Team 'How Does That Make You Feel?': SemEval2016 Task 4 Proposal*

Evan Akers – M08637241
Kyle Arens – M06181631
Ryan Benner – M04890768
Joe Hirschfeld – M08704926

Anonymous NAACL submission

Abstract

This document is a proposal for the SemEval2016 Task 4 Sentiment Analysis in Twitter. It is made up of five subtasks, labeled A through E, covering both labeling tweets on a 3-point and 5-point scale as well as quantification and prediction of an entire grouping of Tweets (Preslav Nekov et al., 2016). We approached this project by developing a classifier for the 5-point classification and quantification subtasks, then generalizing this for the 3-point subtasks.

1 Task Description

Twitter provides a convenient platform for us to share information, news, and opinions. As such, it can provide a rich dataset of Tweets about a variety of topics, such as politics, products, and many other areas. This has made it popular for various natural language processing research projects (Preslav Nekov et al., 2016). This paper describes our approach to the SemEval2016 Task 4 challenges.

The SemEval2016 Task 4 are described as follows:

Subtask A: Given a tweet (irrespective of topic) and decide on a 3-point scale whether the tweet conveys a positive, negative, or neutral sentiment.

Subtask B: Subtask B is similar to Subtask A, but takes a given tweet and a topic, classifies a tweet on a binary scale, positive or negative, in regard to the given topic.

Subtask C: The same challenge as Subtask B, but evaluates the tweets on a 5-point scale: {very positive, positive, neutral, negative, very negative}.

Subtask D: Related to Subtask B, this task requires us to estimate the distribution of a set of tweets about a given topic across a binary positive or negative spectrum.

Subtask E: Similar to Subtask D but an extension of Subtask C, this task asks us to estimate the distribution of tweets about a topic across the 5-point scale (described earlier in Subtask C). The official description of the Subtasks may be read on SemEval2016's Task4 website (Preslav Nekov et al., 2016).

Due to the similarities in these subtasks as well as the given data, we will approach Subtask A as a learning task to get familiar with the approaches of sentiment analysis, and then tackle Subtasks B and D together as well as Subtasks C and E together.

2 Approach

Our general approach to solving the problems of Subtasks A through E consists of Preprocessing, Annotating/Feature Abstraction, a processing through Neural Nets.

* This document has been adapted from the instructions for earlier ACL and NAACL proceedings, including those for NAACL-HLT-15 by Matt Post and Adam Lopez, NAACL-HLT-12 by Nizar Habash and William Schuler, NAACL-HLT-10 by Claudia Leacock and Richard Wicen- towski, NAACL-HLT-09 by Joakim Nivre and Noah Smith, for ACL-05 by Hwee Tou Ng and Kemal Oflazer, for ACL-02 by Eugene Charniak and Dekang Lin, and earlier ACL and EACL formats. Those versions were written by several people, including John Chen, Henry S. Thompson and Donald Walker. Additional elements were taken from the formatting instructions of the *International Joint Conference on Artificial Intelligence*. Microsoft Word formatting was added by Alexander Mamishev (Mamishev, 2013).

2.1 Preprocessing

We prepared our data for use in our feature abstraction layer, by substituting all references to Twitter's embedded shortened links (http://t.co…) to "httptco".

Our current preprocessing is vastly minimal compared to other groups (who included various techniques such as correcting potential typos, separating hashtags into their individual words, and expanding abbreviations) we believe there might be relevance in the manner of *how* someone refers to another entity and the abbreviations used that indicate positive, negative, or neutral sentiment (Balikas and Amini, 2016; Räbiger et al., 2016; Yadav, 2016). For example, if a person is happy with Microsoft, they might type Microsoft, utilizing correct capitalization whereas a negative sentiment tweet might ignore proper capitalization or spelling altogether.

After preprocessing we passed our tweets to our tokenization and feature abstraction layer. We assigned the following features to each token in a tweet.

1) The count of each of the following punctuations [".", "..", "…", "?", "!"] are used in the sentence the token is a part of.
2) Whether or not the word in the token is spelled correctly.
3) Whether or not the token is a hashtag.
4) The part of speech of the token.

From this, we then took our tweets and passed them into a convolutional neural network to learn to predict sentiment given the above feautres.

In our final project, before passing our tweets to our convolutional neural net, we will break down tweets that contain conjunctions, believing that depending on the conjunctions, either the LHS or RHS might hold more weight toward predicting the positivity/negativity of a tweet's sentiment. Then all tweets lacking conjunctions and the tweet "halves" created from the above "splitting" on conjunctions will be passed into a second neural net, where we will train on the aforementioned feature set.

We believe our approach on specifically splitting tweets up over conjunctions is relatively unique to how teams approach this problem. We are aware that is has been used in some tools, such as VADER but haven't seen much explicitly done over training on tweets with contractions (Hutto and Gilbert, 2015).

Presently, we train each subtask using the appropriate amount of classifiers as dependent upon the subtask. However, we don't yet accurately weight the topic for sentiment analysis.

After training our model, we used the DevTest data to see how well our features could predict sentiments and estimate the distribution of positive and negative tweets.

