ALANIS at SemEval-2018 Task 3: A Feature Engineering Approach to Irony Detection in English Tweets

Kevin Swanberg‡ and **Madiha Mirza**† and **Ted Pedersen**† and **Zhenduo Wang**†*
‡Department of Writing Studies
†Department of Computer Science
University of Minnesota
Duluth, MN 55812, USA
{swanb034,mirz0022,tpederse,wang7211}@d.umn.edu

Abstract

This paper describes the ALANIS system that participated in Task 3 of SemEval-2018. We develop a system for detection of irony, as well as the detection of three types of irony: verbal polar irony, other verbal irony, and situational irony. The system uses a logistic regression model in subtask A and a voted classifier in subtask B, both of which rely on manually developed features to identify ironic tweets. ALANIS placed 34th of 43 systems in subtask A and 26th of 31 systems in subtask B.

1 Introduction

With the invention and growth of various social networking sites, irony and other creative linguistic devices have become increasingly prevalent in online content. Particularly when considering microblogging platforms like Twitter, which encourage users to share their thoughts and opinions on a wide variety of topics, the use of irony can be extremely common. This can have strong implications for various problems in natural language processing, which often have difficulty in processing this ironic content (e.g., (Liu, 2012; Ghosh and Veale, 2016; Maynard and Greenwood, 2014)), thus motivating the development of an accurate irony detection system.

While irony has many varying definitions, it is defined by the SemEval task organizers as a trope or figurative language whose actual meaning differs from what is literally enunciated. Our system, ALANIS (Automated Location and Naming of Ironic Sentences), uses a manually developed feature set and a logistic regression classifier for subtask A and a voted classifier for subtask B, achieving mean accuracies of .650 and .607 respectively on the training set and .512 and .434 respectively on the test set. The F1-scores are .469 and .276 on the test set.

2 Task Description

SemEval Task 3 involves two subtasks. In subtask A, a tweet simply must be identified as ironic or non-ironic. In subtask B, three types of ironic content must be individually differentiated from non-ironic content. These three types of irony are verbal polar irony, other verbal irony, and situational irony. The task organizers provided us with a training set of 3,834 tweets for both subtasks.

3 ALANIS

Our system, ALANIS, uses manually developed features indicative of ironic content, and passes a feature matrix for each tweet to a logistic regression classifier in subtask A and a voted classifier system which employs a logistic regression, SVM, and Random Forest classifier in Subtask B. It use the scores from each of these classifiers to "vote" on the correct label for a tweet.

3.1 Feature Selection

We explored two types of features, structural features and affective features. Structural features included sentence semantic similarity, irony-rich word lists, indicative parts of speech, and content features. Affective features included sentiment polarity and subjectivity. These features were used to assign scores for each tweet, creating a feature matrix.

3.2 Structural Features

Our system combines a number of structural features that are identified as indicative of ironic content by previous solutions.

Sentence Semantic Similarity is a measurement of similarity in meaning between two sentences. This is a structural feature employed in a

*Authors are ordered alphabetically by their first name.

Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018), pages 507–511
New Orleans, Louisiana, June 5–6, 2018.

system designed by (Farías et al., 2016) with some success. Ironic tweets with multiple sentences should show a sharp change in meaning between sentences. To implement this feature we employed WordNet synsets. The similarity of the sentences is computed based on the semantic similarity of the words contained in the two sentences.

Irony-Rich Word Lists: Our system takes advantage of a number of words claimed to be indicative of irony. This involved several manually developed word lists. Most importantly, we used discourse markers, which are phrases that are indicative of discourse segments. Examples of these include *however, on the other hand*, and *in my opinion*. These have been cited as being more common in ironic content (Farías et al., 2016). Our system employs a list of 53 discourse markers. Also based on Farias et al, we measure intensifiers, like *very* and *really* that make adjectives stronger.

In addition to these, we build on our curated lists of irony-indicative words with features like swear words and top words, as well as textual markers of laughter like *lol* and *haha*, which was noted to be common in ironic content (Buschmeier et al., 2014). Another word list included interjections to detect irony. Interjections are word that express feeling rather than meaning, for example, words like *wow*, *gosh*, and *jeez*.

Indicative Parts of Speech: We also built features that measured the prevalence of several parts of speech thought to be indicative of ironic content. These included adjectives, adverbs, prepositions, and named entities. All of these features were identified by the NLTK[1] POS tagger in order to count their occurrence. These counts were then normalized for the length of the tweet. Adjectives and adverbs occur more frequently in ironic tweets than non-ironic according to (Kreuz and Caucci, 2007). We hypothesized that prepositions and named entities would occur more often in situational irony, due to the likely need to explain the situation.

Content Features: ALANIS also employs a number of features relevant to the content of the tweet in order to identify irony. These include Word Count, Punctuation, and URLs. According to Farias et al, ironic tweets tend to have excessive punctuation to catch the eyes of readers and to stress a point. Examples include "It is really worth it!!!" or "Okay...". Thus, heavy punctuation sometimes implies irony. Farias et al. also identified that ironic tweets are likely to contain fewer words than non-ironic tweets, thus motivating the use of the word count feature.

