Sarcasm and Irony Detection

Roman Purgstaller
Graz University of Technology - Institute of Interactive Systems and Data Science

ABSTRACT

Detecting the linguistic nuances of irony and sarcasm in language is not only challenging task for machine learning algorithms, but also for humans. A variety of approaches has been proposed in this area of research. In this study we build a german dataset using ironic, sarcastic and regular tweets. We focus on the feature engineering process as well as the evaluation of different machine learning models. We achieved our best results using only unigram features. Our model distinguishes between ironic, sarcastic and regular tweets with 52% accuarcy on a balanced dataset. Our model achieved up to 70% seperating sarcastic from regular tweets. Another focus in this study is analysing the underlying feature set. Sarcastic and ironic tweets often contain positive adjectives as well as polarizing topics.

KEYWORDS

machine learning, text classification, data mining

1 INTRODUCTION

The distinction between irony and sarcasm, or even the distinction between ironic or sarcastic statements and regular statements is not only a challenge for machine learning models, but also for humans. Especially with the advance of social media platforms, it is increasingly hard to interpret specific statements.

Various previous studies have been published on this subject [8, 12], using different approaches. Most studies have been published using a dataset for the english language. We weren't able to find studies exploring german data.

We propose an approach using german tweets, to distinguish between ironic, sarcastic and other non-ironic and non-sarcastic tweets. In this paper we focus on the following areas of research:

- Mining and building a german text corpus
- Creating feature sets using a Bag-of-Words and dictionary based approach
- Creating and analysing various classification algorithms

We aim to find answers to the following questions:

- Can we compete with existing studies in this area of research?
- What are good classification algorithms for this problem?
- Is it possible to build a machine learning model, containing actual insights what users consider to be sarcastic or ironic?
- Is it possible for a classification model to identify not only single words, but phrases indicating sarcasm and irony?

The remainder of this paper is structured as follows: In section 2 we cover the theoretical background. We will then outline the created dataset in section 3 and the feature engineering process in section 4. After covering the evaluation process in sections 5, 6 and 7 we will outline new insights as well as possible enhancements to our work in section 8.

2 RELATED WORK

The distinction between sarcasm and irony is well studied in literature, linguistics or psychologie [7, 11]. For text classification, on the other hand, this area of research is considered a difficult problem.

Verbal irony is considered a rhetorical technique, in which the speakers mean the opposite of what they actually say. Sarcasm can be considered a specific form of irony. While irony is often used to lighten up the mood, sarkasm is often intented to express scorn and mockery. Some publications argue that there is a high similarity between sarcasm and irony [4, 15].

One approach for sarcasm and irony detection is to recognize the sentiment of the speaker using sentiment analysis. There have been also attempts to use a Bag-of-Words and a dictionary approach [8, 12, 13]. A dictionary approach often takes the length of a text, special characters or capitalized words into account. Other studies for sarcasm detection in social media focus on contextual features as well as taking the relations between users and the content they produce into account [2].

Our Study is particularly inspired by the work of ling et al. [12]. In their study they used a a Bag-of-Words as well as a dictionary based approach. The study revealed that word-based features have a high impact on the classification results.

3 DATASET

To build a corpus of sarcastic, ironic and non-ironic, non-sarcastic tweets we used hashtags, authors assigned to their tweets. We mined tweets which are labeled by the author to be sarcastic (# sarkastisch, # sarkasmus) and tweets that labeled as ironic statements (# ironie, # ironisch) as well as regular tweets containing neither sarcastic nor ironic hashtags.

The assumption we adopt from other studies is that the author is the best judge of whether a tweet is meant sarcastic or ironic. We performed a brief examination of the final corpus. We found scenarios where the tweets are labeled ironic or sarcastic but are not meant this way, leading to noise in the observed corpus. Examples of the mined tweets in our dataset include the following:

(1) *Nur weil ich mich mit # Twitter nicht auskenne, bedeutet noch lange nicht, dass ich # ironie nicht erkenne*
(2) *Die # Ironie dabei: Die Handwerker werden eine Heizung abmonteiren und ein Loch durch die Außenwand brechen*
(3) *vielleicht solltest # Sarkasmus dazu schreiben*

Some tweets labeled as ironic sometimes simple contain the word "ironisch" in the statement without ironic meaning. Authors sometimes also explain ironic situations, while the statement is not meant to be ironic. Nevertheless, the overall majority seem to intent to express sarcasm or irony when labeling their tweets as such.

The final corpus consists of 9000 tweets, 3000 from each category.

4 FEATURE ENGINEERING

We combined two different approaches when creating our feature set. We used a Bag-of-Words approach combined with dictionary based features.

Apart from plain text, tweets can contain URLs, references to other users as well as hashtags. Before vectorization, we preprocessed each tweet by removing all URLS as well as usernames for the Bag-of-Words approach.

