Word Classification for Sentiment Polarity Estimation Using Neural Network

Hidekazu Yanagimoto, Mika Shimada, and Akane Yoshimura

School of Engineering, Osaka Prefecture University
Osaka, Japan, 599-8531
{hidekazu,shimada,yoshimura}@cs.osakafu-u.ac.jp

Abstract. Though there are many digitalized documents in the Internet, the almost all documents are unlabeled data. Hence, using such numerous unlabeled data, a classifier has to be construct. In pattern recognition research field many researchers pay attention to a deep architecture neural network to achieve the previous aim. The deep architecture neural network is one of semi-supervised learning approaches and achieve high performance in an object recognition task. The network is trained with many unlabeled data and transform input raw features into new features that represent higher concept, for example a human face. In this study I pay attention to feature generation ability of a deep architecture neural network and apply it to natural language processing. Concretely word clustering is developed for sentiment analysis. Experimental results shows clustering performance is good regardless of an unsupervised learning approach.

Keywords: Natural Language Processing, Deep Architecture Neural Network, Feature Extraction.

1 Introduction

Though there are many digitalized documents in the Internet, the almost all documents are unlabeled data. To label data much human power is needed since now only human understands information and classify them into appropriate clusters. Though a classifier is constructed to classify them automatically, many labeled data is needed to construct a high performance classifier. If you construct a classifier using such numerous unlabeled data, this problem vanishes. This approach is a semi-supervised learning approach.

In pattern recognition research field many researchers pay attention to a deep architecture neural network. The deep architecture neural network is one of semi-supervised learning approaches and achieves high performance in an object recognition task. The network is trained with many unlabeled data and transforms input raw features into new features, which show higher concept, for example a human face. Since the pattern recognition simulates one of human activities using a deep architecture neural network, in this study the neural network is applied to language understanding.

S. Yamamoto (Ed.): HIMI/HCII 2013, Part I, LNCS 8016, pp. 669–677, 2013.

Word classification for sentiment analysis is developed as a task of language understanding. The sentiment analysis determines article polarity using natural language processing. The polarity means positive, negative, or neutral from the viewpoint of an author. A polarity dictionary is used to develop the sentiment analysis system and it is important to construct a high quality polarity dictionary. Since the dictionary construction needs much human power to judge many words polarities, an automatic dictionary construction method is desired. Hence, my aim is that a deep architecture neural network is applied to the polarity dictionary construction. Before the goal this study checks whether a neural network can be applied to the dictionary construction or not constructing word classification system.

In Section 2 related works are explained. In Section 3 the proposed method is described from the viewpoint of neural network architecture and learning methodology. In Section 4 some experiments are explained and performance of the proposed method is discussed.

2 Related Works

A proposed method is related to neural network researches and feature extraction researches. Hence, in this section related works, that is neural network approaches and kernel methods, are explained.

A neural network is one of models to simulate a brain simply and was proposed by Rosenblatt[1]. Then Rumelhart et al. proposed backpropagation algorithm to train multi-layer feedforward neural network[2]. Because of a backpropagation algorithm neural network are used in various kind of research fields. The neural networks are generally shallow architecture neural networks, for example three-layer or four-layer neural networks but deeper neural networks are not used. Vanishing gradient problem[3] limits the number of layers in neural networks. The vanishing gradient problem means the backpropagation algorithm cannot propagate error in deeper layer's weight. Hence, though you use a deep architecture neural network, you cannot achieve good performance since you cannot adjust weights in the neural network.

To overcome this problem deep learning[4] was proposed. The deep learning is a methodology to train a deep neural network, which is many layers neural network. To solve the vanishing gradient problem the deep learning approach uses a layer-wise learning approach. These days the deep learning is applied to pattern recognition and natural language processing.

In pattern recognition researches Lee et al.[5] applied a deep architecture neural network to object recognition and could extract good characteristics from unlabeled enormous image data. The architecture of their neural network integrated local image features into high-level features, for example outline of cat face and human's back shot. In ILSVRC2012 Large Scale Visual Recognition Contest a deep convolutional neural network achieved the highest performance by Hinton et al[6].

In natural language processing researches deep architecture neural networks are used to construct a language model. Since a hidden Markov model is generally

used as a language model, conditional probabilities is determined from training data in the neural network. This approaches are called a neural network language model and proposed by Bengio and Arisoy[7,8]. In natural language processing a document is represented as a discrete vector based on term frequency. On the other hand in neural network language model a document is represented as a continuous vector transforming discrete vectors into continuous vectors using a neural network. After then documents are classified for text classification, information retrieval, and information filtering.

