Is Topic the Culprit? The Influence of Topic Modeling in Irony Classification

Ivan Rep, Marino Tomaš, Zvonimir Petar Rezo

University of Zagreb, Faculty of Electrical Engineering and Computing
Unska 3, 10000 Zagreb, Croatia
{ivan.rep2, marino.tomas, zvonimir-petar.rezo}@fer.hr

Abstract

Irony detection is the task of recognizing words and sentences whose intended meanings are contrary to their literal meanings. In recent years there has been growing interest in Twitter sentiment analysis. This contributed to the demand for automated irony detection since irony can invert tweet sentiment. In this paper, we arrange tweets into topics and analyze the impact of topic modeling in irony detection. We also compare feature importance among different topics. To explore this we use logistic regression and the SemEval-2018 Task 3: Irony detection in English tweets dataset. We show that results slightly improve when including tweet topics and that feature importance differs among different topics.

1. Introduction

Irony is the use of words to express something other than and opposite of the literal meaning. It is widely used on social media platforms such as Twitter and is hard to detect because it depends on many different factors. Irony detection is important in tasks such as sentiment analysis. Sentiment analysis is often used to estimate public opinion about something or someone. The problem with irony is that it can flip sentiment, for example, consider the following tweet:

Trump was so great as president

A good indicator if this tweet is ironic would be the information about users' political preferences, but this kind of data is rarely available. In this paper, we analyze if the contextual topic of a tweet is a good predictor of irony since some topics might have a higher proportion of ironic tweets as opposed to some other topics. We use vectorized representations and word embeddings to arrange tweets into topics using unsupervised topic models.

Other important predictors of irony are features extracted from tweets. We argue that users use different textual expressions in ironic tweets such as all-uppercase words, emojis, ellipses, etc... For example, consider the following tweet

I cant wait to go to work :) ...

Here we see an example of an ironic tweet where the user expresses irony through emojis and ellipses.

In this paper, we use the logistic regression classifier to predict irony. We also extract hand-crafted features from tweets and compare feature importance among different topics. The rest of this paper is organized as follows. Section 2. presents related work on irony detection. In Section 3. we describe the SemEval-2018 dataset and give insight into the preprocessing and feature extraction procedure. Section 4. describes the topic models used and their hyperparameters. Finally, in Section 5. and Section 6. we discuss results and present conclusions.

2. Related Work

In the past few years, more attention was given to irony detection in general. Vastly different approaches have been used to tackle this problem. In Davidov et al. (2010), authors utilize a semi-supervised sarcasm identification algorithm for sarcasm detection in Amazon reviews and tweets. The main features for the model are based on punctuation and surface patterns, such as high-frequency words and content words. Nozza et al. (2016) perform topic-irony detection based on probabilistic topic models. Paper results give indications that the topic models could greatly help in irony classification. Nimala et al. (2021) use a tweet-based dataset and works on finding distributions of sentiment and sarcasm for captured topics and extracting sentiment-based sarcasm prevalent topics. Bouazizi and Ohtsuki (2015) use a combination of sentiment, punctuation, and pattern-related features along with lexical and syntactic features to achieve good results.

Based on experience from related work, we decided to use a variety of features that have already been tried out for similar problems. The distinction between our approach and the previous ones is that we take topics into account and give a further analysis of the feature importances.

3. Data

3.1. Dataset

The dataset we used in this work was taken from *SemEval-2018 Task 3: Irony detection in English tweets* competition as described in Van Hee et al. (2018). It was constructed by collecting tweets with #not, #irony, and #sarcasm hashtags. Afterward, all occurrences of these three hashtags were removed. The dataset contains raw tweets written in English with annotations signifying whether a tweet is ironic or not. The data is already split into a train and test set which consist of 3834 and 784 examples. The number of ironic and non-ironic tweets per set is displayed in Table 1. During our experiments, we use a validation set that consists of 30% of the train set examples.

The advantage of the dataset used in this paper is that the irony label was manually annotated, while in most other similar work dataset entries were labeled based on

Table 1: Dataset structure

	Ironic	Non-ironic
Train set	1911	1923
Test set	311	473

whether the entry contained an irony-related hashtag (#sarcasm #irony).

3.2. Preprocessing

The preprocessing pipeline consists of tokenization, hashtag segmentation, twitter specific symbol removal, and stopword removal. Tokenization is done using NLTK[1] tokenizer specifically designed for tweets. This also lowercases the tweets, removes the Twitter handles, and replaces repeated character sequences of length 3 or greater. Hashtags in the tokenized tweets are then replaced with their word segmented substitute using the ekphrasis[2] library. We then further remove the emojis, emoticons, upright slashes, underscores, hyperlinks, ellipses, stopwords, and words shorter than 3 characters. This is done without using libraries, with the exception of emoji removal. To achieve this we use the emoji[3] library.

Finally, we create lemmas for the preprocessed text using SpaCy[4] medium-sized pipeline for English and keep them for later use in topic modeling. If any of the tweet lemmas result in an empty list, we remove those examples from the dataset. Before applying any classification algorithm, the data is scaled using scikit-learn[5] StandardScaler.

