Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

FAKULTÄT
FÜR MATHEMATIK, INFORMATIK
UND NATURWISSENSCHAFTEN

Master Thesis

Comparative Argument Mining

Mirco Franzek

5franzek@informatik.uni-hamburg.de
Studiengang Informatik
Matr.-Nr. 6781911

Erstgutachter: Prof. Dr. Chris Biemann
Zweitgutachter: Dr. Alexander Panchenko

Abgabe: April 2017

I am C3P0, protocol droid, human-cyborg relations.
I am fluent in over 6 million forms of communication. – *C3PO*

Contents

1 Introduction

1.1 Motivation: An Open-Domain Comparative Argumentative Machine (CAM)

1.2 Related Work

1.2.1 Argumentation Theory

[Habernal et al., 2014] presented a comparison between the results of two different annotation studies. One used the Claim/Premise-Model, while the other one used the Toulmin model. They emphasized that there is no "one-size-fits-all" model. (ERWEITERN)

(BEGRÜNDEN: warum kein modell genutzt)

1.2.2 Argument Mining

[Lippi and Torroni, 2016] gave a summary of the research topic "Argument Mining" in general. They introduced five dimensions to describe Argument Mining problems: granularity of input, the genre of input, argument model, the granularity of target and goal of analysis. Furthermore, the typical steps of Argument Mining Systems are defined. First, the input must be divided into argumentative (e.g. claim and premise) and non-argumentative parts. This step is described as a classification problem. Second, the boundaries of the argumentative units are identified; this is understood as a segmentation problem. Third, the relations between argumentative units are identified. For instance, claims and premises are connected with a "support" relation.

A system which is capable of recognising comparative sentences and their components such as the compared entities, the property on which the entities are compared to and the direction of comparison was presented in [Fiszman et al., 2007]. The evaluation showed that the outcome has a high quality (SCORE?). However, the presented system is specific to the domain of studies to drug therapy. The system uses patterns generated from sentences (WHICH SENTENCES), as well as domain knowledge. Therefore, the methods cannot be transferred for the problem of this thesis.

[Park and Blake, 2012] presented a domain-specific approach on argumentative sentence detection. The problem is formulated as a binary classification task (a sentence is either comparative or not). As in [Fiszman et al., 2007], the features are tailored for medical publications. Lexical features capture the presence of specific words, many of them bound to the medical domain. The analysis of 274 sentences resulted in syntactic features. (BEISPIEL) Similar to [Fiszman et al., 2007], the features cannot be directly transferred to other domains.

A recent publication on Comparative Argument Mining is [Gupta et al., 2017], where a set of rules for the identification of comparative sentences (and the compared entities) is derived from *Syntactic Parse Trees*. With those rules, the authors achieved a F1 score of 0.87 for the identification of comparative sentences. The rules were obtained from 50 abstracts of biomedical papers. Such being the case, they are domain dependent.

Because this thesis deals with user-generated content from the web, publications dealing with similar data are of interest.

The challenges occurring while processing texts from social media are described in [Šnajder, 2017]. In this publication, social media is broadly defined as "less controlled communication environments [...]". Besides the noisiness of text, missing argument structures and poorly formulated claims are mentioned. It is expected that the text used in this thesis will have the same shortcomings. Additionally, [Šnajder, 2017] emphasized that analyzing social media texts can delivery reasons behind opinions.

In addition to the challenges mentioned above, [Dusmanu et al., 2017] also points to the specialized jargon in user-generated content like hashtags and emoticons. With this in mind, [Dusmanu et al., 2017] classified tweets about the "Brexit" and "Grexit" either as argumentative or as non-argumentative. Besides features used in other mentioned papers, new features covering hashtags and sentiment are added. They achieved a F1 score of 0.78 (using Logistic Regression) for the classification. It must to be said that the data set is small (SIZE) and the domain is rather specific.

Many publications on argument mining are dealing with a classification problem of some kind. Publications dealing with the identification of argument structures are of relevance for this thesis, as they provide valuable insights on the suitability of features and algorithms.

[Aker et al., 2017] summarized and compared features used in other publications for identification of argumentative sentences. In addition, a Convolutional Neural Network (as described in [Kim, 2014]) was tested. Two existing corpora and six different classification algorithms were used. As a result, structural features are most expressive; Random Forest is the best classifier.

[Stab and Gurevych, 2014] described a two-step procedure to identify components of arguments (such as claim and premise) and their relationships (like "premise A supports claim B"). The identification step is formulated as a multi-class classification. For the identification of argumentative components, a F1 score of 0.72 is reported.

