Evaluating Performance of Semi-Supervised Self Training in Identifying Fake Reviews

Kaushik Varadha Rajan, Nivasse Ajagane, Shubham Srivastav
North Carolina State University , Raleigh NC 27606, USA
{kaushi, najagan, ssrivas6}

1 Background and Introduction

The main objective of our project is to build classifiers using Semi-Supervised learning methods. We will then use this classifier to identify "fake" restaurant reviews posted on Yelp. Yelp is a website which publishes crowd-sourced reviews about local businesses including restaurants [7]. Yelp uses its own proprietary algorithm for filtering "fake" reviews. For the purpose of this project, we would be assuming Yelp classification as pseudo ground truth. Semi-supervised learning is a class of supervised learning tasks and techniques that also make use of unlabeled data for training - typically a small amount of with a large amount of unlabeled data. Supervised learning methods are effective when there are sufficient labeled instances to construct classifiers. Labeled instances are often difficult, expensive, or time consuming to obtain, because they require empirical research. When it comes to restaurant reviews, we have a large supply of unlabeled data. Often semi supervised learning achieves a better accuracy than supervised learning which is only trained on the labeled data [6].

There are various approaches that can be used for semi-supervised learning. These include Expectation Maximization, Graph Based Mixture Models, Self-Training and Co-Training methods. In our project, we will be focusing on applying the Self-Training approach to Yelp's reviews. In self-training, the learning process employs its own predictions to teach itself. An advantage of self-training is that it can be easily combined with any supervised learning algorithm as base learner [6].

We will be using three different supervised learning methods - Naïve Bayes, Decision Trees and Logistic Regression as base learners. We would then be comparing the accuracy of each of the semi-supervised learning methods with its respective base learner. The base learners would be using both behavioral and linguistic features.

1.1 Related Work

Extensive studies (Mukherjee et al., 2013; Liu et al., 2013) have been done on determining the effectiveness of existing research methods in detecting real-life fake reviews on a commercial website like Yelp and in trying to emulate Yelp's fake review filtering algorithm.

Apart from this, Liu et al., (2013) proposed a novel model to detect opinion spamming in a Bayesian framework and model the spamicity of reviewers by identifying certain behavioral features. The key motivation is based on the hypothesis that opinion spammers differ from others on behavioral dimensions [2].

Research has also been done (Jafar et al., 2015) in the application of semi supervised learning to a pool of unlabeled data and augmenting performance of supervised learning algorithm. They have studied the semi-supervised self-training algorithm with decision trees as base learners.

2 Proposed Method

Extensive studies have already been done on detecting spam using supervised learning techniques. Mukherjee et al., (2013) have built upon this by using Yelp's classification of the reviews as pseudo ground truth. Additionally, Li et al., (2011) have used semi supervised co-training on manually labeled dataset of fake and non-fake reviews.

For our project, we will be focusing on applying semi-supervised self-training to Yelp's reviews by using Yelp's classification as pseudo ground truth. Our approach is inspired from the above two state of art research on review classification.

We aim to come up with a new solution that will help increase the performance of semi-supervised approach – the idea being that semi-supervised learning methods could improve upon the performance of supervised learning methods in the presence of unlabeled data.

To test this hypothesis, we implemented the self-training algorithm using Naïve Bayes, Decision Trees and Logistic Regression as base learners and compared their performance.

2.1 Collection of Data

We built a Python crawler to collect restaurant reviews from Yelp. Reviews were collected for all restaurants in a particular zip code in New York. We collected both the recommended and non-recommended reviews as classified by Yelp. The dataset consists of approximately 40k unique reviews, 30k users and 140 restaurants. The following attributes were extracted:

- Restaurant Name
- Average Rating
- User Name
- Review Text
- Rating
- Date of Review
- Classification by Yelp (Recommended / Not Recommended)

2.2 Preprocessing of Data

We carried out the following steps during preprocessing:

