Semi-Supervised Learning For Sentiment Analysis

John Miller, Aran Nayebi, Amr Mohamed
{millerjp, anayebi, amr1} @stanford.edu

Abstract

We leverage vector space embeddings of sentences and nearest-neighbor methods to transform a small amount of labelled training data into a significantly larger training set using an unlabelled corpus. The quality of the larger training set is measured by prediction accuracy on a benchmark sentiment analysis task. Our results indicate it is possible to achieve accuracy within 3-5% of the baseline using only 5-8% the amount of labelled data.

1 Introduction

Many statistical learning approaches to sentiment analysis require large amounts of labelled training data. Acquiring large corpora of labelled data is often a time consuming and expensive process. Additionally, the Internet contains vast stores of data that could be useful for sentiment classification, but there is not an obvious way to use this data without first hand annotating each sentence.

The goal of this project is to explore how to use a small set of labelled data and a large set of unlabelled data to construct models with accuracy comparable to those trained on large sets of exclusively labelled data.

Given a large training set with just a few pieces of labelled data, the naive approach is to simply ignore the unlabelled data and train a classifier only on hand-annotated sentences. Intuitively, however, one should be able to achieve improved accuracy by also considering the unlabelled data. One might hope to find some way to use the labelled data to find good labels for the unlabelled data. In general, this problem is as difficult as the original inference task, so we focus on fast, heuristic procedures.

At a high level, our approach will be to find similar sentences in the unlabelled data and assign them a label that matches a reference sentence in the seed set. To make the notion of "similarity" between sentences precise, we represent each sentence as a vector in a high dimensional vector space and use Euclidean distance between sentences as a proxy for similarity.

To construct embeddings that capture the semantics of a particular sentence, we build embeddings directly from distributed representation of words, e.g. [2].

Given some fixed sentence embedding, we use the seed set to assign labels to the closest neighboring sentences in this high dimensional space. Then, equipped with this expanded training set, we use the resulting sentence embeddings as features in a standard sentiment analysis task.

The performance of our semi-supervised inference procedure is evaluated against the baseline training set, which uses the entire labelled corpus. Despite receiving significantly less training data, our results indicate that it is possible to achieve accuracy within $3-5\%$ of the baseline with a seed set 5% of the size of the baseline and a small number of nearest neighbors.

2 Preliminaries

2.1 Data

We apply our semi-supervised approach to the benchmark sentiment analysis task introduced in [5]. The data for this task is the Stanford Sentiment Treebank [5]. The Stanford Sentiment Treebank consists of 11,855 single sentences extracted from www.rottentomatoes.com movie reviews, and includes a total of 215,154 unique phrases that have been annotated by three human judges.

Each sentence is labelled with one of 5 sentiment classes, 0 for "very negative", 1 for "negative", 2 for "neutral", 3 for "positive", and 4 for "very positive". This dataset is then decomposed into training, development, and test sets of 8544, 1101, and 2210 sentences, respectively.

As a simplification, we consider only the more coarse-grained 3-class classification task, where the labels are $\{$Negative, Neutral, Positive$\}$ obtained by bucketing "very negative" and "negative" as well as "positive", and "very positive" into two classes.

From this larger corpus, we allow our algorithms to only access the true labels of a small subset of this data, which we refer to as the "seed set." We stress that the text of the other sentences is known to the algorithms, but their corresponding labels are not.

To ensure the models we develop are robust to changes in the seed set, we generate a new seed set on each run of

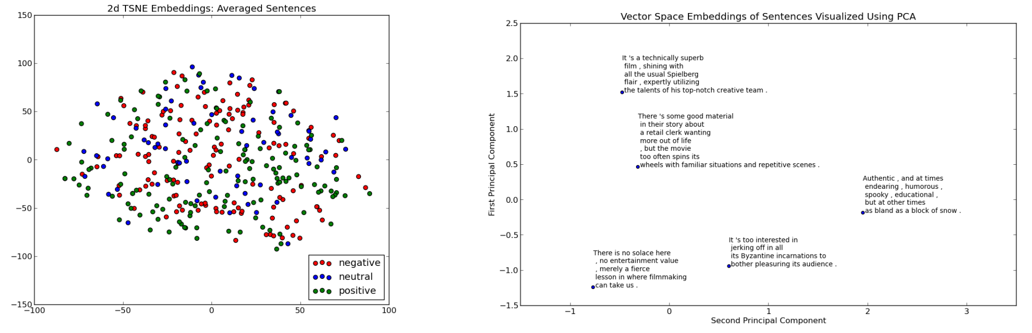


Figure 1: Visualization of Distributed Sentence Representations

the algorithms by randomly sampling the set of sentences.