3.3. Feature Extraction

Multiple features were extracted from tweets in order to make useful inputs for our classifier. The first features we extracted were tweet length in characters without counting whitespaces, tweet length in words, all uppercase word count, all lowercase word count, and emoji count. The first 2 features were calculated using the preprocessed text while the rest were determined using the original tweets. These features were designed as baseline features to give us basic statistics about the tweets.

Next, we looked at some Twitter-specific characteristics. Three different tweet-specific tokens were taken into account: hyperlinks, user mentions, and hashtags. Huang et al. (2018) use hashtag counts for the same task, so we decided to further extend this to other Twitter-specific characters, counting hyperlinks and user mentions as well.

As explained in Chang (2010), hashtags are the primary way Twitter users organize the information they tweet. Even though users often misuse them, they carry important information about the tweet. For simplicity, we added words from hashtags to the remainder of tweet text at the place they occurred in, but future work can include making a separate feature based only on hashtags.

Twitter users often use punctuation and emoticons to enhance the meaning of their tweets. Hence we extracted features based on exclamation (!) mark counts, question (?) mark counts, and ellipsis counts (...). Besides this, we count the number of emoticons since users often express sentiment in this way.

The sentiment of each word was obtained using NLTK's Sentiment Analysis Package. We create features for positive and negative word counts, where positive and negative words were extracted using a polarity threshold of 0.2. Additionally, we take into account mixed sentiment by constructing a feature that shows if a tweet contains negative and positive sentiment in a span of 5 words.

Next, we add the count of named entities in each example for the PERSON, GPE, NORP, and ORG tags. We think that users might often express irony when writing about politicians or similar entities.

4. Topic Model

Topic models are a natural language processing tool used for discovering abstract topics in a collection of documents. In this paper, we try to determine whether knowing the tweet's topic makes any difference in determining the sarcastic nature of the tweet.

Table 2: K-Means silhouette scores

Model	Validation	Test
KMeans, CountVectorizer	0.203	0.171
KMeans, TfIdfVectorizer	0.201	0.181

As a topic modeling baseline, we decided to use clustering algorithms. We use k-means with the count vectorized preprocessed text and k-means with the TF-IDF vectorized preprocessed text. The number of topics in these models was determined using the elbow method on inertia plots for models ranging from 2 to 60 clusters. Both of the values were set to 17. We used the silhouette score for evaluating the clusters, as seen in Table 2. We further manually examined the clustered examples. Both methods gave us poor results so we changed our approach.

In order to obtain more suitable and interpretable topics, we decided to use a deep learning model. Our choice was the BERTopic model from Grootendorst (2022) which uses HuggingFace transformers and a c-TF-IDF weighting scheme. The library used for this is the bertopic[6] library. We run a grid search for determining the hyperparameters for the minimum size of the topic and the number of words per topic to extract. For evaluating the models used in the grid search we use the UMass version of the coherence score metric as described in Röder et al. (2015). This metric measures the degree of semantic similarities between high-scoring words in the topic. We made use of the implementation in the gensim[7] library. Fortunately, this also

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

[2]https://github.com/cbaziotis/ekphrasis

[3]https://github.com/carpedm20/emoji/

[4]https://spacy.io/

[5]https://scikit-learn.org/

[6]https://maartengr.github.io/BERTopic/index.html

[7]https://radimrehurek.com/gensim/

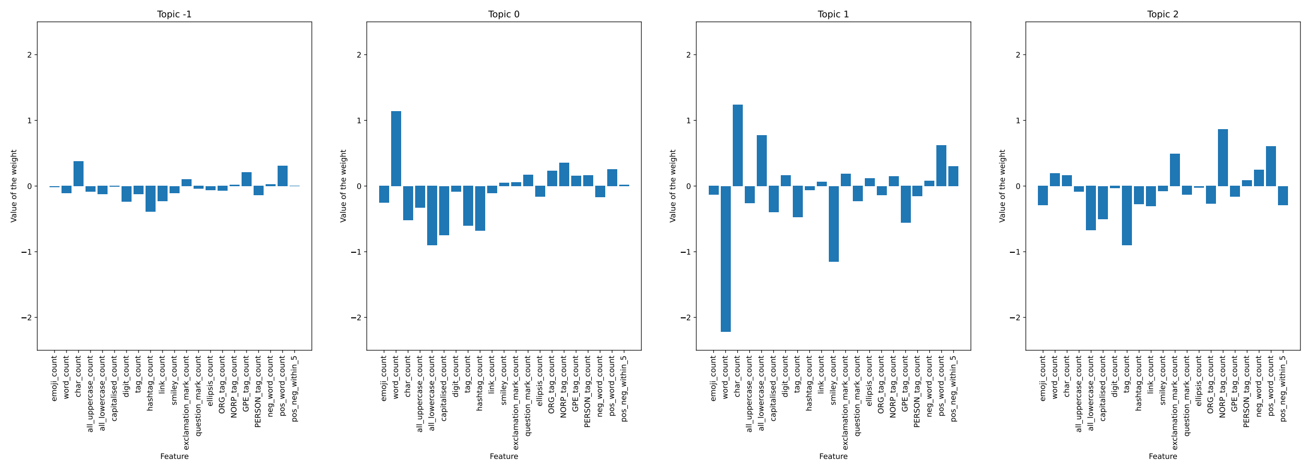

Figure 1: Feature importances for models trained on 4 most frequent topics

somewhat accomplished our goal of minimizing the count of outlier examples. We decided to set the minimum size of the topic to 20 and the number of words per topic to extract to 10 which gave us a coherence score of -17.935 on the validation set and -18.274 on the test set. The rest of the hyperparameters were set to default values.

