KLUEnicorn at SemEval-2018 Task 3: A Naïve Approach to Irony Detection

Luise Dürlich
Friedrich-Alexander Universität Erlangen-Nürnberg / Germany
luise.duerlich@fau.de

Abstract

This paper describes the KLUEnicorn system submitted to the SemEval-2018 task on "Irony detection in English tweets". The proposed system uses a naïve Bayes classifier to exploit rather simple lexical, pragmatic and semantic features as well as sentiment. It further takes a closer look at different adverb categories and named entities and factors in word-embedding information.

1 Introduction

Automatic irony and sarcasm detection has made great advances in recent years, evolving from considering purely lexical information (Kreuz and Caucci, 2007) to sentiment (González-Ibáñez et al., 2011) and semantics (Ghosh et al., 2015). With new approaches that are aware of the context a tweet is produced in, promising results of as much as 87% accuracy (Silvio et al., 2016) have been achieved.

In the following sections, I present a constrained contribution to the SemEval-2018 irony detection task (Van Hee et al., 2018). As useful context for the training data was rather hard to come by, a solely tweet based approach is explored. In the next section, the dataset provided by the task organizers will be discussed. Sections 3 and 4 will elaborate on data preprocessing and the types of features that were tested. Finally, sections 5, 6 and 7 will present experiments on the usefulness of different features to different classifiers, the settings used for the submitted systems and the competition results.

2 Data

To train the system, only the official training set consisting of 3,834 tweets was used. Of these tweets 1,911 were ironic and 1,923 were non-ironic. Depending on the subtask at hand, namely binary irony detection (task A) or the differentiation between different types of irony (task B), the ironic tweets were further categorized as either *verbal irony by means of polarity contrast* (class 1), *other verbal irony* (class 2) or *situational irony* (class 3). This resulted in 1,390 examples for class 1, 316 examples for class 2 and 205 examples for class 3. The tweets still contained the original URLs, that were further analyzed to get an idea of whether or not they could provide useful context information. However, only as much as 14% of the ironic sample even contained URLs and most of these just linked to images and the original tweet on Twitter. As the data did not include the names of the authors or contained any additional context information,[1] context with respect to the authors user profile was not explored further.

3 Preprocessing

As preparation for tagging, segmentation problems – especially arising around emoji and punctuation marks – were corrected, user mentions were anonymized to "*@user*" and URLs replaced by "*http://url.com*". Hashtags were stripped of the "*#*" and segmented using a simple hand-crafted hashtag tokenizer that relies on regular expressions and a dictionary consisting of the *Unix wordlist* and terms filtered from some 190,000 tweets to account for non-standard words and spelling. The tweets were then tagged using the part-of-speech tagger provided by *Ark TweetNLP* (O'Connor et al., 2013) and filtered using regular expressions. Conjunctions, determiners, existential uses of "*there*", numerals, predeterminers, prepositions, pronouns, punctuation, URLs and user mentions were discarded. Finally, some per-

[1] The profiles of users mentioned in the tweets were not considered.

Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018), pages 607–612
New Orleans, Louisiana, June 5–6, 2018.

sisting segmentation issues related to the tagging – e.g. sequences of emoji were not segmented and sometimes assigned the wrong tag – were resolved and proper nouns identified by the tagger were replaced by "^_*NNP*".

4 Features

The features described in this section were either obtained from tagged tweets, raw tweets as string or tokenized raw tweets using the tokenizer provided by *Ark TweetNLP*. For tokenization, some adaptions had to be made to ensure correct segmentation around emoji. In general, the focus was laid on quick and easy-to-extract binary or count-based tweet properties (except for embeddings and named entities). The features used to train the model, can be assigned to the following categories:

Lexical: A bag-of-words was extracted from the tagged tweets using the *TfidfVectorizer* provided by *scikit-learn* (Pedregosa et al., 2011).

As more of a structural feature, tweet length both in terms of words and in terms of characters was exploited.

Pragmatic: The amount of punctuation, quotation marks, character repetition – and as special case ellipsis (expressed by "...") – as well as uppercase words were added to the features by simply counting their occurrence. As Twitter-specific patterns, the presence and number of user mentions, urls and hashtags as well as the number of emoji in a given tweet were noted.

Sentiment: Two sentiment lexicons were used to capture the mean positive, objective and negative sentiment associated with the hashtags, emoji and normal words present in a given tweet:

1. *AFINN* (Nielsen, 2011), a list of about 2,500 entries assigned to a scale ranging from -5 to 5, that also covers some expressions common in texting and microblogging (e.g. "*lol*").

