Statistical Natural Language Processing : Project

Lucas Swiniarski
NetID : ls4411
lucas.swiniarski@nyu.edu

Ruben Stern
NetId : rns365
rns365@nyu.edu

1 Problem Setting

We want to tackle the problem of sarcasm detection. We find this problem interesting for various reasons :

- Sarcasm is a challenging task as it requires a high level of language understanding to be detected. For instance, one sentence can be both sarcastic and not sarcastic depending on the context.
- Even for a normal human being, it can be quite hard to detect sarcasm.
- Sarcasm detection raises interesting questions : How important is the context of a sentence in its sarcastic meaning ? Is sarcasm necessarily situation-bounded ? We could try to answer those questions by conditioning our classifier on different information. For instance, building a classifier to detect sarcasm based on the sentence, then adding a condition on the context of the sentence.

2 Data sets

The data set we are going to use is the Self-Annotated Reddit Corpus (SARC) which comes from [1]. It is a large corpus designed especially for sarcasm detection. It contains 533 million statements among which 1.3 million are labeled as sarcastic. This will allow us to train our model with both balanced and unbalanced methods.

This corpus have been extracted from Reddit, a social news aggregation, web content rating, and discussion website. Reddit's users can submit content such as text posts or direct links. They can then vote submissions up or down that determines their position on the page. Submissions with the most up-votes appear on the front page or the top of a category. In addition, content entries are organized by areas of interest called subreddits. For instance, subreddit topics are news, science, movies, music, books, food, ... Thanks to its 234 million monthly unique users and it 725 million comments, Reddit correspond to a huge source of text data. Indeed, SARC Data set have been built as a subset of comments from January 2009 to April 2017.

Furthermore, each statement in SARC have been labeled by the author (not an independent annotator) and is provided with user, topic and conversation context (previous comments in the page). The common method used by authors for sarcasm annotation consists of putting the marker "/s" at the end of their statement. Such a method implies that these markers are noisy since many users do not make use of the marker, do not know it, or use it only when the sarcastic intent is not obvious. In order to reduce this noise, authors of [1] excluded comments that are answers of sarcastic comments as the label in this case is extremely noisy because authors agree or disagree with the previously expressed sarcasm with their own sarcasm without often labeling it. Another technique they used to decrease the noise of the SARC is to keep only comments from users that know about the "/s" sarcasm annotation which corresponds to users who have already used the marker previously.

In order to do a raw estimation of SARC's noise, they also labelled manually a sample of 500 comments tagged as sarcastic and 500 tagged as non sarcastic, taking into account the full context of

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

these comments. Thus, they tried to estimate false positive rate and false negative rate of their data set. They found out a false positive rate of 1.0% and a false negative rate of 2.0%. The false positive rate remains reasonable while the false negative rate is significant as the sarcasm proportion is 0.25% which highlights a wide definition of sarcasm and indicates a need of methods that can handle noisy data in the unbalanced setting. In the balanced setting, it still corresponds to a small amount of noise.

We also plan to use a data set available on Kaggle: South Park cartoon lines. It contains all the lines (70k) from the cartoon South Park annotated with season, episode and speaker. Although this data set is not labeled with sarcasm, we want to apply our model trained on SARC data in order to quantify how sarcastic each character is and rank them by sarcasm. Then, by doing some visualization and look manually at the produced labels, we want to check the transfer learning ability of our model.

3 Evaluation Metric

As the problem we want to tackle correspond to binary text classification, we will use common evaluation metrics: precision and recall.

4 Survey of related work / baselines

Following [2] in Automatic Sarcasm Detection: A Survey, there are three main approaches to sarcasm detection, a rule-based approach, a statistical approach a deep learning one.

4.1 Rule-based approach

Veale and Hao [3] identifies whether simile (construction of the form : * as a *) is intended to be sarcastic or not. They also use Google search to compute the posterior probability of each simile. The strength of this approach lies on decision understanding : given a test example you know why your classifier has taken a decision.

4.2 Statistical Approaches

In statistical approaches, there are two ways to get better classifiers : Having better featurization of the sentence and its context, and having a better learning algorithm. Such features can be :

- bag of words or word-level n-gram.
- character level n-gram.
- intensity of adjectives and adverbs used.
- min/max/average number of synonyms in a sentence.
- frequency of words used.

For the learning algorithm, most techniques use support-vector machine for classification, Pearson chi-square test for assessing if the correlation between a feature and the target label is meaningful.

4.3 Deep learning approach

The deep learning approach mainly use recurrent neural network for sentence embedding which alleviate the need to featurize the text. Some work has been done to learn user-embedding as some users tend to use sarcasm more frequently. Recent work focus on leveraging the context of a sentence, as it is a good indicator of sarcasm as well.

5 Your Proposed Approach

The first step is to classify sarcasm at a sentence level, using different algorithms :

- Featurizing the sentence, using bag of words, character-level n-gram, and more complex features. Then using a maxent model, or a svm.

- Learning a word embedding and a sentence embedding using a recurrent neural network, and use the sentence embedding to predict sarcasm.

Further models will include using the context of a sentence (previously said sentences) to help predict whether the current sentence contains sarcasm or not.

We also plan the domain transfer ability of our models. Indeed, the idea would be to train them on the SARC data set and apply it to the South Park cartoon lines. Then, we want to use this to quantify how sarcastic is each character and rank them by sarcasm. As this data set is not labeled, we will not be able to evaluate our prediction with precision and recall. Instead, we want to do some visualization and look manually at the produced labels. For instance, using our knowledge of the cartoon, we should be able to assess if our ranking is reasonable. We can also have a look at some of the most confident predictions and see if we agree with them. In addition, we will be able to compare the percentage of sarcastic content from SARC and South Park data sets. This study will then enable us to assess the transfer learning ability of our model and his ability to generalize.

References

[1] "A Large Self-Annotated Corpus for Sarcasm". 2017. URL: https://arxiv.org/abs/1704.05579.

[2] Aditya Joshi et al. "Automatic Sarcasm Detection: A Survey". In: *ACM Computing Surveys* (). URL: https://arxiv.org/pdf/1602.03426.pdf.

[3] Tony Veale and Yanfen Hao. "Detecting Ironic Intent in Creative Comparisons". In: *ECAI, Vol. 215* (2010). URL: http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=854DFFA350ABFD5D05171AF56922B036?doi=10.1.1.385.4410&rep=rep1&type=pdf.