bwbaugh : Hierarchical sentiment analysis with partial self-training

Wesley Baugh
Department of Computer Science
University of North Texas
brianbaugh@my.unt.edu

Abstract

Using labeled Twitter training data from SemEval-2013, we train both a subjectivity classifier and a polarity classifier separately, and then combine the two into a single hierarchical classifier. Using additional unlabeled data that is believed to contain sentiment, we allow the polarity classifier to continue learning using self-training. The resulting system is capable of classifying a document as *neutral*, *positive*, or *negative* with an overall accuracy of 61.2% using our hierarchical Naive Bayes classifier.[1]

1 Introduction

Many people use social networks, such as Twitter, to connect and communicate with others. Users of social networks often share their experiences, such as watching a recent movie or tv show, reading a book, or a newly tried product or service. In addition, social networks provide an avenue for discussion of current events, such as politics. Many people and companies are often concerned with how others perceive their product—which is sometimes themselves, as is the case for politicians—or their service. By understanding and reacting to what the consumer is thinking, they can attempt to maximize their good press as well as to help minimize the bad. It would therefore be useful to use the information users of social networks share to perform sentiment analysis in order to understand how people perceive targets of interest.

In general, sentiment analysis often involves the use of machine learning, especially Naive Bayes, SVM, and MaxEnt classifiers [Jose]. Features general include n-grams and POS tags [Go et al., 2009; Pak and Paroubek, 2010; Jose], as well as sentiment lexicons [Jose]. Go et al. [2009] achieved around 82.5% accuracy for positive-negative polarity detection, Jose achieved around 76% accuracy for subjective-objective classification, and Pak and Paroubek [2010] achieved around 70% accuracy for a combined subjectivity-polarity classifier.

While determining whether a document known to be subjective is positive or negative (polarity detection) is relatively easy, a currently more difficult task in sentiment analysis is identifying whether a document is subjective or objective (subjectivity analysis). Many approaches simply ignore the objective class [Go et al., 2009], which does not work for real world problems as there are a substantial amount of documents that are either partially or wholly objective [Koppel and Schler, 2006].

Many previous methods focus on either subjectivity analysis or polarity detection. Our method incorporates both subtasks into a single overall system in order to perform sentiment analysis.

2 Background

The sentiment analysis in Twitter task of SemEval-2013 [Wilson et al., 2013] provides 9,864 labeled tweets from Twitter to be used as a training dataset. Each instance is labeled as either `positive`, `negative`, or `neutral`, and was annotated through Amazon's Mechanical Turk. The terms of service for Twitter puts restrictions on the

[1] A working demo of the system will be available for a short time at: `http://infertweet.bwbaugh.com`

*Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 539–542, Atlanta, Georgia, June 14-15, 2013. ©2013 Association for Computational Linguistics

type of data that may be re-released, therefore participants SemEval-2013 Task 2 participants were required to download tweets directly from Twitter. Due to deleted or otherwise unavailable tweets, this system was only able to download approximately 8,750 training instances. Additionally, a development dataset was provided with 1,654 labeled tweets, of which 340 are negative, 739 are neutral, and 575 are positive. The provided test set consisted of 3,813 instances, of which 601 are negative, 1640 are neutral, and 1572 are positive.

In related work, Go et al. [2009] generated an automatically labeled noisy gold standard by searching for tweets that contained one of several emoticons[2] (e.g. :) or :() that were mapped to either the positive or negative class depending on the type of emoticon in the text. This system also collected approximately one million tweets using emoticons as a keyword search for matching, however the data remained unlabeled. Though these tweets are unlabeled, they are presumed to be subjective—either positive or negative but not neutral—because of the intuitive association of emoticons with sentiment.

3 Approach

The system uses a custom implementation of Multinomial Naive Bayes as the classifier.[3] We create a hierarchical classifier, which in this case consists of two binary classifiers. The first-level is the subjectivity classifier, which can output objective (neutral) or subjective. If the output of the first level is subjective, then the second-level polarity classifier decides if the instance is positive or negative.

Both classifiers (subjective and polarity) are trained on approximately 8,750 training instances, which come from the released SemEval-2013 training dataset. The subjective classifier is not given any additional training data. The system then uses its current model to classify approximately one million *unlabeled* tweets that are believed to be subjective. The unlabeled tweets were classified one at a time. If the system classified the tweet as subjective, it was used to train the polarity classifier only if the confidence in the predicted label was greater than 0.8. We stopped the system after approximately 910k total training instances were used.

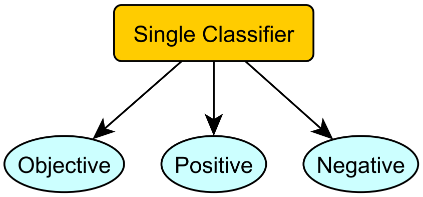

Figure 1: A single multinomial classifier, which can output any class label.