Removing hashtags seems to be double-edged. On the one hand, [5] pointed out that tweets with hashtags tend to be noisy. One the other hand, features expressing sentiment (e.g. "traurig", "freude", "humor") are presumably good indicators of sarcasm and irony. Overall, we achieved slightly better results by not removing hashtags from the corpus.

We experimented with word based unigrams, bigrams and trigrams. For the dictionary approach we used a POS tagger [14] to count the number of adjectives, interjections, nouns and verbs. We also counted the number of capitalized words. Additionally, we added various binary features, checking for punctuation marks, laughing acronyms. We also checked the occurence of a list of positive and negative smileys.

5 EMPIRIVAL EVALUATION

5.1 Evaluation Metrics

In our experiments we primarily looked at accuracy and the f1 score. Since we use multiclass classification and our dataset is well balanced, we use a macro averaged f1 score.

5.2 Hyper Parameter Tuning

For hyper parameter tuning we experimented with exhaustive Grid-search as well as a randomized search for classification algorithms. For validation and to avoid overfitting we use a stratified 3-fold cross validation over the parameter grid for both methods.

The best estimator is evaluated automatically using a macro averaged f1 score.

5.3 Classification

The results from the experiments described in section 4 are used as input vectors for different classification methods.

We tested the following classification algorithms using a variation of hyper parameter settings:

- Passive aggressive classifier
- Ridge classifier
- Nearest centroid
- Linear support vector classification
- Multinomial and bernoulli naive bayes
- k-nearest-neighbor algorithm
- Random forest classifier
- SGD classifier
- Maximum entropy classifier

The linear support vector classification uses an implementation of liblinear [6].

6 IMPLEMENTATION DETAILS

For implementation we used the programming language python. Initially, we accessed official python api [1] using tweepy [16]. Unfortunately, the twitter API only allows to retrieve tweets which are not older than one week. We were therefore only able to build a corpus containing a few hundred sarcastic tweets. To resolve this issue, we used the library "GetOldTweets" [9] to build a larger corpus.

For vectorization and classification we used the scikit-learn machine learning library [3].

For part-of-speech tagging we used a pre-trained classifier for german text [14].

The graphics were created using matplotlib [10].

7 RESULTS

7.1 Vectorization

We experimented with unigrams, bigrams and trigrams, as well as weighting dictionary based features differently.

One approach was removing unigrams from the dataset completely and only use bigrams or bigrams and trigrams for classification. The principal idea was that in sentences such as "Ein Sarkasmus Detektor, das ist wirklich eine grossartige idee. Gut gemacht!" phrases like "grossartige idee", "gut gemacht" could identify sarcasm better then single tokens. However, our results for unigrams indicates that without unigrams, we lose features like "zynismus", "sympathieträger", "sarkasper" indicating sarcasm. We would also lose most interjections. Additionally, some unigrams, considered important by classifiers are just multiple words written as one. For example "isso", "echtallwissend" or "hartaberfair".

While weighting dictionary based features higher doesn't improve most classification algorithms, this method significantly improves the passive-aggressive classifier, as well as the SGD classifier.

Overall, bigrams as well as trigrams couldn't improve results over unigrams. Looking at the top features for classifiers, some contained useful phrases, such as "gefuehlte exakt" or "betroffen zeigen". The issue with most of these phrases is that they could indicate both, sarcasm as well as irony.

7.2 Classificaton Results

Initially, we tested a variation of classification algorithms (section 5.3) using different vectorization methods (section 7.1). We also included tf-idf transformation, which improved our results slightly.

However, the initial results where rather disillusioning. We achieved the best results using only unigrams as well as selecting 80% of features using χ^2 feature selection. The best results with this setting was achieved using a maximum entropy classifier (see figure 1).

Using only sarcastic and non-sarcastic tweets the accuracy is about 15 to 20 percentage points higher (see figure 2). Naive bayes classification could outperform other classifiers in this set of experiments.

7.3 Feature Analysis

Overall, features with high impact on the classification result don't indicate sarcasm or irony. Some positive adjectives like "beruehmt",

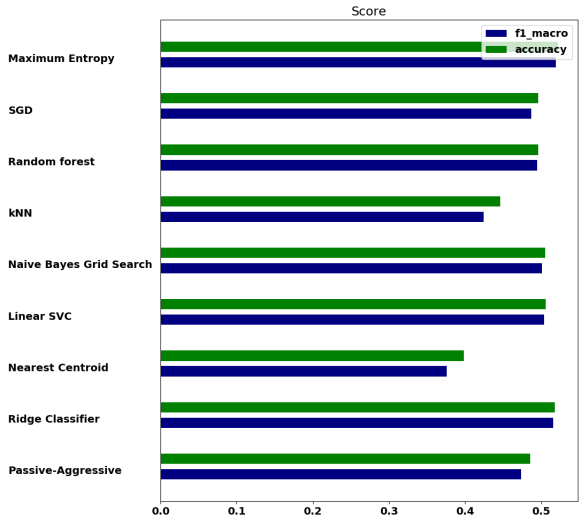

Figure 1: Classifying sarcastic, ironic regular tweets using unigrams and χ^2 feature selection

