Deliverable 2

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. Annotated training tweets from iSarcasmEval were transformed into vectors by using word embeddings generated from the DeepMoji model. These vectors were trained on by multiple classifiers in order to predict the presence of sarcasm. A Linear SVM produced the best F1 score for both the training and test set, of the classifiers evaluated. The resultant F1 score for the training set (0.7754) was markedly improved in comparison to an "always predict false" baseline (0.4286), but only minorly improved for the test set, which scored 0.5371 in comparison to the baseline of 0.4615.

1 Introduction

Sarcasm can be difficult to sense among speakers, even if in-person interaction provides a bevy of accompanying prosodic, facial, and otherwise contextual cues. Predictably, automating the detection of sarcasm in decontextualized text is complicated and demanding. 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 explore whether Random Forest and Support Vector Machines trained on DeepMoji vectors perform competitively on this primary task.

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 will be the F1 score for the sarcastic class. Later, an adaptation task will be 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. A simple accuracy score will be used as an evaluation metric for the adaptation task. Both metrics are in line with the iSarcasmEval metrics.

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 200 tweets (all sarcastic), and 200 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 will be examined and applied for D3.

The description for the shared task can be found

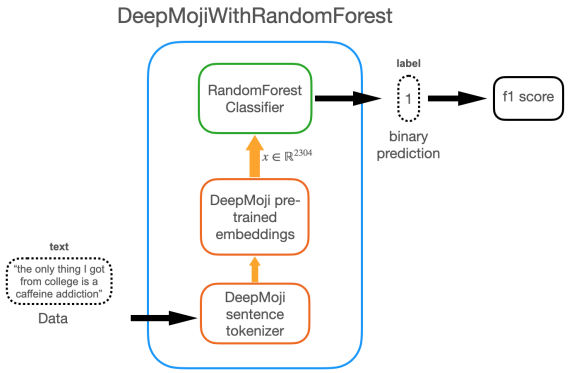

Figure 1: Model architecture for DeepMojiWithRandomForest as a part of system pipeline.

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

Our system is designed to explore a variety of models, which can in principle be diverse in approach (e.g. rule-based, statistical, neural, ensembles). We primarily focus on adapting existing resources into a minimalist architecture for D2. This involves finding word embeddings that will likely be useful to sarcasm-detection, and training a simple classifier on the resulting representations of the primary task data. Figure 1 schematically illustrates this process for one of our models.

3.2 Structure of the codebase

The system for D2 targets the primary task only.

The codebase is designed to run the same binary classification (of tweets as sarcastic or not) on multiple possible models and compare their results all at once. 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 `primary_task.sh` takes a single argument: the path to a setup file

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

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

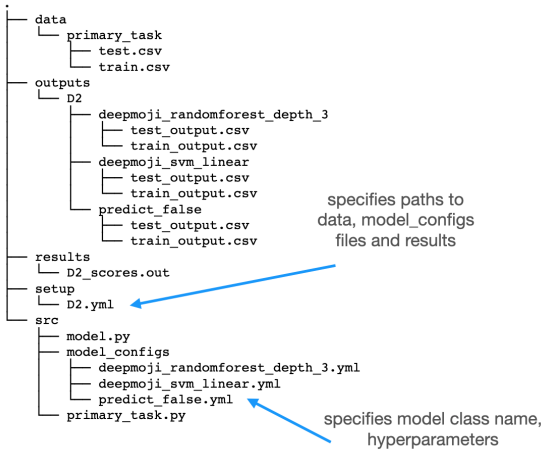

Figure 2: System architecture (folder structure) for evaluating multiple models.

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 process is illustrated in Figure 2.

4 Approach

We focus on obtaining sarcasm-relevant vector representations from DeepMoji (details in **Discussion.**), and finding a simple auxiliary classifer on which to train these representations with non-trivial performance. To provide some sense of measuring performance, we also record the performance of a model that mechanically predicts *not sarcastic* for every example. Due to the unbalanced nature of the data, this model achieves an F1-score of .4286.

The models we specify and report here include:

- **PredictFalse**. This model's `classify` method is solely `return 0`, thus classifying all input as non-sarcastic.
- **DeepMojiWithSVM**. This model uses DeepMoji to convert each tweet to a vector corresponding to its "emotional content", and then trains a scikit-learn SVM.SVC model based on vectors and provided labels. The model's classify method calls the `fit` method of the SVM. Different kernels were used to test the SVM, including the sigmoid, linear, and rbf classifiers.
- **DeepMojiWithRandomForest**. This model is similar to the above, but instead trains and

predicts with a scikit-learn RandomForestClassifier. The depth of decision trees in the classifier was altered across models.