How different datasets represent the argumentative unit of a claim is analysed in [Daxenberger et al., 2017]. After an analysis of the datasets and their annotation scheme, [Daxenberger et al., 2017] conducted two experiments. In the first one, each learner (Logistic Regression, Convolutional Neural Networks and LSTM) was trained and evaluated (10-fold cross-validation) on each dataset one after another. On average, the macro F1 score for identifying claims was 0.67 (all results ranging from 0.60 to 0.80). No significant difference between the results of Logistic Regression and the neural models was found. In isolation, lexical, structural and word embeddings were the best features. Structural features turned out to be the weakest. The second experiment was conducted in a cross-domain fashion. For each pair of datasets, one was used as the training set and the other one as the test set. The average macro F1 score was 0.54. In this scenario, the best feature combination outperformed all neural models. However, it is assumed that there might not be enough training data for the neural models. As the last point, [Daxenberger et al., 2017] noted that all claims share at least some lexical clues.

The role of discourse markers in the identification of claims and premises are discussed in [Eckle-Kohler et al., 2015]. A discourse marker is a word or a phrase which connects discourse units (citation). For instance, the word “as” can show a relation between claim and premise: “As the students get frustrated, their performance generally does not improve”. A similar function for words like “better”, “worse” or “because” is expected in this thesis. [Eckle-Kohler et al., 2015] showed that discourse markers are good at discriminating claim and premises. If claim and premise are merged into one class “argumentative”, this can be used to identify argumentative sentences. The F1 score is not presented, but the accuracy is between 64.53 and 72.79 percent.

A summary of several features for the identification of argumentative sentences can be found in chapter **??**.

1.2.3 Domain-Specific Comparative Systems

The enormous amount of Comparison Portals shows the need for comparisons. Frequently aired television spots empathize the popularity of those portals.

Most of those portals are specific to a few domains and a subset of properties, for example, car insurances and their price. Because of that, those systems have some restrictions. Comparisons are only possible between objects of the domains and predefined properties. Source of the data is usually databases. Humans are involved in gathering, entering and processing the data.

Comparison Portals solely compare and deliver facts. Because of that, they can only give the advice to choose X over Y based on the facts collected. An insurance X might be the best in the comparison (e.g., best price), while the internet is full of complaints about lousy service.

Examples of classical Comparative Portals are *Check24*, *Verivox*, *Idealo*, *GoCompare*, and *Compare*[1], just to name a few.

[1] https://check24.de, https://verivox.de, https://idealo.de, https://gocompare.com,

As an example, Check24 can compare a wide variety of different objects like several insurances, credit cards, energy providers, internet providers, flights, hotels and car tires. After the user entered some details (based on the object type, see figure 1.2.3), Check24 shows a ranking of different service providers. The user can choose different properties to re-rank the list. For instance, to compare different DSL providers, the user has to enter her address, how fast the internet should be and if she wants telephone and television as well. She can then select price, speed, and grade (rating) to sort the resulting list.

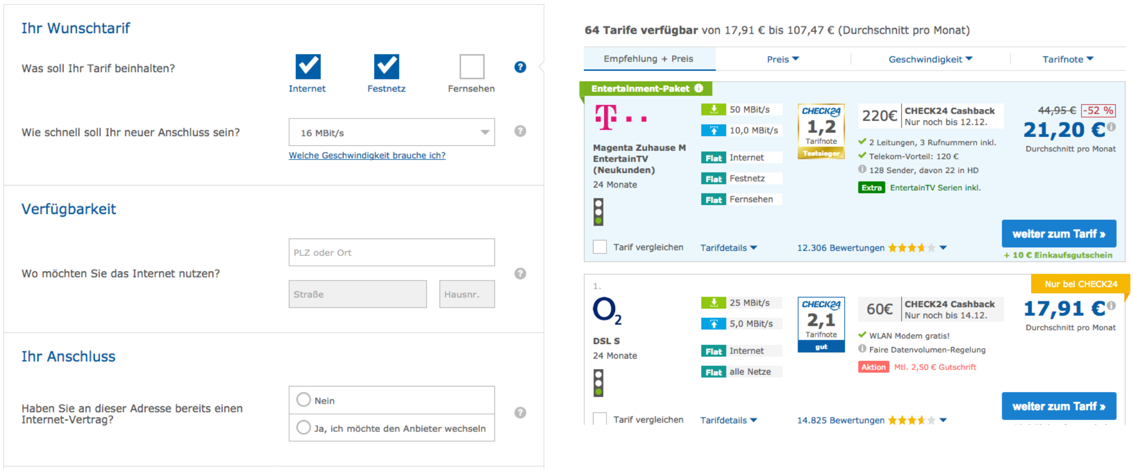


Figure 1.1: Check24 DSL Provider

The other mentioned sites work similarly. They provide more of a ranking than a comparison.

