

Homework 1: Semi-supervised Text Classification

Seokchan Ahn

Department of Computer Science

University of California, Irvine

seokchaa@uci.edu

kaggle username: Seokchan Ahn

Abstract

The objective for this assignment is to train a classifier that identifies the presidential candidate from their speech segments of 140 characters or less. We are given about 4k labeled data and 43k unlabeled data. First I improved the default supervised classifier by 3.62% using hyper-parameter tuning, lemmatization, and tokenization. Then, I tried a semi-supervised approach known as self-training, which gradually expands the labeled data using the previous model. However, the accuracy has decreased as the amount of expanded data increases, because the original model's accuracy was less than 0.5, generating more falsely labeled data.

1 Introduction

Semi-supervised learning is an approach that combines a small amount of labeled data with a large amount of unlabeled data during training. The acquisition of labeled data often requires a lot of time and cost, whereas acquisition of unlabeled data is relatively inexpensive. In such situations, semi-supervised learning can be a cost-effective and practical way.

2 Backgrounds

2.1 Logistic Regression Classifier

Logistic Regression classifier is the baseline supervised machine learning algorithm for classification. It also has a very close relationship with neural networks since a neural network can be viewed as a series of logistic regression classifiers stacked on top of each other (Jurafsky and Martin, 2009). To prevent overfitting, regularization can be applied to objective functions.

2.2 Bag of Words

The bag-of-words model is a model that represents a text as the bag of words it has, disregarding grammar and word order but keeping multiplicity (Harris, 1954).

3 Experiments

3.1 Supervised: Improve the Basic Classifier

3.1.1 Hyper-parameter Search

L-BFGS solver with l2 normalization is used for the logistic regression classifiers, because they are known as fast and perform well with small data.¹ The strength of the regularization can be tuned with C, which is a constant that represent the inverse of regularization strength. The C values in [0.1,...,1.0] were tested for every experiment. For the base model, the best accuracy of 41.8% was achieved when C=0.7 as shown in Fig 1.

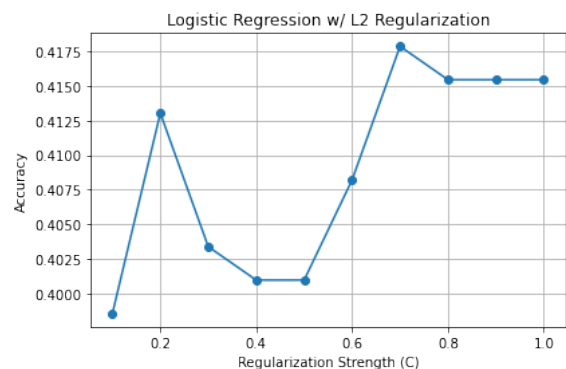


Figure 1: Accuracy by Regularization Strength

3.1.2 Additional Features

A bigram is a sequence of two adjacent words. Bag of n-grams are sometimes helpful in many NLP tasks since they retains some contextual information. However, adding bigram features decreased the performance by 2.7%, increasing the vocabulary size greater than 50k. Thus, I tried limiting the number of features by 10k, but the accuracy was still less than the baseline. TF-IDF is a weighting scheme commonly used in information retrieval, which gives higher weight for frequent, but not common words. But adding TF-IDF feature dropped the accuracy, even more than bigram.

