Deliverable 4

Connie Chen, Mickey Shi, Nathaniel Imel, Sadaf Khan
{conchen, nimel, bigmigs, sadafkha}@uw.edu

https://github.com/mickeyshi/573-affect-research-group

Abstract

Sarcasm detection is an affect-related natural language processing task of particular relevance for processing textual data from social media platforms. In this paper, we report our results on one sarcasm detection task from iSarcasmEval 2022.

In previous papers, we attempted to classify sarcastic and non-sarcastic tweets. In this paper, we adapt the model designed for this past task to a new task: determining which tweet in a given pair is sarcastic, and which is a non-sarcastic rephrase.

For D4, we adapt the best models from D3 to work on within-pair sarcastic classification of tweets, where pairs are a sarcastic tweet and a non-sarcastic rephrase. With the inclusion of further emotion vectors and more preprocessing, we attempted to supplement the D3 model, but additional features did not result in improved performance for the primary task. We find that the model is more accurate on the adaptation task than the primary task, though we note several issues in the evaluation of both tasks that prevent us from drawing strong conclusions.

1 Introduction

Sarcasm can be difficult to sense among speakers, even when in-person interaction provides a bevy of accompanying prosodic, facial, and otherwise contextual cues. Predictably, automating the detection of sarcasm in decontextualized text is a complicated and demanding task. SemEval 2022 includes a set of sarcasm detection challenges, under the title iSarcasmEval. The authors of this paper have drawn from iSarcasmEval's task set, with a primary task of binary classification, where tweets are classified as sarcastic or not, and an adaptation task of binary classification, where pairs of tweets will be assigned a sarcastic and non-sarcastic label. For D2, we examined whether Random Forest and Support Vector Machines trained on DeepMoji vectors perform competitively on this primary task, and found that a linear SVM performed the best. For D3, we preprocessed data further, added polarity features to tweet embeddings, and tried more models. A Voting ensemble composed of AdaBoost, k-nearest neighbors, and MLP classifiers performed the best, outperforming the linear SVM from D2.

2 Task Description

2.1 Primary and Adaptation Tasks

The primary task of a given model in this paper is to determine whether a given tweet is sarcastic or not. The evaluation metric for this task is F1 score for the sarcastic class. An adaptation task, provided by the iSarcasmEval shared subtask C, is to discriminate a sarcastic tweet from a counterpart that is a non-sarcastic rephrase; that is, given two tweets about the same event, where one is sarcastic, and the other is not, determine which tweet is sarcastic. Accuracy is used as an evaluation metric for the adaptation task in keeping with the metrics of the task authors.

2.2 Datasets

The training dataset is composed of English language tweets, which are marked for being 'sarcastic' or not. 'Sarcastic' tweets are further annotated for the 'type' of sarcasm they are (*rephrase, sarcasm, irony, satire, understatement, overstatement, rhetorical question*) in addition to author-generated non-sarcastic 'translations' of the tweet. For example, if

a sarcastic tweet reads *The only thing I got from college is a caffeine addiction*, the author translation is *College is really difficult, expensive, tiring, and I often question if a degree is worth the stress.* The training dataset has 3466 tweets, 865 of which are marked as 'sarcastic.' There are two testing datasets that correspond to the different tasks (primary and adaptation). The primary dataset has 1400 tweets, 200 of which are sarcastic. The adaptation dataset has 867 sarcastic tweets (all the positive examples of the primary dataset) and rephrases of said tweets.

2.3 Resources

Bamman and Smith (2015) provides a methodology for detection of sarcasm within conversational contexts. Joshi et al. (2017) aggregates generally used approaches for sarcasm detection used across a number of studies. These approaches were examined and applied towards D3 and D4.

The description for the shared task can be found at the SemEval 2022 Google site[1], while the GitHub repository hosting the training and test data can be found at Abu Farha et al. (2022a) [2]

Abu Farha et al. (2022b) describes the task, but an affiliated paper has yet to be written. It should eventually be published upon completion of submissions.

3 System Overview

3.1 Modeling

For D2, we focused on exploring a handful of statistical ML classifiers as candidates for our final system model. For D3, we explored additional classifiers, preprocessed our data, and evaluated one approach to feature engineering by appending polarity features. For D4, we implemented our adaptation task, experimented with feature engineering by appending NRClex (Bailey, 2019) emotion features, and further preprocessed our data. Figure 1 illustrates our modeling of the adaptation task using one of our chosen classifiers.

