Active Deep Networks for Semi-Supervised Sentiment Classification

Shusen Zhou, Qingcai Chen and Xiaolong Wang
Shenzhen Graduate School, Harbin Institute of Technology
zhoushusen@hitsz.edu.cn, qincai.chen@hitsz.edu.cn,
wangxl@insun.hit.edu.cn

Abstract

This paper presents a novel semi-supervised learning algorithm called Active Deep Networks (ADN), to address the semi-supervised sentiment classification problem with active learning. First, we propose the semi-supervised learning method of ADN. ADN is constructed by Restricted Boltzmann Machines (RBM) with unsupervised learning using labeled data and abundant of unlabeled data. Then the constructed structure is fine-tuned by gradient-descent based supervised learning with an exponential loss function. Second, we apply active learning in the semi-supervised learning framework to identify reviews that should be labeled as training data. Then ADN architecture is trained by the selected labeled data and all unlabeled data. Experiments on five sentiment classification datasets show that ADN outperforms the semi-supervised learning algorithm and deep learning techniques applied for sentiment classification.

1 Introduction

In recent years, sentiment analysis has received considerable attentions in Natural Language Processing (NLP) community (Blitzer et al., 2007; Dasgupta and Ng, 2009; Pang et al., 2002). Polarity classification, which determine whether the sentiment expressed in a document is positive or negative, is one of the most popular tasks of sentiment analysis (Dasgupta and Ng, 2009). Sentiment classification is a special type of text categorization, where the criterion of classification is the attitude expressed in the text, rather than the subject or topic. Labeling the reviews with their sentiment would provide succinct summaries to readers, which makes it possible to focus the text mining on areas in need of improvement or on areas of success (Gamon, 2004) and is helpful in business intelligence applications, recommender systems, and message filtering (Pang, et al., 2002).

While topics are often identifiable by keywords alone, sentiment classification appears to be a more challenge task (Pang, et al., 2002). First, sentiment is often conveyed with subtle linguistic mechanisms such as the use of sarcasm and highly domain-specific contextual cues (Li et al., 2009). For example, although the sentence "The thief tries to protect his excellent reputation" contains the word "excellent", it tells us nothing about the author's opinion and in fact could be well embedded in a negative review. Second, sentiment classification systems are typically domain-specific, which makes the expensive process of annotating a large amount of data for each domain and is a bottleneck in building high quality systems (Dasgupta and Ng, 2009). This motivates the task of learning robust sentiment models from minimal supervision (Li, et al., 2009).

Recently, semi-supervised learning, which uses large amount of unlabeled data together with labeled data to build better learners (Raina et al., 2007; Zhu, 2007), has drawn more attention in sentiment analysis (Dasgupta and Ng, 2009; Li, et al., 2009). As argued by several researchers (Bengio, 2007; Salakhutdinov and Hinton, 2007), deep architecture, composed of multiple levels of non-linear operations (Hinton et al., 2006), is expected to perform well in semi-supervised learning because of its capability of modeling hard artificial intelligent tasks. Deep Belief Networks (DBN) is a representative

Coling 2010: Poster Volume, pages 1515–1523,
Beijing, August 2010

deep learning algorithm achieving notable success for semi-supervised learning (Hinton, et al., 2006). Ranzato and Szummer (2008) propose an algorithm to learn text document representations based on semi-supervised auto-encoders that are combined to form a deep network.

Active learning is another way that can minimize the number of required labeled data while getting competitive result. Usually, the training set is chosen randomly. However, active learning choose the training data actively, which reduce the needs of labeled data (Tong and Koller, 2002). Recently, active learning had been applied in sentiment classification (Dasgupta and Ng, 2009).

Inspired by the study of semi-supervised learning, active learning and deep architecture, this paper proposes a novel semi-supervised polarity classification algorithm called Active Deep Networks (ADN) that is based on a representative deep learning algorithm Deep Belief Networks (DBN) (Hinton, et al., 2006) and active learning (Tong and Koller, 2002). First, we propose the ADN architecture, which utilizes a new deep architecture for classification, and an exponential loss function aiming to maximize the separability of the classifier. Second, we propose the ADN algorithm. It firstly identifies a small number of manually labeled reviews by an active learner, and then trains the ADN classifier with the identified labeled data and all of the unlabeled data.