- **Cleaning of Data**
  The data that we collected had lots of duplicate records and the first step was to remove these. Following this, we modified the date field of all the records to ensure that the formatting was consistent.
- **Processing of Text reviews**
  The first step here was to remove all the Stop Words. Stop Words are words which do not contain important significance to be used in search queries. These words are filtered out because they return vast amount of unnecessary information [8]. Then we converted the text to lower case and removed punctuations, special characters, white spaces, numbers and common word endings. Finally, we created the Term Document Matrix to find similarity between the text reviews.
  Following is the word cloud of the text reviews:

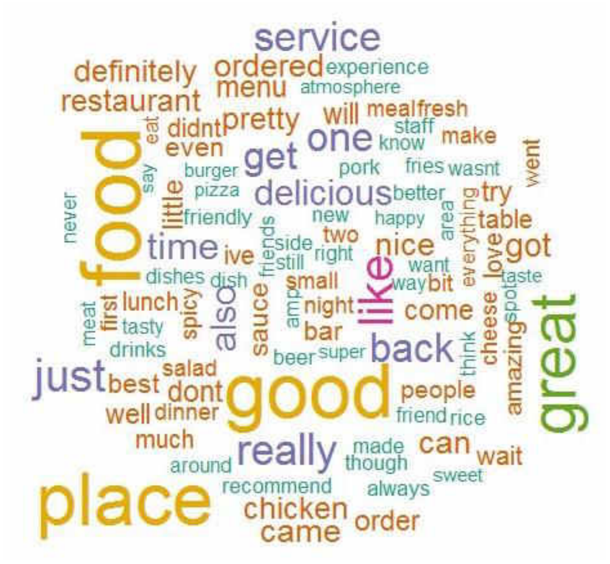

Figure 1**:** Word Cloud of the reviews

- **Calculating Behavioral Dimensions**

Variable/Description	Description
$a; A; r; r_a = (a, r)$	*Author a; set of all authors; a review; review by author a*
$f_{MNR}(a)$	*Maximum number of reviews by author a*
$Max\mathrm{Re}v(a)$	*Maximum number of reviews posted in a day by an author a*
f_{rel}	*Length of the review*
$f_{Dev}(r_a)$	*Reviewer Deviation for a review r by author a*
$*(r_a, p(r_a))$	*The* rating of r_a on product $p(r_a)$ on the 5* rating scale*
f_{cs}	*Maximum content similarity for an author*
$\cos ine(r_i, r_j)$	*Cosine similarity between review i and j*

Table 1: List of Notations

Using the attributes that we extracted, we identified the following four behavioral features that could be used to build our classifier (The notations are listed above) [1], [3]:

- **Maximum Number of Reviews (MNR):** This feature computes the maximum number of reviews in a day for an author and normalizes it by the maximum value for our data.

$$f_{MNR}(a) = \frac{Max\mathrm{Re}v(a)}{\max_{a \in A}(Max\mathrm{Re}v(a))}$$

- **Review Length:** This feature is basically the number of words in each preprocessed text review.

$$f_{rel} = length(r_i)$$

- **Rating Deviation:** This feature finds the deviation of reviewer's rating for a particular restaurant from the average rating for that restaurant (excluding the reviewer's rating) and normalizing it by the maximum possible deviation, 4 on a 5-star scale.

$$f_{Dev}(r_a) = \frac{|*(r_a, p(r_a) - E[*(r_{a' \neq a}, p(r_a)]|}{4}$$

- **Maximum Content Similarity (MCS):** For calculating this feature, we first computed the cosine similarities for every possible pair of reviews that are given by a particular reviewer. Then, we choose the maximum of these cosine similarities to represent this feature.

$$fcs(a) = \max_{r_i, r_j \in R_a, i<j} cosine(r_i, r_j)$$

2.3 Sampling

Using random sampling, we split our data set into training and testing sets in the ratio of 70:30 respectively. Then we divided the training set such that approximately 60 % of the records were unlabeled and the remaining were labeled. Following this, we used subsets of increasing sizes from the labeled data to train the base learner (Naïve Bayes). To generate the subsets of labeled data, we used both simple random sampling and stratified sampling approaches. The results of these approaches are discussed in the Experiment and Results' section.

