Classifying Tweets Based on Climate Change Stance

Natural Language Processing

1 Introduction

Climate change is real yet it has become a polarizing topic. We, along with many others, believe that the language used to describe the issue is to blame. We believe climate change needs a new lexicon, one that replaces phrases that unintentionally trigger resistance or confusion with phrases that elicit support and unity. In order to create this lexicon, we need the ability to first identify and classify text by whether or not it indicates belief in human-caused climate change.

The biggest challenge we face in accomplishing this is the lack of a sufficiently large and good dataset to train a classifier. Our project addresses this challenge by using both labeled and unlabeled data to perform semi-supervised classification on tweets with the aim that this model can be used for further analysis of climate change language and natural language processing in the creation of the new lexicon.

We use Twitter as the main source of data for our classification task. Twitter has diverse communities with differing opinions about climate change, and contains data that is easily accessible to researchers making it an ideal data source.

To classify the tweets, we implemented three semi-supervised multi-class classification algorithms: self training, semi-supervised SVM, and Multinomial Naive Bayes with EM (MNB-EM). Though these methods showed promise, the supervised unigram Multinomial Naive Bayes model ultimately had the best performance on our test set. In this paper, we present these algorithms and analyze our results.

2 Related Work

Tweet classification has mainly been performed using supervised learning algorithms. For instance, Mohammad et al. [1] classified tweets to determine their stance on various categories including climate change. Noteworthy from their paper is the classification of climate change tweets had the worst performance of all the categories they classified: atheism, feminism, Hillary, abortion, and climate change. It also had the fewest data points of all the categories and they were highly imbalanced with 53.7% agreeing climate change is a real concern, 3.8% against and 42.5% neutral. This class imbalance is a similar problem in the labeled datasets we collected, and indicates a general difficulty in obtainining labeled climate change sentiment data.

There have been numerous approaches to implementing semi-supervised SVMs (S3VM). One of which is by Ogawa et al, who proposed an infinitesimal annealing algorithm that addresses the trade-off between the resolution of annealing steps (how much weigh to put on the unlabeled data) and the computation cost during training in most S3VM algorithms [2]. However, we did not use their implementation as we were able to train in a reasonable amount of time using a Quasi-Newton implementation by Gieseke et al. [3] which is one of the mostly widely accepted S3VM implementations. It is a binary classification algorithm that we adapted to be multi-class for our project.

Tsuruok et al. implemented Naive Bayes via EM algorithm with a class distribution constraint introduced during the iteration process of the EM algorithm. This constraint keeps the class distribution of unlabeled data consistent with the class distribution estimated from labeled data, preventing the EM algorithm from converging into an undesirable state [4]. Though this performed well in their experiments, we chose not to add a class constraint to our algorithm because our labeled dataset was imbalanced and our unlabeled dataset did not necessarily reflect the same class distribution. Another variant of the EM with Naive Bayes algorithm was implemented by Nigam et al [5] where they added a weighting factor to modulate the contribution of the unlabeled data. We incorporated this approach in our implementation.

3 Dataset and Features

Class	Description and (Example)	% of Dataset
1	the tweet supports the belief of man-made climate change ("climate change, a factor in texas floods, largely ignored")	57.7%
0	the tweet neither supports nor refutes the belief of man-made climate change ("global warming on mars....")	29%
-1	he tweet does believe in man-made climate change ("scientists were attempting the same global warming scam 60 years ago")	13.3%

Table 1: Descriptions of each class, example data point and dataset breakdown by class

For the unlabeled data, we used the twitter API via TweePy [7] to obtain climate change related tweets by querying hashtags that have been shown in previous research [8] to contain climate change related tweets. These were: globalwarminghoax, globalwarmingisahoax, climatechange, climatechangefraud. Preprocessing the tweets included: removing tweets that were clearly not related to climate change, tweets that only contained links and or emojis, and also removing all punctuation, stop words, and emojis from the tweets then finally converting all the tweets to lowercase.

For our unigram models, our feature vector was a document-term frequency matrix with dimensions equal to the vocabulary size (3,007 words). For the bigram models, our feature vectors were pairs of document-terms in frequency matrices (2,988 word-pairs). In both cases, we only used words/word-pairs that appeared in over 5 tweets.