¹<https://scikit-learn.org>

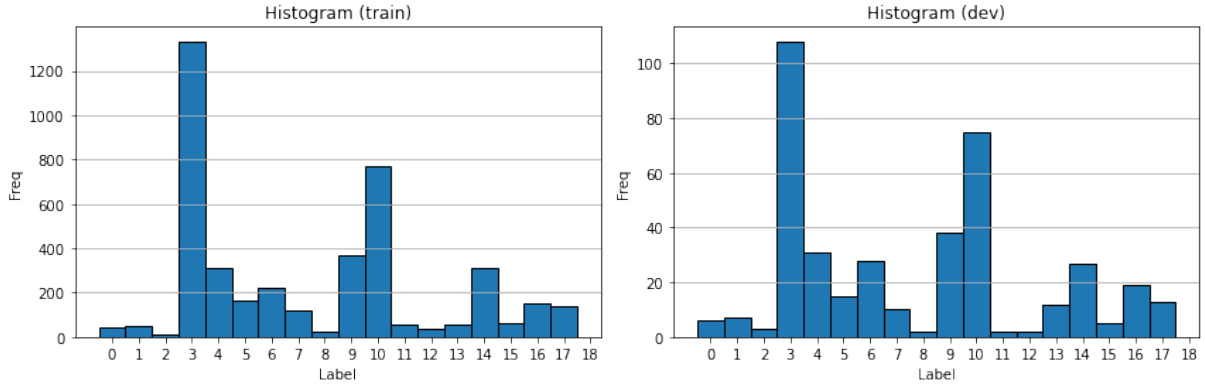


Figure 2: Histograms of Train and Dev Dataset

This might be because the dataset is severely imbalanced across the classes (Fig 2), reducing weights for meaningful words for words in documents of frequent classes.

Method	Dev Accuracy (%)
Baseline	41.8
+Bigram	39.1
+Bigram nfeat=10k	39.6
TF-IDF	37.0

Table 1: Accuracies with various methods

3.1.3 Preprocessing and Tokenization

Lemmatization is to reduce inflectional and derivationally related forms of a word to a common base form (Manning et al., 2008), which can help reducing the sparsity of the words. Tokenization is the process of dividing a string of written language into its component words. Subword tokenization is a technique that splits words into subwords that occurs frequently among the words. It reduces the word scarcity and sometimes an unseen word can be represented with a set of subwords. BPE (Sennrich et al., 2015) one of the most famous subword tokenization algorithm. NLTK (Loper and Bird, 2002) is an open source project that contains NLP libraries such as lemmatizer and tokenizer. Table 2 shows the accuracies with various preprocessing and tokenization algorithms. The best accuracy was achieved when lemmatized, tokenized with the NLTK tokenizer, and C=0.5.

Method	Dev Accuracy (%)
Baseline	41.8
Lemmatization	41.5
NLTK Tokenizer	44.2
+Lemmatization	45.2
BPE Tokenizer	43.9
+Lemmatization	44.9

Table 2: Accuracies with various methods

3.2 Semi-supervised: Exploiting the Unlabeled Data

3.2.1 Data Augmentation

Data augmentation are techniques used to increase the amount of data by creating synthetic data from the unlabeled data or slightly modifying the existing data. To leverage the unlabeled data, I generated the labels using the best supervised models and trained the same model with all the labeled and synthesized data. The results are shown on Table 3.

Method	Dev Accuracy (%)
Baseline	41.8
Lemma+NLTK Tok	45.2
+Synthesized	43.2
Lemma+BPE Tok	44.9
+Synthesized	42.3

Table 3: Accuracies w/ and w/o data augmentation

3.2.2 Self-training

Self-training is to iteratively expand the labeled data by using the classifier to predict on the unlabeled data and retrain the classifier. I tried expanding the predicted labels with probability greater than 0.8 for every iteration and stopped when the model accuracy stalls 5 times. Fig 3 shows the