Another interesting type of websites are Question Answering Portals like *Quora* or *GuteFrage*[2]. Although comparisons are not their primary goal, a lot of comparative questions are present on those sites. On Quora, more than 2.380.000 questions have the phrase "better than" in their title. If *Ruby* and *Python* are added, 10.100 questions remain.[3] Same is true for the German site GuteFrage, though, the numbers are smaller than on Quora.[4]

More interestingly are systems which can compare any objects on arbitrary properties. Two examples are *Diffen* and *Versus*[5].

Versus aggregates freely available data sources like Wikipedia and official statistic reports. For example, the comparison of "Hamburg vs. Berlin" uses Wikipedia for the number of universities, worldstadiums.com for the availability of sport facilities and the

https://compare.com - all last checked: 12.12.2017

[2]https://quora.com, https://gutefrage.net - all last checked: 12.12.2017

[3]Checked via Google on 11th of December. Search phrase: `"better than" site:quora.com` and `ruby python "better than" site:quora.com`

[4]334.000 for `"besser als" site:gutefrage.net` and 78 for `ruby python "Besser als" site:gutefrage.net`

[5]https://diffen.com, https://versus.com - all last checked: 12.12.2017

Economist for the Big Mac Index. Presumably, some human processing is involved as the possible comparisons are limited. For instance, a comparison of Hamburg and Darmstadt is not possible as Darmstadt is not available on Versus. Likewise, “Ruby vs. Python” is not possible, Versus suggests to compare “Rome vs. Pyongyang” instead. Although Versus shows how many users “liked” the objects, it does not give a clear statement which one is better. For instance, it is not possible to check automatically whether Hamburg or Berlin is better for a short city trip. The user must search manually all valid properties like the number of museums, theaters, the price of public transport tickets and so on.

Similar to Versus, Diffen aggregates different data sources (see figure 1.2.3). All in all, the aggregated information is similar to Versus. The comparison is also tabular. Besides the automatically aggregated data, users can add information on their own. Diffen describes itself as “inspired by Wikipedia”[6]. Diffen does not enforce any restrictions on the objects of comparison, but it faces the same problem as Versus as objects are missing. A comparison between Darmstadt and Hamburg is likewise not possible: all cells for Darmstadt in the table are just empty.

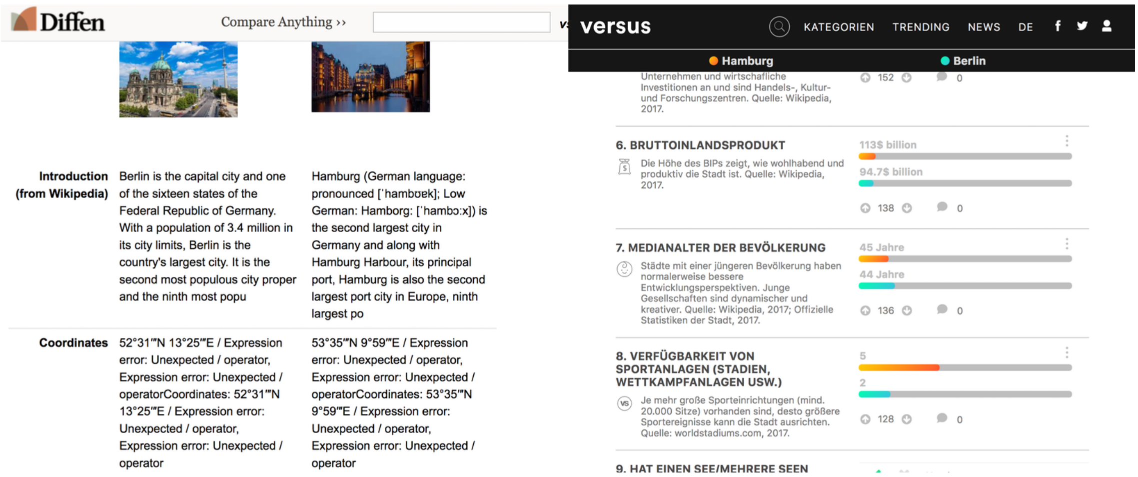


Figure 1.2: “Hamburg vs. Berlin” on Diffen and Versus

Neither Versus nor Diffen provides a comprehensible reason why an object is better than another one. They merely aggregate facts and bring them face to face. Despite the aggregation approach of both systems, many meaningful comparisons are not possible or not helpful (like “Hamburg vs. Darmstadt”, “Java vs. C#”, “Dr Pepper vs. Orange Juice”). Also, the user can not define the properties for the comparison. The sites provide every information available for the objects. For instance, Versus shows 42 properties for “Hamburg vs. Berlin” and only 35 for “Hamburg vs. Munich”.

