Sentiment Analysis Using Semi-Supervised Recursive Autoencoder

Vinay Kumar

April 19, 2015

Abstract

The aim of this project was to use semi-supervised recursive autoencoder provided by [2] and classify the english phrases from movie reviews into five sentiment classes; very positive, positive, neutral, negative and very negative by softmax regression classifier.

1 Introduction and Motivation

Sentiment analysis is a growing field targeting to extract insights or subjective conclusions from the sources like text or large amount of data. Identifying the sentiment from online text gives businesses and organizations insights into what are main issues hitting the public opinion and what what they should focus on improving or changing. Getting significant attention from both business and research communities, sentiment analysis has many potential applications like summarizing user-reviews, brand-management and public relations management of business organizations and governments. Most of the past work has been focused on classifying the data in two classes: positive and negative. In my project, I aim to classify the data in five different classes: very positive, positive, neutral, negative and very negative.

2 Related work

This project is inspired by the work of Mr. Richard Socher, phd scholar at Stanford University in sentiment analysis using his recursive autoencoder who according to his website www.socher.org, "*introduced a novel machine learning framework based on recursive autoencoders for sentence-level prediction of sentiment label distributions*". Socher's model of recursive autoencoder learns vector space representations for multi-word phrases as shown in the below image from his website.

3 Dataset

The sentiment analysis dataset used is from Kaggle, consisting of 8544 sentences which are converted to 156060 English phrases from movie reviews. I did the standard 70-30 percentage split from this dataset for the training set and the test set respectively. This dataset can be found on the url mentioned in [5]

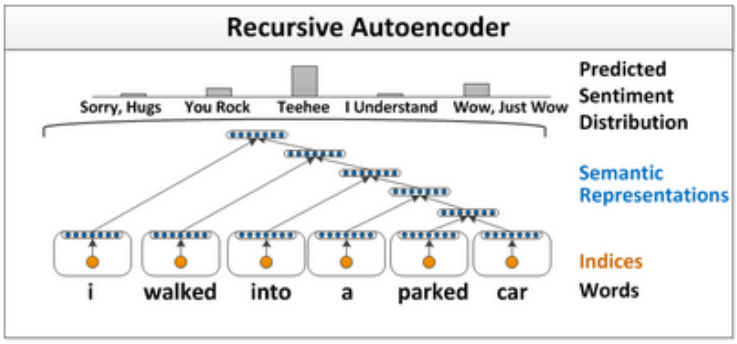

Figure 1: vector space representations for multi-word phrases in socher's model.

4 Methodology

I have used the algorithm for Semi-Supervised Recursive Autoencoder from Richard Socher as described in [1]. The algorithm consists of an unsupervised part and a supervised part.

4.1 Recursive Autoencoder(RAE)

The unsupervised part is a recursive auto-encoder which creates an N-dimensional vector that represents the phrase or simply called as the 'code' for the phrase. For doing this, each word of every phrase is represented as an N-dimensional vector (N randomly chosen to be 50 here), N being is a positive integer. This vector is formed by randomly selecting every element from a uniformly distributed range of [-0.05, 0.05].For every adjacent word-pair in the phrase, the value of p is calculated:

$$p = f(X^{(1)}[v_1; v_2] + b^{(1)}) \tag{1}$$

where $X_{(1)}$ is an Nx2N matrix, v_1 and v_2 are the vectors corresponding to every word in the pair, and $b_{(1)}$ is an Nx1 vector and function f is a sigmoid or tanh function which works as an activation function. After computing p, the original word-vectors are calculated back from p by performing the following evaluation:

$$[v_1', v_2'] = X^{(2)}p + b^{(2)}. \tag{2}$$

where $X^{(2)}$ is a 2NxN matrix and $b^{(2)}$ is a 2Nx1 vector. The autoencoder aims to minimize the below described reconstruction error:

$$E_{rec}([v1; v2]) = \frac{1}{2}\|[v_1; v_2] - [v_1'; v_2']\|^2 \tag{3}$$

The reconstruction error is calculated for every adjacent word-pair in the phrase (e.g.w1,w2 followed by w2,w3), then both word vectors of the pair with the minimum reconstruction error are replaced with their code p. The same reconstruction-error is calculated on the new list of word vectors again and the pair with the lowest error is replaced with the code. The same greedy pattern is followed to the point when only one vector left in the list. This vector is taken as the representative "code" of the phrase. The source code for this is publicly available at Richard Socher's website [2] .

