Sparse Autoencoders in Sentiment Analysis

Mariusz Kleć
Polish-Japanese Institute of Information Technology,
Multimedia Department, Warsaw, Poland.
mklec@pjwstk.edu.pl
http://www.pjwstk.edu.pl

Abstract. This paper examines the utilization of sparse autoencoders in the task of sentiment analysis. The autoencoders can be used for pre-training a deep neural network, discovering new features or for dimensionality reduction. In this paper, sparse autoencoders were used for parameters initialization in deep neural network. Experiments showed that the accuracy of text classification to a particular sentiment may be improved by incorporating sparse autoencoders to this task. Analysis, outcomes, and description of this approach are presented.

Keywords: sparse autoencoder, sentiment analysis, unsupervised pre-training

1 Introduction

Sentiment analysis (SA) is the task of identifying peoples attitude toward a particular topic, based solely on textual information. The text can be considered in terms of authors opinions such as “like” or “dislike,” “for” or “against,” “good” or “bad,” etc. This is a binary approach as the result is represented only by two polar opposites (e.g., “bad” vs “good”). Some researches incorporate a sentiment scale to judge peoples opinions [10]. This scale approach is useful for recognizing bloggers attitude toward a given topic (e.g., presidential election) or how much people like a movie or a particular product. SA is also widely applied in business intelligence, for example, for evaluating products and services. Additionally, it can be very useful in such fields of research as politics, law, sociology, and psychology, and in analyzing trends, identifying ideological orientations, targeting advertising, etc.

1.1 Related Work

This paper describes utilization of sparse autoencoders in SA. An autoencoder (AE) is a special type of neural network (NN) where the output vector has the same dimensionality as the input vector. An AE tries to reconstruct the input in the output layer, passing data through the hidden layers. AEs were used for dimensionality reduction [6], pre-training deep NN; [2] or as a features detector [15]. The most intuitive and widely used example of learning nonlinear features
from data by means of AEs is the recognition of pen stroke edges from images of hand written digits (MNIST)\(^1\) [15] [12]. The AE with greater number of hidden layers than one forms a deep AE. The pre-training of deep NN in turn has its origin in the research on pre-training deep AE for dimensionality reduction [6].

Training deep architectures with backpropagation produces suboptimal results. It is caused by the weakness of gradient descent optimization method where gradients rapidly diminish in magnitude as the depth of the network increases. The solution turned out to be pre-training deep structure, layer-by-layer in an unsupervised way, using Restricted Boltzman Machine (RBM) or simple AEs (with one hidden layer) [7]. In such a case, gradient descent can be used only for fine-tuning the whole pre-trained structure, training the network to minimize the error with respect to the labels [1] [2] [4]. Many researchers used this fact in their experiments in different fields of research, for example, in automatic speech recognition systems [16] [5] and in text classification [14]. For the MNIST database, the error rate was reduced from 1.6% to 1.2% applying such pre-training methods [7] [2].

In [11], the Internet Movie Review Database (IMDb) was used to determine a sentiment of movie reviews. Today, the second version of this database is available as the second version of IMDb (v2.0)\(^2\) which provides a collection of movie review documents, labeled with their overall sentiment polarity (1000 positive and 1000 negative documents). In [9], sentiment polarity was determined by the application of text-categorization techniques to only subjective portions of the documents in this database. They achieved 86.4% accuracy in 10-fold cross-validation test, employing Naive Bayes classifier. It was a clear improvement over 82.8% accuracy obtained using full-length reviews. With SVM, the full review classification accuracy raised to 87.15%. In this paper, sparse AEs were used for pre-training deep NN for sentiment recognition using the IMDb v2.0.

In Section 2 data preprocessing is described. The concept of sparse AEs and the implementation background are described in Section 3. The experiments and results are presented in Section 4, followed by conclusions in Section 5.

