TakeLab at SemEval-2016 Task 6: Stance Classification in Tweets Using a Genetic Algorithm Based Ensemble

Martin Tutek, Ivan Sekulić, Paula Gombar, Ivan Paljak, Filip Čulinović, Filip Boltužić, Mladen Karan, Domagoj Alagić, Jan Šnajder
University of Zagreb, Faculty of Electrical Engineering and Computing
Text Analysis and Knowledge Engineering Lab
Unska 3, 10000 Zagreb, Croatia
name.surname@fer.hr

Abstract

This paper describes our system for the detection of stances in tweets submitted to SemEval 2016 Task 6A. The system uses an ensemble of learning algorithms, fine-tuned using a genetic algorithm. We experiment with various off-the-shelf classifiers and build our model using standard lexical and a number of task-specific features. Our system ranked 3rd among the 19 systems submitted to this task.

1 Introduction

Stance is the overall position held by a person towards an object, idea, or proposition (Somasundaran and Wiebe, 2009). The task of stance detection – the automatic classification of stance expressed in text – has been attracting increasing interest, as it is of practical interest to many stakeholders, ranging from political bodies to companies. Most recent work focused on stance detection in online debates (Somasundaran and Wiebe, 2009; Somasundaran and Wiebe, 2010; Anand et al., 2011; Hasan and Ng, 2014; Sridhar et al., 2015).

Twitter is an outstanding platform for large-scale stance analysis. However, unlike in the case of dedicated online debate platforms, Twitter data is much less structured and dialogical. Furthermore, as pointed out by Rajadesingan and Liu (2014), processing of tweets poses specific challenges stemming from the high volume of data, brevity of messages, and the use of non-standard language.

In this paper, we describe a system for stance classification in tweets, with which we participated in the SemEval-2016 Task 6A. Given a tweet and the topic of the tweet (the *target*), the task was to predict the stance of the tweet author as either in FAVOR of the target, AGAINST it, or NONE. Our system relies on a supervised three-way classifier, using features standardly employed for stance classification and similar tasks, but also a number of task- and target-specific features. The gist of our approach was to start out with the entire "kitchen sink" of features, as well as a number of off-the-shelf classification algorithms, and then perform a comprehensive model optimization on a per-target basis. However, instead of relying on a single model for each target, we trained four different models and combined them into one classifier ensemble for each target using a genetic algorithm. Our system (TakeLab) ranked 3rd among the 19 systems submitted to the SemEval-2016 Task 6A.

2 Related Work

To the best of our knowledge, the only work to have addressed the stance classification in tweets is that of Rajadesingan and Liu (2014), who specifically tackle the data sparseness problem using a semi-supervised approach based on label propagation. Somasundaran and Wiebe (2009) and (2010) address stance detection in two-sided debates using supervised models with opinion-target pairs, as well as sentiment and argumentation trigger words as features. Anand et al. (2011) address the same domain, but also consider the dialogical properties of debates by identifying the rebuttals between posts, while Sridhar et al. (2015) consider the joint stance classification of posts and relations among them. Hasan and Ng (2014) combine stance classification with reason classification in a joint learning framework.

Proceedings of SemEval-2016, pages 476–480,
San Diego, California, June 16-17, 2016.

3 Model

We approach this task as a three-way multiclass supervised classification problem. We start off with a number of learning algorithms and also design a number of lexical and task-specific features. Subsequently, we reduce the algorithm and feature space by employing a series of optimization rounds. Finally, we train and fine-tune an ensemble of the chosen classifiers using a genetic algorithm. Following sections describe these steps in more detail.

3.1 Features

We first preprocess the data: we tokenize the tweets,[1] stem the resulting tokens, and finally eliminate the stop words using the NLTK toolkit (Bird et al., 2009). In some configurations, we use a sentiment lexicon compiled by Han and Baldwin (2011) to replace all positive and negative sentiment-bearing words with dummy labels POS and NEG, respectively.