To summarize, a lot of different comparison portals exist and are widely used. Espe-

[6]https://www.diffen.com/difference/Diffen:About - Last checked: 11.12.2017

cially the domain-specific portals do a good job, but inflexibility dearly buys the performance. First, the portals can only compare objects on predefined properties. Second, the data acquisition is not fully automatic. Domain-unspecific systems are good at aggregating information but do not provide a reasonable explanation to prefer X over Y.

Adding information like comments and product reviews can enrich the comparison with reasons and opinions, such as “Ruby is easier to learn than C” or “Python is more suitable for scientific applications than Erlang as many libraries exist”.

2 Building a data set for Comparative Argument Mining

There's not dataset.

2.1 Common Crawl Text Corpus

The raw data used for the creation of the dataset was derived from CommonCrawl. CommonCrawl is a non-profit organisation which crawls the web and releases the data and metadata with a loose license. This master thesis uses the crawl data from DATE. Furthermore, the data was processed: HTML was stripped out, and the content was splitted into sentences using X. To make the data maintainable, the sentences where imported into an ElasticSearch index. The index has a size of 1.1tb and contains 3,288,963,864 unique sentences.

To get an idea how many sentences in the index may be comparative, searches with cue words was performed. The query 2.1 yields 55,627,400 results, the more specific query `is better than` yields 428,932 results.

Listing 2.1: Candidates for Comparative Sentences

```
{
  "query":{
    "bool":{
      "must":[
        {
          "query_string":{
            "default_field":"text",
            "query":"better OR easier OR faster OR nicer OR wiser OR
                ↪ cooler OR decent OR safer OR superior OR solid OR
                ↪ teriffic OR worse OR harder OR slower OR poorly OR
                ↪ uglier OR poorer OR lousy OR nastier OR inferior OR
                ↪ mediocre"
          }
        }
      ]
    }
  }
}
```

Those numbers indicate that the index contains enough comparative sentences to create machine learning data set.

Lesen [Panchenko et al., 2017]

2.2 Prestudy

Previous to the main study, a pre-study was conducted to assess the quality of the annotation guidelines, the approach of sentence generation and the task itself.

2.2.1 Data Selection and Preprocessing

To obtain comparative sentences from the ElasticSearch index, Query 2.2 was used. The sentence must contain two comparable objects (like "Apple" and "Pear") and at least one cue word. Presence of the cue words “better”, “worse”, “superior” and “inferior” should increase the probability of the sentence to be comparative. In this way, the amount of noisy sentences should be reduced. However, not all comparisons will contain one of the cue words, so 25% of the sentences sentences where obtained without the cue words.

Listing 2.2: Prestudy Sentence Selection Query

```
{
  "query":{
    "bool":{
      "must":[
        {
          "query_string":{
            "default_field":"text",
            "query":"(better OR worse OR superior OR inferior) AND \"<
                ↪ OBJECT_A>\" AND \"<OBJECT_B>\""
          }
        }
      ]
    }
  }
}
```

Ten hand-selected object pairs were used (see table 2.1). It was expected that those pairs will yield differently phrased comparisons, as, for instance, cars are compared in a different way than programming languages. Some sentences contain programming- and computer specific terms, so a need for this knowledge was expressed in the title of the crowdsourcing task.

All retrieved sentences contain each of the objects exactly once.

Table 2.1: Objects of the Annotation Prestudy

First Object	Second Object	# Sentences
Ruby	Python	109
BMW	Mercedes	107
USA	Europe	106
Beef	Chicken	106
Android	iPhone	104
Cat	Dog	104
Football	Baseball	104
Wine	Beer	104
Car	Bicycle	103
Summer	Winter	103
		1050

2.2.2 Task

Using the method described above, 1050 sentences were obtained for the prestudy. The annotators were asked to assigne one of the following classes to the sentences. Each sentence was annotated by three annotators.