[1] https://sites.google.com/view/semeval2022-isarcasmeval

[2] https://github.com/iabufarha/iSarcasmEval

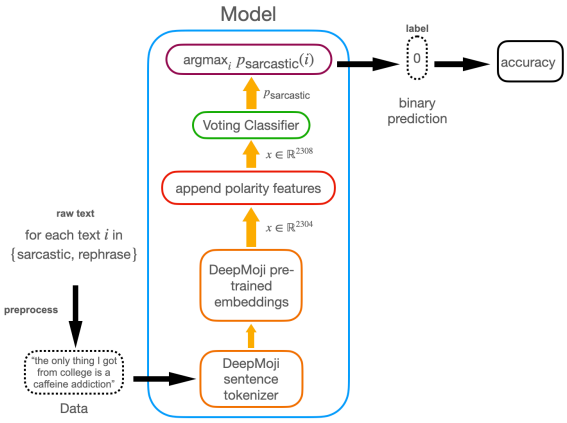

Figure 1: Model architecture for the adaptation task using the Voting classifier.

3.2 Structure of the codebase

The system for D4 targets both the primary task and adaptation task.

The codebase is designed to run the same binary classification (of tweets as sarcastic or not) from D3 on multiple possible models and compare their results all at once. In addition to the primary task, our codebase now also runs sarcasm detection on pairs of sarcastic tweets and their non-sarcastic rephrases as a discriminatory task. We treat the specification of hyperparameters for one model as equivalent to specifying different models. Hyperparameters for models are encoded in YAML files that provide on-the-fly configuration capabilities across test runs.

The main project script `adaptation_task.sh` takes a single argument: the path to a setup file where the paths for data, particular models (and their configs), outputs and results are specified. Each model config file must specify a string name identical to the name of a class implemented in `model.py`. This main script runs both the primary and adaptation tasks.

4 Approach

In D4, we focused on the top 3 performing models from D3: Voting with polarity, KNN with polarity, and AdaBoost with polarity. With these models, we implemented NRCLex emotion vectors along with our previous DeepMoji vectors as well as additional preprocessing changes.

Two preprocessing changes were made: first,

links were substituted with `<link>` tokens, rather than removing them from the dataset entirely. This was to represent the presence of images within the dataset - we hypothesized that an image could contain a meme or joke to contrast against the meaning of the phrase, possibly making a sarcastic comment more clear. Unfortunately, many of the actual images that the tweets linked to have been removed from Twitter, or are at least inaccessible from the link, preventing us from using image recognition on the image as an additional feature vector. Additionally, ampersands were encoded as `;&amp` tokens - we instead replaced all instances of this token with "and" in preprocessing.

Additionally, we attempted further feature engineering by implementing NRCLex emotion vectors that measures emotion affects such as fear, joy, negativity, positivity, and anger.[3] We approached this implementation with the assumption that perhaps lack of a certain emotion like anger or having a generally dispassionate tone could be a signifier of sarcasm.

We train on our primary task and evaluate the results from our adaptation task. In order to classify each sarcastic tweet and non-sarcastic rephrase in the adaptation task, we use sklearn's `predict_proba()` function to obtain the model's probabilistic confidence scores over labels, where the text with the highest probability of being sarcastic in each pair would be classified as sarcastic.

Adding NRCLex vectors slightly decreased our Voting and KNN model performances on the primary task and the adaptation task. Ultimately, we decided to remove the NRCLex vectors because of their negative impact on our results.

4.1 Models

The base models we report here include:

- **Voting with polarity**. This is an ensemble classifier of one AdaBoost, MLP, and KNN classifier each with polarity features.
- **KNN with polarity**. This is a k-Nearest Neighbors classifier with $k = 5$ and polarity features.
- **AdaBoost with polarity**. This is an AdaBoost ensemble classifier that fits at most 50 DecisionTree classifiers and polarity features.