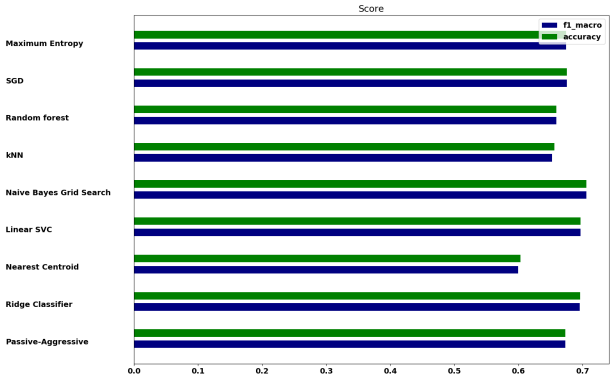

Figure 2: Classifying sarcastig and regular tweets using unigrams and χ^2 feature selection

"motiviert" or "maechtig" could be considered useful for both categories. We often found features for polarizing topics, especially in sarcastic tweets, like political parties or politicians. Even the austrian journalist Florian Klenk (the poor fellow) is not spared in ironic tweets.

Surprisingly, dictionary features didn't have much impact on the classification results. Even with different weighting schemas, we were not able to improve classification performance.

8 DISCUSSION AND FUTURE WORK

In this study we evaluated multiple machine learning models to differentiate between sarcastic, ironic and non-sarcastic, non-ironic tweets. We focused also on analysing and understanding the underlying dataset and the creation of feature sets. Unfortunately our resuts were not able to compete with other studies in this area of research. Yet, we were able to gather some insights from our experiments, which can be summarized as follows:

We are able to find relevant phrases, which might indicate sarcasm or irony using bigrams or trigrams. But most of this phrases are ambiguous. Most phrases might indicate sarcasm, irony or neither. We achieved better results using only unigrams, relying on clearer features like "zynisch".

The maximum entropy classifier outperformed other classification algorithms overall. Naive bayes classification as well as linear support vector machines and ridge classifiers seem to be well suited for this problem as well.

Overall, the created models seem to be unstable. Most features considered important by our models doesn't indicate the corresponding class. This is especially true for bigrams and trigrams. One way to solve this problem might be a manuel review of the tweets to eliminate noisy data [8].

Another useful extension to our work could be a sentiment analysis. Ling et. al. [12] used a dicitionary-based sentiment score and achieved good results.

REFERENCES

[1] [n. d.]. Twitter API. https://developer.twitter.com/en/docs.html. Accessed: 2018-06-14.

[2] Silvio Amir, Byron C Wallace, Hao Lyu, and Paula Carvalho Mário J Silva. 2016. Modelling context with user embeddings for sarcasm detection in social media. *arXiv preprint arXiv:1607.00976* (2016).

[3] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. 2013. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*. 108–122.

[4] Rebecca Clift. 1999. Irony in conversation. *Language in society* 28, 4 (1999), 523–553.

[5] Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Semi-supervised Recognition of Sarcastic Sentences in Twitter and Amazon. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning (CoNLL '10)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 107–116. http://dl.acm.org/citation.cfm?id=1870568.1870582

[6] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research* 9 (2008), 1871–1874.

[7] Raymond W Gibbs. 1986. On the psycholinguistics of sarcasm. *Journal of Experimental Psychology: General* 115, 1 (1986), 3.

[8] Roberto González-Ibánez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in Twitter: a closer look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers-Volume 2*. Association for Computational Linguistics, 581–586.

[9] Jefferson Henrique. [n. d.]. Get old Tweets. https://github.com/Jefferson-Henrique/GetOldTweets-python. Accessed: 2018-06-14.

[10] J. D. Hunter. 2007. Matplotlib: A 2D graphics environment. *Computing In Science & Engineering* 9, 3 (2007), 90–95. https://doi.org/10.1109/MCSE.2007.55

[11] Roger J Kreuz and Sam Glucksberg. 1989. How to be sarcastic: The echoic reminder theory of verbal irony. *Journal of experimental psychology: General* 118, 4 (1989), 374.

[12] Jennifer Ling and Roman Klinger. 2016. An Empirical, Quantitative Analysis of the Differences between Sarcasm and Irony. In *International Semantic Web Conference*. Springer, 203–216.

[13] DG Maynard and Mark A Greenwood. 2014. Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. In *LREC 2014 Proceedings*. ELRA.

[14] Philip Nolte. 2014. ClassifierBasedGermanTagger. https://github.com/ptnplanet/NLTK-Contributions.

[15] Tomáš Ptáček, Ivan Habernal, and Jun Hong. 2014. Sarcasm detection on czech and english twitter. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. 213–223.

[16] Pablo Rivera. [n. d.]. tweepy. http://www.tweepy.org/. Accessed: 2018-06-14.