown in Table 5.1.

Table 5.1: F1 per label for the baseline approaches.

	None	*Neutral*	*First*	*Second*	Micro-avg.
Lexicon	0.22	0.37	0.48	0.38	0.39
SVM	0.09	0.48	0.41	0.36	0.40

5.2 Pre-trained Transformer-based Models

In the following, we focus on approaches using pre-trained transformer-based models, as these achieve outstanding results for many NLP tasks, including text classification and stance detection [12, 14, 19, 64]. For all the following approaches, we evaluate the most common and successful versions including BERT [12], RoBERTa [39], XLNet [81], and T5 [48]. We use the base version of these models because the larger ones (where available) have not shown any improvement in several experiments, but work much less reliable on our hardware due to their larger GPU memory requirement. In the first experiments, we also evaluated Albert [35] and DistilBERT [53] as representatives of those transformer architectures with a significantly reduced number of parameters. Our preliminary experiments showed that they perform fundamentally worse, even though they aim at achieving better results on smaller datasets. Hence, we did not consider them for any of the following approaches. The answer stance (*none*, *neutral*, *first*, *second*) is transformed into a number from 0 to 3 and used as the class label for fine-tuning and evaluating the models. All of the following results are achieved on the test set, after optimizing the hyperparameters for the different models on the train set in a 5-fold cross-validation setting. As previously, we use the library Simple Transformers[1] to implement these models.

We conduct classification experiments of two types: (1) detecting the stance of the complete answer and (2) detecting the stance of the individual sentences and combining them into answer predictions. In the first subsection, we deal with those models that are trained and evaluated on the whole answer. The second general approach is investigated in the second subsection. Lastly, we investigate if we can improve the previous results by combining several models into one ensemble classifier.

5.2.1 Answer Based Models

In the following, the transformer-based models are fine-tuned and evaluated on the whole answer. This is a problem for long answers, as for some models such as BERT the maximum length is 512 tokens. Everything longer than this length will be truncated. However, this has no big impact on the performance for our dataset, because less than 5% of the answers in our dataset are longer than 512 tokens. We analyze different approaches to use the whole answer as input for the models.

[1]`https://github.com/ThilinaRajapakse/simpletransformers`

First, we describe the approaches employing the capability of most transformer-based models to learn with text pairs as input. We construct the pairs out of the comparison targets and the corresponding answers to the comparative question. Secondly, we discuss models that are trained on just the answer, but where all answer targets are masked with special tokens. Finally, we extend these model's central principle of transfer-learning to the task-specific fine-tuning. These transformers are pre-trained on big datasets to develop an understanding of language to perform better after only being fine-tuned on relatively small task-specific datasets. We attempt to use another bigger stance detection dataset for additional fine-tuning to achieve better results on our dataset.

Classification of Target Answer Pairs

One straightforward way to create the input for the transformer-based models is to concatenate the first target and the answer for every dataset instance with the special separator token for the transformer. It is feed into the model as *[CLS] <target1> [SEP] <answer> [CLS]*, where *[CLS]* and *[SEP]* are the classifier and separator token used while pre-training the models. The different transformer-based models are fine-tuned and the hyperparameters are separately optimized to achieve the results shown in Table 5.2. Unlike the other models, T5 was not pre-trained with separator tokens. As a consequence, this model does not deliver competitive results for this setup. The BERT model predicts the stance with an accuracy of 0.40, and XLNet and RoBERTa achieve a higher accuracy of 0.47.

Table 5.2: F1 for different transformer-based models employed on target answer pairs as input.

	None	*Neutral*	*First*	*Second*	Micro-avg.
BERT	0.26	0.50	0.44	0.30	0.40
XLNet	0.11	0.62	0.46	0.41	0.47
RoBERTa	0.33	0.57	0.49	0.36	0.47

A disadvantage of this task setup is that just the first target is used for training the models and predicting the stance. To improve the results, we analyze two alternative setups that use both targets.

Following the proposal by Pan et al. [46] to concatenate multiple text fragments using several separator tokens, we extend the input to *[CLS] <target1>*

[SEP] <target2> [SEP] <answer> [CLS] for each training and test instance. However, this decreases the performance of the models.

The other straightforward strategy to utilize both targets splits each dataset instance into one example per target. This additionally increases the training set size which also might help to improve the results. The models are employed on *[CLS] <target1> [SEP] <answer> [CLS]* as well as *[CLS] <target2> [SEP] <answer> [CLS]* for each instance. Both instances for one answer are either used for training or for evaluating the models. We transform the stance label for the answer into simpler *pro/con* labels for each instance. The preferred target gets a *pro* label and the other one a *con* label. We first exclude any instances with an *neutral* or *none* label as we do not have any information about the stance towards the individual targets. The model is trained and used to predict the labels for the test instances. To evaluate the results, the predictions for the individual stance per target are assembled into an answer stance. Instead of the categorical predictions, we use the probabilities returned by the models. The probability of the *pro* label for the instance with the first target and the probability of the *con* label for the instance with the second target are averaged into the probability for the *first* label as the answer stance. The probability for the *second* label is analogously calculated.

This yields an accuracy of 0.30, which is worse than just predicting the majority class. An analysis of the individual predictions shows that the target in the input had barely any influence on the output of the model. This means that the predictions for both instances per answer are almost the same and the smallest differences in the probabilities decide about the resulting answer stance label. The model seems to be unable to learn the connection between the target and the answer for predicting the stance. This hypothesis was tested by training and evaluating a model on just the answer, i.e., *[CLS] <answer> [CLS]*, and comparing the results with the previous results for target answer pairs. This new model is just slightly less accurate with an accuracy of 0.46 (compared to 0.47) and thus confirms our hypothesis.

When splitting the instance per target, we, therefore, aim to stronger emphasize the importance of the target as the first part of the input. We manage to do this by reintroducing an *none* label and adding additional training instances with no relation between target and answer. We simply take each answer and create additional instances with targets from other randomly chosen answers. Generating up to two additional instances per answer boosts the performance considerably. Adding too many instances with an unrelated target is counter-productive, probably hindering the model to learn the important labels. The model achieves a micro-averaged accuracy of 0.54 for the two included classes. This is an improvement over the previous best model, however, it is only trained and evaluated on non-neutral instances with just three labels. For comparison, if we train and evaluate the model with the original setup on just the non-neutral instances, it reaches an accuracy of 0.52. The model using the split instances can improve the results on the non-neutral instances, however, the difference is small and we found no way to extend this approach to the neutral instances. Including these with a neutral label or labeling both instances per answer as positive or negative does not yield any usable results. Finer grained annotations about each of the targets might help to create sensible training instances, but we assume, based on our experiments, that training each answer with each target separately is not that useful.

Classification of Masked Answers

Instead of providing the question targets explicitly to the model, we can also replace the targets in the answer with special tokens. This could improve the results because it hides the concrete objects behind generic tokens creating more homogeneous training instances. Additionally, this gives us the possibility to pass more external information to the model. For both of the question targets, multiple answer targets can be flagged in the different sentences including paraphrases and synonyms.