Our paper makes several important contributions: First, this paper proposes a novel ADN architecture that integrates the abstraction ability of deep belief nets and the classification ability of backpropagation strategy. It improves the generalization capability by using abundant unlabeled data, and directly optimizes the classification results in training dataset using back propagation strategy, which makes it possible to achieve attractive classification performance with few labeled data. Second, this paper proposes an effective active learning method that integrates the labeled data selection ability of active learning and classification ability of ADN architecture. Moreover, the active learning is also based on the ADN architecture, so the labeled data selector and the classifier are based on the same architecture, which provides an unified framework for semi-supervised classification task. Third, this paper applies semi-supervised learning and active learning to sentiment classification successfully and gets competitive performance. Our experimental results on five sentiment classification datasets show that ADN outperforms previous sentiment classification methods and deep learning methods.

The rest of the paper is organized as follows. Section 2 gives an overview of sentiment classification. The proposed semi-supervised learning method ADN is described in Section 3. Section 4 shows the empirical validation of ADN by comparing its classification performance with previous sentiment classifiers and deep learning methods on sentiment datasets. The paper is closed with conclusion.

2 Sentiment Classification

Sentiment classification can be performed on words, sentences or documents, and is generally categorized into lexicon-based and corpus-based classification method (Wan, 2009). The detailed survey about techniques and approaches of sentiment classification can be seen in the book (Pang and Lee, 2008). In this paper we focus on corpus-based classification method.

Corpus-based methods use a labeled corpus to train a sentiment classifier (Wan, 2009). Pang et al. (2002) apply machine learning approach to corpus-based sentiment classification firstly. They found that standard machine learning techniques outperform human-produced baselines. Pang and Lee (2004) apply text-categorization techniques to the subjective portions of the sentiment document. These portions are extracted by efficient techniques for finding minimum cuts in graphs. Gamon (2004) demonstrate that using large feature vectors in combination with feature reduction, high accuracy can be achieved in the very noisy domain of customer feedback data. Xia et al. (2008) propose the sentiment vector space model to represent song lyric document, assign the sentiment labels such as light-hearted and heavy-hearted.

Supervised sentiment classification systems are domain-specific and annotating a large scale corpus for each domain is very expensive (Dasgupta and Ng, 2009). There are several solutions for this corpus annotation bottleneck.

The first type of solution is using old domain labeled examples to new domain sentiment clas-

sification. Blitzer et al. (2007) investigate domain adaptation for sentiment classifiers, which could be used to select a small set of domains to annotate and their trained classifiers would transfer well to many other domains. Li and Zong (2008) study multi-domain sentiment classification, which aims to improve performance through fusing training data from multiple domains.

The second type of solution is semi-supervised sentiment classification. Sindhwani and Melville (2008) propose a semi-supervised sentiment classification algorithm that utilizes lexical prior knowledge in conjunction with unlabeled data. Dasgupta and Ng (2009) firstly mine the unambiguous reviews using spectral techniques, and then exploit them to classify the ambiguous reviews via a novel combination of active learning, transductive learning, and ensemble learning.

The third type of solution is unsupervised sentiment classification. Zagibalov and Carroll (2008) describe an automatic seed word selection for unsupervised sentiment classification of product reviews in Chinese.

However, unsupervised learning of sentiment is difficult, partially because of the prevalence of sentimentally ambiguous reviews (Dasgupta and Ng, 2009). Using multi-domain sentiment corpus to sentiment classification is also hard to apply, because each domain has a very limited amount of training data, due to annotating a large corpus is difficult and time-consuming (Li and Zong, 2008). So in this paper we focus on semi-supervised approach to sentiment classification.

3 Active Deep Networks

In this part, we propose a semi-supervised learning algorithm, Active Deep Networks (ADN), to address the sentiment classification problem with active learning. Section 3.1 formulates the ADN problem. Section 3.2 proposes the semi-supervised learning of ADN without active learning. Section 3.3 proposes the active learning method of ADN. Section 3.4 gives the ADN procedure.

3.1 Problem Formulation

There are many review documents in the dataset. We preprocess these reviews to be classified, which is similar with Dasgupta and Ng (2009). Each review is represented as a vector of unigrams, using binary weight equal to 1 for terms present in a vector. Moreover, the punctuations, numbers, and words of length one are removed from the vector. Finally, we sort the vocabulary by document frequency and remove the top 1.5%. It is because that many of these high document frequency words are stopwords or domain specific general-purpose words.

After preprocess, every review can be represented by a vector. Then the dataset can be represented as a matrix:

$$\mathbf{X}=\left[\mathbf{x}^1,\mathbf{x}^2,\ldots,\mathbf{x}^{R+T}\right]=\begin{bmatrix} x_1^1, x_1^2,\ldots, x_1^{R+T} \\ x_2^1, x_2^2,\ldots, x_2^{R+T} \\ \vdots,\vdots,\ldots,\vdots \\ x_D^1, x_D^2,\ldots, x_D^{R+T} \end{bmatrix} \quad (1)$$

where R is the number of training samples, T is the number of test samples, D is the number of feature words in the dataset. Every column of $\mathbf{X}$ corresponds to a sample $\mathbf{x}$, which is a representation of a review. A sample that has all features is viewed as a vector in $\mathbb{R}^D$, where the i^{th} coordinate corresponds to the i^{th} feature.

