Irony detection in English tweets

Yu-Chen Lin
705315195
ericlin8545@cs.ucla.edu

Po-Chun Yang
605297984
a29831968@g.ucla.edu

Abstract

Social media has stimulated people to develop the ability to use creative and figurative language use like irony. Since the major definition of irony is a trope or figurative language use whose actual meaning differs from what is literally enunciated, the detection of irony is quite valuable for natural language processing tasks such as text mining, author profiling, detecting online harassment, and sentiment analysis. In this project, we conduct experiments on several combinations of data preprocessing techniques with four different classifiers, observe the results, and discuss them.

1 Introduction

Development of the social media has stimulated creative and figurative language use like irony. This frequent use of irony on social media has important implications for natural language processing tasks. Although different definitions of irony coexist, it is often identified as a trope or figurative language use whose actual meaning differs from its surface. As such, modeling irony has a large potential for applications in various research areas, including text mining, author profiling, detecting online harassment, and perhaps one of the most popular applications at present, sentiment analysis.

In this project, we participate in SemEval-2018 Task 3: Irony detection in English tweets (Van Hee et al., 2018). This task consists of two subtask, subtask A and subtask B, subtask A is a binary classification problem and subtask B is a multiclass classification. For subtask A, we are given several tweets and we need to identify this tweet as irony or not. For example, *"Love these cold winter mornings :grimacing face: best feeling everrrrrrr !"* is irony but *"My fingers smell like lavashak :)"* is not. On the other hand, subtask B classifies tweets into 4 different categories, including ironic by clash, situational irony, other irony, and not ironic. In this project, we chose to develop the subtask A, which is a binary classification problem.

In this project, we focus on two technical parts, the data pre-processing and the use of different classifiers. For the data pre-processing technique, we adopt the techniques and features such as TF-IDF, Word Embedding (GloVe), Sentiment Analysis, Punctuation Analysis, Part of Speech (POS), Synonym, and Frequent Words. And for the classifiers, we try four different classifiers, including Random Forest, Gaussian Naive Bayes, Support Vector Machine, and Decision Tree. In the evaluation section, we do the experiments on several combinations of the data pre-processing techniques with different classifiers. As a result, we could observe which technique or classifier can achieve better accuracy in this problem.

2 Motivation

Irony is a literary technique that the true meaning of the text is different from the surface form. For computers to understand and analyze the true meaning behind texts, it's essential to detect irony in texts. If computers can not identify irony inside the texts, they might interpret the sentnece with wrong meaning and this would lead to bad results. Thus, we want to dive into the irony detection in our final project with different techniques. From the experiments, we hope to find out better ways to conduct irony detection. In the end, we want to help computers identify irony inside our languages and understand our text better.

3 Methodology

In our project, we separate the methodoloies into two sections, which are **Data Pre-processing** and **Model-training**. For the data pre-processing part, we conduct several data analysis techniques on our raw data and tried to extract different types of fea-

tures from our data. And for the model-training part, we adopted four different classifiers and tried to find a suitable one for our task.

3.1 Data Pre-processing

For data pre-processing, we tried the following techniques to extract features from the data, TF-IDF, Word Embedding, Sentiment Analysis, Punctuations Analysis, Part of Speech (POS) Analysis, and Synonym Analysis.

3.1.1 TF-IDF

TF-IDF is short for term frequency–inverse document frequency, and it's a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. The definition of TF is the number of times a term occurs in a document, and the IDF is incorporated which diminishes the weight of terms that occur very frequently in the document set and increases the weight of terms that occur rarely. In our project, we used the *TfidfVectorizer* tool from scikit-learn to extract the TF-IDF features from our data.

3.1.2 Word Embedding (GloVe)

Word embedding is the collective name for a set of language modeling and feature learning techniques in natural language processing (NLP) where words or phrases from the vocabulary are mapped to vectors of real numbers. And GloVe is one of the word embedding techniques developed as an open-source project at Stanford University (Pennington et al., 2014). With GloVe, we can turn the tokens in our sentences into a vector and train the machine learning model with the vectors.

3.1.3 Sentiment Analysis

Since the sentiment in sentence might be a feature that can be used to identify irony, we used the *senti_synsets* tool from nltk to identify positive and negative tokens in our tweets. And we used the number of positive and negative tokens as features in our project.

3.1.4 Punctuations Analysis

As we observed, punctuation might be an important feature to identify irony, especially exclamation mark since people often want to emphasize their ironic tone with exclamation mark. So we also consider the number of punctuations especially exclamation mark in our project as features.

3.1.5 POS Analysis

A part of speech (POS) is a category of words that have similar grammatical properties. In our project, we also adopted POS structure in tweets as a feature. We count the numbers of noun, verb, adjective, and adverb into consideration in the project.

3.1.6 Synonym

A synonym is a word or phrase that means exactly or nearly the same as another word or phrase in the same language. We also consider the use of synonym in our project. We count the numbers of synonym in a single tweet and take the number as feature in the experiments.

In our project, we tried different combinations of the above techniques to find a better one that can help us train a good model to detect irony.

3.2 Model-training

To train our model, we used four different classifiers - Decision Tree, Random Forest, Gaussian Naive Bayes, and Support Vector Machine (SVM). We want to find out which classifier is the best fit for our task.

3.2.1 Decision Tree

A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility (Safavian and Landgrebe, 1991). Decision Tree has several advantages such as easy to understand and interpret, fast for inference, and automatic feature selection. Besides, it can also work with numerical and categorical features and has no assumptions about the shape of data. However, it tends to overfit. A small change in the data can cause a large change in the structure of the decision tree causing instability. And the training time is often longer.