We take the annotated targets in the answer to replace the corresponding tokens in the answer with a unique placeholder per target. The model is trained and evaluated on just these masked answers, i.e. *[CLS] <masked_answer> [CLS]*, with the answer stance as the label. Our first experiments show the major impact of the chosen markers. The results vary extremely depending on the placeholders used. For the models pre-trained with masked tokens, i.e. BERT, XLNet, and RoBERTa, a string in the same form as the masked tokens works best. For instance, using *[FIRST_OBJECT]* (and *[SECOND_OBJECT]*) leads to better results than using a natural language placeholder (i.e. *first object*) or using one token without the brackets (i.e. *FirstObject*). For the T5 models, which are not pre-trained with masked tokens, one single unique token such as *FirstObject* delivers the best results. The best results the models achieve are shown in Table 5.3. The results confirm our intuition that the models learn the answer stance better when more information about the targets and more homogeneous training instances are provided, although some context information is lost due to the masking of the targets. Every model type manages to achieve a higher accuracy than previously in the answer target pair setup. As before, the models based on XLNet and RoBERTa (and now also T5) reach a much higher accuracy than the BERT model. The gap in performance between the models even increases. We also experimented with supplementing the answer with the target(s) or question, but this is not successful in further improving the results.

Table 5.3: F1 for different models trained and tested on the masked answers.

	None	*Neutral*	*First*	*Second*	Micro-avg.
BERT	0.16	0.45	0.52	0.25	0.43
XLNet	0.38	0.58	0.59	0.58	0.57
T5	0.21	0.62	0.61	0.62	0.60
RoBERTa	0.40	0.57	0.68	0.58	0.60

Two-Step Fine-Tuning

One key factor for the success of transformer models in NLP is the pre-training on big datasets. This allows us to fine-tune these models on relatively small task-specific corpora and nevertheless achieve state-of-the-art results for many tasks. As our dataset is small considering the complexity of the task, we attempt to take this idea one step further. We investigate if it is possible to improve our classification results by using another dataset to sample additional similar training instances.

We decide to use the VAST dataset [2] as the additional fine-tuning corpora. This specific dataset was chosen because it resembles our corpus closely. The dataset was constructed as a zero/few-shot stance detection dataset, i.e., each target only occurs once or very few times in the corpus (the mean number of examples per topic is 2.4). Additionally, the stance is given for comments consisting out of several sentences. Both of these characteristics are rare for other stance detection datasets. The main difference to our problem is that we are looking for a stance towards two targets and the VAST corpus is designed for single-target stance detection. Therefore, we only analyze the effect of additional fine-tuning using target answer pairs as input.

We investigate two different setups to use this additional dataset, once as training data extension and once as the first training set in a two-step fine-tuning. For the first setup, we combine the dataset with our training set and fine-tune the models on both at the same time. As the alternative, we first train the models on the VAST corpus and afterward fine-tune them on our training set. We conduct these experiments with the RoBERTa model as it was the most effective model in the previous setup. The first setup changes the results to the worse, while the second one slightly improves the accuracy from 0.47 to 0.49. The results from tuning the hyperparameter seem to indicate that the results are more stable than previously and that fewer epochs with the original training set are needed to reach the same result quality. Even though we are able to slightly increase the accuracy, we cannot confirm that this approach is effective for improving the results as we observed fluctuations of 0.02 in the accuracy even for models trained with the same hyperparameters.

5.2.2 Sentence Based Models

The task of detecting the stance for a whole answer is very complex, as the stance usually is not uniform over all sentences. As typically the pros and cons are weighed against each other, some sentences might express a positive stance towards one of the objects and other parts might argue for the other one. To determine the overall stance the more prominent position has to be extracted. We aim to dismantle the difficult task of detecting the stance for the whole answer into predicting the stance for each sentence and combine these predictions into one total stance. Our idea is to transfer the good results from several different sentence-based stance detection tasks [47, 55] to the sentences in our answer and find a suitable model to combine these predictions into one overall stance.

The sentences in one answer can be divided into three different categories depending on how many targets they include. Around 11% of the sentences contain both targets, 33% contain one of the two targets and the remaining 56% contain neither object. We suppose that we can best predict the stance for the sentences if we use different classifiers for each of these sentence types. As we need external data to train the classifiers, we can look for data as similar as possible to each of these types. The closer the training data is to the data to be analyzed the better the results will be. In the following subsections, we deal with classifiers for each of these sentence types. The last subsection puts the different models together to predict the stance for the answer and evaluates how well the answer stance can be detected.

Two Target Sentences

The sentences with two objects are a minority and around half of the answers do not contain even one sentence of such type. Nevertheless, we assume that these sentences are very important for detecting the stance because they often directly compare these two objects. As we do not have stance annotation for the individual sentences in our dataset we need another corpus to train the models on. Panchenko et al. [47] created the CompArg dataset with preference labels for 7,199 comparative sentences. Each of these sentences compares exactly two objects. The preference label describes the preference regarding the comparison expressed in the comparative sentences. The corpus was sampled from the Common Crawl by looking for pairs of objects and comparative cue phrases such as "better" or "faster". These pairs of objects were compiled by combining objects of the same class (e.g., programming languages) or those with high similarity. The idea behind these pairs is that they are more likely to be targets of comparisons. The cue phrases were used to increase the probability to extract comparative sentences even further. Despite these efforts, 73% of the sentences still contain no comparison and are marked by the label *none*. Around 19% have a more positive stance towards the first object in the sentence (*better*). The remaining 8% express a stance favoring the second object (*worse*).

To select the classifier for the sentences in our dataset, we evaluate different models on this dataset. We assume, that the comparison results are transferable to the sentences in our answers, i.e., the more accurate a classifier is on the CompArg data the more accurate it is as well on the sentences with both targets in our corpus. Besides creating the dataset, Panchenko et al. [47] also propose classifiers for it. Their most accurate classifier is based on decision trees learned with XGBoost [9] on InferSent [11] representations of just the part of the sentences between the two targets. Ma et al. [40] propose for the CompArg dataset a new model called ED-GAT to improve on the results of this classifier. This model is a deep graph attention network that is trained on dependency parse graphs with BERT embeddings as the nodes. We assume that it might be possible to achieve the same or even better results with a simpler and more general transformer model. We train and evaluate the different transformer models on the CompArg dataset once with the input *[CLS] <target1> [SEP] <sentence> [CLS]* and once with masked sentences, i.e., *[CLS] <masked_sentence> [CLS]*. For the masked sentences, we rely on the previous experiments about the masks and replaced the targets in the sentences depending on the model with the best working placeholder (e.g., *[FIRST_OBJECT]* for BERT, RoBERTa, and XLNet and *FirstObject* for T5). We use the train/test split used by Panchenko et al. [47] and optimize the hyperparameters with a 5-fold cross-validation strategy on the train set. For both input types, the most accurate model is based on RoBERTa. The one trained and tested on pairs achieves a micro-averaged accuracy of 0.84 and the one working on the masked sentences achieves an accuracy of 0.91. This improves on the results by Panchenko et al. [47] and Ma et al. [40] by 0.06 and 0.04, respectively. In Table 5.4 the results are compared in more detail. It should be noted that the other models (BERT, T5, XLNet) employed on the masked sentences also outperform the ED-GAT model and are only slightly less accurate than the RoBERTa model. Besides the clear improvement in accuracy, these models themselves are simpler and not as task-specific as the ED-GAT classifier. Another interesting result of these experiments is, that the way the dataset instances are fed into the model plays a huge role. The transformer models trained on sentence-pairs are worse than decision trees employed on InferSent embeddings. In contrast, the same models trained on the masked sentences outperform all other classifiers.

Table 5.4: Comparison of the F1 of different classifiers on CompArg.