5 Results

Additionally, to compare against the shared task's leaderboards, we also generated the accuracy values for the adaptation task. Our best model, Voting + Polarity, outperforms the leading result.[4] However, there is reason to be cautious about the robustness of these evaluation results for reasons about to be discussed.

6 Qualitative Analysis

6.1 NRCLex features

The addition of the NRCLex vectors to the DeepMoji+VaderSentiment model from D3 resulted in decreased performance across all categories. We speculate this is because the DeepMoji already provided a fine-grained emotion vector with more precision than the seven categories provided by the NRCLex tokenization process.

6.2 Adaptation task evaluation

6.2.1 Nature of the task

The model appeared to perform much better on the adaptation task than the primary task, when comparing F1 scores. This may seem initially surprising, as adaptation task performances typically drop in comparison to a primary task for which a model is specifically designed. However, in our case, the greater performance on the adaptation task can be partially explained by the differing stipulations of the primary and adaptation task. In the primary task, the classifier performs a binary classification independently on each provided tweet. In the end, the F1 score is calculated across these many independent classification events. With regards to the adaptation task, however, pairs of tweet are classified at the

[3] http://sentiment.nrc.ca/lexicons-for-research/

[4] First place user 'emma' reports an accuracy of 0.8700 and an F-score of 0.8694; we report an accuracy of 0.9066 and an F1 of 0.9066. See 'Task C - English' at https://codalab.lisn.upsaclay.fr/competitions/1340#results.

	Voting w/ polarity	KNN w/ polarity	AdaBoost w/ polarity
Removing NRClex	0.906	0.697	0.723
Adding NRClex	0.902	0.691	0.723

Table 1: Comparison of performance prior after adding and removing NRCLex

Classifier	Primary Dev	Primary Test	Adaptation Test
PredictRandom	0.4608	0.4152	0.4636
Voting + Polarity	0.8333	0.6061	0.9066
KNN + Polarity	0.7077	0.6066	0.6971
AdaBoost + Polarity	0.6721	0.5667	0.7232

Table 2: F1 score across classifiers for primary task

same time, based on which of the two tweets has the larger confidence score. If these tweets were being individually classified, it could just so be that both tweets would be typically labelled sarcastic, or neither; however, by using the probabilistic measure and forcing one tweet to be labelled sarcastic at a time, in addition to the knowledge that at least one of the tweets in the pair is, indeed, sarcastic, the task is greatly simplified.

Another way to think of this is to consider the performance of a random classifier on such a task. If a classifier randomly assigned the 'sarcastic' label per pair, it would result in an even distribution of true positives, true negatives, false positives, and false negatives, achieving an F1 score of 0.5. This is already greater than some of our initial F1 scores for the primary task! Comparatively, when assigning a probabilistic confidence score from a model that is reasonably well-trained on detecting sarcasm, we see accuracy and recall skyrocket as the score doesn't need to be as high as it would be in the primary task to force a 'sarcastic' labelling; it just needs to be higher than the other score. The gentle nudge in directing the sarcasm classifier is less stringent that the model's threshold for classifying a tweet as 'sarcastic' in the primary task.

6.2.2 Problematic data

The relative sentence length of the words were also taken into account as seen in Table 4. We found that, in general, sentence length for the rephrases was significantly lower than the original sentences, and we suspect that the classifier may, in actuality, be training on non-sarcasm related details such as **string length** and other markers of **sentence complexity**. A counterargument to this might be sarcastic constructions inherently tend toward complex phrases, in order to better employ contrast between successive phrases or use lengthy imagery, and that selecting against complexity is thus justifiable.

The semantic link between the sarcastic tweets and the non-sarcastic rephrases was somewhat tenuous. Ideally, the rephrase and the original tweet should have the same semantic meaning, and this generally the case. However, certain examples in the dataset stood out:

- (sarcastic tweet) 'Replace Pelosi #Nancy'
- (rephrase) 'He should not have been elected president and donald trump won the presidency.'

