Mama Edha at SemEval-2017 Task 8: Stance Classification with CNN and Rules

Marianela García Lozano[*,†], Hanna Lilja[*], Edward Tjörnhammar[†], and Maja Karasalo[*]

[*]FOI Swedish Defence Research Agency
[*]{garcia, hanna.lilja, maja.karasalo}@foi.se
[†]KTH Royal Institute of Technology
[†]{mgl, edwardt}@kth.se

Abstract

For the competition SemEval-2017 we investigated the possibility of performing stance classification (support, deny, query or comment) for messages in Twitter conversation threads related to rumours. Stance classification is interesting since it can provide a basis for rumour veracity assessment. Our ensemble classification approach of combining convolutional neural networks with both automatic rule mining and manually written rules achieved a final accuracy of 74.9% on the competition's test data set for Task 8A. To improve classification we also experimented with data relabeling and using the grammatical structure of the tweet contents for classification.

1 Introduction

The task of determining the veracity of a rumour is sometimes a difficult one, even with the reasoning power of a human being. This paper presents an approach to an automatic analysis of discussion elements with respect to rumours. Discussion structure and analysis can well play a part in a broader effort to assess rumour veracity, and the expectation is that the results presented here is one step of the way towards that end goal.

The research presented in this paper is a submission to SemEval-2017, Task 8 (RumourEval: Determining rumour veracity and support for rumours), Subtask A (SDQC) (Derczynski et al., 2017). The objective of this subtask is to classify the relation between a tweet and the rumour it is related to in terms of support, deny, query or comment.

Our approach to this classification task is building three different classifiers and combining the predictions in an ensemble method. The general idea is that different types of classifiers may learn different concepts and hence complement each other, resulting in a better prediction capability for the joint classifier. Furthermore we tested the accuracy in applying our ensemble approach to both originally labeled and relabeled data.

The remainder of this paper is organized as follows: Section 2 describes the data given in the task and our observations on irregularities in the data labeling. Section 3 contains a description of the process and used methods employed for the experiments. Sections 4 and 5 describe the results and discusses the findings. Finally, we conclude the work in Section 6.

2 Data Inconsistencies

The data used in the SemEval-2017 Task 8A is a subset of the PHEME data set of social media rumours (Zubiaga et al., 2016).

When studying the dataset, it was discovered that for some tweets, the annotation is somewhat inconsistent, e.g., tweets with very similar contents have sometimes received different annotations. For example, despite being nearly identical replies to the same source tweet, the first is annotated with (C) and the second with (S):

> I just feel sick RT @[user]: At least 12 dead in Paris shooting. Updated story:[link]
>
> Awful. RT @[user]: At least 12 dead in the Paris shooting. [link]

Other found issues were that tweets were sometimes labeled (Q) although they contained no query. Some tweets were labelled with respect to its direct predecessor in the conversation thread, as opposed to with respect to the source tweet. Also, some tweets were annotated (S) for simply

Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval-2017), pages 481–485,
Vancouver, Canada, August 3 - 4, 2017.

expressing empathy or concern, without any input on the veracity of the rumour.

3 Method

For solving the task we had an approach of multiple parallel data pipelines, for our workflow schema, see Figure 1. The following sections and subsections describe the parts of the figure.

3.1 Data Processing

Chosen parts of the raw data were extracted into new subsets of attributes, tailored for each type of classifier. Among the extracted data attributes were the text content of the tweets, metadata related to the tweets (such as time of posting) and metadata related to the users (such as number of followers). In terms of training, development and test data we used the data split provided by the SemEval Task organizers.

To investigate whether the inconsistencies in the annotation would affect the results of the experiments, we constructed a separate data set in which we tried to remedy the found inconsistencies.iIn uncertain cases, the original annotation was used.

3.2 Feature Engineering

Most of the metadata attributes from the raw data could be used as features for classification without further processing. However, some attributes (the text data in particular) required preprocessing and feature engineering in order to be a useful representation of the data.