A deep architecture neural network creates new features from primitive raw input data essentially. This characteristic is similar to a kernel method[9]. The kernel method maps features in input space onto features in higher-dimension space using nonlinear function, which is called a kernel function. In the kernel method the kernel functions are defined previously since the functions have to satisfy Mercer's theorem. On the other hand, the deep architecture neural network makes more complex features from input raw features according to a training data distribution.

Finally Restricted Boltzmann Machine (RBM)[10] and Sparse Autoencoder[11] are described. A deep architecture neural networks uses RBM and Sparse Autoencoder as a part of the network. To train hidden layers in a deep architecture neural network it is not realistic to use labeled data because of the vanishing gradient problem. They are trained without labeled data since their cost functions are errors between an input patten and an output pattern. In RBM neurons in a hidden unit are independent each other in determining outputs of neurons in a visible unit since their connections in RBM is restricted. Sparse Autoencoder is trained as fire of neurons in a hidden layer is sparse.

3 Feature Generation Using Neural Network

In this section the proposed method is described from the viewpoint of neural network architecture and learning methodology.

3.1 Neural Network Architecture

The proposed method uses a deep architecture neural network to generate a continuous feature vector from a discrete feature vector. The neural network cannot be trained with a backpropagation approach since the network has deep architecture.

Fig. 1 shows an example of a deep architecture neural network used in this study. The network is constructed combining some Restricted Boltzmann Machines (RBMs). A hidden layer in a previous RBM is a visible layer in the following RBM.

3.2 Learning Methodology

RBM is trained using a Contrastive Divergence k alrogithm[12,13].

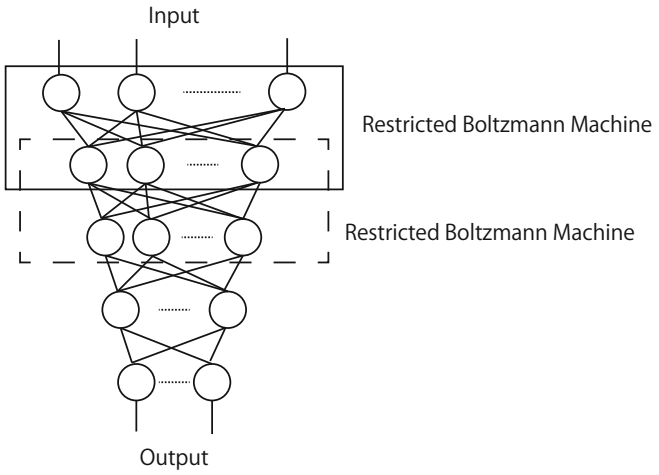

Fig. 1. This figure shows a deep architecture neural network in the proposed method. Each layer is regarded as a Restricted Boltzmann Machine and trained with RBM training method.

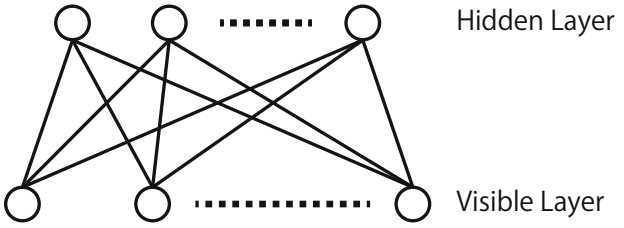

Fig. 2. This figure shows a typical Restricted Boltzmann Machine

In Fig 2 general RBM is illustrated. The RBM has 2-layer architecture, which is a hidden layer and a visible layer. Since neurons in the hidden layer are not observed outside, nobody cannot control them. The RBM obtains input data from neurons in the visible layer and keeps processing input data until the RBM is stable. Moreover, the RBM has no connection among neurons in the same layer. States of neurons in the same layer is defined independently when all states of neurons in another layer are known. Hence, neurons in the same layer is called conditional independence.

To introduce Contrastive Divergence k (CD-k) an energy function is defined. States of each neurons in the visible layer denotes $\mathbf{x}$ and states in the hidden layer denotes $\mathbf{h}$. A weight between h_i and x_j is W_{ij}. $\mathbf{b}$ and $\mathbf{c}$ denote weights of visible neurons and hidden neurons respectively. Hence, the energy function is defined below.

$$\text{Energy}(\mathbf{x}, \mathbf{h}) = -\mathbf{b}^{\mathrm{T}}\mathbf{x} - \mathbf{c}^{\mathrm{T}}\mathbf{h} - \mathbf{h}^{\mathrm{T}}W\mathbf{x} \tag{1}$$

And using the energy function, a joint distribution of RBM is defined.