The annotators where asked to assign one of the four classes (see table 2.2) to each sentence.

Table 2.2: Classes for the Prestudy

Class	Description
BETTER	The first object in the sentence (object A) is better than the second one (object B)
WORSE	The first object is worse
UNCLEAR	Neither BETTER nor WORSE fits, but the sentence is comparative
NO_COMP	The sentence is not comparative

In a first step, 100 sentences were annotated. To ensure the quality, twelve additional sentences were setup as test sentences. If one annotator failed three test sentences, he was removed from the task.

The sentences were preprocessed: the first object was replaced by OBJECT_A, the second by OBJECT_B. Examples are shown in table 2.3. The removal was done so that the annotators can concentrate on the comparative structure of the sentence and are not biased by the objects.

This test step delivered valuable insights. First, the amount of test sentences was to small. Users might see the same test sentence twice. Second, the phrasing of the annotation guidelines was to confusing, especially the distinction between NO_COMP and UNCLEAR as well as their class names. Third, the complete removal of the original objects is suspected to partly obscure the sense of the sentences.

Table 2.3: Sentences for the first step

Sentence	Expected Class
This is potentially useful for OBJECT_A, PHP, JS and OBJECT_B.	NO_COMP
Also keep in mind that OBJECT_A blends will give you worse mileage than OBJECT_B	WORSE
Snowboarding during OBJECT_A is a lot better than during OBJECT_B.	BETTER

In a second step, 200 new sentences were annotated, again with three annotations per sentences. This time, 51 test questions were used, so that it is less likely that annotators will see the same question twice. Furthermore, the preprocessing was changed. Instead of removing the original objects, :[OBJECT_A] was appended to the first object, :[OBJECT_B] to the second object. Also, each object was highlighted in a different color. Example sentences are shown in table 2.4. In this way, the annotators could quickly see the objects of interest while the sense of the sentence remains intact.

Table 2.4: Sentences for the second step

Sentence	Expected Class
I'd go with **python:[OBJECT_A]** or **ruby:[OBJECT_B]**.	NO_COMP
I prefer **ruby:[OBJECT_A]** over **python:[OBJECT_B]** on windows.	BETTER
I've tried **python:[OBJECT_A]**, and can see why people like it, but **ruby:[OBJECT_B]** suits my style better.	WORSE
i think this car is a far better deal than the **bmw:[OBJECT_A]** 5 series or **mercedes:[OBJECT_B]** 320e.	UNCLEAR

2.2.3 Results

Each sentence was annotated by three annotators. Figure 2.1 shows the class distribution.

Figure 2.1: Class Distribution in the prestudy

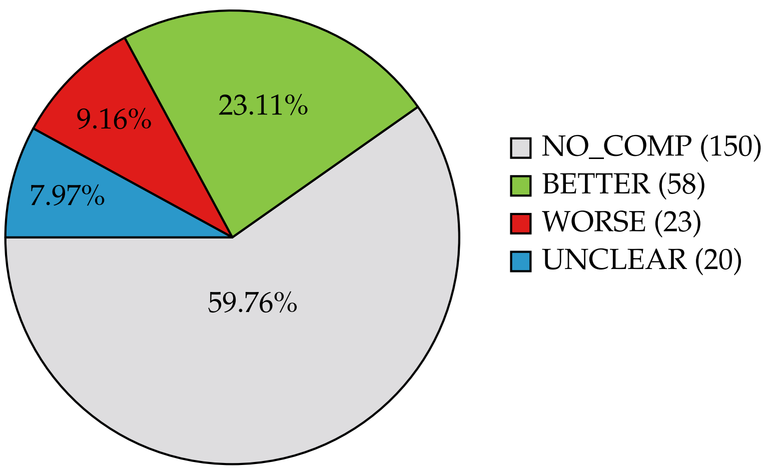
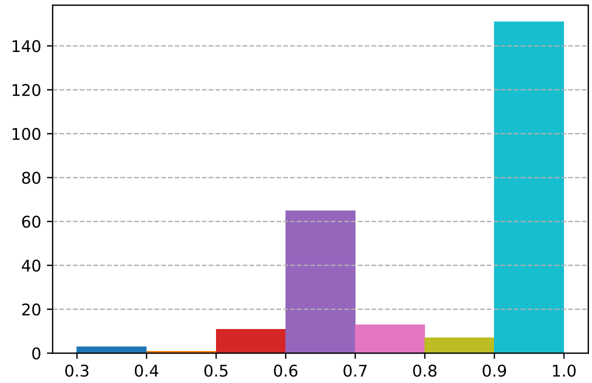