The L labeled samples are chosen randomly from R training samples, or chosen actively by active learning, which can be seen as:

$$\mathbf{X}^L=\mathbf{X}^R(\mathbf{S}),\mathbf{S}=[s_1,s_2,...,s_L]\ 1\le s_i\le R \quad (2)$$

where $\mathbf{S}$ is the index of selected training reviews to be labeled manually.

Let $\mathbf{Y}$ be a set of labels corresponds to L labeled training samples and is denoted as:

$$\mathbf{Y}^L=\left[\mathbf{y}^1,\mathbf{y}^2,\ldots,\mathbf{y}^L\right]=\begin{bmatrix} y_1^1, y_1^2,\ldots, y_1^L \\ y_2^1, y_2^2,\ldots, y_2^L \\ \vdots,\vdots,\ldots,\vdots \\ y_C^1, y_C^2,\ldots, y_C^L \end{bmatrix} \quad (3)$$

where C is the number of classes. Every column of $\mathbf{Y}$ is a vector in $\mathbb{R}^C$, where the j^{th} coordinate corresponds to the j^{th} class.

$$y_j^i=\begin{cases}1 & \text{if } \mathbf{x}^i\in j^{\text{th}}\text{ class} \\ -1 & \text{if } \mathbf{x}^i\notin j^{\text{th}}\text{ class}\end{cases} \quad (4)$$

For example, if a review $\mathbf{x}$ is positive, y=[1, -1]'; else, $\mathbf{y}$ = [-1, 1]'.

We intend to seek the mapping function $\mathbf{X}^L\to\mathbf{Y}^L$ using the L labeled data and R+T-L unlabeled data. After training, we can determine $\mathbf{y}$ by the trained ADN while a new sample $\mathbf{x}$ is fed.

3.2 Semi-Supervised Learning

To address the problem formulated in section 3.1, we propose a novel deep architecture for ADN method, as show in Figure 1. The deep architecture is a fully interconnected directed belief nets with one input layer $\mathbf{h}^0$, N hidden layers $\mathbf{h}^1$, $\mathbf{h}^2$, …, $\mathbf{h}^N$, and one label layer at the top. The input layer $\mathbf{h}^0$ has D units, equal to the number of features of sample data $\mathbf{x}$. The label layer has C units, equal to number of classes of label vector $\mathbf{y}$. The numbers of units for hidden layers, currently, are pre-defined according to the experience or intuition. The seeking of the mapping function $\mathbf{X}^L \to \mathbf{Y}^L$, here, is transformed to the problem of finding the parameter space $\mathbf{W}=\{\mathbf{w}^1, \mathbf{w}^2,\ldots,\mathbf{w}^N\}$ for the deep architecture.

The semi-supervised learning method based on ADN architecture can be divided into two stages: First, AND architecture is constructed by greedy layer-wise unsupervised learning using RBMs as building blocks. All the unlabeled data together with L labeled data are utilized to find the parameter space $\mathbf{W}$ with N layers. Second, ADN architecture is trained according to the exponential loss function using gradient descent method. The parameter space $\mathbf{W}$ is retrained by an exponential loss function using L labeled data.

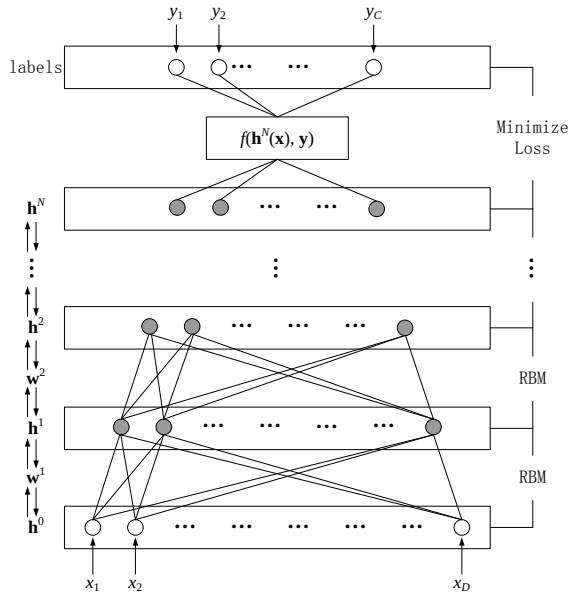

Figure 1. Architecture of Active Deep Networks

For unsupervised learning, we define the energy of the state ($\mathbf{h}^{k-1}$, $\mathbf{h}^k$) as:

