Sentiment Analysis Using Semi-Supervised Recursive Autoencoder

Vinay Kumar(12806)

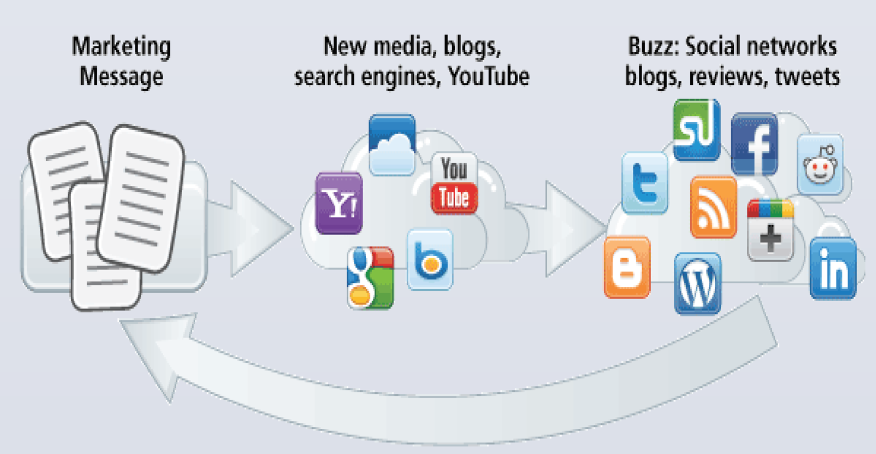
CSE, IIT Kanpur

INTRODUCTION

Sentiment analysis is a growing field targeting to extract insights or subjective conclusions from the sources like text or large amount of data.

Applications: 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.

Past work: Most of the past work has been focused on classifying the data in two classes: positive and negative. In my project, I have classified the data in five classes: very positive, positive, neutral, negative and very negative.



DATASET

I am using the sentiment analysis dataset from Kaggle [3], which contains phrases and sentences from Rotten Tomatoes movie reviews. This dataset [3] consists of 8544 sentences which is converted to 156060 English phrases from movie reviews.

Kaggle. 2014. Sentiment Analysis on Movie Reviews. https://www.kaggle.com/c/sentimentanalysis-on-movie-reviews

METHODS

I am using the Semi-Supervised Recursive Autoencoders algorithm from Richard Socher. The algorithm consists of an unsupervised part and a supervised part.

Recursive Autoencoder(RAE): The unsupervised part is a recursive auto-encoder that creates an N-dimensional vector that represents the phrase or simply called as the 'code'.

Supervised Learning Classifier(Multinomial Logistic Regression): In the second part after obtaining an N-dimensional vector or 'code' for each phrase from the unsupervised recursive auto-encoder, supervised learning is to be used to classify this vector into a particular class.

Google Word2Vec: Instead of using the randomly initialized N-dimensional vectors, I will use google word2vec tool [4] which maps words with similar meaning to similar positions in the N-dimensional vector space.

MULTINOMIAL LOGISTIC REGRESSION

Softmax Regression: This model generalizes logistic regression to classification problems where the class label *y* can take on more than two possible values.

MINFUNC

I used *minFunc* which is a Matlab function for unconstrained optimization of differentiable real-valued multivariate functions using line-search methods.

COST FUNCTION

$$p(y^{(i)} = j|x^{(i)};\theta) = \frac{e^{(\theta_j-\psi)^T x^{(i)}}}{\sum_{l=1}^{k} e^{(\theta_l-\psi)^T x^{(i)}}} = \frac{e^{\theta_j^T x^{(i)}} e^{-\psi^T x^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T x^{(i)}} e^{-\psi^T x^{(i)}}} = \frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T x^{(i)}}}.$$

CLASS PROBABILITIES

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=1}^{k} 1\left\{y^{(i)} = j\right\} \log \frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T x^{(i)}}}\right]$$

Notice that this generalizes the logistic regression cost function, which could also have been written:

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}(1-y^{(i)})\log(1-h_\theta(x^{(i)})) + y^{(i)}\log h_\theta(x^{(i)})\right] = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=0}^{1} 1\left\{y^{(i)} = j\right\} \log p(y^{(i)} = j|x^{(i)};\theta)\right]$$

RESULTS

I have the following results for Using Recursive Auto-Encoder with Multinomial Logistic Regression.

My Overall Accuracy: 85.8%

PAST WORK:

One-vs-all LR (RAE): 85.3%

SVM: 85.6%

Stanford Model(Socher): 86.4%

CLASSWISE-ACCURACY

Class	Accuracy
V.Negativ	95.10%
Negative	83.40%
Neutral	73.10%
Positive	80.60%
V. Positiv	94.60%

ISSUES & RESOLUTIONS

Randomized N-dimensional vector : Using **word2vec** to map similar words nearby to exploit the similarity of meaning/context

Belong to more than one class -> Unnecessary errors : Using **Softmax regression** instead of logistic regression

Altering the values of random parameter, ***α***, the parameter that controls the relative weighting between **the reconstruction error**.

Using MinFunc: When using Matlab's mnrfit to train a multinomial logistic regression classifier recently, I found it rather memory-consuming. I used *minFunc* for optimization of multivariate functions.

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

[3] Google. 2014. word2vec. https://code.google.com/p/word2vec/

[4] PENG QI http://qipeng.me/software/mnlr.html

CONCLUSION

Softmax regression gave a slight improvent over one-vs all regression.

Using pre-trained word vectors instead of randomly initialized vector gave a better mapping to the model of the semantic order of words.Changing the values of random parameters like **α**,can also improve the accuracy slightly.

We could try increasing the training dataset size. Because recursive neural network is a more complex algorithm (non-linear), it 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 would be more.