4 Methods

4.0.1 Multinomial Naive Bayes Unigram

As a baseline, we built a multiclass Multinomial Naive Bayes model (MNB), which has been shown to have good performance in text classification problems and is also robust to concept drift, which is key, because the language used to talk about climate change is continually changing.

For a dataset with $\{(x^{(i)}, y^{(i)}); i = 1...n\}$ where $x^{(i)} = (x_1^{(i)}, x_2^{(i)}, ..., x_d^{(i)})$ and $\{y^{(i)} \in [-1, 0, 1]\}$ The likelihood for the three-class MNB takes the form:

$$\mathcal{L}(\phi_y, \phi_{k|y}) = \prod_{i=1}^{n} p(x^{(i)}, y^{(i)}) = \prod_{i=1}^{n} \left(\prod_{j=1}^{d} p(x_j^{(i)}|y^{(i)}; \phi_{k|y}) \right) p(y^{(i)}; \phi_y)$$

where $\phi_y = j$ is the multinomial probability that example y belongs to class j and $\phi_{k|y=j}$ is the probability that word k is present in messages of class j.
Solving for the maximum likelihood parameters with Laplace smoothing yields:

$$\phi_{y=j} = \frac{1 + \sum_{i=1}^{n} \mathbb{1}(y^{(i)} = j)}{n + k} \qquad \phi_{k|y=j} = \frac{1 + \sum_{i=1}^{n} \sum_{j=1}^{d_i} \mathbb{1}(x_j^{(i)} = k|y^{(i)} = j)}{|V| + \sum_{i=1}^{n} \{y^{(i)} = j\} d_i}$$

where $\phi_y = j$ and $\phi_{k|y=j}$ are as described above, and $|V|$ is the size of the vocabulary.

4.0.2 Multinomial Naive Bayes Bigram

Instead of considering each word of each tweet individually and independently, we also consider counts of bigrams of words as features for MNB. For each example tweet $x^{(i)}$, $z_j^{(i)} = (x_j^{(i)}, x_{j+1}^{(i)}), j = 1, ..., d_i - 1$, where d_i is the number of words in tweet i. The bigrams $z^{(i)}$ are treated independently, and take the place of the examples $x^{(i)}$ from unigram Naive Bayes, allowing for maximum likelihood parameter estimation.

4.0.3 Self Training

We implemented self training, a semi-supervised algorithm that incorporates the unlabeled data using the methodology described below.

1. Train unigram Naive Bayes model on training set
2. Use the trained model to predict labels of unlabeled data
3. Assign predicted labels that have at least 50% probability as true labels and add them as new training examples
4. Repeat the training of the Naive Bayes model on the updated training set until no more update are being made to the training set of the max number of iterations is reached.
5. Make predictions using the the last trained model

4.0.4 Multi Class Semi-Supervised SVM

Semi-supervised SVM (S3VM) or Transductive SVM (TSVM) is a semi-supervised learning algorithm that inherits the large margin concepts of SVMs.[2]. The main task is to find an optimal class label for the unlabeled data and then using them to find the separating hyperplanes that maximize the margin. We extended the implementation of Quasi-Newton Semi-SVM implementation (QN-S^3VM) by Gieseke et al. [3], shown to the right, from a binary classification implementation to a one-vs-one multi-class implementation. We train a model (with RBF kernel) for each of the class combinations: -1 and 0, -1 and 1, and 0 and 1. During prediction, we assign the class that was chosen the most by the three models. This approach was chosen over a one-vs-all implementation because it is much less sensitive to imbalanced data sets like ours.

In the algorithm at the right, equations (7) and (8) are defined in the source paper[3], λ is a regularization parameter, λ' is a cost parameter that determines influence of unlabeled data, c_k refers to candidate solutions, β_k is the step length, and the sequence α_i is an annealing sequence which is used to increase the influence of the unlabeled data on the model.

Algorithm 1 QN-S^3VM

Require: A labeled training set $T_l = \{(\mathbf{x}_1, y'_1), \ldots, (\mathbf{x}_l, y'_l)\}$, an unlabeled training set $T_u = \{\mathbf{x}_{l+1}, \ldots, \mathbf{x}_n\}$, model parameters λ', λ, an initial (positive definite) inverse Hessian approximation $\mathbf{H}_0$, and a sequence $0 < \alpha_1 < \ldots < \alpha_\tau$.