$$E\left(\mathbf{h}^{k-1},\mathbf{h}^k;\theta\right)=-\sum_{s=1}^{D_{k-1}}\sum_{t=1}^{D_k} w_{st}^k h_s^{k-1} h_t^k - \sum_{s=1}^{D_{k-1}} b_s^{k-1} h_s^{k-1} - \sum_{t=1}^{D_k} c_t^k h_t^k \quad (5)$$

where $\theta=(\mathbf{w},\mathbf{b},\mathbf{c})$ are the model parameters: w_{st}^k is the symmetric interaction term between unit s in the layer $\mathbf{h}^{k-1}$ and unit t in the layer $\mathbf{h}^k$, k =1,…, N-1. b_s^{k-1} is the s^{th} bias of layer $\mathbf{h}^{k-1}$ and c_t^k is the t^{th} bias of layer $\mathbf{h}^k$. D_k is the number of unit in the k^{th} layer.

The probability that the model assigns to $\mathbf{h}^{k-1}$ is:

$$P\left(\mathbf{h}^{k-1};\theta\right)=\frac{1}{Z(\theta)}\sum_{\mathbf{h}^k}\exp\left(-E\left(\mathbf{h}^{k-1},\mathbf{h}^k;\theta\right)\right) \quad (6)$$

$$Z(\theta)=\sum_{\mathbf{h}^{k-1}}\sum_{\mathbf{h}^k}\exp\left(-E\left(\mathbf{h}^{k-1},\mathbf{h}^k;\theta\right)\right) \quad (7)$$

where $Z(\theta)$ denotes the normalizing constant. The conditional distributions over $\mathbf{h}^k$ and $\mathbf{h}^{k-1}$ are:

$$p\left(\mathbf{h}^k\mid\mathbf{h}^{k-1}\right)=\prod_t p\left(h_t^k\mid\mathbf{h}^{k-1}\right) \quad (8)$$

$$p\left(\mathbf{h}^{k-1}\mid\mathbf{h}^k\right)=\prod_s p\left(h_s^{k-1}\mid\mathbf{h}^k\right) \quad (9)$$

the probability of turning on unit t is a logistic function of the states of $\mathbf{h}^{k-1}$ and w_{st}^k:

$$p\left(h_t^k=1\mid\mathbf{h}^{k-1}\right)=\text{sigm}\left(c_t^k+\sum_s w_{st}^k h_s^{k-1}\right) \quad (10)$$

the probability of turning on unit s is a logistic function of the states of $\mathbf{h}^k$ and w_{st}^k:

$$p\left(h_s^{k-1}=1\mid\mathbf{h}^k\right)=\text{sigm}\left(b_s^{k-1}+\sum_t w_{st}^k h_t^k\right) \quad (11)$$

where the logistic function is:

$$\text{sigm}(\eta)=1/1+e^{-\eta} \quad (12)$$

The derivative of the log-likelihood with respect to the model parameter $\mathbf{w}^k$ can be obtained by the CD method (Hinton, 2002):

$$\frac{\partial \log p(\mathbf{h}^{k-1})}{\partial w_{st}}=\left\langle h_s^{k-1}h_t^k\right\rangle_{P_0}-\left\langle h_s^{k-1}h_t^k\right\rangle_{P_M} \quad (13)$$

where $\langle\cdot\rangle_{P_0}$ denotes an expectation with respect to the data distribution and $\langle\cdot\rangle_{P_M}$ denotes a distribution of samples from running the Gibbs sampler initialized at the data, for M full steps.

The above discussion is based on the training of the parameters between two hidden layers with one sample data $\mathbf{x}$. For unsupervised learning, we construct the deep architecture using all labeled data with unlabeled data by inputting them one by one from layer $\mathbf{h}^0$, train the parameter between $\mathbf{h}^0$ and $\mathbf{h}^1$. Then $\mathbf{h}^1$ is constructed, we

can use it to construct the up one layer $\mathbf{h}^2$. The deep architecture is constructed layer by layer from bottom to top, and in each time, the parameter space $\mathbf{w}^k$ is trained by the calculated data in the k-1th layer.