2.2 Features: Distributed Sentence Representation

Each sentence s is represented as a vector $s \in \mathbb{R}^{300}$. To construct this representation, we use a semantic vector space model of language, where each word is represented by a real-valued vector. These vector representations are taken from a database of 6 billion word vectors trained on a corpus of 840 billion tokens using the GloVe model [2].

To obtain a sentence representation for sentence $s = s_1 s_2, \ldots, s_n$, we take the average of the word vectors $s_1, s_2, \ldots, s_n$ where s_i is the GloVe vector corresponding to the i-th word in the sentence. As demonstrated in [5], this naive word vector averaging often outperforms bag of words and bag of n-gram models. Equipped with this representation, the distance between two sentences is then simply the ℓ_2 distance.

To gain some intuition about the topology of the resulting vector space of sentences, we visualize a random subset of the sentence embeddings using both t-SNE [4] and principal components analysis. The corresponding plots are given in Figure 1.

One salient observation from Figure 1 is that the vector space representation appears to encode semantic qualities of the individual sentences. Examining the local geometry in the t-SNE plot reveals that sentences with similar sentiment tend to appear together in local clusters. Furthermore, examining the PCA plot shows a desirable ordering of the examples with respect to sentiment. After projecting the data into two dimensions, the sentences are roughly ordered along the first principal component with respect to sentiment. Sentences with positive sentiment are positive along this axis, neutral sentences are near 0, and negative sentiment sentences are negative.

This result makes intuitive sense since we expect the words typically in positive reviews to be somewhat distinct from the words typically in a negative review and appear together in a local cluster in the GloVe vector space. Hence, the corresponding vector averages and the clusters for each of these classes should also be distinct.

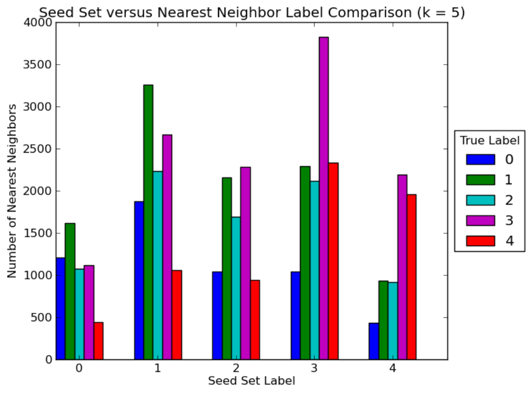


Figure 2: Distribution of Nearest Neighbors Sentiment

3 Models

3.1 Extending the Training Set

Motivated by the discussion in the previous section, we propose the following approach to extend the small seed set of labelled data into the full training set. Let S denote the set of sentences, and let I denote the seed set.

To construct a larger training set, we find the k-nearest neighbors for each sentence $s \in I$, as measured by Euclidean distance. Then we add each of the nearest neighbors to the training set with the same label as s.

We emphasize that the true label of these sentences is unknown to the model, and the assigned label is only a heuristic guess. Both the size of the initial seed set I and the number of nearest neighbors, k, are parameters in the model, and we will study classification accuracy as these numbers vary.

As a justification for this approach, consider Figure 2. This plot gives the number of nearest neighbors with a particular sentiment label for each label in the seed set. For each seed set label, the most common nearest-neighbor label is precisely the original label with the distribution being particularly accurate for "positive" and "very positive sentences." As should also be clear, this procedure does introduce some noise into the training set. We discuss the effect of this noise and its impact on classification accuracy in Section 5.

After constructing the expanded training set, we train a sentiment analysis classifier on the resulting sentences. We briefly describe the top two performing models for this task, as measured by accuracy on the full, labelled training set.

3.2 Regularized SoftMax Regression

Regularized SoftMax Regression generalizes logistic regression to the multi-class setting. The model is parameterized by $\theta_i \in \mathbb{R}^{300}$ for classes $i = 1, 2, \dots, k$ and a regularization parameter $\lambda \in \mathbb{R}$, which is chosen via 10-fold cross-validation. The objective function is given by

$$J(\theta) = \left\{ \sum_{i=1}^{m} \sum_{j=1}^{k} \mathbf{1}_{y^{(i)}=j} \log \frac{\exp \theta_j^T x^{(i)}}{\sum_{j=1}^{k} \exp \theta_j^T x} \right\} + \frac{\lambda}{2} \sum_{i=1}^{k} \sum_{j=0}^{n} \theta_{ij}^2$$

To improve the rate of convergence, we implemented softMax and optimized $J(\theta)$ using adaGrad, which improved convergence by an average of 1200 iterations [3].