3 Preliminary Results

3.1 Subtask A

By using the F_1^{PN} metric defined by Nakov *et alii* and using their provided script to score our result on the 2016 Subtask A devtest data, we achieved a score of 0.3296. Note that higher is better (Nakov *et al.,* 2016).

3.2 Subtask B

By using the ρ^{PN} metric defined by Nakov *et alii* and using their provided script to score our result on the 2016 Subtask BD devtest data, we achieved a score of 0.512. Note that higher is better and baseline should be around 0.5 (Nakov *et al.,* 2016).

3.3 Subtask C

By using the MAE^{M} metric defined by Nakov *et alii* and using their provided script to score our result on the 2016 Subtask CE devtest data, we achieved a score of 1.28. Note that lower is better (Nakov *et al.,* 2016).

3.4 Subtask D

By using the KLD metric defined by Nakov *et alii* and using their provided script to score our result on the 2016 Subtask BD devtest data, we achieved a score of 0.408. Here, the lower the score the better (Nakov *et al.,* 2016).

3.5 Subtask E

By using the EMD metric defined by Nakov *et alii* and using their provided script to score our result on the 2016 Subtask CE devtest data, we achieved a score of 0.378. Note that lower is better (Nakov *et al.,* 2016).

4 Timeline

Week of…	Task
10/22	Implement conjunction separation and evaluation
10/29	Incorporate a word-level sentiment tagger as a feature.
10/29	Strip the "@" from mentions in tweets, to increase spellchecker accuracy
11/5	Identify and implement new areas of improvements in text-preprocessing.
11/5	Identify and implement other grammar-based features to use.
11/12	Evaluate current features and heuristics, prune as necessary.
11/19	Write final report
11/27	Submit Proposal

Acknowledgments

Evan Akers wrote the spell-checking feature extraction function.

Kyle Arens developed the data parsing algorithm to break down input data into data structures consumed internally by our algorithms. He extended this into creating the "data flow" pipeline from reading the input files to creating the neural net data objects. This included reading the input data, performing the text preprocessing on that data, and transforming into the output data format. Kyle wrote the word indexing function that took the list of unique words and the counts from the input set and transformed that into a list of significant words and their corresponding indices for use in our vector graphs consumed by the neural net. Next, he wrote the word to vector function which takes the previously mentioned list of significant words, all of the input tweets, and list of features and abstracts them into three multi-dimensional graphs that match the input data structure the neural net requires. Kyle worked with Joe to create the conjunction net (not used in our proof of concept, but will be included for final submission) and verified the mathematical algorithms within the neural net. Kyle improved code-readability by refactoring variable names and documenting all code written.

Ryan Benner initially contributed at a higher level to this project, by doing background research on semantic analysis and what other groups have done previously for SemEval-2016 Task 4. After this, working with Joe, Ryan came up with the initial preprocessing and target feature extractions we used in our preliminary results. He was directly responsible for most of the feature implementation annotation on tokens, writing the following algorithms for annotation: hashtag annotation, punctuation annotation, and part-of-speech annotation. These functions contributed to the data-flow pipeline. He also contributed to the data pipeline by filtering out "Not Available" tweets. Ryan created the word count function for creating a list of unique words counts, denoted by their token-form and associated part-of-speech. After this, Ryan wrote the project proposal document. Ryan also obtained the result data from running the prediction algorithms and compiled them for use in this report. Early in the development process, Ryan fixed minor bugs that occurred as we constructed the data-flow pipeline.

Joe Hirschfeld wrote various parts of our code base. First, he authored the scripts to download the data from Twitter and filter it according to the Twitter terms of service to create both algorithm- and score-ready versions of all the data we used for training and dev-testing. Next, he authored both our conjunction-engine related neural net as well as the feature-extraction and sentiment-analysis neural net. As part of the sentiment-analysis neural net, Joe took the data parsed and massaged by Kyle and Ryan's code, and transformed them into various vectors and matrices for use by the sentiment engine. During the neural net development process, he worked with Kyle on the math-related algorithms of the neural net and afterward, spent his time tuning the various hyper-parameters on the sentiment neural net. During the development cycle, Joe took time to clean up the code database, doing necessary refactors to keep the code clean and maintainable.

References 1,7,10,13

C.J. Hutto and E.E. Gilbert. 2014. VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. In *Eighth International Conference on Weblogs and Social Media (ICWSM-14)*, Ann Arbor, MI.

Georgios Balikas and Massih-Reza Amini. 2016. TwiSE at SemEval-2016 Task 4: Twitter Sentiment Classification. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, San Diego, US.

Preslav Nakov, Alan Ritter, Sara Rosenthal, Fabrizio Sebastiani, and Veselin Stoyanov. 2016. SemEval-2016 Task 4: Sentiment Analysis in Twitter. Association for Computational Linguistics. In *Proceedings of the 10th*

International Workshop on Semantic Evaluation (SemEval-2016), San Diego, US.

Stefan Räbiger, Mishal Kazmi, Yücel Saygin, Peter Schüller, and Myra Spiliopoulou. 2016. SteM at SemEval-2016 Task 4: Applying Active Learning to Improve Sentiment Classification. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, San Diego, US.

Vikrant Yadav. 2016. thecerealkiller at SemEval-2016 Task 4: Deep Learning based System for Classifying Sentiment of Tweets on Two Point Scale. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, San Diego, US.