Initialize $\mathbf{c}_0$ via supervised model.
for $i = 1$ **to** τ **do**
  $k = 0$
  while termination criteria not fulfilled **do**
    Compute search direction $\mathbf{p}_k$ via (7)
    Update $\mathbf{c}_{k+1} = \mathbf{c}_k + \beta_k \mathbf{p}_k$
    Update $\mathbf{H}_{k+1}$ via (8)
    $k = k + 1$
  end while
  $\mathbf{c}_0 = \mathbf{c}_k$
end for

QN-S^3VM Algorithm From Gieske et al.

4.0.5 Semisupervised Multinomial Naive Bayes with Expectation Maximization

The final model we implemented was training a multi class Multinomial Naive Bayes with Expectation Maximization (MNB-EM) classifier.

We can treat the likelihood of the labeled and unlabeled examples as a product of the likelihoods for the labeled and unlabeled data. In addition to labeled examples $\{(x^{(i)}, y^{(i)}); i = 1...n\}$ described in section **4.0.1** we also have unlabeled data $\{(x^{(l)\prime}); l = 1...k\}$ where $x^{(l)\prime} = (x_1^{(l)\prime}, x_2^{(l)\prime}, ..., x_d^{(l)\prime}{}_l)$. We weight the labeled training data loss with α, a hyperparameter we tune to determine how much we "trust" the labeled data more than the unlabeled data. The combined log likelihood takes the following form:

$$\ell(\phi_y, \phi_{k|y}) = \alpha \sum_{i=1}^{n} \sum_{j=1}^{d} \log p(x_j^{(i)} | y^{(i)}; \phi_{k|y}) + \sum_{i=1}^{n} \log p(y^{(i)}; \phi_y) + \sum_{l=1}^{k} \log \sum_{y^{(l)\prime}} Q_l(y^{(l)}) \frac{\prod_{m=1}^{d} p(x_m^{(l)\prime} | y^{(l)}; \phi_{k|y}) p(y^{(l)\prime}; \phi_y)}{Q_l(y^{(l)})}$$

where $Q_l(y^{(l)}) = p(y^{(l)} | x^{(l)}; \phi_y, \phi_{k|y})$

In the E-step of the model, we estimate

$$Q_l^{(t)}(y^{(l)\prime} = j) = p(y^{(l)\prime} = j | x^{(l)\prime}; \theta^{(t)}) = \frac{\prod_{m=1}^{d} p(x_m^{(l)\prime} | y^{(l)\prime} = j; \phi_{k|y}) p(y^{(l)\prime} = j; \phi_y)}{\sum_{y^{(l)\prime}} \prod_{m=1}^{d} p(x_m^{(l)\prime} | y^{(l)\prime}; \phi_{k|y}) p(y^{(l)\prime}; \phi_y)}$$

Next, in the M-step of the model, we estimate the parameters $\phi_y, \phi_{k|y}$. Note that $p(x_m^{(i)} = k | y^{(i)} = c; \phi_{k|y=c}) = \phi_{k|y=c}$ and $p(y^{(i)}; \phi_{y=c}) = \phi_{y=c}$ for both the labeled and unlabeled data. When estimating the parameters, we have the additional Lagrange multiplier constraints that $\sum_{k=1}^{|V|} \phi_{k|y} = 1$ and $\sum_{c=-1}^{1} \phi_{y=c} = 1$.

With Laplace smoothing, we obtain the following M-update parameters:

$$\phi_{k|y=c} = \frac{1 + \alpha \sum_{i=1}^{n} \sum_{j=1}^{d} 1(x_j^{(i)} = k \cap y^{(i)} = c) + \sum_{l=1}^{k} Q_l(y^{(l)\prime} = c) \sum_{m=1}^{d} 1(x_m^{(l)\prime} = k)}{|V| + \alpha \sum_{i=1}^{n} \sum_{j=1}^{d} 1(y^{(i)} = c) + \sum_{l=1}^{k} Q_l(y^{(l)\prime} = c) \sum_{m=1}^{d} 1}$$

$$\phi_{y=c} = \frac{1 + \alpha \sum_{i=1}^{n} 1(y^{(i)} = c) + \sum_{l=1}^{k} Q_l(y^{(l)'} = c)}{C + \alpha n + k}$$

where $C = 3$ is the number of categories for the model.