According to the $\mathbf{w}^k$ calculated above, the layer $\mathbf{h}^k$ can be got as below when a sample $\mathbf{x}$ is fed from layer $\mathbf{h}^0$:

$$h_t^k(\mathbf{x}) = \mathrm{sigm}(c_t^k + \sum_{s=1}^{D_{k-1}} w_{st}^k h_s^{k-1}(\mathbf{x})) \quad t = 1,\ldots,D_k \quad k = 1,\ldots,N-1 \quad (14)$$

The parameter space $\mathbf{w}^N$ is initialized randomly, just as backpropagation algorithm. Then ADN architecture is constructed. The top hidden layer is formulated as:

$$h_t^N(\mathbf{x}) = c_t^N + \sum_{s=1}^{D_{N-1}} w_{st}^N h_s^{N-1}(\mathbf{x}) \quad t = 1,\ldots,D_N \quad (15)$$

For supervised learning, the ADN architecture is trained by L labeled data. The optimization problem is formulized as:

$$\arg\min_{\mathrm{h}^N} \quad \mathrm{f}\left(\mathrm{h}^N\left(\mathbf{X}^L\right), \mathbf{Y}^L\right) \quad (16)$$

where

$$\mathrm{f}\left(\mathrm{h}^N\left(\mathbf{X}^L\right), \mathbf{Y}^L\right) = \sum_{i=1}^{L}\sum_{j=1}^{C} \mathrm{T}\left(\mathrm{h}_j^N\left(\mathbf{x}^i\right) y_j^i\right) \quad (17)$$

and the loss function is defined as

$$\mathrm{T}(r) = \exp(-r) \quad (18)$$

In the supervised learning stage, the stochastic activities are replaced by deterministic, real valued probabilities. We use gradient-descent through the whole deep architecture to retrain the weights for optimal classification.

3.3 Active Learning

Semi-supervised learning allows us to classify reviews with few labeled data. However, annotating the reviews manually is expensive, so we want to get higher performance with fewer labeled data. Active learning can help to choose those reviews that should be labeled manually in order to achieving higher classification performance with the same number of labeled data. For such purpose, we incorporate pool-based active learning with the ADN method, which accesses to a pool of unlabeled instances and requests the labels for some number of them (Tong and Koller, 2002).

Given an unlabeled pool $\mathbf{X}^R$ and a initial labeled data set $\mathbf{X}^L$ (one positive, one negative), the ADN architecture $\mathbf{h}^N$ will decide which instance in $\mathbf{X}^R$ to query next. Then the parameters of $\mathbf{h}^N$ are adjusted after new reviews are labeled and inserted into the labeled data set. The main issue for an active learner is the choosing of next unlabeled instance to query. In this paper, we choose the reviews whose labels are most uncertain for the classifier. Following previous work on active learning for SVMs (Dasgupta and Ng, 2009; Tong and Koller, 2002), we define the uncertainty of a review as its distance from the separating hyperplane. In other words, reviews that are near the separating hyperplane are chosen as the labeled training data.

After semi-supervised learning, the parameters of ADN are adjusted. Given an unlabeled pool $\mathbf{X}^R$, the next unlabeled instance to be queried are chosen according to the location of $\mathbf{h}^N(\mathbf{X}^R)$. The distance of a point $\mathbf{h}^N(\mathbf{x}^i)$ and the classes separation line $h_1^N = h_2^N$ is:

$$\mathbf{d}^i = \left|h_1^N\left(\mathbf{x}^i\right) - h_2^N\left(\mathbf{x}^i\right)\right| / \sqrt{2} \quad (19)$$

The selected training reviews to be labeled manually are given by:

$$s = \left\{ j : \mathbf{d}^j = \min\left(\mathbf{d}\right)\right\} \quad (20)$$

We can select a group of most uncertainty reviews to label at each time.

The experimental setting is similar with Dasgupta & Ng (2009). We perform active learning for five iterations and select twenty of the most uncertainty reviews to be queried each time. Then the ADN is re-trained on all of labeled and unlabeled reviews so far with semi-supervised learning. At last, we can decide the label of reviews $\mathbf{x}$ according to the output $\mathbf{h}^N(\mathbf{x})$ of the ADN architecture as below:

$$y_j = \begin{cases} 1 & \text{if } h_j^N(\mathbf{x}) = \max\left(\mathbf{h}^N(\mathbf{x})\right) \\ -1 & \text{if } h_j^N(\mathbf{x}) \neq \max\left(\mathbf{h}^N(\mathbf{x})\right) \end{cases} \quad (21)$$

As shown by Tong and Koller (2002), the BalanceRandom method, which randomly sample an equal number of positive and negative instances from the pool, has much better performance than the regular random method. So we incorporate this "Balance" idea with ADN method. However, to choose equal number of positive and negative instances without labeling the entire pool of instances in advance may not be practicable. So we present a simple way to approximate the balance of positive and negative reviews. At first, count the number of positive and negative labeled data respectively. Second,

for each iteration, classify the unlabeled reviews in the pool and choose the appropriate number of positive and negative reviews to let them equally.