3.2.1 Preprocessing

Tweets have limited space (only 140 characters) and hence, symbols, abbreviations, slang and contractions are used in an effort to increase the information ratio. The downside of this is that the tweets tend to appear noisy to NLP software and some preprocessing is usually needed. The measures we applied were, e.g., splitting contractions and stemming the words. For the stemming we employed the Python package "Snowballstemmer". Also, in an effort to avoid training the classifiers on data that could be too context specific, e.g., a link or a user name. We elected to remove mentions and links, replacing them with a simple "@" and "http://".

3.2.2 Grammatical Representation Generation

We investigated whether the grammatical structure could be useful as a replacement of the tweets themselves. We chose to use a pretrained model on the English language for the initial tweet to Part Of Speech (POS) Tagging, i.e., Parsey McParseface[1] (Andor et al., 2016). Further; We relied on CoNLL-U (Nivre, 2015), a unicode version of CoNNL-X (Buchholz and Marsi, 2006) as our data format for the POS tagging. Input tweets were not stemmed but otherwise preprocessed.

We utilized the same rule encoding as (Feng et al., 2012) and chose the one level neighbourhood semantic rules, i.e. the $\hat{r}*$ notational case, i.e., unlexicalized production rules combined with the grandparent node.

3.2.3 Word2vec Embeddings

We used word2vec vectors learned using the skip-gram model which predicts the linear context of the words surrounding the target words (Mikolov et al., 2013)[2].

3.3 Classification

In our method we used an ensemble approach consisting of combining several classifiers and using their individual strengths in the final voting.

3.3.1 CNN

In our experiments we followed (Kim, 2014)'s architecture[3]. We experimented with different CNN model versions, see all paths that lead to CNN in Figure 1. We used a non static setting for the word embeddings which, as previously mentioned, came from the pretrained word2vec corpus. Due to the skewed nature of the training data with two classes being significantly more common than the other two we adjusted the weights to reflect this relationship, i.e., $w[S, D, Q, C] = [0.157, 0.396, 0.399, 0.048]$. We used no regularization.

[1]The environment containing the trained Parsey McParseface model is available at docker hub as `edwtjo/syntaxnet:conll`

[2]The pretrained corpus `GoogleNews-vectors-negative300.bin.gz` can be downloaded from `https://code.google.com/archive/p/word2vec/`.

[3]As a starting point for the CNN-implementation we used Denny Brick's version of (Kim, 2014)'s Theano implementation. The code can be found at `https://github.com/dennybritz/cnn-text-classification-tf`

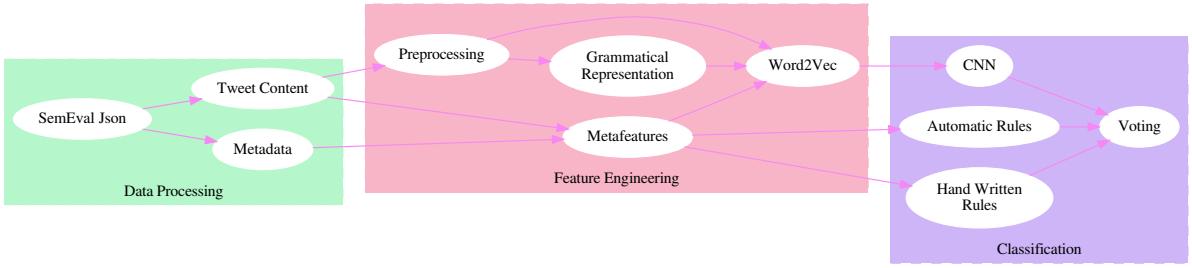

Figure 1: Workflow schematic for our data pipeline

3.3.2 Automatic Rule Mining

Classification through automatic rule mining was done with the WEKA (Witten et al., 2016) implementation of PART (Frank and Witten, 1998). PART was used with the confidence factor set to 0.25 and the minimum number of instances per rule set to 2. For this classifier the data was represented in terms of a number of metadata features: i) Number of ".", "?", "!", and negations in the tweet. ii) Number of statuses, followers and friends, and favourites of the user. iii) Whether the user is verified or not. iv) Whether the tweet was a reply or not and its conversation depth. The conversation depth of a tweet is here defined as 0 for a source tweet, 1 for a reply to a source tweet, and so forth.

