A Noisy Sparse Convolution Neural Network Based on Stacked Auto-encoders

Yulin Ding1,2, Xiaolong Zhang*1,2, Jinshan Tang1,2,3

1School of Computer Science and Technology, Wuhan University of Science and Technology, Wuhan, China
2Intelligent Information Processing and Real-time Industrial Systems Hubei Province Key Laboratory, Wuhan, China
3School of Technology, Michigan Technological University, Houghton, MI, USA

*E-mail: xiaolong.zhang@wust.edu.cn

Abstract—Stacked auto-encoder is mainly used for image classification and it can extract valid information from data through unsupervised pre-training and supervised fine-tuning. This paper is intended to improve the accuracy of image classification, we constructed a 6-layer stacked convolution neural network (CNN) based on stacked auto-encoders. The constructed CNN can extract effective features for image classification through greedy layer-wise training. In order to make the constructed CNN to have strong robustness to noise, we added a noisy-layer in the pre-training stage. Adding the sparsity constraint can make the training of the CNN more effective, and can also reduce data redundancy. For classification applications, our experiments show that the final classification results of the proposed model is superior to the combination of auto-encoders and the noisy auto-encoders.

Keywords—deep learning; convolution neural networks; stacked auto-encoders; noisy auto-encoders; sparsity constrain

I. INTRODUCTION

Recently, deep learning technology has caused a great attention and has wide applications in different fields. Deep learning can be divided into two types: supervised learning and unsupervised learning. Deep convolution neural networks (CNN) [1] is a supervised learning while deep Belief Nets (DBN) [3] is an unsupervised learning.

Deep CNN is a wide deep learning technology being used in image recognition. It has several advantages. For example, training of a deep CNN needs less training parameters and it also has strong stability. This structure of CNN is also highly invariant to translation, scale, tilt and twist.

Besides Deep CNN, there are many other deep learning technologies based on Auto-encoders. For example, in 2010, Vincent et al [2] proposed the Stacked Denoising Auto-encoders (sDAE). The basic idea of sDAE is to use the deep belief network to stack multiple denoising auto-encoders. In sDAE, noise is added to the input, and is only used for initial training in each layer. The sDAE is very effective for feature extraction and it has been shown to be superior to deep belief network to some extent.

Based on the idea of stacked auto-encoders, we propose a noisy sparse stacked convolution neural network algorithm (denoted by nssCAE). The proposed algorithm combines the advantages of both auto-encoder and local connection of CNN. This combination can make the deep neural network be more stable in the identification of displacement, various forms of distorted two-dimensional graphics, etc. The proposed nssCAE is also robust to noise interference. Thus, it can achieve better classification performance than other methods.

II. RELATED WORK

A. Auto-encoder

Auto-encoder (AE) [8] is an unsupervised learning method which reconstructs the input signal at the output side. It consists of two parts: encoder part and decoder part. An auto-encoder typically employs a three layers network. The encoding part is from the input layer to the hidden layer, and the decoding part is from hidden layer to the output layer.

Let \( f(x), g(x) \) be the encoding and decoding functions, respectively, which can be represented as [8]

\[
h = f(x) := \sigma(wx + b_1) \quad (1)
\]

\[
y = g(h) := \sigma(\widetilde{w}h + b_2) \quad (2)
\]

where \( \sigma \) is an active function and the weight \( \widetilde{w} \) is usually set to \( w^T \). The input \( x \) is treated as the target value of the output \( y \), and the network will be trained by updating the network parameters \( \theta = \{w, b_1, b_2\} \) to minimize the error between \( x \) and \( y \). For error function, cross-entropy function is generally used. Let \( L(x, y) \) be cross-entropy function and it can be written as follows [8].

\[
L(x, y) = -\sum_{i=1}^{n} [x_i \cdot \ln(y_i) + (1 - x_i) \cdot \ln(1 - y_i)] \quad (3)
\]

In summary, the AE cost function on training set \( S = \{x_{i}^{'}\}_{i=1}^{N} \) can be represented by
\[
J(\theta) = \sum_{x \in \mathcal{S}} L(x, g(f(x))). \quad (4)
\]

By minimizing the cost function (using gradient descent algorithm), a compressed representation of a given input is found. The representation can capture the relatively important features from the input data and it will be benefit to classification.