3.4 ADN Procedure

The procedure of ADN is shown in Figure 2. For the training of ADN architecture, the parameters are random initialized with normal distribution. All the training data and test data are used to train the ADN with unsupervised learning. The training set $\mathbf{X}^R$ can be seen as an unlabeled pool. We randomly select one positive and one negative review in the pool to input as the initial labeled training set that are used for supervised learning. The number of units in hidden layer $D_1 \cdots D_N$ and the number of epochs Q are set manually based on the dimension of the input data and the size of training dataset. The iteration times I and the number G of active choosing data for each iteration can be set manually based on the number of labeled data in the experiment.

For each iteration, the ADN architecture is trained by all the unlabeled data and labeled data in existence with unsupervised learning and supervised learning firstly. Then we choose G reviews from the unlabeled pool based on the distance of these data from the separating line. At last, label these data manually and add them to the labeled data set. For the next iteration, the ADN architecture can be trained on the new labeled data set. At last, ADN architecture is retrained by all the unlabeled data and existing labeled data. After training, the ADN architecture is tested based on Equation (21).

The proposed ADN method can active choose the labeled data set and classify the data with the same architecture, which avoid the barrier between choosing and training with different architecture. More importantly, the parameters of ADN are trained iteratively on the label data selection process, which improve the performance of ADN. For the ADN training process: in unsupervised learning stage, the reviews can be abstracted; in supervised learning stage, ADN is trained to map the samples belong to different classes into different regions. We combine the unsupervised and supervised learning, and train parameter space of ADN iteratively. The proper data that should be labeled are chosen in each iteration, which improves the classification performance of ADN.

Active Deep Networks Procedure

Input: data $\mathbf{X}$
number of units in every hidden layer $D_1 \cdots D_N$
number of epochs Q
number of training data R
number of test data T
number of iterations I
number of active choose data for every iteration G
Initialize: $\mathbf{W}$ = normally distributed random numbers
$\mathbf{X}^L$ = one positive and one negative reviews

for i = 1 to I
Step 1. Greedy layer-wise training hidden layers using RBM
for n = 1 to N-1
for q = 1 to Q
for k = 1 to $R+T$
Calculate the non-linear positive and negative phase according to (10) and (11).
Update the weights and biases by (13).
end for
end for
end for
Step 2. Supervised learning the ADN with gradient descent
Minimize $f(h^N(\mathbf{X}),\mathbf{Y})$ on labeled data set $\mathbf{X}^L$, update the parameter space $\mathbf{W}$ according to (16).
Step 3. Choose instances for labeled data set
Choose G instances which near the separating line by (20)
Add G instances into the labeled data set $\mathbf{X}^L$
end
Train ADN with Step 1 and Step 2.

Output: ADN $h^N(\mathbf{x})$

Figure 2. Active Deep Networks Procedure.

4 Experiments

4.1 Experimental Setup

We evaluate the performance of the proposed ADN method using five sentiment classification datasets. The first dataset is MOV (Pang, et al., 2002), which is a widely-used movie review dataset. The other four dataset contain reviews of four different types of products, including books (BOO), DVDs (DVD), electronics (ELE), and kitchen appliances (KIT) (Blitzer, et al., 2007; Dasgupta and Ng, 2009). Each dataset includes 1,000 positive and 1,000 negative reviews.

Similar with Dasgupta and Ng (2009), we divide the 2,000 reviews into ten equal-sized folds randomly and test all the algorithms with cross-validation. In each folds, 100 reviews are random selected as training data and the remaining 100 data are used for test. Only the reviews in the training data set are used for the selection of labeled data by active learning.

The ADN architecture has different number of hidden units for each hidden layer. For greedy

layer-wise unsupervised learning, we train the weights of each layer independently with the fixed number of epochs equal to 30 and the learning rate is set to 0.1. The initial momentum is 0.5 and after 5 epochs, the momentum is set to 0.9. For supervised learning, we run 10 epochs, three times of linear searches are performed in each epoch.

We compare the classification performance of ADN with five representative classifiers, i.e., Semi-supervised spectral learning (Spectral) (Kamvar et al., 2003), Transductive SVM (TSVM), Active learning (Active) (Tong and Koller, 2002), Mine the Easy Classify the Hard (MECH) (Dasgupta and Ng, 2009), and Deep Belief Networks (DBN) (Hinton, et al., 2006). Spectral learning, TSVM, and Active learning method are three baseline methods for sentiment classification. MECH is a new semi-supervised method for sentiment classification (Dasgupta and Ng, 2009). DBN (Hinton, et al., 2006) is the classical deep learning method proposed recently.