	Better	*Worse*	*None*	Micro-avg.
RoBERTa-Pair	0.71	0.54	0.90	0.84
InferSent+XGBoost [47]	0.75	0.43	0.92	0.85
ED-GAT$_{\mathbf{BERT}}$ [40]	0.78	0.56	0.93	0.87
RoBERTa-Mask	0.87	0.67	0.95	0.91

We fine-tune a RoBERTa model with the best hyperparameters on the whole CompArg dataset to use it on the sentences in our corpus. To be able to evaluate the performance on these, we annotate the stance of the sentences in our dataset containing two objects with the same labels. The model achieves an accuracy of just 0.62. A more detailed comparison of the results on CompArg and our sentences is shown in Table A.4. One reason for the lower accuracy is the different class distribution, as only 56% (CompArg: 73%) of the sentences belong to the class *none*. However, as we can see in the detailed comparison, the F1 score is significantly lower even for the individual classes. We suppose the very uniform structures of the sentences in CompArg make the task of detecting the stance easier and the trained model not fully capable for our sentences. The sentences in CompArg are sampled by looking for certain cue phrases and pairs of simple nouns or names. Therefore, these sentences have much less variety than those in our dataset. Nevertheless, we get a classifier that works better than a model trained on our own annotated sentences. For comparison purposes, we fine-tuned a RoBERTa model on our own annotated sentences, but only achieved an accuracy of 0.45. We assume this is because around 500 sentences are not enough to successfully train the model.

One Target Sentences

A third of all sentences in the answers in our dataset contains exactly one compared object. This kind of sentence can be used to differentiate one object from the other, which might be an essential part to determine the overall stance. As previously, we need another dataset to train a classifier on for this type of sentence. In contrast to the sentences with two objects, it is more difficult to select a dataset, as this category of sentences is more generic. We analyze three datasets that promise to be a good fit.

The Claim Stance Dataset created by Bar-Haim et al. [5] contains around 2,400 claims about controversial topics and a label (*pro*, *con*) describing the stance towards these topics. The dataset also includes the stance towards the claim target, which is explicitly used in the claim and is either consistent or contradictory to the topic target. Because the targets in our task are also contained in the analyzed text, we use the claim target and the corresponding label to train our models. This is different from the originally intended task of predicting the stance towards the topic target.

Stab et al. [60] constructed the UKP Sentential Argument Mining Corpus, which contains more than 25,000 arguments annotated either as a supporting, attacking, or no argument with respect to a topic. In contrast to a stance dataset, it does not contain the stance towards these topics but might help identify arguments in the sentences in our dataset. These would closely relate to the stance in the same sentence.

The third corpus is the previously mentioned VAST dataset published by Allaway and McKeown [2]. It was proposed as a dataset for zero/few-shot stance detection and contains the stance (*pro*, *con*, *neutral*) of approximately 23,500 comments towards a vast variety of targets. The length of the labeled text in this dataset is higher as the comments can be longer than a single sentence.

To decide which dataset we use to train the classifier to predict the stance of the sentences in our dataset, we evaluate the prediction results on our dataset. As we have no stance labels for the sentences with one object, we combine the sentence prediction in a simplified manner into an overall answer stance. We average the predictions for all sentences with one object within one answer to obtain the overall stance prediction. The predictions for the sentences with the first target are averaged with the inverted predictions for the sentences with the second one. While a good result is no proof for a good performance on our sentences, we show later that this method works surprisingly well and is therefore a usable approximation. The different transformer models get fine-tuned with optimized hyperparameters on each of the three different datasets and then evaluated on our whole dataset.

The model trained on the VAST dataset achieves by far the best total result with a micro-averaged accuracy of 0.51 on our dataset. The models trained on the Claim Stance Dataset and UKP Sentential Argument Mining Corpus have an accuracy of 0.33 and 0.32. Adding one or both of these corpora to the VAST dataset worsen the results. We suppose that there are several factors that make the VAST dataset more suitable for training the models, even though it contains not just single sentences as comments. We assume that mostly the large size of the dataset and the zero/few-shot setup help reach good results.

Our model is based on RoBERTa and trained on pairs of target and comment. The alternative approach of masking the comments is not feasible as most comments do not contain the target. We additionally consider the classifier proposed by Allaway and McKeown [2] for their own dataset as one possible model. This model, called TGA Net, is a neural network that is based on the BERT embeddings of the comments and an attention vector for the target calculated by topic clustering. To decide between these classifiers we compare them on the test sets (zero-shot, few-shot, altogether) of the VAST dataset. Our transformer-based model is able to improve the results compared to the TGA Net model for all three test sets. On the complete test set, it achieves a macro-averaged[2] F1 score of 0.70, which is 0.04 higher than the accuracy reached by TGA Net. A more detailed comparison of the results is shown in Table 5.5.

Table 5.5: Comparison of the F1 achieved by the TGA Net model [2] and our best model on the complete (few- and zero-shot) test set of VAST. * indicates that we calculated this value from the other given F1 scores for the sake of completeness ($F1_{Macro} = \frac{1}{3} * (F1_{Neutral} + F1_{Pro} + F1_{Con})$)

	Neutral	*Pro*	*Con*	Macro-avg.
TGA Net [2]	0.83*	0.57	0.59	0.67
RoBERTa-Pair	0.83	0.65	0.61	0.70

As the classifier for the one target sentences in our dataset, we, therefore, use a RoBERTa model after fine-tuning it on the whole VAST dataset with the previous best hyperparameters. For the second target, we always invert the prediction, i.e. a positive stance becomes negative and vice versa.

Context Sentences

The third category is the largest one and contains all those sentences that do not contain any compared objects. To use this kind of sentence for the overall

[2]We use the macro-averaged accuracy here to make our results comparable to the original paper.

stance detection we need to identify the target towards which a potential stance is expressed in the sentence. Since there are none of the compared objects explicitly mentioned, we cannot match the stance target to one of the objects. Nevertheless, these sentences might convey a stance about one of the compared objects for instance by using pronouns. Due to our notion that objects are usually named before they are discussed in more detail, we assume that these sentences mostly talk about the previously mentioned object. We use this idea to predict the target of such sentences. We use the previously discussed classifier for the sentences with one compared object to classify these sentences as well. As the target we use the lastly named object. Sentences before any of the objects are mentioned or those after sentences with both objects are ignored.

Combine into Answer Classification

The last step to detect the stance for the whole answer based on sentence predictions is to combine the sentence predictions into an overall stance prediction. The straightforward baseline approach for this is averaging the results over all sentence predictions and choosing the prevailing prediction.

Employing simple classifiers such as decision trees or using threshold values on the number of predictions for each label does not improve the results. Our intuition that the stance at the beginning and end is more important, because the answers tend to have a verdict as the first or last sentence of the answers can also not be confirmed. Weighting the predictions depending on the positions does not lead to better results.

A more complex approach is based on the work by Wachsmuth et al. [67] on combining local sentiments into a global sentiment. We try to use their approach for stance detection instead of sentiment prediction. Mao and Lebanon [41] introduce the concept of sentiment flow as the idea that the argumentation of reviews can be represented by the sequence of local sentiments. Wachsmuth et al. [67] propose a classifier based on this notion and present results that suggest that sentiment flow can be used domain-independent to predict the global sentiment of reviews out of local sentiments. They employ a random forest classifier based on a collection of features. Besides baseline features such as Bag-of-Words, the local sentiment frequency, and the distance to centroids obtained by a sentiment flow clustering, they use the distance to certain flows after applying a range of operations on the sentiment flow. We evaluate if this approach is also helpful for more diverse text and the stance towards a comparison instead of just sentiments in reviews. We use all calculated features because they worked best for most of the setups evaluated in the original paper.