2. *SentiWordNet* (*SWN*) (Esuli and Sebastiani, 2006), a much larger resource that provides different sentiment scores for the different meanings of a word, but restricted to more standard words.

To circumvent too complex disambiguation for the senses in *SWN*, the mean sentiment scores of the possible meanings were taken and whenever an *AFINN* entry existed, scores were reweighted in favor of the *AFINN* sentiment. The scores on emoji were obtained using the emoji aliases provided by the *Python emoji* package.[2]

Semantic: Inspired by the use of word embeddings to contrast a tweet's sarcastic reading with its non-sarcastic representation proposed by (Ghosh et al., 2015), separate models were trained on the ironic and non-ironic instances within the training set. The models were obtained using *word2vec* as provided by *gensim* and the model parameters were set to 100 dimensions and a window size of 5. Hierachical softmax was used for training. To obtain the literal and ironic representations of a tweet, the sums of ironic and non-ironic word embeddings were calculated and the embedding vectors were normalized to length 1 respectively.

Other: In an attempt to capture ironic tweets referring to specific numbers or amounts in a more simplified way than Kumar et al. (2017), who also take the deviation of a given number in the context of a unit of measurement with respect to the mean number encountered with that unit into account, information about the presence of certain number expressions was added to the feature vector.

To get a more fine grained representation of the adverbs used in tweets, a list of different adverb categories and corresponding adverbs was collected from *Wiktionary*[3]. The list contains 19 different categories that are illustrated in table 1. A possible advantage of this representation could be that location or temporal location adverbs – that might be informative in situational irony – can be distinguished from adverbs modifying verbs or adjectives, possibly more useful to spot verbal irony containing e.g. hyperbole.

To record references to entities, the named entity recongnizer provided by *Stanford CoreNLP* (Finkel et al., 2005) was used. After the submission for task A, some more features were added, namely the number of modals, negations and contrasting conjunctions or adverbs.

Weighting and Filtering: To account for less informative features, F-tests were performed on the features and only those that were among the 15% most significant were selected.

[2] https://pypi.python.org/pypi/emoji
[3] https://en.wiktionary.org/wiki/Category:English_adverbs

	Category	Example
1	act-related adverbs	*accidentally*
2	aspect adverbs	*still, yet*
3	conjunctive adverbs	*hence*
4	degree adverbs	*fairly*
5	demonstrative adverbs	*here*
6	focus adverbs	*especially*
7	interrogative adverbs	*why*
8	location adverbs	*there*
9	manner adverbs	*ironically*
10	pronomial adverbs	*therefore*
11	sentence adverbs	*apparently*
12	domain adverbs	*linguistically*
13	evaluative adverbs	*alarmingly*
14	speech-act adverbs	*honestly*
15	modal adverbs	*actually*
16	suppletive adverbs	*well*
17	duration adverbs	*always*
18	frequency adverbs	*constantly*
19	temporal location adverbs	*now*

Table 1: Adverb categories based on Wiktionary

Group	Features	# of Features
1	length in words, length in characters	2
2	character repetition, quotation marks, uppercase	4
3	presence and number of hashtags, URLs and user mentions, presence of emoji	7
4	number of negations, modals, comparison, adverbs contrasting conjunctions / adverbs, amounts / numbers and different types of adverbs and entities	41
5	embedding dimensions	200
6	sentiment	9
7	full feature set (group 1 - 6)	263
8	bag-of-words	A: 1,408 B: 1,455

Table 2: Feature group description

5 Feature Evaluation:

In order to gain insight on the usefulness of the features, a set of experiments[4] was performed, in which the features were assigned specific groups and a selection of classifiers was either trained on the group alone or on all features but those in the group. 10-fold cross-validation was performed on the training set comparing a Gaussian naïve Bayes classifier, support vector machines, a decision tree and a random forest classifier. The groups are reported in table 2.[5]

Results when training on the features without the bag-of-words, displayed in table 3, show that – with the exception of group 5 – most of the groups do not seem to make a big contribution to the rest of the feature set and their exclusion does not lead to substantial drops in performance. For group 5, a decrease in performance of as much as 10% can be observed for the random forest classifier compared to the performance on all features recorded in table 4.

Training on selected features from just one of the groups at a time shows that groups 3 and 5 are already very informative and can produce f-scores

Without group	NB	DT	SVM	RF
1	**69.03**	57.61	67.64	68.48
2	**69.03**	57.98	67.64	68.33
3	68.83	57.87	**69.81**	68.04
4	**68.99**	57.87	67.58	67.50
5	**65.59**	56.50	62.65	60.22
6	**69.01**	58.10	67.65	67.74

Table 3: F1-score when omitting one group at a time for binary irony detection

[4]Note that these experiments only focussed on task A.