5 Experiments, Results and Discussion

5.1 Results and Discussion

Our goal was to accurately predict what class a tweet falls into, so we use accuracy (i.e. fraction of correctly classified examples) on the test set as our evaluation metric. Our models obtained the accuracies shown in Table 2.

Model	Train	Validation	Test
Unigram Naive Bayes (NB)	76.8%	67.6%	66.4%
Bigram Naive Bayes (NB)	72.8%	62.5%	60.6%
MNB with Self Training	66.7%	60.9%	61.6%
Semi-Supervised SVM	59.2%	59.0%	55.2%
MNB-EM	81.9%	65.2%	63.9%

Table 2: Performance of each model on classifying training, validation, and test set data.

To better understand our models' performance, we created confusion matrices based on their predictions on the test set.

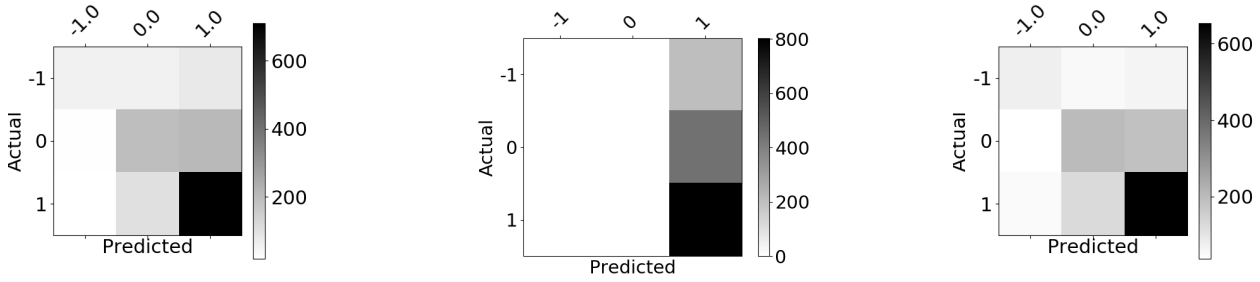

Figure 1: MNB, S3VM, and MNB-EM Test set confusion matrices

Overall, the accuracies reported for all models on the training, validation, and test sets are relatively low compared to the success of text-based classifiers for tasks like spam classification, where classification models can routinely achieve >90% accuracy. This was despite tuning hyperparameters for the semi-supervised models (methods described in section **5.2**), and contrary to expectations that incorporating unlabeled data and using semisupervised learning would improve our results.

Overall, all NB models suffered from overfitting on the training data, with training accuracies significantly higher than validation and test accuracies. MNB and MNB-EM models were able differentiate between the three classes with moderate success. However, the S3VM model predicted the majority class for almost all examples. This is because unlike class-weighted SVM, S3VM does not assign higher misclassification penalties to training instances of the minority class and thus, is susceptible to predicting only the majority class.

Since the labeled and unlabeled tweets are over 3 years different in age, there may be systematic differences in how language related to "climate change" and "global warming" has changed over time. These differences may have resulted in poorer performance of the semisupervised classification models, relative to supervised classification when evaluating the models on the validation and test sets (which were drawn from the same pool of tweets as the labeled set).

There is also a strong overlap between the words most associated with each class (i.e. the words that signal class membership). We believe this makes it particularly difficult to model the decision boundary separating the classes, and explains why the model is unable to classify tweets with high accuracy. From the unigram MNB model, the words that were most associated with each class were:

-1 (negative belief): climate, global, change, warming, climatechange, globalwarming, believe, like, people, made, science
0 (neutral belief): climate, change, global, warming, warmin, like, change, believe, chang, trump, one
1 (positive belief): climate, change, global, warming, change, believe, trump, people, via, real, us

5.2 Experiments

Probability Threshold During Self Training: During self training, we experimented with different thresholds for the probability that prediction on an unlabeled data point should obtain in order to be added to the training set. Surprisingly, adding points with higher confidence values actually decreased the performance of our model on the validation set. This was because all high probability classifications were for class 1, which was already over represented. Thus, adding these points to our training set was leading to further class imbalance and hurting our model's performance. From this we set our model's threshold to 0.5 which was the highest confidence that still yielded good accuracy.