We compare the results obtained through this method with the baseline results. While evaluating the different sentence classifiers we saw that the one for the sentences with one target on its own produces the best results, followed by the one for sentences with two objects. The classifier for the context sentences is the worst one. So, we decide to try three different ways to produce the sentence stance predictions. First, we predict the stance only for sentences with one object, second, we predict it for sentences with one or two objects, and finally, for all sentences. In Table 5.6 we compare the baseline method, i.e., simply averaging the sentence predictions, with the stance flow classifier for each of these three ways to generate the sentence predictions.

Table 5.6: Comparison of the accuracy of both combination approaches for the global stance. The results are based on predictions for all sentences (all), all sentences with targets (one or two), and just the sentences with one target.

	Sentence predictions		
	All	**One & two targets**	**One target**
Baseline (avg.)	0.48	0.51	0.52
Stance Flow	0.48	0.51	0.52

The results show that independent of the way to infer the global stance, it is not helpful to predict the local stance for the three sentence categories. It works best to just predict the stance for the sentences containing exactly one object. Furthermore, the results show that using transformer-based models for sentence-based prediction is a competitive alternative to predicting the stance for the whole answer. We assume that it might be possible to further increase the effectiveness by improving the classifiers for the individual sentences, especially for those with two or no targets. Our models for these sentences do not help to improve the results for detecting the stance of the whole answer.

The transfer of the sentiment flow model from sentiment analysis on reviews to stance detection on answers to comparative questions is not successful, as it does not improve on the baseline. It might be possible to increase the results by finding a better model for combining the local stances into a global stance.

5.2.3 Ensemble Classifier

After building and optimizing several different classifiers we investigate if we can improve the classification results even further by building an ensemble classifier on top of these different models. We evaluate different methods to combine the best models for each input. RoBERTa-Pair is trained on the target answer pairs, RoBERTa uses the masked answer and RoBERTa-One predicts the stance for all sentences with one object and averages them as previously described into one overall stance.

We manage to achieve the best results with an ensemble that predicts the maximum of the weighted average of the predictions from each classifier. To weight the predictions of each individual classifier we use the accuracy they achieved during the 5-fold cross-validation on the train set. However, this ensemble does not manage to outperform the best individual classifier. The results are shown in Table 5.7. To make these final results reproducible, we use manual seeds (see Table A.1) for initializing the weights of the final layers of these transformer-based models. Because we fine-tuned the models with this manual seed, we see some minor differences to the previously reported results for the classifiers.

Table 5.7: Comparison of the F1 of the ensemble classifier, the best individual classifiers, and the baseline.

	None	*Neutral*	*First*	*Second*	Micro-avg.
Baseline (SVM)	0.09	0.48	0.41	0.36	0.40
RoBERTa-Pair	0.30	0.50	0.47	0.39	0.44
RoBERTa-One	0.15	0.33	0.61	0.58	0.51
RoBERTa-Mask	0.35	0.64	0.62	0.63	0.62
Ensemble	0.29	0.64	0.63	0.63	0.62

5.3 Analysis for Practical Application

After designing, evaluating, and comparing different approaches to detect the stance of the answers, we analyze the classifiers for practical applications. First, we analyze the best classifiers using the classified targets instead of the annotated ones to show a more realistic view of the accuracy of these models. Second, we investigate if our models are capable to be employed in our envisioned use-case as part of search engines helping to show users the relevant information for comparative questions.

5.3.1 Using Classified Targets

To evaluate the different approaches to detect the stance, we have previously used the manually annotated targets. To get a more realistic impression of the performance of the stance detection, we evaluate these models using the targets identified by the classifiers designed in Chapter 4. In Table 5.8 we compare the accuracy of the stance detection methods using annotated and classified targets. As the identification of the answer targets is considerably less accurate than the question target identification, the two stance classifiers relying on answer targets, RoBERTa-Mask and RoBERTa-One, suffer more from using the classified targets. The accuracy of RoBERTa-Mask is drastically reduced as the answers are not correctly masked. The RoBERTa-One model also relies on the answer targets to identify the sentences with one compared object. The accuracy of the other two models does not change significantly. Nevertheless, RoBERTa-Mask still achieves the best results by a slight margin.

Table 5.8: Comparison of the micro-averaged accuracy for stance detection with annotated and classified targets.

	Accuracy	
	Annotated targets	**Classified targets**
Baseline	0.40	0.39
RoBERTa-Pair	0.44	0.44
RoBERTa-One	0.51	0.40
RoBERTa-Mask	0.62	0.45

5.3.2 As Part of Search Engines

We initially motivated our research of stance detection on answers to comparative questions as a possible way to help search engines show more relevant information to the user (cf. Section 1). The envisioned use-case for the classifier is to categorize answers to comparative question queries depending on their stance, enabling a better presentation of results compared to just a list of relevant websites. As changing the presentation of search results is an impactful decision, this task is highly precision-oriented. The web is full of text, enabling us to accept a lower recall for the classifier. We investigate how we achieve the best precision in exchange for a lower recall using the classified targets.

The precision-recall curves for our three best classifiers (RoBERTa-Pair, RoBERTa-Mask, RoBERTa-One) are shown in Figure 5.1. Regardless of the targeted recall, RoBERTa-Mask delivers the best precision even using classified targets. The precision-recall curves per label for this classifier are shown in Figure 5.2.

To increase the precision, we set a threshold for the probability and ignore any prediction the model makes with a probability lower than this threshold. We choose the threshold such that we achieve a pre-determined value for the precision on the cross-validated train set resulting in a certain recall. To evaluate the results we calculate the precision and recall achieved on the test set using the same threshold. The resulting values are shown in Table 5.9. For comparison, we show the same values for annotated targets in Table A.5.

Table 5.9: Precision and recall on the cross-validated train set and the test set using threshold values to achieve a certain precision on the cross-validated train set. Evaluated on our dataset using the classified targets.

	Train set		**Test set**	
Threshold	**Precision**	**Recall**	**Precision**	**Recall**
0.976	**1.00**	0.03	0.81	0.02
0.972	**0.95**	0.06	0.81	0.03
0.972	**0.90**	0.06	0.81	0.03
0.967	**0.80**	0.09	0.77	0.08

While we can definitely use thresholds to increase the precision, the results on unseen data are not fully predictable, as the precision and recall values differ strongly between train and test set. However, we assume, this is at least partially caused by the small size of the test set.

As the web contains such a vast amount of content, the very low recall might even be acceptable. More importantly, as the results on the test set show, we are not able to reliably deliver a certain high precision. As long as this is the case, our classifier is far from being used productively in a search engine scenario, even ignoring any potential other problems regarding the broader context, e.g., question identification and answer retrieval.

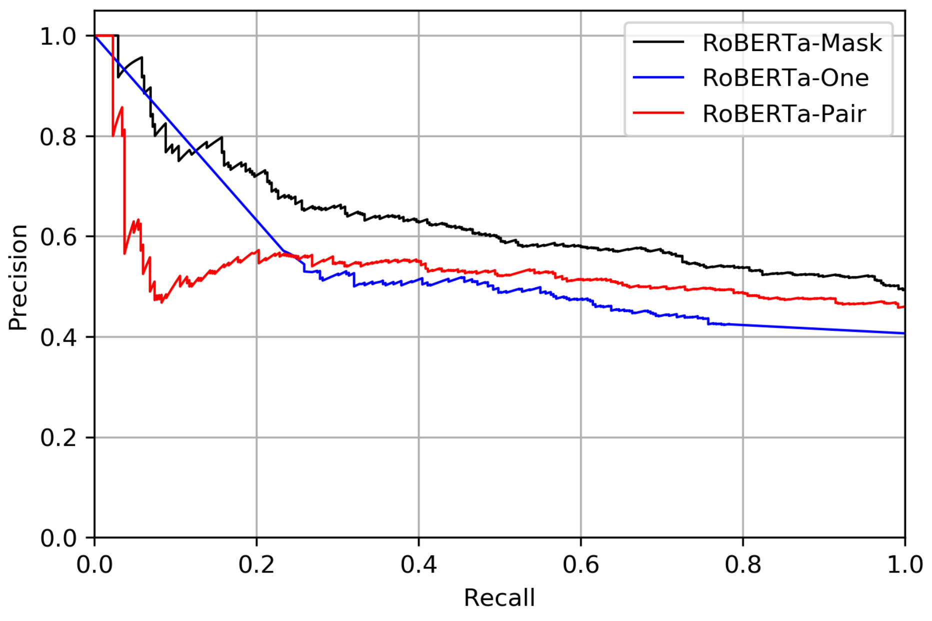


Figure 5.1: The precision-recall curves for RoBERTa-Mask, RoBERTa-Pair, and RoBERTa-One on the 5-fold cross-validated train set using micro-averaged precision.