Figure 2.2: Confidence histogram



Crowdflower has a trust value for each annotator. This trust value and the number of votes per class gives a value of confidence for each label.[1]

As presented in figure 2.2, a majority (151) of the labelings has a confidence greater or equal to 0.9, and 15 sentences a confidence below 0.6; the mean is 0.86. Detailed numbers on the confidence are shown in table 2.3

Figure 2.3: Confidence

Type	Value
Average Confidence	0.86
Standard Derivation	0.17
Lowest Confidence	0.35
Highest Confidence	1.00
25th percentile average	0.67
50th percentile average	1.00

The most difficult sentence is with a confidence of 0.35 for the class *WORSE* was

> Google shouldn't have mandated an inferior map app on the iphone:[OBJECT_A] (as opposed to android:[OBJECT_B]).

It was labelled as *BETTER* (trust: 0.72), *WORSE* (trust: 0.85) and *NO_COMP* (trust: 0.82). The class *WRONG* is correct here, as the object "iphone" is inferior to "android" on the aspect of "map app".

The following sentence was assigned to *BETTER* (0.37 confidence), although it should belong to *UNCLEAR*.

> Not to mention that the iphone:[OBJECT_A] and android:[OBJECT_B] phones deliver a far superior user experience overall

[1] How the confidence is calculated in detail can be found at https://success.crowdflower.com/hc/en-us/articles/201855939-How-to-Calculate-a-Confidence-Score (Last checked: 19.12.2017)

However, the annotator for *UNCLEAR* only had 0.87 trust, while the one for *BETTER* had 1 (third one was *NO_COMP* with 0.82 trust).

All things considered, the result of the prestudy is satisfactory. The annotators agreed in the majority of decisions.

2.3 Main Study

2.3.1 Task Description

2.3.2 Data Generation

Three domains were fixed for the sentences of the main study. The domains were chosen in a way that a majority of people can decide whether a sentence contains a comparison or not.

The most specific domain was "Computer Science Concepts". It contains objects like programming languages, database products and technology standards such as Bluetooth and Ethernet. Many computer science concepts can be compared objectively, for instance, one can compare Bluetooth and Ethernet on their transmission speed. Some basic knowledge of computer science was needed to label sentences correctly. For example, to compare Eclipse and NetBeans, the annotator must know what an Integrated Development Environment (IDE) is and that both objects are Java IDEs. The need of the knowledge was communicated to the prospective annotators. The objects for this domain were manually extracted from "List of ..." articles from Wikipedia.

The second, broader domain was "Brands". It contains objects from of different types (car brands, electronics brands, and food). As brands are present in everyday life of people, it is expected that anyone can label the majority of sentences containing well known brands such as Coca-Cola or Mercedes. As with computer science, the objects for this domain were extracted from "List of ..." articles from Wikipedia.

The last domain is not restricted to any topic. For each one of 25 randomly selected seed words, ten similar words were extracted using JoBimText, a software package for distributional semantics. The seed words were created using https://randomlists.com[2]. Listing 2.3 shows the result[3] for the seed word *harvard*.

Listing 2.3: Similar words to "Harvard"

```
{
   "results":[
      { "score":688.0, "key":"harvard#NP" },
      { "score":245.0, "key":"yale#NP" },
      { "score":163.0, "key":"princeton#NP" },
      { "score":152.0, "key":"mit#NP" },
      { "score":143.0, "key":"stanford#NP" },
      { "score":133.0, "key":"university#NP"},
      { "score":132.0, "key":"tufts#NP" },
      { "score":130.0,"key":"cornell#NP"},
      { "score":127.0, "key":"nyu#NP" },
```

[2]Last checked: 25.01.2018

[3]http://ltmaggie.informatik.uni-hamburg.de/jobimviz/ws/api/stanford/jo/similar/harvard%23NP?numberOfEntries=10fo
Last checked: 25.01.2018; Some uninteresting fields were removed for brevity

```
        { "score":113.0, "key":"university#NN" }
    ]
}
```

This method covers a wide are of possible comparison patterns.

Especially for brands and computer science, the object lists are long (4493 brands and 1339 for computer science).The frequency of each object was checked using a frequency dictionary to reduce the number of possible pairs. All objects with a frequency of zero and ambiguous objects were removed from the list. For instance, the objects "RAID" (a hardware concept) and "Unity" (a game engine) were removed from the computer science list as they are also regularly used nouns.