We compute two types of features: lexical features and task-specific features. We compute the former after preprocessing and the latter before preprocessing the data. The lexical features are as follows:

- **Word features** – Word unigrams, bigrams, and trigrams, computed as (1) binary vectors, (2) count-based vectors, and (3) tf-idf-weighted vectors. We use a frequency cut-off of 2 and additionally filter based on class entropy with a cut-off set at 1.1;
- **Character features** – Character bigrams and trigrams, computed in the same manner as the word features;
- **Word embeddings** – A 300-dimensional distributional representation of the tweet obtained as as a weighted addition of the distributional vectors of the individual words. We use the freely available[2] skip-gram embeddings of Mikolov et al. (2013) and weight each vector based on the information content of the corresponding word, following Šarić et al. (2012).

We also used the following task-specific features:

- **Counting features** – The average word length, number of retweet symbols, number of hashes, number of emoticons, number of capitalized words, and the number of exclamation marks;
- **Repeated vowels** – Whether the tweet contains at least one sequence of the same vowel longer than two characters. This feature is often employed in Twitter sentiment analysis, e.g., in (Xu et al., 2015);
- **Number of misspelled words** – The number of misspelled words, determined using the freely-available PyEnchant spellchecking library;[3]
- **Scripture citation** – Our analysis of the dataset revealed that, for two out of five targets, namely *Atheism* and *Legalization of Abortion*, the users who quote the scriptures are by and large AGAINST the targets. To capture this regularity, we include a feature that checks whether the tweet matches some of the common scripture citation patterns, such as "*Rom. 14:17*" in "*RT @prayerbullets: Let the righteousness, peace, and joy of the kingdom be established in my life -Rom. 14:17*";
- **Hashtag splitting** – Our analysis also revealed that for some targets the hashtags are highly indicative of the stance. In some cases, however, a hashtag – although highly indicative – occurs quite rarely in the dataset. For example, in the tweet "*Rethink your beach clothes. Bc it may oppress some people!! #thisoppresseswomen*" the hashtag is fully indicative of the stance, but overall it occurs rarely. On the other hand, the unigram "oppress" and the bigram "oppress women" are rather frequent in the dataset and also indicative of the stance. To account for this, we split up each hashtag into its constituent words by employing a simple greedy procedure: we start from the end of the hashtag and work our way towards its start, always taking the longest possible word contained in the dictionary.

3.2 Model Optimization

Algorithm selection. We started off considering a number of different classification algorithms, implemented in the scikit-learn package (Pedregosa et al., 2011): Support Vector Machine (SVM), Random Forest (RF), Logistic Regression (LR), Gradient Boosting (GB), Multinomial Bayes (MB), Extra Trees (ET), and general stochastic gradient descent classifier (SGDC). We then fixed a baseline set of fea-

[1] http://sentiment.christopherpotts.net/code-data/happyfuntokenizing.py

[2] https://code.google.com/archive/p/word2vec/

[3] http://pythonhosted.org/pyenchant/

tures consisting of word unigrams and bigrams, and evaluated all the combinations of classifiers and text representations using a 3-fold cross-validation. We discarded the classifiers that performed considerably worse than the others, leaving us with four classifiers: RF, GB, LR, and SVM.

Ensemble learning. We decided to opt for an ensemble classifier using the four remaining algorithms, motivated by the fact that ensembles generally work better than their base learners as well as the fact that all of our algorithms performed comparably.

With this in mind, our next step was to optimize the hyperparameters of each of the four classifiers to have a fixed ensemble model. Since an exhaustive search would be too time-consuming, we arbitrarily fixed multiple feature sets and hyperparameter ranges.

We define the ensemble model as a linear combination of the output probabilities of the classifiers, and optimize its weights with respect to the F-score. A common way of building the ensemble is stacking, in which one uses the classifiers' predictions as inputs to a meta-level classifier. However, we decided not to use this approach as it would not allow us to optimize for the F-score. Instead, we used a genetic algorithm, which works with arbitrary objectives as fitness functions. As we optimize only four values, running time was not an issue.