3.3.3 Hand Written Rules

The classifier based on hand written rules (HWR) relies on a small set of rules to classify tweets. The rules are designed to favour precision over recall, and were constructed through manual inspection of the training data. The default class is *comment*. One type of rule checks if any predefined key phrases are present in the tweet and if so classifies the tweet accordingly. An example of such a rule is: *If sentence contains phrase 'not believable': → assign class* ***deny*** The second type of rule checks for certain combinations of occurrences of "@"-mentions, "#", and urls in the whole text, as well as "?" in the first 5 tokens of the tweet. An example of this type of rule is: *If sentence contains an* ***url*** *and does* ***not start*** *with an* ***@-mention****: → assign class* ***support***

3.3.4 Voting

The final prediction of the class of a data instance is achieved through a voting procedure. As different classifiers have different strenghts and weaknesses, which vary over classes as well as between precision and recall, not all votes are counted as equal. A vote on a certain class from a classifier with high precision on said class is deemed more important.

4 Results

In addition to the results on the test data, results on the development data are also provided.

	Label		
	original (%)	**our (%)**	**diff (pp)**
S	910 (20.1)	955 (21.1)	+44 (+1)
D	344 (7.6)	230 (5.1)	-114 (-2.5)
Q	358 (7.9)	326 (7.2)	-32 (-0.7)
C	2907 (64.3)	3009 (66.6)	+102 (+2.3)

Table 1: Distribution of tweets between classes using original labels and our new labels.

4.1 Relabeling

Out of 4519 tweets we relabeled 846, or about 19%. The distribution of tweets between classes before and after relabeling is shown in Table 1.

4.2 Development Data Performance

The CNN trained using preprocessed tweet contents, together with each tweets absolute time (counting from its source tweet as point zero), and dynamic word2vec settings reached an accuracy of 0.715 on the development set. The PART rules reached an accuracy of 0.701 on the development set. The hand written rules reached an accuracy of 0.733 on the development set.

After tweaking of the voting schema to make use of the strengths of the different methods an accuracy of 0.758 on the development set was reached. The best accuracy on the development data was achieved with the following priority order of votes from the classifiers: 1) CNN vote on support, 2) HWR vote on deny or query, 3) PART vote on query, 4) CNN vote on deny or query, 5) HWR vote on support, 6) Default class comment.

		Precision				Recall			
Model	Accuracy	S	D	Q	C	S	D	Q	C
HWR	0.751	0.55	0.053	**0.605**	0.783	0.319	0.014	0.217	0.943
PART	0.745	0.423	0.105	0.553	0.802	0.319	0.028	0.396	0.910
PART$_R$	0.732	0.343	0.167	0.558	**0.811**	0.383	0.042	**0.453**	0.875
CNN	**0.752**	**0.622**	0.0	0.333	0.761	0.298	0.0	0.009	**0.977**
CNN$_G$	0.633	0.563	**0.2**	0.286	0.681	**0.391**	**0.182**	0.071	0.850
CNN$_R$	0.745	0.538	0.0	0.333	0.758	0.298	0.0	0.009	0.968
Voting	*0.749*	*0.5*	*0.05*	*0.525*	*0.812*	*0.372*	*0.014*	*0.5*	*0.896*

Table 2: Results on test data set, comparison between classifiers. Subscript R after the classifier name stands for "Relabeled" and subscript G stands for "Grammatical Representation". The best result in each column is marked with bold face.

4.3 Test Data Performance

Table 2 shows performance measures for our classifiers, evaluated on the test data. In the Table we have also included the results of training both with the new labels, marked with subscript R, and training a CNN model with only the grammatical representation of the tweet contents and absolute time, marked with subscript G.

5 Discussion

The data set class distribution is rather skewed, with the vast majority of the tweets classified as comments. In fact, using a simple majority classifier would result in an accuracy of 64.3%, i.e., the ratio of tweets classified as comments. This might be considered an evaluation baseline. In our experiments with relabeling the data to try and remedy the found inconsistencies we changed almost one fifth of the labels. Which could indicate that the labels are highly subjective. A reason for this could be that the overall rumour was not given in the data. A downside of the relabeling was that the two largest classes, i.e., S and C, became even more prevalent.