2.4 Machine Learning Algorithm

In our project we focus on using semi-supervised learning with self-training – a widely used method in many domains and perhaps the oldest approach to semi-supervised learning. We chose to evaluate our classifiers using self-training because it follows an intuitive and heuristic approach. Additionally, the usage of Self-Training allowed us to implement multiple classifiers as base learners (for e.g. Naïve Bayes, Decision Trees, Logistic Regression etc.) and compare their performance.

For the choice of base learners, we had various options. We chose Naïve Bayes, Decision Trees and Logistic regression as our three base learners for the Self-Training algorithm. We chose these options because of the fact that Self-Training requires a probabilistic classifier as input to it. We didn't use non-probabilistic classifiers like Support Vector Machines (SVM) and K-nearest neighbor (k-NN) because of this reason.

We were also considering using co-training as one of our semi-supervised learning approaches. However, Co-Training requires the presence of redundant features so that we can train two classifiers using different features before we finally ensure that these two classifiers agree on the classification for each unlabeled example. For the data-set that we were using, we didn't have redundant features and hence we decided against using Co-Training.

2.4.1 Semi-supervised setting

In semi-supervised learning there is a small set of labeled data and a large pool of unlabeled data. We assume that labeled and unlabeled data are drawn independently from the same data distribution. In our project, we consider datasets for which $n_l << n_u$ where n_l and n_u are the number of labeled and unlabeled data respectively [5].

First, we use Naïve Bayes as a base learner to train a small number of labeled data. The classifier is then used to predict labels for unlabeled data based on the classification confidence. Then, we take a subset of the unlabeled data, together with their prediction labels and train a new classifier. The subset usually consists of unlabeled examples with high-confidence predictions above a specific threshold value [4].

In addition to using Naïve Bayes, we are also planning to use Decision Trees and Logistic Regression as base learners. The performance of each of the semi-supervised learning models would then be compared with its respective base learner.

The outline of the self-training algorithm is given below [5]:

Algorithm 1 : Outline of the Self – Training algorithm

```
Initialize: $L, U, F, T$; $L$: Labeled data; $U$: Unlabeled data;
$F$ : Underlying classifier; $T$ : Threshold for selection;
$Iter_{max}$ : Number of iterations; $\{P_l\}_{l=1}^{M}$ : Prior probability;
$t \leftarrow 1$;
while $(U \mathrel{!}= empty)$ and $(t < Iter_{max})$ do
    – $H^{t-1} \leftarrow BaseClassifier(L, F)$;
    for each $x_i \in U$ do
        – Assign pseudo – label to $x_i$ based on classification confidence
    – Sort Newly – Labeled examples based on the confidence
    – Select a set $S$ of the high – confidence predictions according to $n_l \propto P_l$
        and threshold $T$ // Selection Step
    – Update $U = U - S$; $L = L \cup S$;
    – $t \leftarrow t + 1$
    – Re – Train $H^{t-1}$ by the new training set $L$
end while
Output: Generate final hypothesis based on the new training set
```

3 Plan

The main goal of our project was to test the hypothesis that when the number of labeled data is less, semi-supervised learning methods could improve upon the performance of supervised learning methods in the presence of unlabeled data.

To verify this hypothesis, we compared the performance of semi-supervised self-training against its respective base learners. To do this, we performed the following steps:

- We split the available data set into training and testing sets in the ratio of 70:30.
- On the training set, we created labeled data of varying sizes (from 50 to 2000). For the remaining data, we removed the labels and considered it to be the unlabeled data set.
- We then trained the base learners individually on these sets of labeled data and tested it on the test set noting the accuracy.
- Using these base learners, we built the semi-supervised self-training model individually on the sets of labeled data and again tested it on the test set noting the accuracy.
- Finally, we compared the accuracy for the base learners alone and its corresponding semi supervised self-training model and plotted graphs.

One difficulty that we faced while we designed the experiment was that in our dataset, as per Yelp's classification, we had only 11% of data that was classified as spam by Yelp. To ensure that we preserve this ratio between spam vs non spam data while sampling, we decided to use stratified sampling along with simple random sampling. This was done to check if stratified sampling produced any performance improvements.