We modeled the operators of the genetic algorithm as follows. We initialized the population of size 100 uniformly across the interval $[0, 1]$, and constrained the weights to the same interval by clipping. For crossover, we used tournament selection, randomly selecting three individuals from the population, replacing the worst of the three with the child of the remaining two. The crossover created a child by randomly selecting weights from either of the parents. We used mutation with random uniform noise from the interval $[-0.3, 0.3]$.

Feature selection. Lastly, we needed to find the best feature set for each target. We manually compiled a list of feature sets based on our experience with the task as well as intuition about which features work well in practice. We treated the ensemble as a single classifier, and all the constituents received the same feature sets as the input. However, instead of optimizing the score of the individual classifiers, we cross-validated the whole ensemble with respect to the input features and the F-score, treating the genetic algorithm as a trainable classifier, with the stochastic search as an optimization algorithm.

4 Evaluation

4.1 Task Description

The dataset consisted of 2814 tweets in the train set and 1249 in the test set divided into five targets – *Atheism (ATH)*, *Climate Change is a Real Concern (CC)*, *Feminist Movement (FM)*, *Hillary Clinton (HC)*, and *Legalization of Abortion (LA)*. Possible stances were FAVOR, AGAINST, and NONE, where the latter served both as an indicator of a tweet not being related to the target (e.g., *"Pop may throw in the towel second half.."*), as well as the tweet being neutral towards the target (e.g., *"atheism involves what a person does or does not believe, agnosticism involves what a person does or does not know. #Waterford"*).

The official evaluation measure was the macro-average of F-score for FAVOR and AGAINST across all targets, meaning that weak F-score performance on an unbalanced label distribution for a target could be compensated for by the overall good performance on other targets. Note that the label NONE was ignored during the evaluation. Consequently, misclassifying FAVOR or AGAINST as NONE (or vice versa) was penalized less than misclassifying FAVOR as AGAINST (or vice versa).

Although the task can straightforwardly be approached as a three-way classification problem, it can also be framed as a two-step binary classification problem, first discriminating between NONE and FAVOR+AGAINST, and then between FAVOR and AGAINST. The potential benefit of the two-step approach is that the features may be separated more clearly between classes, while the downside is that the error propagates from the first to the second stage.

The ambiguity of the label NONE posed a problem in approaching the task as a two-step binary classification. The best accuracy on the first step (NONE vs. FAVOR and AGAINST) was around 70%, while the best classifiers reached about 80% accuracy in the second step (FAVOR vs. AGAINST), resulting in an overall much higher error rate than that of the three-way classification model.

4.2 Feature Analysis

As described in Section 3, we used an ensemble of classifiers fit by a genetic algorithm. One of the steps was to determine the best features for each target and for each classifier. For brevity, we provide just a part of the results of our feature analysis study in Table 1. We show the F-score for FAVOR and AGAINST for each target as well as the score across all targets. We consider the following ten feature groups (the first three are the groups used in the submitted system):

- **Group 1** – binary-weighted word unigrams, bigrams, and trigrams, as well as character trigrams and stylistic features (the first four task-specific features). This group was used for the ATH and LA targets;
- **Group 2** – POS/NEG labels, binary-weighted word unigrams, bigrams, and trigrams, character trigrams. This group was used for the CC and FM targets;
- **Group 3** – binary-weighted word unigrams, bigrams, and trigrams, as well as character trigrams, stylistic features, and word embeddings. This group was used for the HC target;
- **Group 4** – binary-weighted word unigrams;
- **Group 5** – word embeddings;
- **Group 6** – binary-weighted character trigrams, word embeddings;
- **Group 7** – frequency-weighted word unigrams and bigrams, character trigrams, stylistic features;
- **Group 8** – no stemming, binary-weighted word unigrams, bigrams, and trigrams, character trigrams, stylistic features;
- **Group 9** – no stemming, frequency-weighted word unigrams, bigrams, and trigrams, character trigrams, stylistic features;
- **Group 10** – frequency-weighted unigrams and bigrams, character trigrams.