2 Data Preparation

The IMDb v2.0 was transformed into a vector that indicates statistics about the occurrence of words in documents. The statistical transformation of TF-IDF ranking was applied. This pre-processing step was performed in WEKA\(^3\) environment. Words were converted to lower case and only those words were chosen that formed continuous alphabetic sequences. Semantically useless words in terms of classification, such as “some,” “the,” “a,” etc., were removed by applying a default stop-words list for the English language. Each word had to appear at least five times to be considered as an attribute. The word frequencies for a

\(^{1}\) http://yann.lecun.com/exdb/mnist/

\(^{2}\) http://www.cs.cornell.edu/people/pabo/movie-review-data

\(^{3}\) http://www.cs.waikato.ac.nz/ml/weka/
document (instance) were normalized based on document length. Next, evaluation of attributes was performed by measuring the information gain with respect to the class. After this evaluation, only 784 best attributes were retained. Next, the data was transformed into Matlab format. Ten subsets of data were created for 10-fold cross-validation test. Each subset included equal number of positive and negative reviews. The instances in each subset were randomly shuffled with number random generator set as rand(‘state’,0) in MATLAB. The preprocessed database, used in this paper, is available upon request.

3 Implementation Background

All the experiments involved NN and AE implementation. Both of them were developed in MATLAB according to guidelines presented in [8] from scratch. The code was also extended to operate with cross-entropy cost function [3]. The regularization tends to decrease the magnitude of the weights, which prevents overfitting. The weight decay parameter of $\lambda$ (lambda) is used to control the relative importance of the regularization calculated as $||W||^2$. Furthermore, mini-batch gradient descent, momentum, and learning rate decay were additionally developed. The gradient calculation is also compatible with minFunc\(^4\) function that uses L-BFGS optimization algorithm for weights update [13].

3.1 Sparsity Constraint

The idea behind sparse AEs is to enforce activations of hidden units to be close to the zero for most of the time during training. It was achieved by applying the measure of Kullback-Liebler (KL) divergence to the cost function. KL divergence measures the difference between the two distributions: the average activations of hidden units over the training set ($\hat{\rho}$) and the target distribution ($\rho$). In other words, we want to enforce $\rho = \hat{\rho}$. The target distribution $\rho$, in all the experiments, was set to 0.1. In order to penalize an average activation of hidden units $\hat{\rho}$ deviating too much from its target value of $\rho$, a special penalty term $\beta$ was introduced to control the weight of the sparsity constraint described above [8].

4 Experiments

The experiments involved classifying movie reviews as either positive or negative. The goal was to obtain the highest accuracy performing 10-fold cross-validation test. Sparse AEs were used for weights initialization (pre-training) of NN. As a baseline, five NNs were trained, gradually extending the number of hidden layers, starting with logistic regression. Only the first hidden layer had higher number of units (1000) than the input (784). Any further layers had equal 500 units, up to the fourth hidden layer. The results were compared to the same

\(^4\) http://www.di.ens.fr/~mschmidt/Software/minFunc.html
architectures where layer-by-layer unsupervised pre-training was performed by means of sparse AEs. Each subsequent AE was pre-trained by data, derived from the previous already-trained one (by feed-forwarding data through the hidden layer). The process of pre-training the shallow architecture of NN is presented in Figure 1.

For training sparse AEs, L-BFGS was used due to its superior effectiveness and speed. Fine-tuning was performed using mini-batch gradient descent optimization method with cross-entropy cost function.

4.1 Evaluation

During cross-validation, the pre-training and fine-tuning steps were performed anew in each fold. In each fold, eight subsets of data took part in the training. The two remaining subsets were intended for validation and test set. The validation set did not take part in the training (neither for AE nor NN). It was used only to determine early stopping functionality during the fine-tuning. When the cost value on validation set has not been decreasing by more than 1e-4, the training was terminated and evaluation result on test set was saved.

The best parameters for training particular sparse AE were evaluated by training it with different combinations of $\beta$ (sparsity target) and $\lambda$ (importance of regularization) through 200 epochs. Then, 10-fold cross-validation tests were performed during the fine-tuning step. The sparse AE, which gave the highest accuracy, was used to initialize given layer in the final NN.
To compare the results of different architectures of NN, each of them was trained with the same parameters of gradient-descent (λ: 1e-4, batch size: 30). Only learning rate was different. For training NN where weights were initialized randomly, the learning rate was equal to 0.07. For fine-tuning NN where weights were pre-trained, the learning rate was slightly lower and was equal to 0.01.