$$P(\mathbf{x}, \mathbf{h}) = \frac{e^{-\text{Energy}(\mathbf{x},\mathbf{h})}}{Z} \tag{2}$$

The Z is a normalized parameter. Moreover, since nobody can observe states in the hidden layer, $\mathbf{h}$ is marginalized out.

$$P(\mathbf{x}) = \frac{e^{-\sum_{\mathbf{h}} \text{Energy}(\mathbf{x},\mathbf{h})}}{Z} \tag{3}$$

To adjust a parameter W, gradient of $P(\mathbf{x})$, $\frac{\partial \log P(\mathbf{x})}{\partial W}$, has to be calculated. However, it is intractable to calculate $\frac{\partial \log P(\mathbf{x})}{\partial W}$ analytically. The CD-k algorithm approximates the gradient using difference between Kullback-Leibler divergences [12]. Since the CD-k algorithm needs states of every neurons after RBM carries out k times, the states are estimated using Gibbs sampling. Fig. 3 shows CD-1 algorithm, which is special version of CD-k, using a pesudocode. In this study CD-1 algorithm is used.

```
Initialize weights W randomly
for each epoch do
    for each data x_i of size D do
        Compute μ_i = E[h|x_i, W]
        Sample h_i ~ p(h|x_i, W)
        Sample x_i' = p(x|h_i, W)
        Compute μ_i' = E[h|x_i', W]
        Accumulate g = g + x_i μ_i^T − x_i' h_i'^T
    Update parameters W = W + (α/D) g
```

Fig. 3. The pseudocode shows CD-1 algorithm. When you use CD-k algorithm, you repeat sampling process k times.

A deep architecture neural network is constructed combining trained RBMs hierarchically. After a deep architecture neural network is trained using CD-1 algorithm, discrete feature vectors are transformed into continuous feature vectors. Hence, the outputs of the network are regarded as new feature vectors of input data. When an aim is classification, outputs of the network is inputs of a classifier.

The feature vectors are transformed based on a function constructed with numerous unlabeled data. Using RBM especially, input data are generally mapped onto lower-dimension feature space. Hence, since relevant data are located in neighborhood of lower-dimension feature space, the network is applied to classification task.

4 Experiments

Experiments are carried out using real stock market news, T&C news to evaluate performance of the proposed method.

4.1 Dataset

In experiments T&C news, which is one of stock market news delivery services in Japan, are used as a corpus. The corpus consists of 62,478 articles in 2010 including stock price news, business performance reports, comments of analysts and so on. 100 articles are labeled as positive, negative, or neutral by a stock market specialist. Positive articles include information on increasing stock price. On the other hand, negative articles includes information on decreasing stock price. Neutral articles do not affect stock price.

Since my aim is to estimate sentiment polarity of words, adjectives and some kinds of noun are extracted from the all articles as features. These words include author's intent and are used in sentiment analysis frequently. The number of features is 2,604. All articles are represented as a binary vector which denotes whether the articles include the selected words or not. If you use term occurrence frequency, almost all vectors are binary vectors since the news is very short.

After word extraction 39,269 articles are represented as feature vectors using extracted words and 71 labeled articles are included. To train a deep architecture neural network we use the 71 labeled articles and 10,000 articles that are selected from 39,198 randomly. Hence, 10,071 articles are used to train the network. Since in training phase label is not used at all, the network is trained with an unsupervised learning approach. The 71 articles are used to evaluate performance of classification with a deep architecture neural network. To evaluate performance of word classification clusters are checked manually and discussed from the viewpoint of understandability of clusters for human.

4.2 Results

A deep architecture neural network has 2,604 neurons in an input layer, 3 hidden layers, which have 1,000 neurons, 500 neurons, and 250 neurons respectively, and 100 neurons in an output layer. Hence, the network consists of 4 RBMs.

After applying the trained neural network to binary vectors of the training articles, a similarity distribution among 10,071 articles changes. Fig. 4 shows two distributions of similarity among all articles. You find the two distributions are very different. The left distribution with binary vectors shows all articles are less relevant each other though there are many articles with the same polarity.

First, clusters constructed with new feature vectors are discussed with the 71 labeled articles. Table 1 shows how many articles the proposed method improves. This result shows the proposed method improves many relevance rankings though in some of articles the proposed method worsens relevance ranking.

Finally, word clusters are discussed. The word clusters are constructed analyzing the first hidden layer. The highest weights among a neuron in hidden layer and all neurons in input layer denotes strong relevance among words. Hence, the clusters are constructed the weights between the input layer and the hidden layer. Cluster A is a cluster of positive words and Cluster B is a cluster of negative words.