[5]The bag-of-words was restricted to uni- and bigrams with a minimum document frequency of 5.

of 67.42% and 69.76% respectively. As we can see in table 4, the best score is still obtained when selecting from the entire feature set and training a random forest.

Taking a look at the importance weights assigned by the random forest classifier, it emerges that the embeddings range among the top 220 ranks and carry 91% of the importance weight. They are thus quite important for classification. Tweet length in characters is identified as the most important feature followed by positive word sentiment scores, which might indicate that the assumption by Clark and Gerrig (1984), that ironic utterances are more likely to convey negative sentiment through literally positive one, also holds for the observed tweets. Regarding adverb categories, demonstrative adverbs appear to be most informative.[6] Generally, it can be noted that every group contributes to the top 250 important features with at least one or two features.

Group	NB	DT	SVM	RF
1	48.88	51.12	50.99	**53.20**
2	50.12	47.07	**51.67**	48.66
3	67.42	67.42	67.42	67.42
4	**51.08**	41.36	43.67	42.14
5	68.76	58.22	**69.76**	67.31
6	**65.57**	53.15	47.25	54.21
7	**69.05**	56.77	67.25	68.18
8	**56.77**	51.77	53.60	54.76
all	68.61	60.52	68.18	**70.06**

Table 4: F1-score when training on one group at a time in binary irony detection

6 Submission Settings

For the submission to task A, the parameters for the *TfidfVectorizer* were set to uni-, bi- and trigrams and a minimum document frequency of 2. The feature vectors did not account for modals, negations and contrasting conjunctions or adverbs since these were added to the feature set after the submission deadline for task A. As the two classes were balanced in training and test data, the priors of the Gaussian naïve Bayes classifier were set to 0.5 each.

[6]On the full feature set, location, degree, interrogative, conjunctive, temporal location and focus adverbs rank among the top 50 most important features when ignoring word embedding dimensions.

For task B, the bag-of-words was based on unigrams only with a minimum document frequency of 4. Binary features reporting the presence of hashtags, URLs and user mentions were not included. To distinguish different types of irony as well as non-irony, a two-step classification approach was adopted, first deciding whether a tweet was ironic and then labelling it as either situational or verbal irony with or without polarity change. No priors were defined for the second Gaussian naïve Bayes classifier.

7 Results

With respect to the competition results, the system did not perform very well, getting to rank 27 for task A and 23 for task B. Results compared to a benchmark system and random forest with the best settings are reported in table 5 for task A and in table 6 for task B.[7] Note that *NB* in table 6 refers to a single Gaussian naïve Bayes classifier trained on the same features as KLUEnicorn*.

The results for task A indicate that the model cannot compete with the benchmark system provided by the task organizers (a linear SVM trained on bag-of-words only). Possible reasons for that might be the restrictions imposed during preprocessing and feature extraction – a minimum document frequency of 5 might not be feasible on such a small amount of tweets and summarizing all the mentioned user names under the same token instead of at least keeping the more frequent ones as well as discarding certain parts-of-speech such as personal pronouns for example, might not be beneficial to the model. The quality of the word embeddings, trained on a relatively small amount of data, represents another issue.

System	Accuracy	Precision	Recall	F1-score
Benchmark	63.52	53.25	65.92	**58.91**
KLUEnicorn	59.44	49.14	64.31	55.71
KLUEnicorn*	65.56	57.20	52.41	54.69
RF	60.84	59.54	59.80	54.79

Table 5: Results on test data – Task A

For task B, table 6 suggests, that the submitted system still performs worse than the benchmark, yet the version taking all features into ac-

[7]KLUEnicorn* refers to a version of the system trained on a selection of the 15% most significant out of all features including the bag-of-words.

System	Accuracy	Precision	Recall	F1-score
Benchmark	56.89	41.64	36.35	34.08
KLUEnicorn	34.69	32.14	35.39	29.82
KLUEnicorn*	49.11	42.51	48.62	**40.42**
NB	47.96	31.54	35.60	30.84
RF	51.53	40.67	33.40	30.11

Table 6: Results on test data – Task B

count shows a better performance, outperforming the benchmark by 6% in terms of f-score.

Looking at the predictions in particular, we can observe that the negative class is predicted with a rather high precision (71.55%) for task A, while in task B, non-irony is detected with a high recall of almost 80%. Apparently, the model is best at predicting non-irony. In task B, the model struggles most when predicting situational irony, achieving an f1-score of only 11%. This is not very surprising, given the small amount of examples for class 3 in the training data.