There was a great difference in the performance of the models on the development and test data, e.g., the CNN model went from an accuracy of 71.5% to 75.2%. This indicates that the development data might not be representative for the test data and that methods developed for this task should take care to avoid being too domain specific.

The CNN based models show consistently best accuracy for a few epochs, on average around 8 epochs, and after that the models become overfitted. We tested with regularization to avoid overfitting, but to no avail. This only prolongs the number of epochs necessary for training. Note that we did not do experiments with more than 200 epochs. A theory is that the data set is simply so small that it is easy to overfit a CNN-model and the number of epochs thus must be carefully monitored.

For this competition it became apparent, during the development phase and tuning of hyper parameters, that the grammar structure of tweets had little impact on the overall accuracy of the model, see CNN$_G$ results in Table 2. There are many possible reasons for this but the most likely is that tweets follow a very reduced grammatical structure due to their inherent shortness and as such exhibit similar distributions over the possible output classes. But, there was one class in which the CNN$_G$ model had much better precision than all other models, i.e., the elusive deny class.

The hand written rules performed remarkably well considering their simplicity. A possible explanation for this is of course that the person constructing the rules will make use of general language and world knowledge that a machine may not have access to. A potential problem with this approach is that it might be difficult to significantly improve the performance, at least without a substantial manual effort. Some metadata and text meta features are less intuitive for a human to manually write rules based upon. The PART classifier could more easily handle these features and so discover rules that would have been difficult for a human to come up with.

The main goal of the voting procedure is to exploit the strengths of different types of classifiers. We decided to rank votes per classifier and class, rather than a more complex weighting scheme or a simple majority vote. A motivation for this was

the tendency of the classifiers to favour precision over recall (in the case of HWR) or to be heavily biased towards the comment class due to the class imbalance of the data set.

6 Conclusions

The key contributions of this paper are, among other, approaching the task of stance classification with an ensemble method combining CNNs with both automatic rule mining and manually written rules. Interesting feature engineering was done by, among other things, relabeling the data and using the grammatical structure of the tweet contents.

Utilizing each method's strengths the results were weighted with a voting system developed for the task. The submitted system achieved a final accuracy of 74.9% on the competition's test data set, placing the team at a fourth place on the SemEval-2017 RumourEval task 8A.

Acknowledgments

PhD. M. Rosell suggested relabeling the data.

References

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. 2016. Globally normalized transition-based neural networks. *arXiv preprint arXiv:1603.06042* .

Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X Shared Task on Multilingual Dependency Parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*. Association for Computational Linguistics, pages 149–164.

Leon Derczynski, Kalina Bontcheva, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Arkaitz Zubiaga. 2017. Semeval-2017 task 8: Rumoureval: Determining rumour veracity and support for rumours. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, Vancouver, Canada, pages 69–76. http://www.aclweb.org/anthology/S17-2006.

Song Feng, Ritwik Banerjee, and Yejin Choi. 2012. Syntactic Stylometry for Deception Detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*. Association for Computational Linguistics, pages 171–175.

Eibe Frank and Ian H. Witten. 1998. Generating Accurate Rule Sets Without Global Optimization. Morgan Kaufmann, pages 144–151.

Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. *CoRR* abs/1408.5882.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *Advances in Neural Information Processing Systems*. pages 3111–3119.

Joakim Nivre. 2015. Towards a Universal Grammar for Natural Language Processing. In *International Conference on Intelligent Text Processing and Computational Linguistics*. Springer, pages 3–16.

Ian H. Witten, Eibe Frank, Mark A. Hall, and Christopher J. Pal. 2016. *The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques"*. Morgan Kaufmann, 4 edition.

Arkaitz Zubiaga, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Peter Tolmie. 2016. Analysing How People Orient to and Spread Rumours in Social Media by Looking at Conversational Threads. *PloS one* 11(3):e0150989.