S3VM Hyperparameter and Model Tuning: For the QN-S^3VM model all hyperparameters used were the default hyperparameters from Gieske et al [3]. We experimented with different λ and λ' values by performing a linear search over possible values but none yielded significantly different results from the default values. We experimented with a linear kernel and RBF kernel for the one-vs-one classifiers, and used the RBF kernel for final model evaluation.

EM Hyperparameter Tuning: α is a hyperparameter that represents the "trust" we have for the unlabeled data compared to the labeled data. We vary this parameter from 1 (assigning equal weight to labeled and unlabeled training data) to 100 (assigning 100-times more weight to labeled data than to unlabeled data). The training accuracy is low for smaller values of α suggesting that differences between the labeled and unlabeled data sets may mistrain the classifier. On the other hand, the largest α values have high training accuracies, but relatively low validation accuracies, suggesting that the model is overfitting to the labeled training data. The results of this hyperparameter tuning are shown in Table 3. Based on this analysis, we found the optimal value of $\alpha = 20$.

Alpha	Train Accuracy	Val Accuracy
1	67.3%	60.7%
5	76.8%	62.3%
20	81.9%	65.2%
50	82.6%	65.1%
100	82.9%	64.5%

Table 3: Hyperparamter tuning of α for MNB-EM model.

Downsampling Analysis: We wanted to know the sensitivity of our model to the amount of labeled data used, as well as whether there is systematic bias between the labeled and unlabeled data which decreases model performance on validation and test data. We can test both of these questions by downsampling the amount of training data used (for both MNB and MNB-EM), while keeping the amount of unlabeled data the same (for MNB-EM). Results shown in Figure 2.

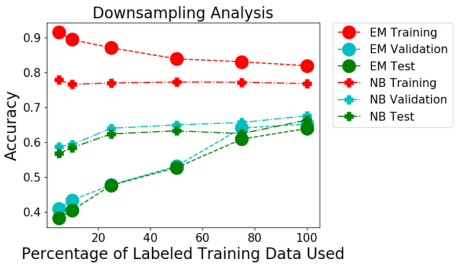

Figure 2: Accuracy as a function of labeled data downsampling, for both MNB-EM and MNB models. MNB data shown with crosses, MNB-EM data with circles.

First, we see the performance of the supervised MNB model (lines with + markers) is relatively consistent even if only 20% of the data were used. This suggests that the labeled data set is already sufficiently large, and performance will not improve by incorporating more of the same data.

Next, note that MNB-EM (circle markers) has higher training accuracy than supervised MNB, suggesting that the unlabeled data helps the model discriminate among training examples. However, the validation and test accuracies are consistently lower for MNB-EM, compared to MNB, showing that the unlabeled data introduces bias that decreases the performance of the model on unseen examples, and that there are systematic differences between the labeled and unlabeled data. The effect is minimized when all the labeled data is used, i.e. when impact of unlabeled data is minimized.

6 Conclusion and Future Work

We set out to classify tweets based on stance on climate change. To improve the results of our supervised classification task, we experimented with three semi-supervised classification models: self training, multi class semi-supervised SVM, multi class Multinomial Naive Bayes with EM. However, the methods we attempted resulted in worse performance than the supervised classification baseline. We believe this is because the unlabeled data differed systematically from the labeled training examples, and in addition that the decision boundaries between the classes are difficult to model.

In the future, we would improve our models in several ways. First, we would improve the MNB-EM algorithm by treating classes as mixtures of sub-classes, each with their own word distributions that may be captured in an MNB model. This method is proposed by Nigam et al [5] and would be a helpful because there are likely many types of people in each class, with each type using language slightly differently. We would also experiment with an auxiliary deep generative model for semi-supervised learning proposed by [9]. Lastly, because we believe the 2016 labeled and 2019 unlabeled data sets may have systematic differences, we would want to generate a small, labeled dataset from 2019 tweets, to use in semisupervised classification.