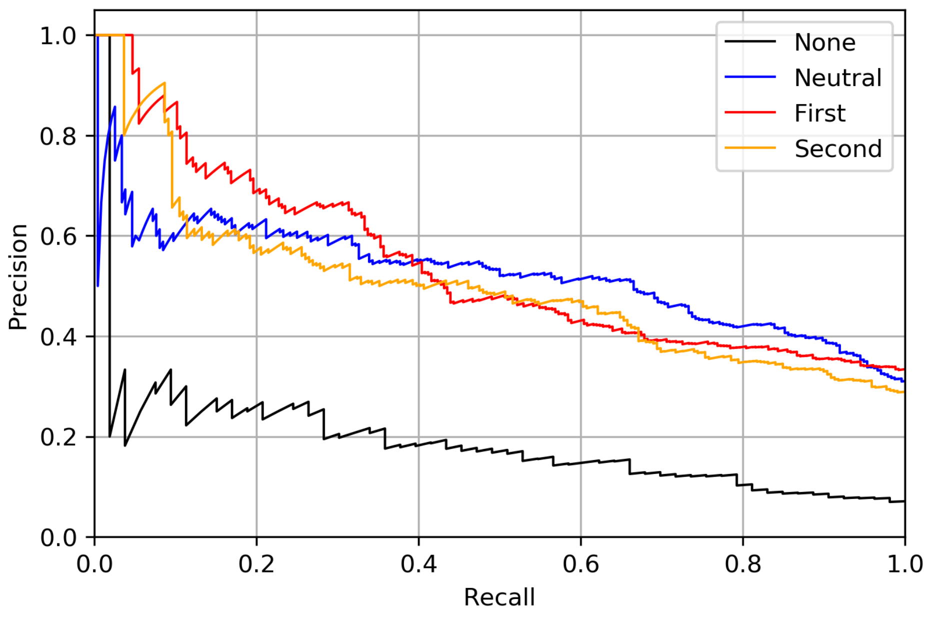


Figure 5.2: The precision-recall curves per label for RoBERTa-Mask on the 5-fold cross-validated train set.

Chapter 6

Conclusion

We tackle the task of identifying the stance of answers towards compared objects in argumentative comparative questions. We focus on questions that ask for a comparison of exactly two options, thus, the answer's stance always has two targets, making stance detection difficult.

As one of our contributions, we created a stance detection dataset for answers to comparative questions. The dataset contains 1,034 comparative questions sampled from CQA websites with their respective answer. We manually annotated the compared objects and the stance towards these.

Furthermore, we designed and evaluated methods to identify the stance targets in the questions and answers. We successfully employed a transformer-based model to identify the targets in the question and used a classifier based on similarity measures to find corresponding targets in the answer.

Our main contribution is the design and evaluation of an extensive selection of approaches to detect the stance in the answer. We compared feature-based baseline classifiers with more advanced transformer-based models. Our experiments focus on improving the effectiveness of the transformer-based models and explore several ideas to accomplish this.

These experiments are also relevant for other tasks as we show by improving on previous results on the CompArg [47] and VAST [2] dataset. In both cases, we manage to achieve better results with a general-purpose transformer-based model compared to a complex task-specific classifier.

Our idea of using other datasets for additional fine-tuning to improve the results for small datasets was not successful. While the VAST dataset was shown to be similar enough to produce a good classifier for the individual sentences it was not useful in improving the results when used for additional fine-tuning.

We investigated the idea of using sentence-based classifiers to detect the stance for the whole answer. While the application of the idea of sentiment flow was not successful, simply averaging the predictions for the single sentences lead to an accuracy not significantly worse than our most effective model.

The most effective classifier for our dataset is a RoBERTa model trained and evaluated on the answer, where all targets have been replaced with unique placeholders. On the answers to comparative questions, this multi-class classifier achieves a micro-averaged accuracy of 0.62 for detecting the stance represented by four different labels (*none*, *neutral*, *first*, *second*).

Such a stance classifier could be employed to help users find more relevant information for argumentative comparative questions. For instance, search engines could categorize answers to comparative questions regarding their stance and give users a more direct presentation of arguments and the prevailing stance regarding the comparison. However, we showed that even our best classifier fails to achieve the very high precision needed for a productive application in this context.

A promising direction for future work can be the advancement of the sentence-based classifier. Stance annotated sentences closer resembling the sentences in our dataset could enable the training of a significantly better sentence stance classifier. Additionally, using more advanced methods to identify the target for sentences containing no compared objects could deliver usable stance predictions for these sentences. These improved classifiers or, additionally, other methods to combine the sentence predictions might improve the overall answer stance prediction.

Another area for further research is superlative comparative questions. We focus on non-superlative questions, as the targets for stance detection are more evident. It would be interesting to investigate how the targets and stance for superlative questions can be identified.

Appendix A

Tables

Table A.1: The hyperparameters for the transformer-based models to use with the library Simple Transformers to reproduce the relevant results in this thesis sorted by section.

Section	Name	model_name	num_train_epochs	train_batch_size	learning_rate	manual_seed
3.2						
	comparative argumentative	bert-base-uncased	3	8	3E-5	
	comparative	bert-base-uncased	3	8	3E-5	
	argumentative	bert-base-uncased	3	8	4E-5	
	superlative	bert-base-uncased	3	8	3E-5	
4.1						
		bert-base-uncased	50	8	5e-5	
4.2						
		bert-base-uncased	3	16	2e-5	
5.2.1.1						
		bert-base-uncased	10	16	2e-5	
		xlnet-base-cased	10	16	2e-5	
		roberta-base	10	16	2e-5	
5.2.1.2						
		bert-base-uncased	10	16	2e-5	
		xlnet-base-cased	10	16	3e-5	
		t5-base	10	16	4e-5	
		roberta-base	10	16	2e-5	
5.2.1.3						
		roberta-base	10	16	2e-5	
5.2.2.1						
	RoBERTa-Pair	roberta-base	10	16	2e-5	
	RoBERTA-Masked	roberta-base	10	16	3e-5	
5.2.2.2						
	RoBERTa-Pair	roberta-base	10	32	3e-5	
5.3.1						
	RoBERTa-Pair	roberta-base	10	16	2e-5	12341
	RoBERTa-One	roberta-base	10	32	2e-5	12341
	RoBERTa-Mask	roberta-base	10	16	2e-5	12341

Table A.2: The number of the total, selected, and annotated questions from the different Stack Exchange domains. The selected questions are those that are classified as comparative, argumentative, and non-superlative. From this set, the annotated questions were randomly chosen.

