HW1: Classification

Alex Lin
alexanderlin01@college.harvard.edu

Melissa Yu
melissayu@college.harvard.edu

February 1, 2018

1 Introduction

Sentiment classification is an important problem that has been well-studied in the realm of Natural Language Processing. Given a sentence (i.e. a collection of ordered words), we wish to determine if the writer is expressing a positive sentiment or a negative one. Over the past few years, several models have been developed to tackle this problem. In this study, we consider four representative models - Multivariate Naive Bayes [1], Logistic Regression [2], Continuous Bag-of-Words [3], and Convolutional Neural Network [4]. We also consider an extension in which we combine [1] and [4]. Using the Stanford Sentiment Treebank dataset, we implement these four models and extensively tune their hyper-parameters to try to determine which one performs the best. Our results surprisingly show that the simplest model seems to achieve the highest accuracy, with Multivariate Naive Bayes being the most successful.

2 Problem Description

As stated in the previous section, given a sentence s with words $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{|s|}$, we wish to determine if s has positive sentiment $y = 1$ or negative sentiment $y = 2$. All words belong to a vocabulary $\mathcal{V}$. Here, we assume that each $\boldsymbol{x}_i$ is a real-valued vector that corresponds to word i.

It is common for $\boldsymbol{x}_i$ to be a one-hot encoding with size $|\mathcal{V}|$. However, $\boldsymbol{x}_i$ can also take on more interesting forms such as a continuous vector as trained by models such as `word2vec`. We will see both types of encodings for $\boldsymbol{x}_i$ in this study.

3 Model and Algorithms

We trained the four different models specified in the instructions. These included (1) a Multivariate Naive Bayes unigram classifier, (2) a logistic regression model over word types, (3) a continuous bag-of-words neural network with embeddings, and (4) a convolutional neural network. We also tried a combination model in which we combined (1) and (4) in an attempt to achieve better accuracy.

3.1 Multivariate Naive Bayes

3.2 Logistic Regression

The logistic regression model involved takes in a vector representation of a sentence $\boldsymbol{x}$, applies a linear function with weights $\boldsymbol{w}$ and bias b, and uses a sigmoid transformation to return the probability of that sentence having positive sentiment p. That is,

$$p = \sigma(\boldsymbol{w} \cdot \boldsymbol{x} + b)$$

The form of $\boldsymbol{x}$ we choose here is simply a binary vector with size $|\mathcal{V}|$ that has a value of 1 if the associated word appears in the sentence and has a value of 0 otherwise.

We train the weights and bias using the Adam optimizer, which converges to lower losses more quickly than stochastic gradient descent. Our chosen learning rate is 0.01. Using training batches of size 100, we optimize the loss function as the negative log-likelihood of the parameters $L(\boldsymbol{w}, b)$. For this model, we primarily experiment with varying a regularization penalty λ applied to the weights $\boldsymbol{w}$, which changes the overall loss function to

$$L(\boldsymbol{w}, b) + \lambda \|\boldsymbol{w}\|_2$$

3.3 Continuous Bag-of-Words

3.4 Convolutional Neural Network

For the convolutional neural network, we first utilize the classic `word2vec` transformation to convert between words in sentences and their continuous vector representations. For each word in a sentence, we have a 300-by-1 vector. Then, we apply three sets of convolutions with kernel sizes of $h = 3, 4, 5$ and 100 out-channels each. The resultant values for each kernel size and out-channel are first put through a rectified linear activation function and then pooled over time. Thus, each original sentence becomes mapped to a 300-feature vector. This vector is then passed through an affine transformation with weights $\boldsymbol{w}$ and bias b, giving us a single value that we pass through a sigmoid activation to generate p, the probability of the original sentence having positive sentiment.

In the spirit of Kim's paper, we also utilize regularization techniques. First, we use dropout with probability 0.5 in the final layer to mitigate co-adaptation of convolutional weight vectors. Next, we also restrict all weights to have a 2-norm of at most 3. This prevents large weights and overfitting. In our experiments, we mainly vary across different optimizers (Adam, SGD, Adadelta), different regularization parameters, and adding in different convolutions.

3.5 Multivariate Naive Bayes + Convolutional Neural Network

This model is simply an aggregation of Multivariate Naive Bayes and the Convolutional Neural Network. Noting that both models give us discriminative probability distributions $p(y|\boldsymbol{x})$ over the sentiment of a sentence, we thought of the following natural way to combine the two predictions: For a given sentence, if the two models agree on a classification, then we simply output that consensus. If they differ, then we choose the model that is "more sure" of its answer; in other words, we output the proposed class of the model who has a higher probability estimate $p(y|\boldsymbol{x})$. We hoped that this aggregation of the two models could improve the overall accuracy.

4 Experiments

We performed several experiments to tune the hyper-parameters of our models. Perhaps surprisingly, Multivariate Naive Bayes seemed to perform the best on this dataset. In-depth explanations of our tuning procedure and associated results can be found in this section.

Model	Acc.
Baseline (Single Class)	0.538
Multivariate Naive Bayes	
Logistic Regression	0.797
Continuous Bag-of-Words	
Convolutional Neural Network	0.802
Multivariate Naive Bayes + Convolutional Neural Network	0.805

Table 1: Classification accuracies of our models.

4.1 Multivariate Naive Bayes

4.2 Logistic Regression

4.3 Continuous Bag-of-Words

4.4 Convolutional Neural Network

4.5 Multivariate Naive Bayes + Convolutional Neural Network

5 Conclusion

End the write-up with a very short recap of the main experiments and the main results. Describe any challenges you may have faced, and what could have been improved in the model.