The remaining objects were combined to pairs. For each type, all possible combinations were created. For brands and computer science, the type is the source list. For the unrestricted domain, the seed word was used. This procedure guarantees that only meaningful pairs are created. The ElasticSearch Index was then queried for entries containing both objects of each pair. For 90% of the queries, the marker terms where added to the query. This was done to check whether there is a chance that those two objects were compared. All pairs were the query yielded at least 100 sentences were kept. Those pairs are frequent enough and have a high chance of generating comparative sentences.

From the sentences of those pairs, 2500 for each category were randomly sampled as candidates for the crowdsourcing campaign. 250 sentences were manually labelled to check if there are enough comparative sentences. Those labels were discarded for the crowdsourcing campaign. The label distribution of the 250 sentences is presented in the figures FIGURE NUMBERS.

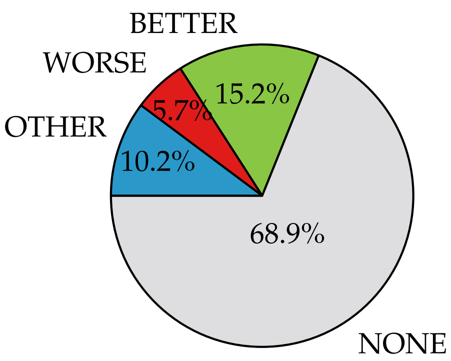


Figure 2.4: Brands

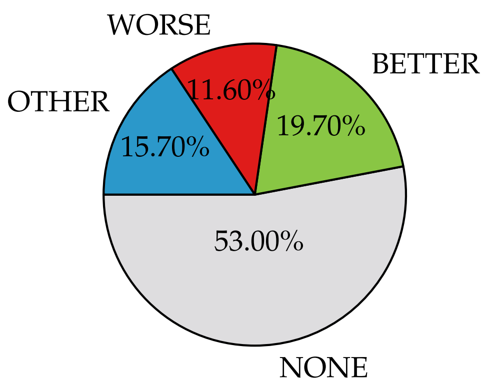


Figure 2.5: Computer Science

In all samples, at least 30% of the sentences are comparative. This number shows that the sampling method is sufficient to sample sentences for the crowdsourcing campaign.

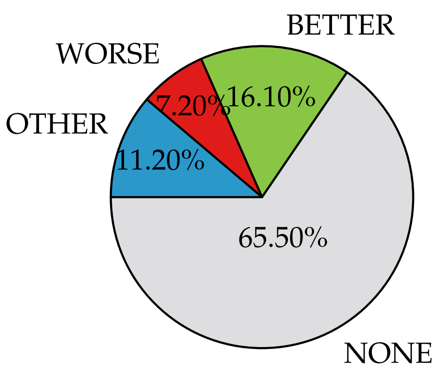


Figure 2.6: Unrestricted

3 Classification of Comparative Sentences

3.1 Description of the experiments

The goal of this thesis is to train a machine learning model in a way that it can decide whether a sentence contains a comparison between two defined objects or not.

To achieve this goal, the task was formulated in three ways of different granularity. In all tasks, the machine learning model was provided with the sentence and the objects.

In the first experiment, the model was trained to classify a sentence into one of the four categories described above. As stated below, the class OTHER adds uncertainty to the model. This problem is handled in next experiments. In the second experiment, all sentences with this class were removed prior training. In the third experiment, the class OTHER was joined with the class NONE. Thus, in those experiments, the model has to decide between three classes.

In the fourth experiment, BETTER, WORSE and OTHER are joined to the class ARG while the fifth experiment joins BETTER and WORSE to ARG and OTHER with NONE. Thus, the fourth and fifth experiments are binary classification tasks.

3.2 Evaluation

The evaluation of the results was done with stratified k-fold cross-validation where k is 3. The overall F1 score of each fold is the weighted average of the F1 scores of each class, where the weights are the number of examples per class. It was produced using the `classification_report` function of sklearn. While discussing the features, single F1 scores are presented. Those scores are the unweighted average of the F1 scores of each fold.

Following [Daxenberger et al., 2017], the results were also evaluated with training on two of the domains and evaluating on the leftover one.

3.3 Algorithms

The classification was performed with SkLearn ([Pedregosa et al., 2011]).

3.4 Features

Several features and feature combinations were tested. The results are presented in table X and Y. As described above, the F1 scores in the tables are the unweighted averages of the three F1 scores of each fold.