4.2 ADN Performance

For MOV dataset, the ADN structure used in this experiment is 100-100-200-2, which represents the number of units in output layer is 2, in 3 hidden layers are 100, 100, and 200 respectively. For the other four data sets, the ADN structure is 50-50-200-2. The number of unit in input layer is the same as the dimensions of each datasets. All theses parameters are set based on the dimension of the input data and the scale of the dataset. Because that the number of vocabulary in MOV dataset is more than other four datasets, so the number of units in previous two hidden layers for MOV dataset are more than other four datasets. We perform active learning for 5 iterations. In each iteration, we select and label 20 of the most uncertain points, and then re-train the ADN on all of the unlabeled data and labeled data annotated so far. After 5 iterations, 100 labeled data are used for training.

The classification accuracies on test data in cross validation for five datasets and six methods are shown in Table 1. The results of previous four methods are reported by Dasgupta and Ng (2009). For ADN method, the initial two labeled data are selected randomly, so we repeat thirty times for each fold and the results are averaged. For the randomness involved in the choice of labeled data, all the results of other five methods are achieved by repeating ten times for each fold and then taking average on results.

Through Table 1, we can see that the performance of DBN is competitive with MECH. Since MECH is the combination of spectral clustering, TSVM and Active learning, DBN is just a classification method based on deep neural network, this result proves the good learning ability of deep architecture. ADN is a combination of semi-supervised learning and active learning based on deep architecture, the performance of ADN is better than all other five methods on five datasets. This could be contributed by: First, ADN uses a new architecture to guide the output vector of samples belonged to different regions of new Euclidean space, which can abstract the useful information that are not accessible to other learners; Second, ADN use an exponential loss function to maximize the separability of labeled data in global refinement for better discriminability; Third, ADN fully exploits the embedding information from the large amount of unlabeled data to improve the robustness of the classifier; Fourth, ADN can choose the useful training data actively, which also improve the classification performance.

Type	MOV	KIT	ELE	BOO	DVD
Spectral	67.3	63.7	57.7	55.8	56.2
TSVM	68.7	65.5	62.9	58.7	57.3
Active	68.9	68.1	63.3	58.6	58.0
MECH	76.2	74.1	70.6	62.1	62.7
DBN	71.3	72.6	73.6	64.3	66.7
ADN	**76.3**	**77.5**	**76.8**	**69.0**	**71.6**

Table 1. Test Accuracy with 100 Labeled Data for Five Datasets and Six Methods.

4.3 Effect of Active Learning

To test the performance of our proposed active learning method, we conduct following additional experiments.

Passive learning: We random select 100 reviews from the training fold and use them as labeled data. Then the proposed semi-supervised

learning method of ADN is used to train and test the performance. Because of randomness, we repeat 30 times for each fold and take average on results. The test accuracies of passive learning for five datasets are shown in Table 2. In comparison with ADN method in Table 1, we can see that the proposed active learning method yields significantly better results than randomly chosen points, which proves effectiveness of proposed active learning method.

Fully supervised learning: We train a fully supervised classifier using all 1,000 training reviews based on the ADN architecture, results are also shown in Table 2. Comparing with the ADN method in Table 1, we can see that employing only 100 active learning points enables us to almost reach fully-supervised performance for three datasets.

Type	MOV	KIT	ELE	BOO	DVD
Passive	72.2	75.0	75.0	66.0	67.9
Supervised	77.2	79.4	79.1	69.3	72.1

Table 2. Test Accuracy of ADN with different experiment setting for Five Datasets.

4.4 Semi-Supervised Learning with Variance of Labeled Data

To verify the performance of semi-supervised learning with different number of labeled data, we conduct another series of experiments on five datasets and show the results on Figure 3. We run ten-fold cross validation for each dataset. Each fold is repeated ten times and the results are averaged.

We can see that ADN can also get a relative high accuracy even by using just 20 labeled reviews for training. For most of the sentiment datasets, the test accuracy is increasing slowly while the number of labeled review is growing. This proves that ADN reaches good performance even with few labeled reviews.

5 Conclusions

This paper proposes a novel semi-supervised learning algorithm ADN to address the sentiment classification problem with a small number of labeled data. ADN can choose the proper training data to be labeled manually, and fully exploits the embedding information from the large amount of unlabeled data to improve the robustness of the classifier. We propose a new architecture to guide the output vector of samples belong to different regions of new Euclidean space, and use an exponential loss function to maximize the separability of labeled data in global refinement for better discriminability. Moreover, ADN can make the right decision about which training data should be labeled based on existing unlabeled and labeled data. By using unsupervised and supervised learning iteratively, ADN can choose the proper training data to be labeled and train the deep architecture at the same time. Finally, the deep architecture is re-trained using the chosen labeled data and all the unlabeled data. We also conduct experiments to verify the effectiveness of ADN method with different number of labeled data, and demonstrate that ADN can reach very competitive classification performance just by using few labeled data. This results show that the proposed ADN method, which only need fewer manual labeled reviews to reach a relatively higher accuracy, can be used to train a high performance sentiment classification system.

