NLP Final Project Update

Colin Pillsbury
Swarthmore College
Department of Computer Science
cpillsb1@swarthmore.edu

Tai Warner
Swarthmore College
Department of Computer Science
twarner2@swarthmore.edu

1 Literature Review

1.1 Gabrilovich and Markovitch

We started with the Gabrilovich and Markovitch paper, titled "Text Categorization with Many Redundant Features", which focuses on the feasibility of feature selection in text categorization contexts. Specifically, the paper looks at how in problems with redundant features, decision tree algorithms like C4.5 can outperform SVMs, but with significant feature selection, SVMs can beat C4.5 by a slight margin. As context, the paper cites several earlier studies that suggest that the majority of features in a bag of words model are relevant for classification, and that SVMs perform best with no additional feature selection. Using data from Open Directory Project, they construct classification problems where there are significant amounts of redundant features. They give the example of the task of predicting whether an article is talking about Boulder, Colorado or Dallas, Texas. Here, there are obviously proper nouns such as place names or local businesses that would be very helpful for distinguishing the two, but many of the other features would not be helpful.

They look at a variety of feature selection techniques (Information Gain, Document Frequency, Chi-square, etc.) and find that while C4.5 performs better when neither algorithms are performing feature selection, SVMs (with a linear kernel) performed better than C4.5 after doing feature selection. The methodology seemed fairly sound, although they do gloss over things like the specific parameters for their C4.5 model. Additionally, it does seem a little surprising that C4.5 would do better after feature selection, because we thought decision trees would ignore irrelevant features anyway. Some of the graphs they include are a little difficult to interpret (missing axis labels?). Also, we are taking the resulting p-values with a grain of salt, knowing that p-values can be pretty easily manipulated using p-hacking.

This paper is similar to our topic in that it focuses on feature reduction for bag of words models, but their work seems to be more focused on a specific subset of text classification problems (tasks with lots of redundant features), which our task does not seem to align with. However, it is useful to know that there is a lot of existing research that suggests that SVMs in general perform best without doing feature selection.

1.2 Ljunberg

The Ljunberg paper, titled "Dimensionality reduction for bag-of-words models: PCA vs LSA", compares the use of PCA and LSA for text classification using SVMs. A lot of the paper is just giving background information about PCA, LSA, and text classification in general. The task he looks at is author classification. The data set is 8500 "text fragments" from Project Gutenberg, from 13 different authors. Ljunberg mentions that his ultimate goal would be to look into working on unsupervised text clustering, where he says dimensionality reduction would be required. We didn't see the exact connection here, but we could see how it would be useful to have lower dimensionality when calculating distances for clustering.

The methodology seems pretty sound, albeit not super interesting. Ljunberg shows that with a baseline SVM classifier, PCA with 50 principal components does better than LSA with 50 components, and about as well as LSA with 200 components. All of the results are of course worse than an SVM not doing dimensionality reduction. Ljunberg is hesitant to make conclusions, noting that this was not a very thorough exploration. But, he does say that it seems like PCA is able to better compact features while maintaining enough information to still make an accurate model.

This paper definitely has some overlap with things we were looking into doing. However, as we will talk about in the updated methodology, we may be pivoting away from focusing on dimensionality reduction. The paper does seem to confirm the conclusions about SVM performance mentioned in the Gabrilovich and Markovitch paper, namely that SVMs for bag of words perform best without any feature selection.

1.3 Lopuszyski and Bolikowski (2014)

Focused on sorting Wikipedia articles as well as abstracts to scientific journals by topic. Models each document as a set-of-tags rather than a bag-of-words. One thing they explicitly do not do is use the actual text from Wikipedia to run their classification on; they only use the titles of the articles to extract keywords and keyphrases from. It is surprising that this works well, because there is so little information in the article title, and it seems like some articles would get misclassified due to its title being a misnomer, whatever that may mean for the computer's method of extracting the keywords/phrases. This gave us the idea of looking for certain syntactic structures in the text, for example "___ of ___" or "___ 's ___", which might match many useful keywords in determining partisanship based on buzzword content which may interest one party more than the other.

1.4 Brodley and Friedl

The Brodley and Friedl paper, titled "Identifying Mislabeled Training Data", looks at techniques for filtering mislabeled training data in supervised contexts. One of the techniques that they look at is single algorithm filters, which are similar to removing outliers are in regression. At a high level, you use your classifier to filter out incorrectly labeled point, and then retrain that classifier on the new cleaned data. They also look at ensemble filters, which are pretty similar except use multiple classifiers to filter. They consider the use of majority voting and consensus voting for filtering mislabeled points.

One interesting issue that they address is the problem of distinguishing between mislabeled data points and data points that are simply exceptions. While it makes sense to fix mislabeled points, changing points that are exceptions could hurt a model's performance. The methods used seem sound, and the results reasonable. These techniques could definitely apply to our task, as the training articles are unilaterally labeled based on their source and not their content, meaning there will certainly be some mislabeled data.

2 Current Methodologies

We want to compare a few different types of classifiers against each other, while both using the data as is, and creating an augmented dataset to train and test on. These two datasets will each be used for training and testing SVM, KNN, and Naive Bayes classifiers, and it will be interesting to see which kinds of classifiers do better on the sparse data which technically has more information in it versus the lower dimensional data.

For the relabeling data approach, we could iteratively train a model, relabel data, and retrain a model on the new data. Each round we would evaluate the performance of the model by using a held-aside data set. With this approach we would probably not add many features beyond the regular bag of words features that we used earlier. However, we are planning on pivoting from the relabeling data approach at this point, to instead focus on the other methods mentioned.

For the ensemble classifier approach, we could test a wide variety of combinations of classifiers to see which ensembles perform best. With this approach we would probably create some new features to go along with the bag of words features.

3 Updated Results

We compared a couple different types of classifiers against one another with our MultinomialNB classifier from Lab 8 as a baseline. From Scikit-Learn we used `KNeighborsClassifier` from `sklearn.neighbors`, and `SVC` from `sklearn.svm`, as well as `sklearn.naive_bayes.MultinomialNB` from before. We also had heard of a classifier called XGBoost which has been used to great success in Kaggle competitions, so we downloaded that and compared its results, too. We wanted to test these new models on the full training and test sets, but found that these models ran much slower than MultinomialNB, so we don't have full results yet. We are planning on doing more thorough testing and parameter tuning in the future, and perhaps trying different classifiers if we are not able to get these to run in a reasonable amount of time.

Model (evaluation	Accuracy	Precision	Recall	F1 Score
Most Frequent Class (cross validation)	0.500	0.000	0.000	0.000
Most Frequent Class (held-aside data)	0.500	0.000	0.000	0.000
Multinomial Naive Bayes (cross validation)	0.800	0.778	0.840	0.808
Multinomial Naive Bayes (held-aside data)	0.610	0.572	0.871	0.811