URLs were also employed as a feature in our system. (Schifanella et al., 2016) found that ironic tweets often contain images and often the interpretation of the irony depends on the image. For example, a photo of a warm, sunny beach with the caption "Terrible weather we're having." However, the task data does not immediately give us images, only a link to images (which may not in fact exist online anymore), so the simplest way to identify this was to just check if a tweet had a URL.

Popular Hashtags and Keywords: Twitter hashtags and keywords are a good measure of public opinion on trending topics and current events. Through these hashtags, users express a wide variety of opinions, including irony. For example, the following hashtags were among the top Twitter hashtags for 2016: #GOPDebate, #PrayforJapan, #WomensRightsAreHumanRights. Our system finds hashtags that contain words related to global issues, sports, entertainment, and fashion using a manually created list of top hashtags.

3.3 Affective Features

While most work on irony detection (e.g., (Carvalho et al., 2009; Barbieri and Saggion, 2014; Vanin et al., 2013)) focus on the structural features, (Farías et al., 2016) show that introducing affective information can also improve state-of-the-art accuracy.

In our work, we included two commonly used sentiment features, polarity and subjectivity. Sentiment polarity reflects the general positivity of a piece of text, while subjectivity is measured against objectivity. Each of the two features is assigned a score within the range $[-1, 1]$. We used the TextBlob package in python [2] to implement the scoring functions for these features.

4 Classifiers

The classifiers we used for our system included Naive Bayes, logistic regression, Support Vector Machine (SVM) and Random Forest. We chose these because they are generally robust classifiers. As we added features to our feature list, we also

[1] http://www.nltk.org/

[2] http://textblob.readthedocs.io/en/dev/

kept track of the performance of the the logistic regression, Random Forest, and SVM classifiers, while Naive Bayes was only used for the bag of words baseline.

In ALANIS, all the classifiers take the tweet features matrix as input and have as output a binary label vector for the categorical result. We separate the data into a training set and a test set using cross-validation. We train the classifiers with the training set to optimize the parameters including the hyperplane and kernel. Then we evaluate the trained classifier on the test set.

Support Vector Machine: We find that SVM is relatively powerful when the feature list is short, compared with other classifiers. Also, we noticed that SVM classifier scores the highest recall, which means that it detects the most ironic tweets.

Logistic Regression Classifier: We find that logistic regression is stable in terms of total accuracy. It becomes the most accurate classifier for our long final feature list. The logistic regression classifier does not show any tendency in detecting irony or avoiding error. Because of its linear kernel, we are able to get the trained weights for each feature. This helps us know the capacities of features and select them better. We rank the importance of features according to the magnitude of their weights. See Table 3 for details.

Random Forest Classifier: This classifier does not perform well compared to the others. However, it is worth mentioning that when the feature list grows long, it retains a higher accuracy than SVM.

Voting System: We find that the confusion matrices of the classifiers are different. This means the classifiers have different specialties. Therefore, we combined them in order to get a synthesized result. We make a voting system with the classifiers discussed previously. The voting system uses majority rule.

5 Experimental Results

Our final result is that logistic regression is most accurate classifier for subtask A and the voting system is most accurate for subtask B. Table 1 shows the average cross validation scores on the training set, while Table 2 shows the scores when our system is trained on the whole training set and evaluated on the test set.

BOW+NB stands for the Bag of Words Naive Bayes baseline, while LR, RF, and Vote represent logistic regression, Random Forest, and the Voting System. Based on these results, we used logistic regression or subtask A and the voting system for subtask B.

Model	TaskA Acc	TaskB Acc
BOW+NB	0.572	0.285
LR	**0.650**	0.603
SVM	0.596	0.545
RF	0.622	0.584
Vote	0.644	**0.607**

Table 1: Performance of classifiers on training set

Model	TaskA Acc/f1	TaskB Acc/f1
LR	0.512/0.469	-
Vote	-	0.434/0.276

Table 2: Performance of classifiers on test set

The result in Table 1 are based on 5-fold cross validation with the training data. As such we expected comparable results when we applied our classifiers to the test data. However as can be seen in Table 2 this is not the case. The results for LR decline from .65 to .51, and for Vote from .607 to .434. While some variation is to be expected, this was surprising to us. We hypothesize one of two possible explanations. First, our methods may have overfit the training data and so do not generalize well to other data. However, since we employed cross validation we are not certain how likely this explanation proves to be. The second explanation may be that the test data is in some way different from the training data, to the extent that a model learnt on the training data may not fare well on the test data. We have not yet analyzed the test data closely enough to resolve this question, but consider this to be an important step in understanding our results.

6 Discussion and Future Work

We tried to interpret the importance of the features by the magnitude of their weights in logistic regression. The weights of the features' performance in subtask A and B are shown in Table 3. In interpreting this table, the further a feature is from 0, the stronger the feature's impact is on our classifier. For instance, of our features, stop words and laughters are relative weak features. Conversely, intensifiers, discourse markers, adjective/adverbs, and prepositions are much stronger features.