Domain	**Number of questions**		
	Total	**Selected**	**Annotated**
Academia	32,044	199 (0.62%)	100
Computer Science	36,105	85 (0.24%)	50
Gardening & Landscaping	13,691	65 (0.47%)	50
Music: Practice & Theory	19,114	145 (0.76%)	50
Seasoned Advice (Cooking)	23,215	273 (1.18%)	100
Software Engineering	55,725	700 (1.26%)	130
Software Recommendations	21,234	102 (0.48%)	50
Super User (Computers)	444,609	2,039 (0.46%)	100
Travel	41,149	225 (0.55%)	100

Table A.3: Example questions and answers with their stance labels. The question and answer targets are in bold.

Question	Answer	Stance
Which is better **the ps3** or **the xobo360**?	**The Ps3** will be the dominating console..It will be more graphically pleasing, and will be able to handle a lot more information at a much quicker rate than **Xbox 360** ..night and day in comparison..not to mention **PS3** will have a wider range of entertainment for the system..The early start for Microsoft's new **Xbox 360** is their only chance..but it cannot compare to what Sony is touting that **the PS3** will be doing..Sony does do nice work..They have invested a Billion dollars into the PS2 but mostly **the PS3**... 2 thumbs up for **Sony's PS3**!!	*First*
Is it better **to use hot water** or **cold** with the garbage disposal running?	**Cold** While I agree that **the hot water** should break the grease down, a garbage disposal manual will tell you **to run cold water** down the drain for a few seconds after running the disposal. * always **use cold water** when operating the disposer to solidify fatty & greasy waste so they will be chopped up & flushed down the drain. * it's safe **to run hot water** from the sink through the garbage disposer. However, **use cold water** when you are operating the disposer.	*Second*
Is it better to buy **a condo** or **a house** in Las Vegas?	**Condo** is less maintenance, but **a house** will be worth more later. There's also a huge **Condo** boom on "the strip" right now. Tons of new highrises are going up daily, only time will tell!	*Neutral*
VISE or **Installer-Maker** for Snow Leopard?	Use Package Maker from Apple. It's free and is included with the Developers Kit from Apple. I believe it will handle what issues you have, but if not check out LanRev Install Ease. InstallVise is a pain for anyone that is trying to create a automated software clone. PackageMaker & LanRev Install Ease will create Packages, which can contain multiple versions of your software, for multiple versions of the OS. And they can be used in Apple's SUI, InstaDMG, and etc for Automated OS creation...	*None*

Table A.4: Comparison of the F1 of RoBERTa-Mask on CompArg [47] and the sentences with two objects in our dataset.

	Better	*Worse*	*None*	Micro-avg.
CompArg [47]	0.87	0.67	0.95	0.91
Our two object sentences	0.44	0.23	0.78	0.62

Table A.5: Precision and recall on the cross-validated train set and the test set using threshold values to achieve a certain precision on the cross-validated train set. Evaluated on the data using the annotated targets.

	Train set		**Test set**	
Threshold	**Precision**	**Recall**	**Precision**	**Recall**
0.978	**1.00**	0.03	1.00	0.01
0.978	**0.95**	0.03	1.00	0.01
0.976	**0.90**	0.03	1.00	0.01
0.963	**0.80**	0.06	0.77	0.09

Appendix B

Algorithms

```
Input: Question, Answer
Output: Target in answer or none if not extractable
if question has multiple answers or answer has multiple sentences or
  answer starts with "yes" or "no" then
    return None;
else if answer is one token long or root of answer preposition then
    return answer;
else if answer is noun phrase then
    objects = split answer on ",", " and " and " or ";
    return objects;
else if lemma of root of answer is "be" then
    first phrase, second phrase = split answer on "be" lemma;
    if one phrase has a significant overlap with question and other
      phrase is noun phrase then
        return noun phrase;
    else
        return None;
    end
else
    return None;
end
```

Algorithm B.1: Extract targets from answer.

```
Input: Question
Output: Question Targets
qt = tokenize question with spacy;
if qt contains one of the following [‘vs’, ‘vs.’, ‘or’] then
    x = token at position of [‘vs’, ‘vs.’, ‘or’] in the question;
    target_1 = token left of x;
    target_2 = token right of x;
    if target_1 is part of noun chunk then
        target_1 = get whole noun chunk;
    if target_2 is part of noun chunk then
        target_2 = get whole noun chunk;
else if qt contains token dependency tagged as conj then
    x = first token tagged as conj;
    target_1 = left dependency subtree of dependency parent of x;
    target_2 = dependency subtree of x;
else if question contains ‘difference’ and ‘between’ then
    sep = find position of [‘and’, ‘&’, ‘or’, ‘vs’] in question after
      ‘between’;
    target_1 = words between position of ‘between’ and sep;
    target_2 = words after sep;
return target_1, target_2
```

Algorithm B.2: Rule-based approach to identify the question targets.

Bibliography

[1] R. Abbott, B. Ecker, P. Anand, and M. A. Walker. Internet argument corpus 2.0: An SQL schema for dialogic social media and the corpora to go with it. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, May 23-28, 2016*. European Language Resources Association (ELRA), 2016. 2

[2] E. Allaway and K. R. McKeown. Zero-Shot Stance Detection: A Dataset and Model using Generalized Topic Representations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020*, pages 8913–8931. Association for Computational Linguistics, 2020. 1, 2, 5.2.1, 5.2.2, 5.5, 6

[3] P. Anand, M. Walker, R. Abbott, J. E. Fox Tree, R. Bowmani, and M. Minor. Cats Rule and Dogs Drool!: Classifying Stance in Online Debate. In *Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis (WASSA 2.011)*, pages 1–9. Association for Computational Linguistics, 2011. 2

[4] I. Augenstein, T. Rocktäschel, A. Vlachos, and K. Bontcheva. Stance Detection with Bidirectional Conditional Encoding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016*, pages 876–885. The Association for Computational Linguistics, 2016. 2

[5] R. Bar-Haim, I. Bhattacharya, F. Dinuzzo, A. Saha, and N. Slonim. Stance Classification of Context-Dependent Claims. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 251–261. Association for Computational Linguistics, 2017. 2, 5.1, 5.2.2

[6] Y. Bilu and N. Slonim. Claim Synthesis via Predicate Recycling. In *Proceedings of the 54th Annual Meeting of the Association for Computa-*

tional Linguistics (Volume 2: Short Papers), pages 525–530. Association for Computational Linguistics, 2016. 2

[7] A. Bondarenko, P. Braslavski, M. Völske, R. Aly, M. Fröbe, A. Panchenko, C. Biemann, B. Stein, and M. Hagen. Comparative Web Search Questions. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining*, pages 52–60. ACM, 2020. 1, 1, 2, 3, 3.2

[8] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, et al. Universal Sentence Encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174, 2018. 5.1

[9] T. Chen and C. Guestrin. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794. ACM, 2016. 5.2.2

[10] A. N. Chernodub, O. Oliynyk, P. Heidenreich, A. Bondarenko, M. Hagen, C. Biemann, and A. Panchenko. TARGER: Neural Argument Mining at Your Fingertips. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Volume 3: System Demonstrations*, pages 195–200. Association for Computational Linguistics, 2019. 1

[11] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes. Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680. Association for Computational Linguistics, 2017. 5.1, 5.2.2

[12] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019. 3.2, 5.2

[13] V. Dittmar. Erkennen und Verstehen von vergleichenden Fragen, 2020. 2, 3.2, 4.1

[14] C. Dulhanty, J. L. Deglint, I. B. Daya, and A. Wong. Taking a Stance on Fake News: Towards Automatic Disinformation Assessment via Deep