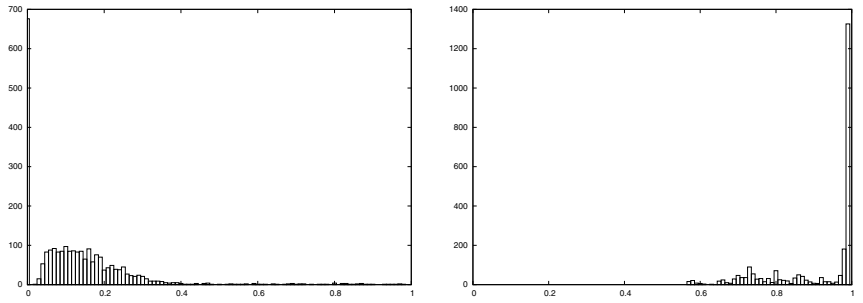

Fig. 4. The left graph shows a distribution of similarity among all articles using binary vectors for the articles. The right graph shows a distribution of similarity among all articles using continuous vectors generated with the proposed method.

Table 1. The table shows the number of ranking improvement for top 10 relevant articles between original features and new features the proposed method generated. Increasing the number of the same polarity articles in top 10 relevant ones, increase a score in "Improvement".

Improvement	Worsening	No change
36	17	18

Table 2. The table shows some clusters obtained with the proposed method. Original data is written in Japanese.

Cluster A	Cluster B
accounting	upswing
press release	average
adjustment	evolution
significant	drop-off
prediction	not(negation)
increased profit	trade
progress	caution
excellent condition	attention

4.3 Discussion

The proposed method can capture more essential relevance among articles than existing bag-of-words approach from Fig. 4 and Table 1. Since the proposed method trains the neural network with numerous unlabeled data and transform input data into lower dimension space, the neural network has a function gathering relevant words in the same neuron. The function affects performance of word clustering. From Table 2 the function works well in this network since some clusters include the same polarity words. However, there are many clusters

including both positive words and negative words, too. Not introducing polarity information expressly in training phase causes bad clusters. Hence, applying the proposed method to classifying articles according to their polarity, more discussion is needed.

5 Conclusion

I proposed word classification using a deep architecture neural network. From some experiments it was confirmed that the proposed method could capture essential relevance among text data and construct some clusters including the same polarity words. The result shows a deep architecture neural network can be applied to natural language processing, too.

Since this study is the first step to apply a neural network to sentiment analysis, a classifier to estimate article polarity will be constructed or be applied to polarity dictionary construction. And I would like to discuss performance of the proposed method from the viewpoint of their applications.

Acknowledgement. I would like to thank Centillion Japan Co. Ltd. for giving us market news for our experiments.

References

1. Rosenblatt, F.: The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. Psychological Review 65(6), 386–408 (1958)
2. Rumelhar, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagation errors. Nature 323, 533–536 (1986)
3. Hochreiter, S.: The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. Int. Journal of Uncertainty, Fuzzines and Knowledge-Based Systems 6(2), 107–116 (1998)
4. Bengio, Y.: Learning Deep Architecture for AI. Foundations and Trends in Machine Learning 2(1), 1–127 (2009)
5. Lee, Q.V., Ranzato, M., Monga, R., Devin, M., Chen, K., Corrado, G.S., Dean, J., Ng, A.Y.: Building High-level Features Using Large Scale Unsupervised Learning. In: Proc. of the 29th Int. Conference on Machine Learning (2012)
6. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet Classification with Deep Convolutional Neural Networks. In: Advances in Neural Information Processing, vol. 25. MIT Press, Cambridge (2012)
7. Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C.: A Neural Probabilistic Language Model. Journal of Machine Learning Research 3, 1137–1155 (2003)
8. Arisoy, E., Sainath, T.N., Kingsbury, B., Ramabhadran, B.: Deep Neural Network Language Model. In: Proc. of NAACL-HLT 2012, pp. 20–28 (2012)
9. Shoelkopf, B., Smola, A.J.: Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. The MIT Press (2001)

10. Hinton, G.E., Salakhutdinov, R.R.: Reducing the Dimensionality of Data with Neural Network. Science 313, 504–507 (2006)
11. Lee, H., Battle, A., Raina, R., Ng, A.Y.: Efficient Sparse Coding Algorithm. In: Advances in Neural Information Processing Systems, vol. 19, pp. 801–808 (2006)
12. HInton, G.E., Osindero, S., Teh, Y.W.: A Fast Learning Algorithm for Deep Blief Nets. Neural Computation 18, 1527–1554 (2006)
13. Muraphy, K.P.: Machine Learning: A Probabilistic Perspective. The MIT Press (2012)