In order to understand the different performance

Feature	Weight	Feature	Weight
intensifier	1.05	subjectivity	0.20
discourse	0.90	named entity	0.18
adj./adv.	0.81	swear words	0.18
preposition	0.81	URLs	0.15
polarity	0.54	word count	0.12
politcal	0.39	similarity	0.11
interjections	0.34	stopwords	0.06
celebrity	0.33	laughter	0.01
punctuation	0.32		

Table 3: Weights (in absolute) of features

of the classifiers, we made confusion matrices for all the classifiers and then also did a weight analysis for logistic regression since it employs a linear kernel. From the confusion matrices, we found that although the classifiers have their specialties, the voting system does not always work out well. We believe that this is because SVM and Random Forest are both much weaker than the logistic regression classifier (shown in Table 1) so they neutralize logistic regression's advantages.

In subtask B, we need to label a tweet with 0 (non-ironic) or 1,2,3 (three different subcategories of irony). However, the difference among these subcategories are so subtle that our features do not capture them very well. Overall these classifiers have a hard time with multi-class classification. Since all three classifiers have more similar results for task B, the voted system is more successful

We review our system and the output of our system and find several possible explanations. From the confusion matrix, we can see that class 2 (7%) and 3 (5.9%) are relatively rare. This makes the task very hard for classifiers because of lack of information to train on for class 2 and 3. However, because the majority of data in subtask B falls into class 0 and class 1 we are still able to get a high accuracy (0.6+). If the data was spread more evenly between the four classes our system would likely perform better. When analyzing individual tweets from class 1, 2 and 3, we found that their feature lists are more similar to each other than to class 0 (non-ironic). This means we miss features that are relevant for identifying different types of irony, making our feature-based classifier ill suited to this task.

To see the system's effectiveness, it is often helpful to consider some indicative examples of the system in action. Consider examples (1) and (2) below. Our system successfully classifies (1) as ironic, but fails to classify (2) as ironic.

1. Feeling like crap. And being treated horribly too. It's a great day. #iwanttogohome

2. Hey there! Nice to see you Minnesota/ND Winter Weather

Our system likely successfully classifies (1) for a number of reasons. First, the word count of the sentences is low, which seems typical of ironic tweets. It has strong sentiment polarity between the sentences. The first two sentences are negative, and the last sentence is positive. There is also a strong shift in sentence similarity between sentences.

However, (2) is classified incorrectly. This is likely because it is identified by our system as similar sentiment in both sentences. There is also very little punctuation or emojis, and there are no indicative words, like discourse markers or interjections in the tweet, causing our system to fail.

These results demonstrate that a manually-selected feature-based system, using both structural and affective features can achieve reasonable success in identifying ironic content. This system is successful even when used with non-conventional language such as that seen in Twitter data. Our mean accuracy scores of .650 and .607 on the two subtasks on the training set demonstrates both a reasonable success, and an opportunity for future work in irony detection by extending the feature set further or even applying a deep learning approach to the problem when enough data is available.

7 Acknowledgments

This project was carried out as a part of CS 8761, Natural Language Processing, a graduate level class offered in Fall 2017 at the University of Minnesota, Duluth by Dr. Ted Pedersen.

References

Francesco Barbieri and Horacio Saggion. 2014. Automatic detection of irony and humour in twitter. In *ICCC*. pages 155–162.

Konstantin Buschmeier, Philipp Cimiano, and Roman Klinger. 2014. An impact analysis of features in a classification approach to irony detection in product reviews. In *Proceedings of the 5th Workshop*

on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis. Association for Computational Linguistics, Baltimore, Maryland, pages 42–49.

Paula Carvalho, Luís Sarmento, Mário J. Silva, and Eugénio de Oliveira. 2009. Clues for detecting irony in user-generated contents: Oh...!! it's "so easy" ;-). In *Proceedings of the 1st International CIKM Workshop on Topic-sentiment Analysis for Mass Opinion*. ACM, New York, NY, USA, TSA '09, pages 53–56.

Delia Irazú Hernańdez Farías, Viviana Patti, and Paolo Rosso. 2016. Irony detection in twitter: The role of affective content. *ACM Trans. Internet Technol.* 16(3):19:1–19:24.

Aniruddha Ghosh and Tony Veale. 2016. Fracking sarcasm using neural network. In *WASSA@ NAACL-HLT*. pages 161–169.

Roger J Kreuz and Gina M Caucci. 2007. Lexical influences on the perception of sarcasm. In *Proceedings of the Workshop on computational approaches to Figurative Language*. Association for Computational Linguistics, pages 1–4.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies* 5(1):1–167.

Diana Maynard and Mark A Greenwood. 2014. Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. In *LREC*. pages 4238–4243.

Rossano Schifanella, Paloma de Juan, Joel R. Tetreault, and Liangliang Cao. 2016. Detecting sarcasm in multimodal social platforms. *CoRR* abs/1608.02289.

Aline A. Vanin, Larissa A. Freitas, Renata Vieira, and Marco Bochernitsan. 2013. Some clues on irony detection in tweets. In *Proceedings of the 22Nd International Conference on World Wide Web*. ACM, New York, NY, USA, WWW '13 Companion, pages 635–636.