B. Denoising Auto-encoders

Denoising Auto-encoders (DAE) [4] is an improved version of AE. The basic idea of DAE is to add noise to the training data and uses artificial noisy data to train a compressed representation of the original input. Fig. 1 shows the basic idea of DAE algorithm. By adding noise to the original input \( x \), we can get the partially corrupted data \( \tilde{x} \). In DAE, the choice of noise types is important. Currently, two ways are mainly employed for generating noisy data. One is to add Gaussian noise and the other one is to add binary masking noise [4].

Fig. 1. Denoising auto-encoders algorithm structure.

C. Stacked Denoising Auto-encoders

Stacked denoising auto-encoders [2] are the extension of DAE by stacking the network structure. In the structure of a stacked auto-encoder, the corrupted input layer is used for reducing the impact of noise on training at each layer. Once the training in this layer is finished, the output will be used as the input for the next layer. In a stacked auto-encoder, the stacking of the learning can make the network learn more representative features.

D. Noisy Auto-encoders

Stacked noisy auto-encoders (SNAE) [6] is an extension of DAE. The basic idea is that a noise is embedded into the term between the encoding and decoding process. A noisy auto-encoder is more robust than a DAE and can prevent a network from over-fitting. This has shown the robustness of the intermediate layer feature was more important than the robust reconstruction.

In a noisy auto-encoder, the choice of noise types is also important. Besides the two noise models described in the denoising auto-encoders, the noise used in hidden layer can also use dropout noise [9].

III. ALGORITHM

In a stacked auto encoders, the training is not very efficient because of the large number of parameters. In order to make the training more effective, we combine CNN with stacked encoder in this paper and developed a new learning structure called noisy sparse stacked convolution auto-encoders. The improvement of the training efficiency of the combination lies in that the features in CNN are extracted by a set of untrained weights and are used as the input to the next level for training directly.

A. Stacked Convolution auto-encoders

In convolution auto-encoders, the weights are locally connected and thus can be shared among all the locations in the input. The advantage can reduce the number of training parameters in the network. However, in the fine-tuning stage, when back-propagation algorithm [5] is used to train the network, the increase of the depth of the network and the magnitude of the gradient of the back propagation will drastically decrease and this will cause a very small derivative of the weights in the first few layers. Therefore, the change of weights will be very small when a gradient descent method is used to train the network. This will affect the learning effectiveness from samples. In order to solve this issue, greedy layer-wise training methods of stacked auto-encoders are used to reduce the effects of gradient diffusion. The structure using greedy layer-wise training methods of stacked auto-encoders can find the duplicate, important local features.

Besides the weight sharing, the new structure is similar to the auto-encoder. For convenient discussion, let us introduce some mathematics used in the previous research. Based on this mathematics, we will develop our algorithm.

For an input \( x \), the hidden layer of the \( m \textsuperscript{th} \) feature map is described as follows

\[
h_m = \sigma(x \ast w_m + b_1) \quad (10)
\]

where \( \sigma \) is the active function (tanh function is used in this paper), \( \ast \) means 2D convolution.

The output of the convolution layer can be described as

\[
y = \sigma\left(\sum_{m \in H} h_m \ast \tilde{w}_m + b_2\right) \quad (11)
\]

where \( \tilde{w}_m \) represents the transposition of weights in two dimensions, \( H \) is the number of hidden layer feature maps. The reconstruction error function (see equation (3)) is used in this paper.

B. Noisy Auto-encoders

The previous research work [6] has shown that adding noise to the input layer and the hidden layer can improve the robustness of the network and can also prevent network from over-fitting. The basic idea can be explained as follows: adding noise to the hidden layer can be understood as to strengthen the hidden memory layer, so that the network has better robustness. In dealing with test sets, adding noise can make the network find common features more effectively and thus enhance the classification effect. In this paper, we follows...
this idea by adding double-layer noise to the stacked convolution auto-encoders. Fig. 2 is the convolution auto-encoder with noisy auto-encoders.

Fig. 2. The structure of convolution auto-encoder with noisy auto-encoders.

C. Sparsity Constrain

When a CNN trained by layer-wise, there may be some redundancy in the network structure. In order to avoid this kind of redundancy, we will not add any independent constraints to the convolution kernel. Instead, we propose adding sparsity constraint to the fine-tuning process so as to improve this efficiency. The fine-tuning process is measured by the whole network loss as a direct error, thus the classification accuracy is expected to be improved effectively.