Bidirectional Transformer Language Models for Stance Detection. *CoRR*, abs/1911.11951, 2019. 2, 5.2

[15] R. Eckart de Castilho, É. Mújdricza-Maydt, S. M. Yimam, S. Hartmann, I. Gurevych, A. Frank, and C. Biemann. A Web-based Tool for the Integrated Annotation of Semantic and Syntactic Structures. In *Proceedings of the Workshop on Language Technology Resources and Tools for Digital Humanities (LT4DH)*, pages 76–84. The COLING 2016 Organizing Committee, 2016. 3.3

[16] A. Faulkner. Automated Classification of Stance in Student Essays: An Approach Using Stance Target Information and the Wikipedia Link-Based Measure. *Proceedings of the 27th International Florida Artificial Intelligence Research Society Conference, FLAIRS 2014*, pages 174–179, 2014. 2

[17] C. Fellbaum. WordNet. In *Theory and Applications of Ontology: Computer Applications*, pages 231–243. Springer, 2010. 4.2

[18] J. L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378, 1971. 3.3

[19] E. Graells-Garrido, R. Baeza-Yates, and M. Lalmas. Every Colour You Are: Stance Prediction and Turnaround in Controversial Issues. *12th ACM Conference on Web Science*, 2020. 2, 5.2

[20] S. Gupta, A. A. Mahmood, K. Ross, C. Wu, and K. Vijay-Shanker. Identifying Comparative Structures in Biomedical Text. In *BioNLP 2017*, pages 206–215. Association for Computational Linguistics, 2017. 2

[21] I. Habernal and I. Gurevych. Exploiting Debate Portals for Semi-Supervised Argumentation Mining in User-Generated Web Discourse. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2127–2137. Association for Computational Linguistics, 2015. 2

[22] Y. HaCohen-Kerner, Z. Ido, and R. Ya'akobov. Stance Classification of Tweets using Skip Char Ngrams. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 266–278. Springer, 2017. 2

[23] K. S. Hasan and V. Ng. Stance Classification of Ideological Debates: Data, Models, Features, and Constraints. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 1348–1356. Asian Federation of Natural Language Processing, 2013. 2

[24] M. Honnibal and I. Montani. spaCy 2: Natural Language Understanding with Bloom Embeddings, Convolutional Neural Networks and Incremental Parsing. To appear, 2017. 4.1

[25] M. Hu and B. Liu. Mining and Summarizing Customer Reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, page 168–177. Association for Computing Machinery, 2004. ISBN 1581138881. 5.1

[26] Y. Igarashi, H. Komatsu, S. Kobayashi, N. Okazaki, and K. Inui. Tohoku at SemEval-2016 Task 6: Feature-based Model versus Convolutional Neural Network for Stance Detection. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 401–407. Association for Computational Linguistics, 2016. 2

[27] S. Iyer, N. Dandekar, and K. Csernai. First quora dataset release: Question pairs, 2017. 3.2

[28] A. Jain and P. Pantel. Identifying Comparable Entities on the Web. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009*, pages 1661–1664. ACM, 2009. 2

[29] N. Jindal and B. Liu. Identifying Comparative Sentences in Text Documents. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, page 244–251. Association for Computing Machinery, 2006. ISBN 1595933697.

[30] N. Jindal and B. Liu. Mining Comparative Sentences and Relations. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 2*, AAAI'06, page 1331–1336. AAAI Press, 2006. ISBN 9781577352815. 2

[31] G. A. Kennedy. Aristotle On Rhetoric: A Theory of Civic Discourse. *Philosophy and Rhetoric*, 26(4):322–327, 1993. 2

[32] D. Küçük and F. Can. Stance Detection: A Survey. *ACM Comput. Surv.*, 53(1), 2020. ISSN 0360-0300. 2

[33] A. Kulkarni, N. R. Uppalapati, P. Singh, and G. Ramakrishnan. An Interactive Multi-Label Consensus Labeling Model for Multiple Labeler Judgments. In *AAAI*, pages 1479–1486, 2018. 5

[34] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein, I. Polosukhin, M. Kelcey, J. Devlin, K. Lee, K. N. Toutanova, L. Jones, M.-W. Chang, A. Dai, J. Uszkoreit, Q. Le, and S. Petrov. Natural Questions: a Benchmark for Question Answering Research. *Transactions of the Association of Computational Linguistics*, 2019. 3.2

[35] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In *International Conference on Learning Representations*, 2019. 5.2

[36] S. Li, C. Lin, Y. Song, and Z. Li. Comparable Entity Mining from Comparative Questions. *IEEE Transactions on Knowledge and Data Engineering*, 25(7):1498–1509, 2013. 2

[37] Y. Li and C. Caragea. Multi-Task Stance Detection with Sentiment and Stance Lexicons. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6299–6305. Association for Computational Linguistics, 2019. 2

[38] C. Liu, W. Li, B. Demarest, Y. Chen, S. Couture, D. Dakota, N. Haduong, N. Kaufman, A. Lamont, M. Pancholi, et al. IUCL at SemEval-2016 Task 6: An Ensemble Model for Stance Detection in Twitter. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 394–400, 2016. 2

[39] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR*, abs/1907.11692, 2019. 3.2, 5.2

[40] N. Ma, S. Mazumder, H. Wang, and B. Liu. Entity-Aware Dependency-Based Deep Graph Attention Network for Comparative Preference Classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5782–5788. Association for Computational Linguistics, 2020. 2, 5.2.2, 5.4

[41] Y. Mao and G. Lebanon. Isotonic Conditional Random Fields and Local Sentiment Flow. In *Advances in neural information processing systems*, pages 961–968, 2007. 2, 5.2.2

[42] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 3111–3119, 2013. 4.2

[43] S. Mohammad, S. Kiritchenko, P. Sobhani, X. Zhu, and C. Cherry. SemEval-2016 Task 6: Detecting Stance in Tweets. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 31–41. Association for Computational Linguistics, 2016. 2

[44] A. Murakami and R. Raymond. Support or Oppose? Classifying Positions in Online Debates from Reply Activities and Opinion Expressions. In *Coling 2010: Posters*, pages 869–875. Coling 2010 Organizing Committee, 2010. 2

[45] T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder, and L. Deng. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016)*, volume 1773 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2016. 3.2, 3.3

[46] X. Pan, K. Sun, D. Yu, J. Chen, H. Ji, C. Cardie, and D. Yu. Improving Subject-Area Question Answering with External Knowledge. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 27–37, 2019. 5.2.1

[47] A. Panchenko, A. Bondarenko, M. Franzek, M. Hagen, and C. Biemann. Categorizing Comparative Sentences. In *Proceedings of the 6th Workshop on Argument Mining*, pages 136–145. Association for Computational Linguistics, 2019. 1, 2, 3.4, 5.2.2, 5.2.2, 5.4, 6, A.4

[48] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. 5.2

[49] A. Rajadesingan and H. Liu. Identifying Users with Opposing Opinions in Twitter Debates. 2014. ISBN 978-3-319-05578-7. 2

[50] N. Reimers and I. Gurevych. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing.* Association for Computational Linguistics, 2019. 5.1

[51] R. Rinott, L. Dankin, C. Alzate, M. M. Khapra, E. Aharoni, and N. Slonim. Show Me Your Evidence - an Automatic Method for Context Dependent Evidence Detection. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 440–450, 2015. 2

[52] Y. Samih and K. Darwish. A Few Topical Tweets are Enough for Effective User-Level Stance Detection. *CoRR*, abs/2004.03485, 2020. 2