Additionally, certain rephrases simply had no relevance to the original tweet:

- (sarcastic tweet) 'if you see me crying in the self-service car wash in my rosati's uniform, no you didn't'
- (rephrase) 'No.' *[Author's note: This is the full tweet.]*

Additionally, as with the primary task, detecting sarcasm using context appears incredibly difficult without some sort of world knowledge. For example, the following pair:

Classifier	Adaptation Test
PredictRandom	0.4638
Voting + Polarity	0.9066
KNN + Polarity	0.7001
AdaBoost + Polarity	0.7232

Table 3: Accuracy score across classifiers for adaptation task

- (sarcastic tweet) 'Always glad to live in Kent.'
- (rephrase) 'It's absolutely going to be awful living In Kent after Brexit considering the issues with Lorries.'

Improvements to the model could incorporate some kind of dictionary/POS modeling for neologisms, such as place names, company names, and celebrities.

7 Discussion

Submissions to the iSarcasmEval task have been closed for several months, and the F1 scores of the other participants' models on the same subtask have been made public on the task's CodaLab.[5]. While our final test F1 score of 0.6066 from our polarity-enhanced KNN is not incredibly successful purely in terms of F1, the highest F1 score achieved on the challenge was 0.6052, followed by 0.5691, and 0.5295, from a total of 43 submissions. Our acquired F1 score ranks higher than any F1 in this list. These results are competitive, and indicate primarily that DeepMoji's emotion embeddings, run through ensemble approaches, have decent, if not human-level performance on tweets.

Additionally, for the adaptation task, the accuracy of 0.90657 on the test set was better than any submitted task. As mentioned before, however, we may be training on non-sarcasm related factors, such as sentence complexity of the rephrase when compared to the original tweet, and there were fewer submissions for the adaptation task when compared to the primary task.

This was despite our approach being somewhat hamstrung by DeepMoji's datedness when compared to other affect models. We found that we were unable to incorporate BERTweet and RoBERTa, as intended, and certain other emotion features such as Emo2Vec, because DeepMoji's python compatibilities were irreconcilable with these new packages.

Some problems we ran into were long classification time - retraining the models took time, especially on the 2000-length vector for the ensemble methods. We solved this by caching the vectors, so that successive runs of classifiers would not have to repeat the same preprocessing/sentence encoding pipeline. Further optimizations to this process could include reusing base classifier models across ensemble methods, instead of repeatedly training AdaBoost for its own classifier and the voting ensemble, for instance.

8 Conclusion

We built a working, end-to-end emotion-based classifier to ascertain sarcasm in text, using the BiLSTM-based framework DeepMoji as a basis and building upon it with ensemble classifiers.

Future validation of this model could include a more extensive test set, and better examination of the intent of the task. While our model ostensibly performs well on the annotator-submitted judgments for sarcasm, we're uncertain whether its judgments of sarcasm really align with a robust sample of speaker intuitions.

Sarcasm is a discourse-level phenomena. Its detection is often predicated on a shared point of reference, including knowledge about prevailing beliefs, political attitudes, historical knowledge, or even interpersonal dynamics. Humans themselves notably struggle with detecting sarcasm on social media, particularly when content is shared inter-generationally or

[5] https://codalab.lisn.upsaclay.fr/competitions/1340#results

Classifier	Correct Sarc	Correct nonsarc	Incorrect sarc	Incorrect nonsarc
PredictRandom	96.09	74.71	96.01	71.62
Voting + Polarity	96.74	73.73	89.37	66.52
KNN + Polarity	98.44	75.36	90.46	67.67
AdaBoost + Polarity	95.12	74.63	98.47	68.95

Table 4: Avg str len across D4 classifiers

across other types of 'closed environments' online. With all of this in mind, it is no surprise that automating sarcasm detection is a task with many confounding variables. It may behoove future models for sarcasm detection to attempt to capture world information as well as characterize the suggested beliefs of those who post sarcastic content. It it possible that the likelihood of sarcasm goes up with younger demographics, or among particular online affiliations. All of this would require collecting a much more robust database. However, this may be the only way to capture and manage all the subjectivities that arise with the use of sarcasm.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022a. isarcasmeval dataset. https://github.com/iabufarha/iSarcasmEval.

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022b. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Mark M. Bailey. 2019. Nrclex.

David Bamman and Noah Smith. 2015. Contextualized sarcasm detection on twitter. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 9, pages 574–577.

Aditya Joshi, Pushpak Bhattacharyya, and Mark J Carman. 2017. Automatic sarcasm detection: A survey. *ACM Computing Surveys (CSUR)*, 50(5):1–22.