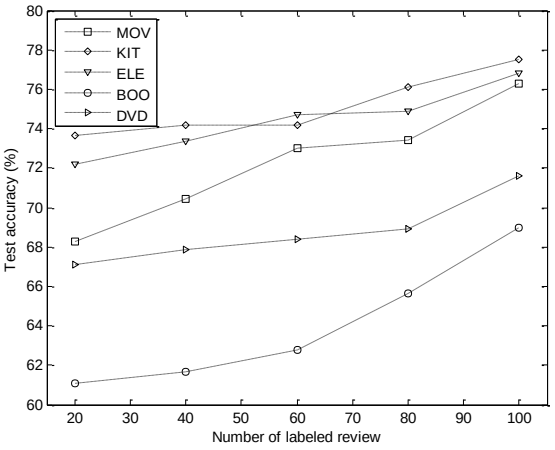

Figure 3. Test Accuracy of ADN with Different Number of Labeled Reviews for Five Datasets.

Acknowledgement

This work is supported in part by the National Natural Science Foundation of China (No. 60703015 and No. 60973076).

References

Bengio, Yoshua. 2007. *Learning deep architectures for AI*. Montreal: IRO, Universite de Montreal.

Blitzer, John, Mark Dredze, and Fernando Pereira. 2007. Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification. In *45th Annual Meeting of the Association of Computational Linguistics*.

Dasgupta, Sajib, and Vincent Ng. 2009. Mine the Easy, Classify the Hard: A Semi-Supervised Approach to Automatic Sentiment Classification. In *Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*.

Gamon, Michael. 2004. Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis. In *International Conference on Computational Linguistics*.

Hinton, Geoffrey E. . 2002. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8): 1771-1800.

Hinton, Geoffrey E. , Simon Osindero, and Yee-Whye Teh. 2006. A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, 18: 1527-1554.

Kamvar, Sepandar, Dan Klein, and Christopher Manning. 2003. Spectral Learning. In *International Joint Conferences on Artificial Intelligence*.

Li, Shoushan, and Chengqing Zong. 2008. Multi-domain Sentiment Classification. In *46th Annual Meeting of the Association of Computational Linguistics*.

Li, Tao, Yi Zhang, and Vikas Sindhwani. 2009. A Non-negative Matrix Tri-factorization Approach to Sentiment Classification with Lexical Prior Knowledge. In *Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*.

Pang, Bo, and Lillian Lee. 2004. A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts. In *42th Annual Meeting of the Association of Computational Linguistics*.

Pang, Bo, and Lillian Lee. 2008. *Opinion mining and sentiment analysis* (Vol. 2).

Pang, Bo, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment Classification using Machine Learning Techniques. In *Conference on Empirical Methods in Natural Language Processing*.

Raina, Rajat, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y. Ng. 2007. Self-taught learning: transfer learning from unlabeled data. In *International conference on Machine learning*.

Ranzato, Marc'Aurelio, and Martin Szummer. 2008. Semi-supervised learning of compact document representations with deep networks. In *International Conference on Machine learning*.

Salakhutdinov, Ruslan, and Geoffrey E. Hinton. 2007. Learning a Nonlinear Embedding by Preserving Class Neighbourhood Structure. In *Proceedings of Eleventh International Conference on Artificial Intelligence and Statistics*.

Sindhwani, Vikas, and Prem Melville. 2008. Document-Word Co-regularization for Semi-supervised Sentiment Analysis. In *International Conference on Data Mining*.

Tong, Simon, and Daphne Koller. 2002. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2: 45-66.

Wan, Xiaojun. 2009. Co-Training for Cross-Lingual Sentiment Classification. In *Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*.

Xia, Yunqing, Linlin Wang, Kam-Fai Wong, and Mingxing Xu. 2008. Lyric-based Song Sentiment Classification with Sentiment Vector Space Model. In *46th Annual Meeting of the Association of Computational Linguistics*.

Zagibalov, Taras, and John Carroll. 2008. Automatic Seed Word Selection for Unsupervised Sentiment Classification of Chinese Text. In *International Conference on Computational Linguistics*.

Zhu, Xiaojin. 2007. *Semi-supervised learning literature survey*. University of Wisconsin Madison.