Both the DeepMoji models are set to 'balance' their labels, to prevent the unbalanced distribution in our primary task data from biasing the classifiers.

5 Results

Results from our preliminary classifiers were summarized in Table 1. The Linear SVM performed best on the test data, with an F1-score of 0.5371. Most of the models trained were only slightly better than the PredictFalse baseline, and the non-linear SVM models performed noticeably worse.

Classifier	Train F1-score	Test F1-score
PredictFalse	0.4286	0.4615
SVM (RBF)	0.5708	0.4450
SVM (Linear)	0.7754	0.5371
SVM (Sigmoid)	0.5706	0.4445
RF (Depth 3)	0.6401	0.5086
RF (Depth 4)	0.6815	0.5179
RF (Depth 5)	0.7411	0.5116

Table 1: Summary of results across classifiers

Classifier	Train (s)	Total (s)
SVM (RBF)	111.82	111.90
SVM (Linear)	81.30	81.39
SVM (Sigmoid)	111.47	111.56
RF (Depth 3)	6.99	7.07
RF (Depth 4)	7.15	7.22
RF (Depth 5)	7.37	7.45

Table 2: Summary of runtime across classifiers

		Predicted	
		non-sarc.	sarc.
True	non-sarc.	732	468
	sarc.	63	137

Table 3: Number of sarcastic and non-sarcastic predicted vs actual examples on the test dataset for Linear SVM.

Random forest classifiers trained faster than SVM models by several factors. The Linear SVM model trained faster than its other SVM counterparts by a moderate amount as well.

While our best-performing classifier was the Linear SVM, the proportional number of false positives for this classifier was fairly significant (> than half of the true negatives), so we will focus on reduction of these false negatives in future deliverables.

6 Discussion

There are multiple approaches to automated sarcasm detection. As we were looking to detect sarcasm in tweets, we chose an approach that utilized DeepMoji, a BiLSTM-with-Attention model trained on tweets to produce word embeddings. We leveraged these word embeddings to produce vectors per tweet in the dataset. We then trained and tested various classifiers in order to determine which produced the best F1 score on both the training and test sets.

We found that the Linear SVM obtained the highest performance, which performed slightly better than the PredictFalse baseline model on the primary task. The same can be said of the Random Forest models' performance with the hyperparameters of `max_depth` 3, 4, and 5. The ultimate success of Linear SVM above other classifiers was surprising, especially against a Sigmoid SVM, as normally more sophisticated modeling is expected from the latter. Additionally perplexing is that the Linear SVM took less time to train and run than its other SVM counterparts, and yet delivered the best results.

Several unanticipated issues were run into in the course of using DeepMoji, or, more specifically, its pyTorch implementation, TorchMoji (as hosted by huggingface).[3]

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.[4]. While our test F1 score of 0.5371 from a Linear SVM is not notably successful, it appears the high-

[3]DeepMoji requires Python 2, while TorchMoji is compatible with Python 3. However, TorchMoji only operates up till Python 3.5, causing issues for researchers who did not have devices that could run Python 3.5. As Python 3.5 becomes increasingly unsupported, the ability to use TorchMoji will become impossible without further updates from the developers' ends. This is part of a larger problem in that TorchMoji (and DeepMoji) have not been updated since 2018. This is potentially the cause of another bug in the current state of the repository, wherein a particular method call is missing the appropriate number of variable assignments. This error must be manually fixed in order for our program to run.

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

est F1 score achieved on the challenge was 0.6052, followed by 0.5691, and 0.5295, from a total of 43 submissions. Therefore, our acquired F1 score ranks about third in this list, which is fairly high among the other attempts. With this in mind, our model performed competitively but still has room for improvement.

7 Conclusion

It is unclear whether the Linear SVM classifier—or our general approach of training a simple classifier using DeepMoji embeddings—can achieve substantial improvement in performance with further refinement. If we do proceed with our current approach, possible plans for improvement could include exploring other models such as Naive Bayes, AdaBoost, or k-NN as well as tweaking their respective hyperparameters.

Relatedly, the RBF and Sigmoid SVM classifiers' poor performances in comparison to the Linear SVM suggests that most of the data could be linearly separable and transforming the data with kernel tricks is detrimental to the training process. Therefore, it would be reasonable to explore the performance of other linear classifiers moving forward.

Additional preprocessing on the sentences prior to generating the sentence vectors could remove filler words, hyperlink information, and other non-affect-related data prior to affect analysis.

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.

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.