We get the result that standard error is 2.243445 through the 6 observed value. Coefficient of determination R_2 is very close to 1. It shows the test value and the actual value is very close, and we know our camera calibration value is precision.

Conclusions

This article uses the RAC algorithm to make computer vision automatic calibration, and taking pictures for instance simulation of real life. Measure different object in the image, and then compared with the actual value, verify the algorithm performance. By this method, we can know this method is automatic calibration. Through the analysis of the goodness of fit, the error between measured value and the actual value is small. Calibration precision is high, and has certain market application prospect.

References

1. Jiang G.W., Chao C.Z., Fu S.H., Yu Q.F. (2010) Like Machine Precision Calibration Technique based on Controllable Rotation. *Journal of optics*, 11(9), p.p.1308-1314.
2. Xie Y.H., Wang C.J., Huang S.H. (2004) A New Technology to Realize the Automatic Computer Vision Calibration. *Journal of huazhong university of science and technology* (*natural science edition*), 13 (11), p.p.13-15.
3. Wu T., Xu X.J., Zhao R. (2005) Camera Calibration Method to Explore in the Space Robot. *Journal of hefei university of technology (natural science edition)*, 8(6), p.p.20-25.
4. Ma Y., Wang Z.H., Gong L. (2012)CCD Camera Parameter Calibration Method based on RAC Calibration Technology Research. *Optics*, 7(12), p.p.11-15.
5. Chen L, Chan C., Duan X.W., Zhang L. (2009) Binocular Positioning in the Application of Digital Camera Image Calibration. *Computer and digital engineering,* 4(2), p.p.155-157.
6. Bai X., Lin H., Zou L. (2012) Considering the Radial Distortion of the Parameters of the Robot Vision System Calibration. *Computer engineering and design*, 8(7), p.p.4407-4411.

Defects Detection Based on Deep Learning and Transfer Learning

Liu Ri-Xian[1, 2]

[1] College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, Zhejiang, China
[2] JinHua Polytechnic, Jinhua 321017, Zhejiang, China

Yao Ming-Hai

College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, Zhejiang, China

Wang Xian-Bao

College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, Zhejiang, China

Abstract
Defect detection is an important step in the field of industrial production. Through the study of deep learning and transfer learning, this paper proposes a method of defect detection based on deep learning and transfer learning. Our method firstly establishes Deep Belief Networks and trains it according to the source domain sample feature, in order to obtain the weights of the network according to source domain samples. Then, the structure and parameters of the source domain DBN is transferred to the target domain and target domain samples are used to fine-tune the network parameters to get the mapping relationship between the target domain training sample and defect-free template. Finally, the defects of testing samples will be detected by compared with the reconstruction image. This method not only can make full use of the advantages of DBN, also can solve over-fitting in DBN network training through parameters transfer learning. These experiments show that DBN is a successful approach in the high-dimensional-feature-space information extraction task, which can perfectly establishes the mapping relationship, and can quickly detect defects with a high accuracy.
Key words: DEEP LEARNING, TRANSFER LEARNING, DEFECT DETECTION, DBN

1. Introduction

In industrial processes, scratches, chips, cracks and other visual defects may occur on product surfaces due to factors like raw materials, equipment and processing technologies, which affect the quality and performance of products to certain extents. Defect detection, as an important aspect of industrial production, provides an important guarantee for the quality of products. Currently, many Chinese companies adopt the traditional manual approach for inspection of surface quality. Manual inspection is a type of contact inspection, which not only has low degree of automation, high labor intensity, low efficiency and high missed detection probability1, but also features greatly varying standards, which thus cannot meet the requirements of modern industry. With the development of computer vision technology, machine vision has gained application in the industrial product inspection. Machine vision-based defect detection system detects the color, shape and texture of products by machine vision technology, which is a non-contact inspection that does not require direct contact with products. Since the products inspected are not subject to secondary pollution, it is a truly environmentally-friendly inspection system.

Many researchers at home and abroad are now working on a variety of product defect recognition algorithms. There are two major methods for defect recognition: image processing-based method and pattern recognition & classification method. Image processing technology-based [3] defect recognition firstly employs a range of image preprocessing methods such as median filtering, iterative thresholding and Sobel operator to effectively extract image features, and then identifies and classifies defects with support vector machines or other classifiers. Pattern recognition & classification-based method [1,2,4], on the other hand, firstly extracts feature parameters with distinct categorical differences in shape, color and texture, respectively, based on the visual defect image characteristics of products to constitute defect feature vectors, and then identifies product defects using BP neural networks or other classifiers. The above two defect detection methods achieve recognition, classification and detection of defects both through a "feature representation" + "classifier" framework, where the collected capsule images are preprocessed firstly, and the corresponding features are extracted thereafter to perform classification detection with classifiers. The advantage is that it can well handle the detection of a particular defect of products; however, it cannot be used universally for different visual defects of products. Visual defect recognition methods vary largely for different products; moreover, product feature representation is extracted through the manually designed features, which is not expandable.