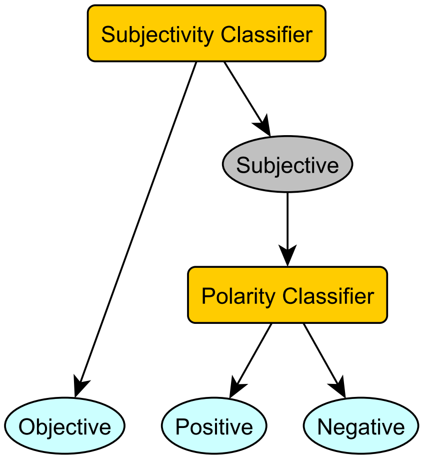

Figure 2: A hierarchical classifier, which in this case consists of two binary classifiers. The first level is a subjectivity classifier, with an output of either *subjective* or *neutral*. The second level is a sentiment polarity classifier, with an output of either *positive* or *negative*.

The core features extracted are unigrams and bigrams. Bigrams had an additional __start__ and __end__ token at the beginning and end of the full text of the training instance.

As part of a preprocessing step, we attempted to find URLs in the text and replace them with a special __URL__ token. We shortened characters repeated more than twice, such that "haaaaaaate" would become "haate". We attempted to find dates in the text and replace them with a special __DATE__ token.

[2] The term *emoticon* comes from a blending of the words "emotion" and "icon".

[3] The machine learning components (Multinomial Naive Bayes) were written for this system as a Python library, and will be available on GitHub: https://github.com/bwbaugh/infer. That toolkit was then used as a foundation for writing the code for the system, which will also be available on GitHub: https://github.com/bwbaugh/infertweet.

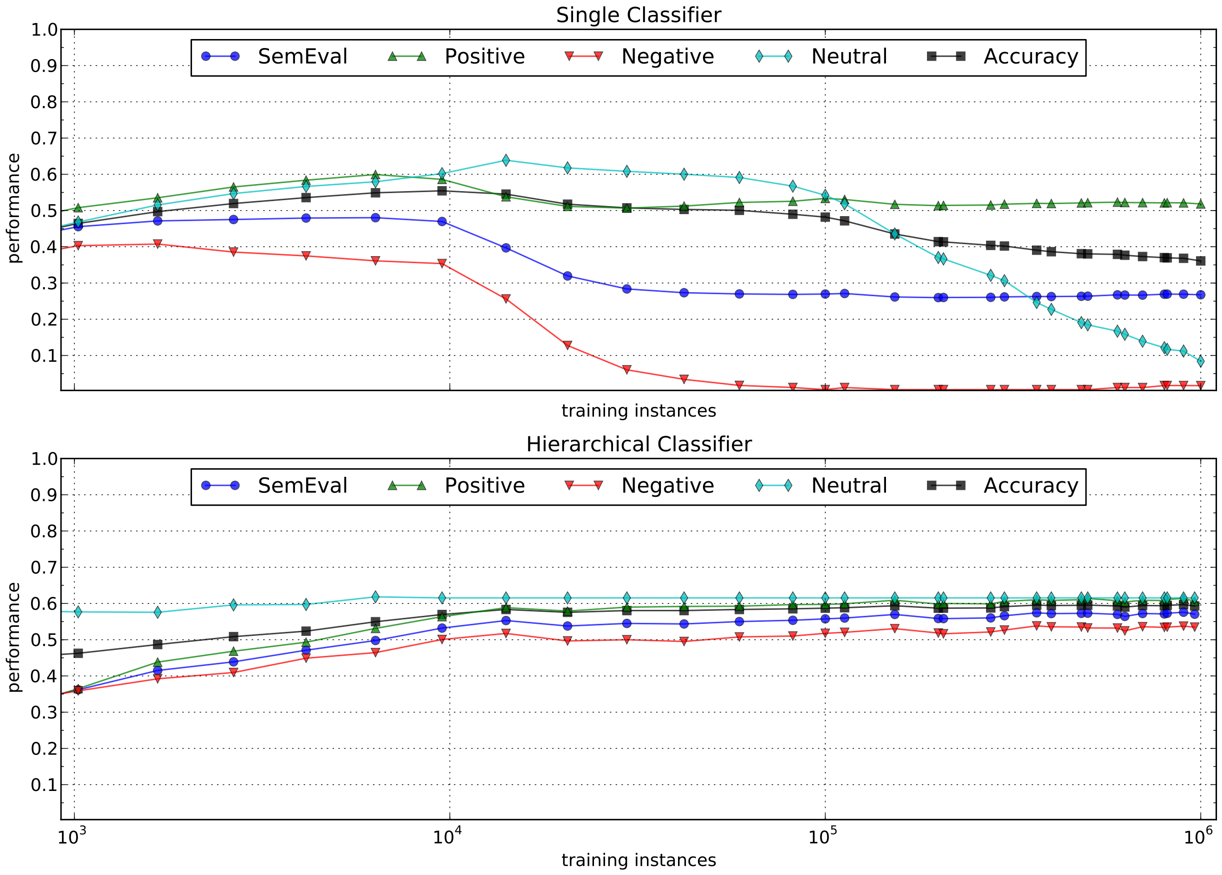

Figure 3: Performance of the single (non-hierarchical) and hierarchical classifiers on the development set vs. the number of training instances. The performance metric for *positive*, *negative* and *neutral* is F-measure, *SemEval* is the simple average of the positive and negative performance, and *accuracy* is the overall number of correct instances. The first 8,750 instances are labeled, while the rest are unlabeled instances that were added using self-training.