Let the input of an hidden layer be \( x \) and \( h_m(x) \) denote the \( m \)-th neuron activation in this layer. Let the average activation of the \( m \)-th neurons on the hidden layer over training set \( S = \{x_i\}_{i=1}^n \) is represented by \( \hat{\rho}_m \), which is defined as

\[
\hat{\rho}_m = \frac{1}{N} \sum_{i=1}^{N} h_m(x^{(i)}).
\]  

In order to make the fine-tuning neurons satisfy sparsity constrain, we can set \( \hat{\rho}_m = \rho (m = 1, 2, 3, ..., j) \). Here \( \rho \) is a sparsity parameter and is set to be \( \rho = 0.05 \) in this paper, and \( j \) is the number of neurons.

When the difference between \( \hat{\rho}_m \) and \( \rho \) becomes larger, Kullback-Leibler divergence function is taken for punishment.

\[
KL (\hat{\rho}_m \| \rho) = \rho \times \ln \left( \frac{\rho}{\hat{\rho}_m} \right) + (1 - \rho) \times \ln \left( \frac{1 - \rho}{1 - \hat{\rho}_m} \right) \tag{13}
\]

This value of this function will increase as the difference between \( \hat{\rho}_m \) and \( \rho \) increases. We can reduce the gap between \( \rho \) and \( \hat{\rho}_m \) by minimizing this function value to achieve sparseness constraint, and also obtain more effective features.

D. Algorithm Description

The proposed algorithm (nssCAE) is applied to the classification task, and its training process is composed of two stages: pre-training and fine-tuning. The algorithm is described as follows:

Input: the picture vector in data set.
Output: the test set error rate.

1) Construct a model of a convolution layer.
2) Reconstruct the convolution layer in 1) using auto-encoder. Using equation (3) as the reconstruction error function in the pre-training stage of the model.
3) Adding noise into the current input layer and the next convolution layer.
4) For each hidden layer in nssCAE;
   a) Set the pretraining epochs as the pre-training iteration times;
   b) Set the batch size;
   c) Execute operations in 2), 3) to complete the training of the CNN;
   d) Insert a pooling layer after the convolution layer.
5) Get the parameter set \( \theta = \{w, b_1, b_2\} \) in each hidden layer after the network is trained.
6) Output the average cost per epoch.
7) Adding sparsity constrain to the fine-tuning stage.
8) Through the iterations, find the best validation loss. After the parameters are updated, calculate the error rate on the test set.
9) Until the iteration number is more than a preset maximum number or the done looping parameter is set to true.
10) Output the best error rate of the test set.

The nssCAE algorithm is mainly divided into two stages: pre-training and fine-tuning. Steps 1) ~ 6) describe the pre-training process, which is mainly used to determine the training parameters \( \theta \) of the network. Steps 7) ~ 10) describe the fine-tuning process. The nssCAE and softmax regression layer are treated as a whole to optimize the network.

<table>
<thead>
<tr>
<th>TABLE I. PARAMETER DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter Name</td>
</tr>
<tr>
<td>convolution layer active function</td>
</tr>
<tr>
<td>reconstruct active function</td>
</tr>
<tr>
<td>sparsity active function</td>
</tr>
<tr>
<td>pre-training learning rate</td>
</tr>
<tr>
<td>tuning learning rate</td>
</tr>
<tr>
<td>corruption layer 1</td>
</tr>
<tr>
<td>corruption layer 2</td>
</tr>
<tr>
<td>tuning threshold</td>
</tr>
<tr>
<td>( \rho ) in KL divergence</td>
</tr>
</tbody>
</table>
Table I shows the various parameter values in the algorithm. In this experiment, the parameters are mostly empirically selected, only as a reference value, does not represent the optimal values, and which can be changed according to the actual situation.

IV. EXPERIMENT

A. Data Sets

Table II lists the six data sets used in the experiments. The mnist dataset is a handwritten digital dataset containing 50,000 training examples, 10,000 valid examples, and 10,000 test examples. The mnist dataset consists of images with the size of 28 by 28. The other five data sets are acquired from mnist dataset. The number of category is 10, indicating that each dataset is a multiple classification task, and a total of 10 digits (0-9) are identified. In order to compare the proposed algorithm with sparse marginalized denoising auto-encoders (SmDAE) [7] and SNAE [6], these experiments used the same number for training, validation, and test sets.