The following comparisons were made:

- Semi-Supervised Vs Supervised using Naïve Bayes – We aim to implement the base learner as Naïve Bayes classifier and use it in the self-training algorithm.
- Semi-Supervised Vs Supervised using Decision Trees.- We aim to implement the base learner as Decision Tree classifier and use it in the self-training algorithm
- Semi-Supervised Vs Supervised using Logistic Regression. We aim to implement the base learner as Logistic Regression classifier and use it in the self-training algorithm

4 Experiment and Results

Stratified Sampling and Simple Random Sampling

While performing Stratified sampling, we have maintained the same ratio of class labels (recommended vs not recommended) in the labeled dataset as the original dataset

The following graphs show the results of individual base learners vs. the semi-supervised self-training method for varying labeled data sets:

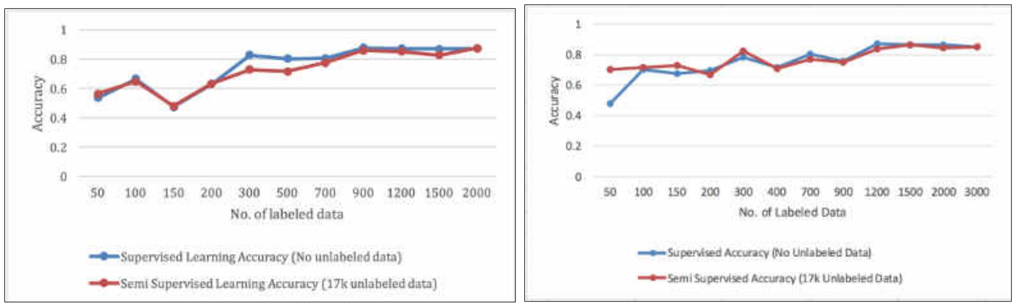

Figure 2**:** Semi-supervised vs Supervised using Naïve Bayes (Stratified Sampling on the left and Simple Random Sampling on the right.)

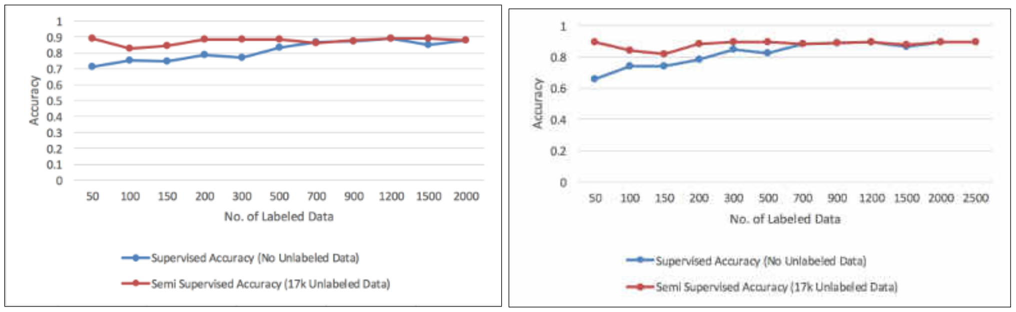

Figure 3: Semi-supervised vs Supervised using Decision Tree (Stratified Sampling on the left and Simple Random Sampling on the right.)

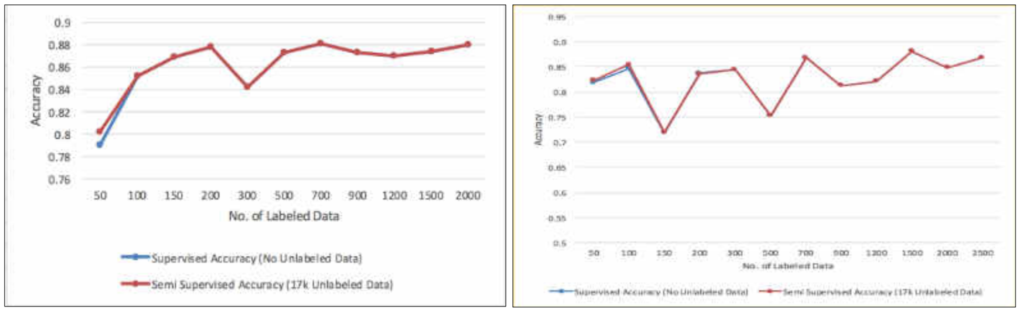

Figure 4: Semi-supervised vs Supervised using Logistic Regression (Stratified Sampling on the left and Simple Random Sampling on the right.)