4 Experiments

4.1 Design

The system was incrementally trained one tweet at a time, with the performance checked every so often by using the current model to classify the development set instances. Once all of the labeled training data had been used, the subjectivity classifier was given no additional training instances, and the remainder of the subjectively charged unlabeled data was used to train the polarity classifier.

Variables experimented on included: extracting n-grams up to size 4 and trying all combinations; mapping Twitter usernames to a special token; mapping substrings recognized as a date to a special token; combining a negation token such as “not” to the following token; deleting characters repeated more than twice; mapping numbers to a special token; counting exclamation points; the confidence threshold above which the predicted label for an unlabeled instance would be used for training.

In addition to collecting unlabeled data using emoticon keywords, we also experimented with using sentences from Wikipedia as neutrally labeled text, as well as using a random subsample of all English-language tweets from the Twitter public stream as a source of unlabeled data for any class.

We also tried using a single non-hierarchical classifier using each source of unlabeled data.

4.2 Results

4.2.1 SemEval-2013 development set

Using additional unlabeled data with the single multinomial classifier always resulted in overall degraded performance.

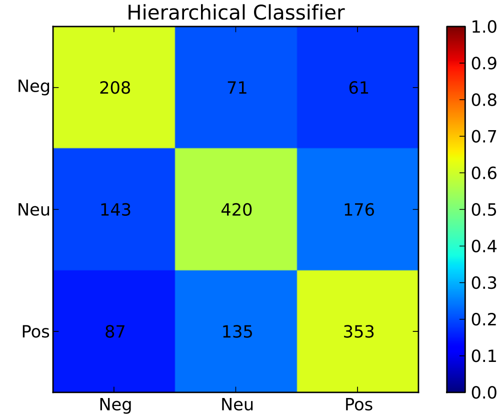

Figure 4: The confusion matrix on the development set produced after training on a total of approximately 970k training instances. Rows are the true labels while columns are the predicted labels.

4.2.2 SemEval-2013 test set

gs \ pred	negative	neutral	positive
negative	324	203	74
neutral	196	1168	276
positive	233	498	841

Table 1: Confusion matrix (hierarchical)

class	prec	recall	fscore
negative	0.4303	0.5391	0.4786
neutral	0.6249	0.7122	0.6657
positive	0.7061	0.5350	0.6088

Table 2: Performance (hierarchical)

The average F-score of the `positive` and `negative` classes is 0.5437, which is the main evaluation metric used by SemEval-2013 Task 2. The overall accuracy is 61.2%.

4.2.3 Discussion

By using a hierarchical classifier, we are able to prevent degradation of the performance of the classifier on neutrally labeled instances by only applying additional training data to the polarity classifier.

The use of additional unlabeled data results in an increase in performance for the hierarchical classifier as seen in Figure 3. However, the increase in performance comes with an exponential increase in the number of unlabeled instances. Using appropriate feature selection for online algorithms, such as feature hashing, a system like this could train indefinitely on additional data from a Twitter stream without running out of memory.

The system's lack of high-quality sources for additional `objective-OR-neutral` data—either labeled or unlabeled—appears to be our biggest obstacle to increasing performance at this time. The poor performance of the single multinomial classifier when given additional unlabeled data can also likely be attributed to this reason. Identifying additional high-quality sources of neutral data would likely go a long way towards improving the overall system performance. Active learning approaches could also be applied with the goal of improving the subjectivity classifier.

5 Conclusion

Using a hierarchical classifier comprised of two Naive Bayes classifiers, we are able to improve the performance of polarity detection with the addition of unlabeled data in an online setting by isolating the subjectivity classifier.

References

Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, pages 1–12, 2009.

Anthony K Jose. Twitter sentiment analysis.

Moshe Koppel and Jonathan Schler. The importance of neutral examples for learning sentiment. *Computational Intelligence*, 22(2):100–109, 2006.

Alexander Pak and Patrick Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In *Proceedings of LREC*, volume 2010, 2010.

Theresa Wilson, Zornitsa Kozareva, Preslav Nakov, Alan Ritter, Sara Rosenthal, and Veselin Stoyanov. Semeval-2013 task 2: Sentiment analysis in twitter. In *Proceedings of the 7th International Workshop on Semantic Evaluation*, 2013.