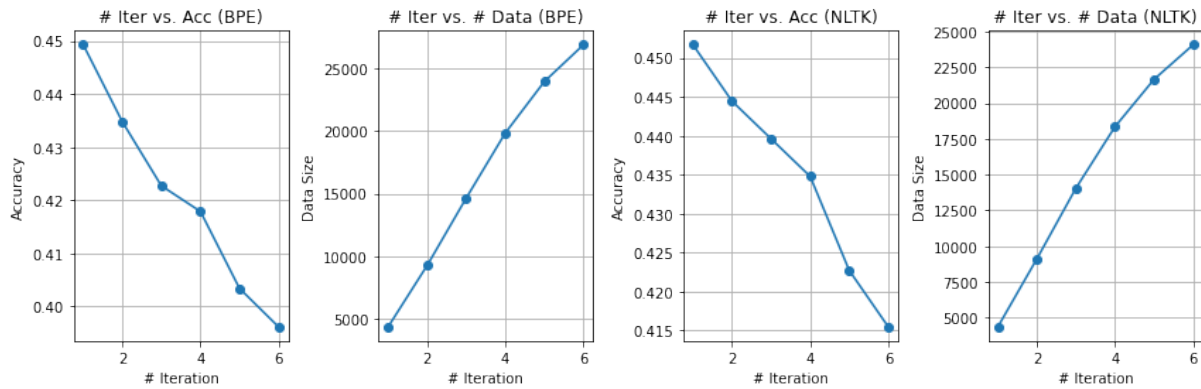


Figure 3: Self-training Accuracy and Data Size

Speaker	Highest Weight Word	Lowest Weight Word
BACHMANN_PRIMARY2012	Israel	you
BIDEN_PRIMARY2008	plan	need
CAIN_PRIMARY2012	presidential	on
CLINTON_PRIMARY2008	CLINTON	freedom
EDWARDS_PRIMARY2008	EDWARDS	every
GINGRICH_PRIMARY2012	positive	our
GIULIANI_PRIMARY2008	...	economy
HUCKABEE_PRIMARY2008	HUCKABEE	my
HUNTSMAN_PRIMARY2012	trust	be
MCCAIN_PRIMARY2008	MCCAIN	here
OBAMA_PRIMARY2008	-	-
PAUL_PRIMARY2012	money	America
PAWLENTY_PRIMARY2012	-	you
PERRY_PRIMARY2012	Washington	what
RICHARDSON_PRIMARY2008	Mexico	president
ROMNEY_PRIMARY2008	,	's
ROMNEY_PRIMARY2012	,	n't
SANTORUM_PRIMARY2012	win	home
THOMPSON_PRIMARY2008	Our	America

Table 4: Highest and Lowest Weighted Features

results started with the previous best two models (Lemma+NLTK Tok and Lemma+BPE Tok). The model performance was decreased as additional data is added. This seems counterintuitive at first, but considering the original supervised models' accuracies are less than 0.5, more than half of the added data would have been falsely labeled. Thus, the overall performance is decreased as the falsely labeled data is added.

Table 4 shows the words that affects a document to be classified to each class the most and the least with the best model. Commonly used words like 'you' and 'America' had the lowest weights and specific keywords like the speaker's name had the highest weights. However, some of the keywords

with high weights were common words such as 'win' or even symbols and these may lead to errors when predicting unseen inputs.

4 Conclusion

In this assignment, I experimented additional features, preprocessing, tokenization to improve the supervised classifiers' performance and semi-supervised approaches. I demonstrated that lemmatization and tokenization can improve the classifier, achieving accuracy of 45.2%, 3.6% higher than the baseline. However, semi-supervised approaches that synthesizes the labels with the supervised classifiers did not improve the accuracy, because the synthesized data contains more wrong labels than

correct labels.

5 Future Works

I showed that synthesizing data with a bad model will not improve the final model's performance. Therefore, introducing new features from the context would be a better approach. As we learned in the class, a context can be represented as a co-occurrence vector. Then, we can cluster the similar words by cosine distance or PMI, and we could get a bag-of-clusters representation of a document, by counting the words in each clusters. This could reduce the sparsity and would help distinguishing similar documents. We could also utilize neural network to train word embeddings or use pretrained word vectors like GloVe (Pennington et al., 2014).

References

- Zellig Harris. 1954. [Distributional structure](#). *Word*, 10(2-3):146–162.
- Dan Jurafsky and James H. Martin. 2009. *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition*. Pearson Prentice Hall, Upper Saddle River, N.J.
- Edward Loper and Steven Bird. 2002. [Nltk: The natural language toolkit](#). *CoRR*, cs.CL/0205028.
- Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, USA.
- Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. [Neural machine translation of rare words with subword units](#). *CoRR*, abs/1508.07909.