4.1 Result Evaluation

4.1.1 Critical evaluation of the Naïve Bayes experiment

- As the size of the labeled data set increases, accuracy of both the models converged to a stable value (Approximately 86%). Thus, Naïve Bayes performed well for both the supervised and semi-supervised training model.
- When number of labeled data was low, Naïve Bayes with simple random sampling performed better with the semi-supervised model than the supervised approach. For stratified sampling, both the models gave similar accuracy. This is in agreement to our initial hypothesis.
- As we increased the number of labeled data, accuracy for the semi-supervised approach was not always better than the supervised approach. This is a deviation from our initial hypothesis. This might be because Naïve Bayes has the strong assumption that the features are conditionally independent. For our project, it is difficult to interpret the interdependencies between behavioral footprints of the reviewers [3].

4.1.2 Critical evaluation of the Decision Tree experiment

- As the size of the labeled data set increases, accuracy of both the models converged to a stable value (Approximately 89%). Thus, Decision Tree performed well for both the supervised and semi-supervised training model.
- For both simple random and stratified sampling, Decision Tree performed better with the semi-supervised model than the supervised approach. This is in agreement to our initial hypothesis.

4.1.3 Critical evaluation of the Logistic Regression experiment

- As the size of the labeled data set increases, accuracy of both the models converged to a stable value (Approximately 88%). Thus, Logistic Regression performed well for both the supervised and semi-supervised training model.
- For both simple random and stratified sampling using Logistic Regression, accuracy for the semi-supervised approach was not always better than the supervised approach. This is a deviation from our initial hypothesis. This might be because of the fact that the self-training algorithm that we're using doesn't work well when the base learner does not produce reliable probability estimates to its predictions [5].

5 Conclusion

Through this project, we learnt that self-training works well when the base learner is able to predict the class probabilities of unlabeled data with high confidence.

Based on the experiments that we performed, we found that in general semi-supervised learning using self-training does improve the performance of supervised learning methods in the presence of unlabeled data.

From the approaches that we tried, we found that semi-supervised self-training using Decision Tree as classifier leads to better selection metric for the self-training algorithm than the Naïve Bayes and Logistic Regression base learners. Thus, Decision tree works as a better classification model for our project.

Since the Decision Tree worked well, we had the idea of implementing Naïve Bayes Tree which is a hybrid of Decision Tree and Naïve Bayes on our data set. Tanha et al., (2015) have conducted a series of experiments which show that Naïve Bayes trees produce better probability estimation in tree classifiers and hence would work well with the self-training algorithm [5].

References

[1] Mukherjee, A., Venkataraman, V., Liu, B. and Glance, N. 2013. What Yelp Fake Review Filter might be Doing? ICWSM. (2013).

[2] Mukherjee, A., Liu, B. and Glance, N. 2012. Spotting fake reviewer groups in consumer reviews. WWW.

[3] Mukherjee, A., Kumar, A., Liu, B., Wang, J., Hsu, M., Castellanos, M., and Ghosh, R. 2013a. Spotting opinion spammers using behavioral footprints. In Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM.

[4] Nigam, K., McCallum, A., Thrun, S., and Mitchell, T. Text classification from labeled and unlabeled documents using EM., 2000.

[5] Tanha, J., Someren, M., and Afsarmanesh, H. "Semi-Supervised Self-Training for Decision Tree Classifiers". Int. J. Mach. Learn. & Cyber. (2015): n. pag. Web.

[6] Tanha, J., Someren, M., and Afsarmanesh, H. “Semi-Supervised Self-Training with Decision Trees: An Empirical Study”. In proceeding of: 3rd IEEE International Conference on Intelligent Computing and Intelligent System, (2011)

[7] "Yelp". Wikipedia. N.p., 2016. Web. 10 Apr. 2016.

[8] "List of English Stop Words - XPO6". XPO6. N.p., 2009. Web. 10 Apr. 2016.