In Table 3 most frequent words per topic are displayed with corresponding frequencies. As we can see, the topics are easily interpretable.

Table 3: Most frequent words per topic and the corresponding frequencies for first 10 topics and outliers

Most frequent words	Count
work, love, day, like	1124
police, black, obama, people	221
fan, game, play, player	182
lol, funny, know, think	161
study, school, class, final	116
sleep, wake, hour, bed	103
twitter, tweet, follow, social	93
christmas, gift, tree, santa	79
food, eat, turkey, drink	74
love, smile, relationship, girl	73
phone, app, service, mobile	72

5. Results

For the baseline model, we trained a logistic regression using baseline features as described in Section 3.3., where we used default values for parameters as defined in the scikit-learn library. To compare if tweet topics improve the model, we trained two logistic regression models, one with all features including topics and another one with all features excluding topics. Grid search was used as the hyperparameter optimization algorithm but the regularization type was set to L1 to improve model interpretability. Additionally, we trained another logistic regression model, where we allowed grid search to select different types of regularization, sacrificing model interpretability in the process. For evaluation metrics, we used accuracy, precision, recall, and F1-score as these were used as the official metrics for the competition. We present the results in Table 4.

Table 4: Test set results using different logistic regression models

Model	Acc	P	R	F1
Baseline	0.568	0.579	0.582	0.566
With topics (L1, L2, Elastic net)	**0.633**	**0.627**	**0.633**	**0.626**
With topics (L1)	0.631	0.626	0.631	0.625
Without topics	0.621	0.616	0.621	0.615

As we can see all models achieve better performance than the baseline model and models with topic features slightly outperform the model without topic features.

Next, we analyzed feature importance among the first four topics in descending order of the example counts. We did this by training a logistic regression and extracting feature weights. Figure 1 shows bar plots of feature weights for each topic. As we can see, feature weights differ significantly among different topics. This suggests that there are underlying differences between contextual topics. Table 5 shows evaluation metrics for these four models.

Table 5: Test set results for 4 most frequent topics

Most frequent words	Acc	P	R	F1
work, love, day	0.642	0.629	0.632	0.630
police, black, obama	0.569	0.567	0.568	0.567
fan, game, play	0.425	0.380	0.376	0.378
lol, funny, know	0.750	0.725	0.796	0.724

These results are inconclusive, since the first and last models achieved better performances, while the second and third achieved worse performances than the best model in

Table 4. This suggests that it is not worth training separate classifiers for each topic.

The obtained results suggest that tweet topics did slightly improve the model. One of the possible reasons why improvements are so small is that other features already contain information about the contextual topic of a tweet or that the other features sufficiently describe the irony within the tweet.

Additionally, there is a significant difference between feature importance in models trained on specific topics.

6. Conclusion

In this paper, we analyzed the significance of the contextual topic of a tweet in predicting irony. We created handcrafted features and tested if including the topic of a tweet improves irony detection. Results suggest that there are slight improvements. We also analyzed feature importance among different topics. Results showed that there is a noticeable difference in feature importance. In future work, we propose experimenting with different topic models and using machine learning algorithms with greater capacity.

References

Mondher Bouazizi and Tomoaki Ohtsuki. 2015. Sarcasm detection in twitter: ”all your products are incredibly amazing!!!” - are they really? In *2015 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6.

Hsia-Ching Chang. 2010. A new perspective on twitter hashtag use: Diffusion of innovation theory. *Proceedings of the American Society for Information Science and Technology*, 47(1):1–4.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Semi-supervised recognition of sarcastic sentences in twitter and amazon. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, CoNLL '10, page 107–116, USA. Association for Computational Linguistics.

Maarten Grootendorst. 2022. Bertopic: Neural topic modeling with a class-based tf-idf procedure. *arXiv preprint arXiv:2203.05794*.

Hen-Hsen Huang, Chiao-Chen Chen, and Hsin-Hsi Chen. 2018. Disambiguating false-alarm hashtag usages in tweets for irony detection. pages 771–777, 01.

K. Nimala, R. Jebakumar, and Mohan Saravanan. 2021. Sentiment topic sarcasm mixture model to distinguish sarcasm prevalent topics based on the sentiment bearing words in the tweets. *Journal of Ambient Intelligence and Humanized Computing*, 12:6801–6810.

Debora Nozza, Elisabetta Fersini, and Enza Messina. 2016. Unsupervised irony detection: A probabilistic model with word embeddings. In *International Conference on Knowledge Discovery and Information Retrieval*, volume 2, pages 68–76.

Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on Web search and data mining*, pages 399–408.

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, June. Association for Computational Linguistics.