3.2.2 Random Forest

Random forest is a supervised learning algorithm that randomly creates and merges multiple decision trees into one "forest" (Liaw and Wiener, 2002). Random forest can solve both classification and regression problems with a decent estimation at both fronts. Besides, its power of handling large-scale data with higher dimensionality is extraordinary. It can output the importance of variable, and this can be a very critical feature. Nevertheless, it can not give precise continuous nature prediction so the performance on regression problems is not so good

as on classification problem. Besides, random forest is like a black box so sometimes we can only try different parameters and random seeds to have better performance.

3.2.3 Gaussian Naive Bayes

Naive Bayes classifier applies Bayes' theorem with strong independence assumptions between the features (Rish, 2001). If the classifier has the assumption that the continuous values associated with each class are distributed according to a Gaussian, then it's a Gaussian Naive Bayes classifier. The advantages of naive Bayes classifier include the efficiency to predict class of test data set, good performance with categorical input variables. But if a categorical variable has a category not seen in training data set, the model will be unable to predict. Besides, naive Bayes classifier is a bad estimator.

3.2.4 Support Vector Machine

Support Vector Machine (SVM) is a supervised machine learning model that uses classification algorithms for two-group classification problems (Suykens et al., 1999). If there is a clear margin between groups, SVM can have really good performance. And it's also effective in high dimensional spaces and the situations that number of dimensions is greated than sample numbers. However, the training time would be terrible if we have large-scale data set. Also the noises can result in bad models.

4 Experimental Settings

In our experiments part, we deployed different combinations of data pre-processing techniques with different classifier. We have 6 different combinations. The first one only contains TF-IDF features. And then we add Word embedding (GloVe), Sentiment analysis, Punctuation analysis, POS analysis, and Synonym cumulatively. So we will have 6 differnt combinatinos. Then I will conduct the experiments of these pre-processing techniques combinations with each classifier. So we will have 6x4 = 24 kinds of results.

5 Results

Table 1 show the results of our experiments and we can see that more features sometimes don't really lead to better performance. For example, when using decision tree as the classifier, the best accuracy comes from only using TF-IDF as our features. And other combinations can not beat this result with decision tree. For the random forest, we observed that the overall accuracy is better than decision tree's results and the variance is not so obvious. Similar results happen when using Gaussian naive Bayes classifier, but the overall results are not so good as random forest. Our best result comes with Support Vector Machine, but there is a special that worth noticing, which is that the accuracy becomes significantly worse with synonym features when using SVM. We think the reason might be that the synonym features are not so fit with SVM.

6 Discussion & Conclusion

From our experiments, we think that random forest could be the best classifier for our task. Although the best result is achieved by SVM, but the overall accuracy of random forest can be better than SVM. And both random forest and SVM are better than Gaussian naive Bayes, then decision tree performs worst in our task. And for the features, we think it's worth discussing. Based on our observations, TF-IDF can performs good in some situation. And GloVe could play a positive role in this task. synonym features might have a bad effect for some models like in SVM. And other features seem not cause too much effects in the accuracy.

For this task, we have thought some future directions that could be tried. For example, we can adopted some cutting-edge techniques introduced by Prof. Chang in class. For example, BERT (Devlin et al., 2019) and Transformer (Vaswani et al., 2017) can be used in this task to analyze tweets since they can both interpret sentences bi-directionaly. And we can also tried some deep learning methods such as CNN, RNN (Mikolov et al., 2010) with more word embedding methods. For example, word2vec (Mikolov et al., 2013) is also a popular word embedding method and there are some pre-trained models can be used on the Internet.

It's pretty interesting to dive into the field of irony detection since even irony is natural to human beings, computers might not have enough ability to understand ironic sentences in our languages. If we could help computers understand irony better, they will have better ability to process natural language processing tasks for us.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of

Classifier	TF-IDF	GloVe	Sentiment	Punctuation	POS	Synonym	Accuracy
Decision Tree	v						0.597
	v	v					0.561
	v	v	v				0.552
	v	v	v	v			0.552
	v	v	v	v	v		0.562
	v	v	v	v	v	v	0.568
Random Forest	v						0.634
	v	v					0.629
	v	v	v				0.634
	v	v	v	v			0.630
	v	v	v	v	v		0.628
	v	v	v	v	v	v	0.637
Gaussian Naive Bayes	v						0.571
	v	v					0.571
	v	v	v				0.571
	v	v	v	v			0.572
	v	v	v	v	v		0.571
	v	v	v	v	v	v	0.570
Support Vector Machine	v						0.637
	v	v					0.649
	v	v	v				0.645
	v	v	v	v			0.646
	v	v	v	v	v		0.650
	v	v	v	v	v	v	0.554

Table 1: Experimental results of different combinations of data pre-processing techniques and each classifier.

deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Andy Liaw and Matthew Wiener. 2002. Classification and Regression by randomForest. *R News*, 2(3):18–22.

Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH*, pages 1045–1048. ISCA.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543.

Irina Rish. 2001. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, pages 41–46. IBM New York.

S. R. Safavian and D. Landgrebe. 1991. A survey of decision tree classifier methodology. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(3):660–674.

J.A.K. Suykens, L. Lukas, P. Van Dooren, B. De Moor, and J. Vandewalle. 1999. Least squares support vector machine classifiers: a large scale algorithm.

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018. SemEval-2018 task 3: Irony detection in English tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 39–50, New Orleans, Louisiana. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.