3.3 Multi-Class Support Vector Machine

We also explored the performance of an open-source Multi-Class SVM implementation [1]. In particular, we used a one-versus-one SVM, which constructs and trains a separate ℓ_1-soft-margin SVM for each pair of classes. At test time, each SVM separately classifies the sentence, and the sentence receives the label of the class with the highest number of votes. For each SVM, the objective is:

$$\min_{\mathbf{w}, \xi, b} \left\{ \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \xi_i \right\}$$

subject to

$$y_i(\mathbf{w} \cdot \mathbf{x_i} - b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \quad \text{for } i = 1, \dots n$$

4 Results

To evaluate the success of our nearest-neighbor methods, we first trained and tuned both classifiers on the entire labelled training set. The table below gives the classification accuracy for various choices of λ and kernels in the SVM. Where parameters are not specified, they are set to the optimal values found via cross-validation. ** indicates the top performing models.

Baseline Model	Accuracy
**SoftMax ($\lambda = 0.05$)	0.763
SoftMax ($\lambda = 10^{-4}$)	0.761
SoftMax ($\lambda = 0$)	0.758
**SVM (Linear Kernel)	0.759
SVM (Gaussian Kernel)	0.734
SVM (Polynomial Kernel)	0.591
SVM (Sigmoid Kernel)	0.632

The following table gives the performance of the top-performing models for different seed set sizes $|I|$ and number of nearest neighbors k. Figure 3 shows classification accuracy as a function of the number of examples ($|I| \cdot k$) for various k number of nearest neighbors compared to the baseline, which is the accuracy of the classifier trained on the full dataset. Results are averaged over 25 random trials.

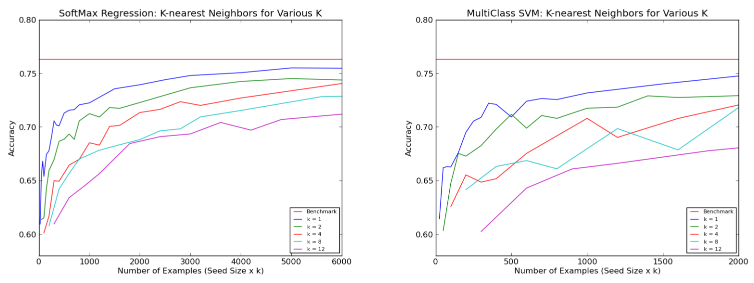
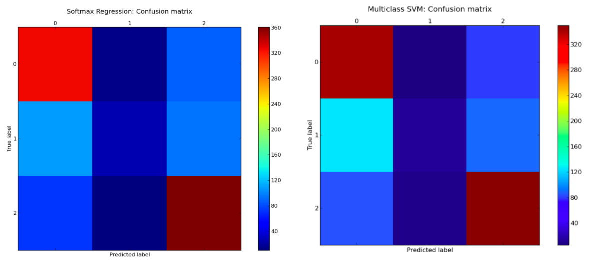


Figure 3: Classification accuracy as a function of seed set size and number of neighbors

Model with Nearest Neighbors	Accuracy
SoftMax($\|I\| = 500, k = 5$)	0.715
SoftMax($\|I\| = 200, k = 10$)	0.691
SVM ($\|I\| = 500, k = 5$)	0.725
SVM ($\|I\| = 200, k = 10$)	0.709

4.1 Error Analysis

To better understand the errors made by both classifiers, consider the confusion matrices given below for each classifier trained on the full labelled training set. Both classifiers are extremely accurate in predicting labels for positive and negative sentences. Further examination reveals that as few as 100-200 labelled sentences and 5-10 nearest neighbors is actually sufficient to separate the positive and negative classes for most examples with either classifier.



On the other hand, both classifiers experience poor performance on the neutral class, with increases in labelled data only slightly improving accuracy. There are several potential reasons for this inequity. First, the t-SNE plot given in Figure 1 suggests that neutral sentences do not exhibit the same clustering properties as positive and negative sentences and are more loosely scattered throughout the space.