4.2 Supervised Learning Classifier

After obtaining an N-dimensional vector or 'code' for each phrase from the unsupervised recursive auto-encoder, any supervised learning classifier like naive bayes or support vector machine could be used to classify this vector into 5 different classes. I have used Softmax regression(multinomial logistic regression) which generalizes the logistic regression to multi-class classification. One more important advantage of using softmax regression is it reduces unnecessary errors by keeping the classification in different classes mutually exclusive(not classifying one phrase into more than one class as opposed to one-vs-all logistic regression used in [3]).

Using MinFunc: When using matlab's in-bulit function *mnrfit* to train a multinomial logistic regression classifier, I found it very much memory and time consuming . So, I used minFunc,[4] which is a matlab function that utilises line-search methods for the optimization of differentiable real-valued multivariate functions.

5 Results

The results I obtained by using softmax regression and recursive auto-encoder are as follows:

Class	**Accuracy(perc.)**	**Correctly labeled phrases**
Very Negative	95.1	44547
Negative	83.4	39051
Neutral	73.1	34192
Positive	80.6	37726
Very positive	94.6	44179

The overall accuracy and the maximum iterations achieved were 85.8 percentage and 70 respectively.. The following confusion-matrix shows the number of phrases correctly classified and mis-classified in other classes. The main reason of mis-classification stems from the fact that the lines between two categories are quite blurry like the words 'happy' or 'excited' could be classified in *positive* or *very positive* class.

Class	Very Negative	Negative	Neutral	Positive	Very Positive
Very Negative	95.1	3.9	0.8	0.1	0.0
Negative	14.4	83.4	2.0	0.1	0.0
Neutral	1.8	10.1	73.1	12.6	2.4
Positive	0.1	1.6	5.4	80.6	12. 3
Very positive	0.0	0.2	1.1	4.1	94.6

6 Conclusion and Future work

Using softmax regression as opposed to one-vs-all logistic regression in [3](which gives an average accuracy of 85.6 percentage) gave a slight improvement mostly

because it reduces unnecessary errors by keeping the classification mutually exclusive, that means no phrase is classified in more than one class For labeling several classes, the maximum iterations of 70 was achieved by the program which points to the idea that if the number of iterations is increased, it may result in improved accuracy. Using pre-trained word vectors instead of randomly initialized vector can give a better mapping of the semantic order of words to the model. Also, changing the values of random parameters like *alpha* (I used it as 0.2), which administers the relative weighting between the reconstruction error, error of the unsupervised part, and the cross-entropy error, error of the supervised learning part can also be experimented upon. The use of bigger dataset as in [2] could promise more accurate results because recursive neural network being a complex algorithm(non-linear), needs more training data to reach a certain upper bound on the error with high probability. Therefore, if we increase the size of the training data, the marginal improvement of neural network could be expected. Finally, a large part of work done in sentiment analysis is the classification of text in the range of positivity. Future works could be more focused on classifying the source/text into subjective classes like classifying/summarizing the content from an online forum into categories like user-reviews, customer help requests, inappropriate content or spam.

7 Acknowldgements

I gratefully acknowledge the guidance of Prof. Amitabha Mukerjee, Computer Science Department at Indian Institute of Technology, Kanpur for his insightful and humble guidance during the course of this project.

References

[1] R. Socher, J. Pennington, E. H. Huang, A. Y. Ng, and C. D. Manning. 2011b. SemiSupervised Recursive Autoencoders for Predicting Sentiment Distributions. In EMNLP Url: http://ai.stanford.edu/ ang/papers/emnlp11- RecursiveAutoencodersSentimentDistributions.pdf

[2] Richar Sochar's website: http://www.socher.org/index.php/Main/SemiSupervisedRecursiveAutoencodersForPredictingSentimentDistributions

[3] Bahareh Ghiyasian and Yun Fei Guo. 2014. Sentiment Analysis Using SemiSupervised Recursive Autoencoders and Support Vector Machines. Stanford.edu

[4] MinFunc matlab code by Mark Schmidt. 2005. url: http://www.cs.ubc.ca/ schmidtm/Software/minFunc.html

[5] Kaggle. 2014. Sentiment Analysis on Movie Reviews. https://www.kaggle.com/c/sentimentanalys is-on-movie-reviews

[6] Peng Qi http://qipeng.me/software/mnlr.html