Every feature was tested on different parts of the sentence: the whole sentence, all words before the first object, all words after the second object and all words between objects. In doing so, the objects either stayed in the sentence, were removed or replaced. Two different replacement strategies were tested: replacing both objects with OBJECT and replacing the first object with OBJECT_A and the second object with OBJECT_B. The replacement approach should test if the objects influence the decision of the classifier. For example, if "Python" is always the "better" object the classifier might become biased.

The following section only describes features which worked out well, leaving out bad combinations (for example, using only the first or last part was not helpful in most cases).

In the first step, n-gram models where tested. Every uni-, bi and trigram in the whole training set was implemented as a binary feature. Restricting on frequency was not helpful. Also, trigrams were not helpful which can be explained by the length of the sentences.

3.5 Results

Table 3.1: Results (3-fold; Linear Support Vector Machine)

	Four classes	Three A	Three B	Binary A	Binary B
Sentence Embeddings (M)					
Unigrams (M)					

3.5.1 Four Labels

3.5.2 Three labels

3.5.3 Binary

3.6 Discussion

4 Conclusion

Bibliography

[Aker et al., 2017] Aker, A., Sliwa, A., Ma, Y., Lui, R., Borad, N., Ziyaei, S., and Ghobadi, M. (2017). What works and what does not: Classifier and feature analysis for argument mining. In *Proceedings of the 4th Workshop on Argument Mining*, pages 91–96.

[Daxenberger et al., 2017] Daxenberger, J., Eger, S., Habernal, I., Stab, C., and Gurevych, I. (2017). What is the essence of a claim? cross-domain claim identification. *CoRR*, abs/1704.07203.

[Dusmanu et al., 2017] Dusmanu, M., Cabrio, E., and Villata, S. (2017). Argument mining on twitter: Arguments, facts and sources. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2307–2312.

[Eckle-Kohler et al., 2015] Eckle-Kohler, J., Kluge, R., and Gurevych, I. (2015). On the role of discourse markers for discriminating claims and premises in argumentative discourse. In *EMNLP*, pages 2236–2242.

[Fiszman et al., 2007] Fiszman, M., Demner-Fushman, D., Lang, F. M., Goetz, P., and Rindflesch, T. C. (2007). Interpreting comparative constructions in biomedical text. In *Proceedings of the Workshop on BioNLP 2007: Biological, Translational, and Clinical Language Processing*, pages 137–144. Association for Computational Linguistics.

[Gupta et al., 2017] Gupta, S., Mahmood, A. A., Ross, K., Wu, C., and Vijay-Shanker, K. (2017). Identifying comparative structures in biomedical text. *BioNLP 2017*, pages 206–215.

[Habernal et al., 2014] Habernal, I., Eckle-Kohler, J., and Gurevych, I. (2014). Argumentation mining on the web from information seeking perspective. In *ArgNLP*.

[Kim, 2014] Kim, Y. (2014). Convolutional neural networks for sentence classification. *CoRR*, abs/1408.5882.

[Lippi and Torroni, 2016] Lippi, M. and Torroni, P. (2016). Argumentation mining: State of the art and emerging trends. *ACM Trans. Internet Technol.*, 16(2):10:1–10:25.

[Panchenko et al., 2017] Panchenko, A., Ruppert, E., Faralli, S., Ponzetto, S. P., and Biemann, C. (2017). Building a web-scale dependency-parsed corpus from commoncrawl.

[Park and Blake, 2012] Park, D. H. and Blake, C. (2012). Identifying comparative claim sentences in full-text scientific articles. In *Proceedings of the Workshop on Detecting Structure in Scholarly Discourse*, pages 1–9. Association for Computational Linguistics.

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

[Šnajder, 2017] Šnajder, J. (2017). Social media argumentation mining: The quest for deliberateness in raucousness.

[Stab and Gurevych, 2014] Stab, C. and Gurevych, I. (2014). Identifying argumentative discourse structures in persuasive essays. In *EMNLP*, pages 46–56.

Eidesstattliche Versicherung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe angefertigt und mich anderer als der im beigefügten Verzeichnis angegebenen Hilfsmittel nicht bedient habe. Alle Stellen, die wörtlich oder sinngemäSS aus Veröffentlichungen entnommen wurden, sind als solche kenntlich gemacht. Ich versichere weiterhin, dass ich die Arbeit vorher nicht in einem anderen Prüfungsverfahren eingereicht habe und die eingereichte schriftliche Fassung der auf dem elektronischen Speichermedium entspricht.
Ich bin mit einer Einstellung in den Bestand der Bibliothek des Fachbereiches einverstanden.

Hamburg, den