[53] V. Sanh, L. Debut, J. Chaumond, and T. Wolf. DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter. *CoRR*, abs/1910.01108, 2019. 5.2

[54] M. Schildwächter, A. Bondarenko, J. Zenker, M. Hagen, C. Biemann, and A. Panchenko. Answering Comparative Questions: Better than Ten-Blue-Links? In *Proceedings of the 2019 Conference on Human Information Interaction and Retrieval, CHIIR 2019*, pages 361–365. ACM, 2019. 1, 2

[55] B. Schiller, J. Daxenberger, and I. Gurevych. Stance detection benchmark: How robust is your stance detection? *CoRR*, abs/2001.01565, 2020. 2, 5.2.2

[56] U. A. Siddiqua, A. N. Chy, and M. Aono. Stance Detection on Microblog Focusing on Syntactic Tree Representation. In *International Conference on Data Mining and Big Data*, pages 478–490. Springer, 2018. 2

[57] P. Sobhani, D. Inkpen, and X. Zhu. A Dataset for Multi-Target Stance Detection. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 551–557. Association for Computational Linguistics, 2017. 2

[58] S. Somasundaran and J. Wiebe. 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, 2010. 2

[59] C. Stab and I. Gurevych. Identifying Argumentative Discourse Structures in Persuasive Essays. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 46–56. Association for Computational Linguistics, 2014. 2

[60] C. Stab, J. Daxenberger, C. Stahlhut, T. Miller, B. Schiller, C. Tauchmann, S. Eger, and I. Gurevych. ArgumenText: Searching for Arguments in Heterogeneous Sources. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 21–25. Association for Computational Linguistics, 2018. 1, 5.2.2

[61] M. Taulé, M. A. Martí, F. M. Rangel, P. Rosso, C. Bosco, V. Patti, et al. Overview of the Task on Stance and Gender Detection in Tweets on Catalan Independence at IberEval 2017. In *2nd Workshop on Evaluation of Human Language Technologies for Iberian Languages, IberEval 2017*, volume 1881, pages 157–177. CEUR-WS, 2017. 2

[62] M. Thomas, B. Pang, and L. Lee. Get out the vote: Determining support or opposition from Congressional floor-debate transcripts. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 327–335. Association for Computational Linguistics, 2006. 2

[63] M. Tutek, I. Sekulić, P. Gombar, I. Paljak, F. Čulinović, F. Boltužić, M. Karan, D. Alagić, and J. Šnajder. TakeLab at SemEval-2016 Task 6: Stance Classification in Tweets Using a Genetic Algorithm Based Ensemble. In *Proceedings of the 10th international workshop on semantic evaluation (SemEval-2016)*, pages 464–468, 2016. 2

[64] J. Vamvas and R. Sennrich. X-stance: A Multilingual Multi-Target Dataset for Stance Detection. In *Proceedings of the 5th Swiss Text Analytics Conference and the 16th Conference on Natural Language Processing, SwissText/KONVENS 2020*, volume 2624 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2020. 2, 5.2

[65] P. Vijayaraghavan, I. Sysoev, S. Vosoughi, and D. Roy. DeepStance at SemEval-2016 Task 6: Detecting Stance in Tweets Using Character and Word-Level CNNs. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 413–419. Association for Computational Linguistics, 2016. 2

[66] R. Vinayakumar, S. Kumar, B. Premjith, P. Poornachandran, and S. K. Padannayil. Deep Stance and Gender Detection in Tweets on Catalan Independence@Ibereval 2017. In *IberEval@SEPLN*, 2017. 2

[67] H. Wachsmuth, J. Kiesel, and B. Stein. Sentiment Flow-A General Model of Web Review Argumentation. In *Proceedings of the 2015 Conference on*

Empirical Methods in Natural Language Processing, pages 601–611, 2015. 2, 5.2.2

[68] H. Wachsmuth, N. Naderi, Y. Hou, Y. Bilu, V. Prabhakaran, T. A. Thijm, G. Hirst, and B. Stein. Computational Argumentation Quality Assessment in Natural Language. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 176–187. Association for Computational Linguistics, 2017. 2

[69] H. Wachsmuth, M. Potthast, K. Al Khatib, Y. Ajjour, J. Puschmann, J. Qu, J. Dorsch, V. Morari, J. Bevendorff, and B. Stein. Building an Argument Search Engine for the Web. In *Proceedings of the 4th Workshop on Argument Mining*, pages 49–59, 2017. 1, 2

[70] M. Walker, P. Anand, R. Abbott, and R. Grant. Stance Classification using Dialogic Properties of Persuasion. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 592–596. Association for Computational Linguistics, 2012. 2

[71] M. A. Walker, P. Anand, R. Abbott, J. E. F. Tree, C. Martell, and J. King. That is your evidence?: Classifying stance in online political debate. *Decision Support Systems*, 53(4):719 – 729, 2012. ISSN 0167-9236. 1) Computational Approaches to Subjectivity and Sentiment Analysis 2) Service Science in Information Systems Research : Special Issue on PACIS 2010. 2

[72] B. Wang, M. Liakata, A. Zubiaga, and R. Procter. TDParse: Multi-target-specific sentiment recognition on Twitter. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 483–493. Association for Computational Linguistics, 2017. 2

[73] L. Wang and W. Ling. Neural Network-Based Abstract Generation for Opinions and Arguments. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 47–57. Association for Computational Linguistics, 2016. 2

[74] R. Wang, D. Zhou, M. Jiang, J. Si, and Y. Yang. A Survey on Opinion Mining: From Stance to Product Aspect. *IEEE Access*, 7:41101–41124, 2019. 2

[75] P. Wei, W. Mao, and D. Zeng. A Target-Guided Neural Memory Model for Stance Detection in Twitter. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2018. 2

[76] W. Wei, X. Zhang, X. Liu, W. Chen, and T. Wang. pkudblab at SemEval-2016 Task 6 : A Specific Convolutional Neural Network System for Effective Stance Detection. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 384–388. Association for Computational Linguistics, 2016. 2

[77] Z. Wu and M. Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 133–138, 1994. 4.2

[78] C. Xu, C. Paris, S. Nepal, and R. Sparks. Cross-Target Stance Classification with Self-Attention Networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 778–783. Association for Computational Linguistics, 2018. 2

[79] R. Xu, Y. Zhou, D. Wu, L. Gui, J. Du, and Y. Xue. Overview of NLPCC Shared Task 4: Stance Detection in Chinese Microblogs. volume 10102, pages 907–916, 2016. ISBN 978-3-319-50495-7. 2

[80] B. Yang, J.-T. Sun, T. Wang, and Z. Chen. Effective Multi-Label Active Learning for Text Classification. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 917–926, 2009. 5

[81] Z. Yang, Z. Dai, Y. Yang, J. G. Carbonell, R. Salakhutdinov, and Q. V. Le. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, pages 5754–5764, 2019. 3.2, 5.2

[82] N. Yu, D. Pan, M. Zhang, and G. Fu. Stance Detection in Chinese MicroBlogs with Neural Networks. volume 10102, pages 893–900, 2016. ISBN 978-3-319-50495-7. 2

[83] G. Zarrella and A. Marsh. MITRE at SemEval-2016 Task 6: Transfer Learning for Stance Detection. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 458–463. Association for Computational Linguistics, 2016. 2

[84] Y. Zhou, A. I. Cristea, and L. Shi. Connecting Targets to Tweets: Semantic Attention-Based Model for Target-Specific Stance Detection. In *International Conference on Web Information Systems Engineering*, pages 18–32. Springer, 2017. 2