Tables 7 and 8 show examples from the test set for task A and B and the corresponding predictions made by the classifier. As we can see, short messages lacking more informative context such as the second example in table 7 or the first example in table 8 are still an issue, whereas tweets containing hashtags that oppose the initial content of the tweet text such as the third example in table 8 can correctly be assigned class 1. With "#not" not being part of the training data, this is more difficult for tweets like the fourth tweet, where only one hashtag is present.

Gold label	Pred.	Tweet
0	0	NOT GONNA WIN http://t.co/Mc9ebqjAqj
0	1	@mickymantell He is exactly that sort of person. Weirdo!
1	1	Just walked in to #Starbucks and asked for a ””tall blonde”” Hahahaha #irony
1	0	@LadySandersfarm: Garner protesters chant 'F*ck Fox News' despite Fox agreeing with them http://t.co/GWIS4hZAI6 #EricGarner #Irony

Table 7: Example predictions KLUEnicorn – Task A

8 Conclusion

In this paper, I described a rather simple system for irony detection based on target tweets only, considering various kinds of features from semantic information to different adverb categories. While all feature groups seem to contribute to performance, the embedded tweets were found to be most informative and to bring a performance gain of 3-10% depending on the classifier. However, the presented system does not do a very good job at detecting irony on the given data set. Both naïve Bayes and random forest cannot compete with the simple baseline when it comes to just identifying irony, but when different types of irony are to be distinguished, a two-step model trained on a selection of all features can outperform the benchmark. For better prediction, more reliable embeddings using more training data should be trained and certain filter settings for preprocessing should be revisited.

Gold label	Pred.	Tweet
0	0	@ChainAttackJay No sugar during christmas time? :(
0	1	Woke Up , showered , made a lunch and got ready for work only to realize that I have the whole weekend off. 😊
1	1	Well got the truck buried today perfect way to start a rainy Wednesday work day off #not #annoyed #pissed
1	2	Loooovvveeeeeee when my phone gets wiped -.- #not
2	2	Just walked in to #Starbucks and asked for a ””tall blonde”” Hahahaha #irony
2	3	and as much as I want to connect .. I like only the people who dont want to .. #Irony #Why oh why?
3	2	People complain about my backround pic and all I feel is like ””hey don't blame me, Albert E might have spoken those words”” #sarcasm #life
3	3	If you wanna look like a badass, have drama on social media #not

Table 8: Example predictions KLUEnicorn* – Task B

References

H. Clark and R. J. Gerrig. 1984. On the Pretense Theory of Irony. *Journal of Experimental Psychology: General*, 1:121–126.

A. Esuli and F. Sebastiani. 2006. SENTIWORDNET: A Publicly Available Lexical Resource for Opinion Mining. In *In Proceedings of the 5th Conference on Language Resources and Evaluation (LREC06*, pages 417–422.

J. R. Finkel, T. Grenager, and C. Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proceed-*

ings of the 43nd Annual Meeting of the Association for Computational Linguistics, pages 363–370.

D. Ghosh, W. Guo, and S. Muresan. 2015. Sarcastic or Not: Word Embeddings to Predict the Literal or Sarcastic Meaning of Words. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1003–1012. Association for Computational Linguistics.

R. González-Ibáñez, S. Muresan, and N. 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, pages 581–586.

R. Kreuz and G. Caucci. 2007. Lexical Influences on the Perception of Sarcasm. *Proceedings of the Workshop on Computational Approaches to Figurative Language*, pages 2–4.

L. Kumar, A. Somani, and P. Bhattacharyya. 2017. "Having 2 hours to write a paper is fun!": Detecting Sarcasm in Numerical Portions of Text. *CoRR*.

F. Å. Nielsen. 2011. *AFINN*. Informatics and Mathematical Modelling, Technical University of Denmark.

B. O'Connor, C. Dyer, K. Gimpel, N. Schneider, and N. A. Smith. 2013. Improved Part-of-Speech Tagging for Online Conversational Text with Word Clusters. In *Proceedings of NAACL*.

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

A. Silvio, B. C. Wallace, H. Lyu, P. Carvalho, and M. J. Silva. 2016. Modelling Context with User Embeddings for Sarcasm Detection in Social Media. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning (CoNLL*, pages 167–177, Berlin, Germany.

C. Van Hee, E. Lefever, and V. Hoste. 2018. SemEval-2018 Task 3: Irony Detection in English Tweets. In *Proceedings of the 12th International Workshop on Semantic Evaluation*, SemEval-2018, New Orleans, LA, USA. Association for Computational Linguistics.