<table>
<thead>
<tr>
<th>Data sets</th>
<th>Description of Data sets</th>
<th>Description of input</th>
<th>category</th>
<th>Train-Valid-Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>mnist</td>
<td>mnist handwritten digital data set</td>
<td>784 dimension gray scale vector, which is in the range [0,1]</td>
<td>10</td>
<td>50000:10000:10000</td>
</tr>
<tr>
<td>basic</td>
<td>Sub data set of the mnist</td>
<td></td>
<td>10</td>
<td>10000:2000:5000</td>
</tr>
<tr>
<td>rot</td>
<td>mnist data set with random rotation operation</td>
<td></td>
<td>10</td>
<td>10000:2000:5000</td>
</tr>
<tr>
<td>bg-rand</td>
<td>mnist data set with random noise background</td>
<td></td>
<td>10</td>
<td>10000:2000:5000</td>
</tr>
<tr>
<td>bg-img</td>
<td>mnist data set with random image background</td>
<td></td>
<td>10</td>
<td>10000:2000:5000</td>
</tr>
<tr>
<td>bg-img-rot</td>
<td>mnist data set randomly added to image background and randomly rotated</td>
<td></td>
<td>10</td>
<td>10000:2000:5000</td>
</tr>
</tbody>
</table>

B. Results

This algorithm is trained on a computer with one CPU, and it is a 6-layer neural network structure. Two convolution layers, both followed by the maximum pooling layer. After that there are a fully connected layer and a softmax classifier. In the pre-training phase, after randomly selected several sets of noise ratios, we found that adding 20% and 10% noise to the input layer and hidden layers respectively can get better results.

Finally, the classification error rate of the test set in the mnist dataset and its sub datasets is compared with other algorithms, and the Table III shows the average results for each algorithms.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>CNN</th>
<th>SmDAE</th>
<th>SNAE</th>
<th>mssCAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>mnist</td>
<td>1.36</td>
<td>1.14</td>
<td>1.13</td>
<td>0.92</td>
</tr>
<tr>
<td>basic</td>
<td>2.92</td>
<td>2.66</td>
<td>2.56</td>
<td>2.22</td>
</tr>
<tr>
<td>rot</td>
<td>11.024</td>
<td>10.32</td>
<td>9.39</td>
<td>9.774</td>
</tr>
<tr>
<td>bg-rand</td>
<td>11.096</td>
<td>10.91</td>
<td>10.86</td>
<td>5.6</td>
</tr>
<tr>
<td>bg-img</td>
<td>17.132</td>
<td>16.36</td>
<td>16.23</td>
<td>9.068</td>
</tr>
<tr>
<td>bg-img-rot</td>
<td>32.784</td>
<td>\</td>
<td>43.35</td>
<td>31.77</td>
</tr>
</tbody>
</table>

The hierarchical model constructed by this algorithm is the same as SmDAE. For both models, we constructed two hidden layers, using the same reconstruction error function to calculate the error between current layer network output and the original input. In both of the model, back propagation algorithm is used to minimize the reconstruction error. In the fine-tuning phase, we used the softmax classifier to get the results. Compared with SNAE, both of these two models add double noise to each hidden layer.

According to the results, the proposed algorithm is superior to the other two algorithms on most datasets. Therefore, this paper speculates that by adding two layers of noise and sparsity constraint simultaneously and by layer-wise training the network, we can get better results.

For the dataset: rot, after several more sets of experiments, we found that, we can reduce the error rate to 9.416% when we set the noise ratios to 10%, 5% and the fine turning learning rate is 0.11. Although the training result is still inferior to the SNAE algorithm, we speculate that the main reason for this is that SNAE constructed a three hidden layer model, and this algorithm only contains two hidden layers. Anyway, this still indicates the experiment results can be improved to some extent by parameter adjustment.

V. SUMMARY

In this paper, we combine the convolution with the stacked auto-encoders and add the noisy and sparsity constraint to construct a noisy sparse stacked convolution auto-encoders to learn the deep data features. We use the characteristics of convolution neural networks with weight sharing, and the greedy layer-wise pre-training method to make the whole network generalization performance better. Besides, the double noise and sparsity constrain can enhance its robustness and reduce redundancy to a certain extent, so as to achieve better classification results.

ACKNOWLEDGMENT

This work was supported in part by National Natural Science Foundation of China (61273225, 61273303, 6137109), Program for Outstanding Young Science and Technology Innovation Teams in Higher Education Institutions of Hubei Province (No. T201202), as well as National “Twelfth Five-Year” Plan for Science & Technology Support (2012BAC22B01).

REFERENCES