Additionally, the labels for neutral sentences, even those made by human judges, are often imprecise. As a canonical example, consider the sentences: “It ’s never dull and always looks good” and “Alas , another breathless movie about same!” The first sentence is seemingly positive, while the second one is seemingly negative, but both are neutral sentences in the treebank.

This ambiguity makes it difficult to train classifiers that can correctly distinguish between a “positive” review and a “neutral” review with positive words or a “negative” review and a “neutral” review with negative words

5 Discussion

As the tables in Section 4 and Figure 3 indicate, extending the subsampled training set using nearest neighbor produced classification accuracies within 3-5% of the baseline achieved on the entire training set, even with $1/16$-th the amount of labelled data.

In the $k = 1$ case, where half of the data is unlabelled, it is somewhat surprising that we can achieve performance roughly equivalent to simply doubly the labelled training set. What is also especially interesting is the $k = 12$ case, where only 8% of the data is originally labelled. Even when the training set is significantly subsampled, extending the training set by a factor of 12 obtains accuracy within 5% of the baseline as the number of examples grows. This suggests, for problems lacking significant amounts annotated training data, our nearest neighbor methods allow one to significantly expand the training set and obtain correspondingly higher accuracy.

One observation from Figure 3 is all of the models have performance that is asymptotically lower than using purely labelled data. This is likely the cost of noise introduced by expanding the training set using nearest neighbors, as Figure 2 indicates that not all of our examples are accurately labelled. While this was not a significant issue in our experiments, it is an open question as to how this approach scales with the problem size.

Finally, all of our methods had difficulty correctly classifying neutral sentences. We posit that neutral sentence vagueness breaks our assumption that the words typically in neutral sentences are unique to neutral sentences. Consequently, unigram averaging is an insufficiently powerful model to capture this ambiguity and does not produce classifiers with good accuracy.

These inaccuracies are a failure of our sentence representation rather than a failure of our nearest neighbor methods. It is likely that, given access to a more powerful sentence representation, nearest neighbor methods would yield the same boost in accuracy obtained by extending the training set, but also provide better classification of neutral sentences.

6 Conclusion

Our results indicate that a small seed-set of labelled sentences and nearest-neighbor methods are often sufficient to achieve classification accuracy within 3-5% of the baseline consisting entirely of labelled data. This suggests, at least in the case of sentiment analysis, it is possible to achieve high accuracy without a substantial investment in the construction of a large, hand-annotated training set.

Viewed in another light, we expect that classifiers tuned and trained on benchmark datasets could be further improved by applying our techniques to large unlabelled bodies of text, e.g. the Wikipedia corpus, and then retraining them on an expanded training set.

Much of the improvements in accuracy come from increased performance on positive and negative sentences. Our experiments suggest 100-200 labelled sentences and 10 nearest neighbors is sufficient to separate these two classes. However, neutral sentences are substantially more difficult, and increases in seed set size only slightly improve accuracy. This is likely because neutral sentences are more evenly distributed in the semantic vector space than the locally clustered positive and negative sentences.

All of the classifiers we tried asymptotically converged to roughly 76% accuracy when trained on the full labelled training set. This appears to be a "fundamental limit" on the capabilities of simple unigram averaging as a form of sentence representation. To improve this number, we must use a more complex sentence representations and perhaps more sophisticated classifiers. In these instances, it is likely that our technique of extending the training set via nearest neighbors will yield improvements in accuracy.

7 Future Work

Our results suggest several extensions to generating a more effective classifier. First, averaging word vectors destroys valuable context and syntactic information. We are exploring different representations of sentences that preserve more of the linguistic structure in an attempt to improve accuracy beyond 76%. Additionally, the seed set has a large impact on the chosen nearest neighbors. It would be interesting to explore how choices in the seed set affect our results.

Acknowledgements

We thank Richard Socher for his guidance and advice.

References

[1] `http://scikit-learn.org/stable/`.

[2] J. Pennington, R. Socher, and C. D. Manning. "GloVe: Global vectors for word representation". In Proc. EMNLP, 2014.

[3] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. JMLR, 12.

[4] L.J.P. van der Maaten and G.E. Hinton. Visualizing High-Dimensional Data Using t-SNE. Journal of Machine Learning Research, 2008.

[5] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts. "Recursive Deep Models for Semantic Compositionality over a Sentiment Treebank". In EMNLP 2013.