4.2 Results

The aim was to compare the results obtained by training NNs with different number of hidden layers. NNs were divided into pre-trained models using sparse AEs, and models where weights were randomly initialized. Table 1 presents average accuracy over 10-fold cross-validation tests. Also, the values of β and λ, used for training the last sparse AE, are presented.

Table 1. Accuracy in percentage from 10-fold cross-validation test. The results where the improvement obtained by pre-training is the highest are given in bold. Each layer was pre-trained based on the pre-trained sparse AE in the previous one. The best parameters for training the last sparse AE are denoted by β (sparsity target) and λ (importance of regularization).

<table>
<thead>
<tr>
<th>Number of units in layers</th>
<th>No pre-training</th>
<th>With pre-training</th>
<th>λ</th>
<th>β</th>
</tr>
</thead>
<tbody>
<tr>
<td>784/2 and 784/2</td>
<td>90.7</td>
<td>90.5</td>
<td>3e-3</td>
<td>4</td>
</tr>
<tr>
<td>784/1000/2</td>
<td>90.45</td>
<td>90.8</td>
<td>3e-3</td>
<td>4</td>
</tr>
<tr>
<td>784/1000/500/2</td>
<td><strong>90.1</strong></td>
<td><strong>91.05</strong></td>
<td>3e-3</td>
<td>4</td>
</tr>
<tr>
<td>784/1000/500/500/2</td>
<td>90.35</td>
<td>91.05</td>
<td>3e-4</td>
<td>4</td>
</tr>
<tr>
<td>784/1000/500/500/500/2</td>
<td>82.05</td>
<td>91</td>
<td>3e-5</td>
<td>5</td>
</tr>
</tbody>
</table>

5 Conclusions

This paper described the transformation of text data into a vector of statistics about the occurrence of words in documents. The vectors were fed into a sparse AE to train its weights and use them as new starting parameters for training NN. Classification for the task of SA was performed.

From the results we can conclude that pre-training on existing data increases the accuracy of sentiment recognition by about 1% in cross-validation test. It is worth noting that applied early stopping criterion is not the most efficient. Observing the process of training, the author noticed that even better results could be achieved when cost value on validation set has increased (especially when the number of hidden layers was higher). This point requires additional improvement to determine the early stopping criterion better. Nevertheless, with this stopping criterion, the results was better (91.05%), compared to the 89.8% achieved using SVN in [9]. In this paper, separation between the data used for
training sparse AEs and the data used for testing was important. The accuracy was calculated on data that did not take part in training NN nor sparse AEs.

Everytime sparse AEs were incorporated into pre-training, the results were better, compared to those where AEs were not taken into consideration. An exception is logistic regression in cross-validation test; when trained with data obtained from the first AE, the accuracy was slightly lower. It might have been caused by the simplicity of the data. Nevertheless, an additional test showed that first AE is able to learn new features from data. The additional logistic regression trainings with different divisions on data obtained from the first trained AE showed that the accuracy increased by 1% even when the input vector (from AE) had higher dimensionality than the data. But when the number of hidden units would be lower than the number of inputs, the sparse AE should also learn compressed representation of the data, as was proved in [6]. This is worth to research in the future, by providing newly discovered features from sparse AE to other machine-learning algorithms like SVM.

This paper does not exhaust all possibility of using sparse AE in SA. The presented approach is quite simple, compared to others that focus on selecting indicative lexical features used for SA. But undoubtedly, this paper increases awareness about using sparse AEs as a method for improving sentiment recognition from text. The intention of the author is to continue the research on deep learning and features detection.

6 Questions

1. Does more data for pre-training sparse autoencoders improve the accuracy of sentiment recognition?
2. What text transformation would be the most efficient for this research?
3. What is the importance of network topology in this research?

Acknowledgments

I would like to thank Krzysztof Marasek, Danijel Koržinek, Thomas Kemp, and Christian Scheible for their support.

References