4.3 Model Variants

In Table 2, we provide the performances (scored with the official evaluation metric) of the top three systems from the official run, namely MITRE, pkudblab, and TakeLab (our submission), as well as of the number of other model variants we produced. We include our single best performing classifier – the Random Forest classifier (Best single) and an ensemble where the output probabilities are simply averaged (Averaging). Additionally, we include an optimistic variant of our best model (Optimistic), where the features and classifier hyperparameters are the same as in our submitted model, but we used the gold labeled test set as the validation set for the genetic algorithm. The result is a rough estimate of the upper-bound F-score of our submitted model on this dataset.

Group	ATH	CC	FM	HC	LA	All
1	**0.686**	0.712	0.645	0.638	0.660	**0.698**
2	0.669	**0.720**	**0.689**	0.621	0.658	0.694
3	0.676	0.706	0.654	0.643	0.645	0.689
4	0.673	0,707	0.654	**0.645**	0.636	0.694
5	0.655	0.706	0.659	0.628	0.649	0.692
6	0.672	0.719	0.657	0.631	0.634	0.697
7	0.660	0.719	0.664	0.636	0.655	0.693
8	0.655	0.715	0.655	0.639	0.611	0.692
9	0.672	0.718	0.660	0.618	0.640	0.695
10	0.656	0.714	0.655	0.628	**0.661**	0.690

Table 1: Feature analysis.

Team/model	F1-score
Optimistic	0.6956
MITRE	0.6782
pkudblab	0.6733
TakeLab*	0.6683
Majority vote	0.6522
Averaging	0.6331
Best single	0.6161

Table 2: Model performances (* marks our submitted model).

We also include a majority vote per target baseline model (Majority vote). The baseline yields an F-score of 0.6522, which surprisingly places it in the 6th place on the task leaderboard.

5 Conclusion

We described the stance classification system with which we participated in the SemEval-2016 Task 6A. Our system uses an ensemble of supervised classifiers, trained using a number of lexical and task-specific features, and optimized using a genetic algorithm. Our system ranked 3rd in the official evaluation run.

There are many possible directions for future work. One is to use the tweet data of users and their social network to augment the training data. Multi-task learning might also be worth investigating, as

some of the topics are semantically related. Furthermore, using external knowledge, such as searching the Wikipedia for keywords related to a topic, may be useful for identifying the tweets related to a topic.

References

Pranav Anand, Marilyn Walker, Rob Abbott, Jean E Fox Tree, Robeson Bowmani, and Michael Minor. 2011. Cats rule and dogs drool!: Classifying stance in online debate. In *Proceedings of the 2nd workshop on computational approaches to subjectivity and sentiment analysis*, pages 1–9. Association for Computational Linguistics.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly Media.

Bo Han and Timothy Baldwin. 2011. Lexical normalisation of short text messages: Makn sens # twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, volume 1, pages 368–378. Association for Computational Linguistics.

Kazi Saidul Hasan and Vincent Ng. 2014. Why are you taking this stance? Identifying and classifying reasons in ideological debates. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 751–762. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Ashwin Rajadesingan and Huan Liu. 2014. Identifying users with opposing opinions in twitter debates. In *Social Computing, Behavioral-Cultural Modeling and Prediction*, pages 153–160. Springer.

Frane Šarić, Goran Glavaš, Mladen Karan, Jan Šnajder, and Bojana Dalbelo Bašić. 2012. TakeLab: Systems for measuring semantic text similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, pages 441–448. Association for Computational Linguistics.

Swapna Somasundaran and Janyce Wiebe. 2009. Recognizing stances in online debates. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 226–234. Association for Computational Linguistics.

Swapna Somasundaran and Janyce Wiebe. 2010. Recognizing stances in ideological on-line debates. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, pages 116–124. Association for Computational Linguistics.

Dhanya Sridhar, James Foulds, Bert Huang, Lise Getoor, and Marilyn Walker. 2015. Joint models of disagreement and stance in online debate. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 116–125. Association for Computational Linguistics.

Hongzhi Xu, Enrico Santus, Anna Laszlo, and Chu-Ren Huang. 2015. LLT-PolyU: Identifying sentiment intensity in ironic tweets. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 673–678. Association for Computational Linguistics.