Deep learning is a new domain in the field of machine learning in recent years. Also known as unsu-

pervised learning, depth learning allows autonomic learning of implicit relationships within the data through multilayered deep neural networks and massive data for learning multilayered representation of data samples, simulates the problem analyzing process of the human brain, and forms more abstract high-level feature representation by combining and learning low-level data, in order to improve the accuracies of subsequent recognition and classification. This paper proposes a deep learning- and transfer learning-based defect detection method through the study on deep learning and transfer learning. This method firstly obtains the mapping relationship from training samples into defect-free template by learning a large number of defect samples, and then implements product defect detection by contrast between reconstructed and defect images. On the other hand, it also achieves transfer cross learning between different products in the DBN training process through parameter transfer learning. The structure of this paper is organized as follows: the first part of the paper describes the principles and methods of deep learning and transfer learning; part two designs and implements the deep learning- and transfer learning-based defect detection methods and processes based on the analysis of deep learning- and transfer learning-based defect detection; part three conducts experiments using capsule and solar cell samples, and then analyzes and compares the experimental results; and part four is the summary and outlook.

2. Study Models

Deep learning is a new research domain in machine learning, which was proposed by Hinton et al. on Science in 2006 [5]. The essence of deep learning is to learn effective features by building learning model with multiple hidden layers and vast amounts of training data, so as to enhance the accuracy of classification or prediction. Its differences from the traditional shallow learning are as follows: 1) deep learning emphasizes the depth of model structure, with hidden layers of deep networks generally counting 5, 6 or even over 10; 2) deep learning highlights the importance of feature learning. In deep learning, learning of features is a data-driven automatic learning of implicit relationships within the data; features learned this way are expansive and expressive. Deep learning models can be divided into supervised and unsupervised deep learning models, where the supervised learning models are represented by Convolutional Neural Network (CNN), and the unsupervised deep learning models are represented by Deep Belief Network (DBN). Application objects of deep learning in the field of signal processing [6-7] include voices, images, videos, texts, etc. In addition, deep learning has also achieved good progress in research directions like human behavior prediction [8], medical diagnosis, drug development, search advertising and autopilot.

2.1 Deep Belief Network (DBN)

2.1.1 Restricted Boltzmann Machine

Restricted Boltzmann Machine (RBM) [10] is a generative stochastic neural network proposed by Smolensky in 1986, which consists of visible layer and hidden layer. Visible layer unit is mainly used to describe an aspect or a feature of observed data, while the hidden layer unit, also known as feature extraction layer, generally has unclear meaning, which can be used to obtain the dependency relationships between variables corresponding to visible layer unit. As an energy-based model, RBM can define an energy function so that every situation in the parameter space has a corresponding scalar energy, which means that the energy function is a mapping from parameter space to energy. Assume that a RBM has n number of visible units $v=(v_1,v_2,\ldots,v_n)$ and m number of $h=(h_1,h_2,\ldots,h_m)$, and that a set of states (v,h) is given, the energy function RBM can be defined by equation (1) as:

$$E(v,h)=-\sum_{i=1}^{n}a_i v_i-\sum_{j=1}^{m}b_j h_j-\sum_{i=1}^{n}\sum_{j=1}^{m}v_i w_{ij} h_j \quad (1)$$

Where a_i is the bias of visible layer, v is i-th visible unit, v_i is the state of i-th visible unit; b_j is the bias of hidden layer, h is j-th hidden unit, h_j is the state of j-th hidden unit; and W_{ij} denotes the connection weight between the visible unit v_i and the hidden unit h_j.

Based on the energy function defined in (1), the joint probability distribution of state (v,h) can be defined by equation (2) as:

$$P(v,h)=\frac{1}{Z}\exp(-E(v,h)) \quad (2)$$

$$Z=\sum_{v,h}\exp(-E(v,h))$$

Where is the normalization constant (also known as the partition function).

According to the intra-layer connectionless condition of RBM, when the state of visible layer units is given, there is a conditional independence between activated states of various hidden units. The probability that a certain neuron on the hidden layer is activated (i.e. value setting of 1) can be defined by equation (3) as:

$$P(h_k=1\mid v)=\sigma(b_k+\sum_{i=1}^{n}v_i w_{ik}) \quad (3)$$

When the state of hidden layer units is given, the probability that the k-th unit v_k on visible layer is activated is defined by equation (4):

$$P(v_k = 1 \mid v) = \sigma(a_k + \sum_{j=1}^{m} w_{kj} h_j) \tag{4}$$

For a practical problem, training of RBM is needed after giving training samples to allow the probability distribution represented by RBM to be consistent with the training data as far as possible. Training of a RBM means to fit the given training data by adjusting RBM parameters W_{ij}, a_i, b_j . Assume that the training sample set is $S = \{v^1, v^2, \ldots v^T\}$, where T is the number of training samples, then maximization of log-likelihood equation (5) is set as the RBM training objective:

$$\theta^* = \arg\max_{\theta} L(\theta) = \arg\max_{\theta} \sum_{t=1}^{T} \ln P(v^{(t)} \mid \theta) \tag{5}$$

2.1.2 Contrastive Divergence

By deriving the partial derivatives of equation (5) with respect to parameters using gradient descent algorithm, respectively, equations (6) (7) (8) are obtained:

$$\frac{\partial \sum_{t=1}^{T} \ln P(v^t)}{\partial w_{ij}} = \sum_{t-1}^{T} [P(h_i = 1) \mid v^t) v_j^t - \sum_{v} P(v) P(h_i = 1 \mid v) v_j] \tag{6}$$

$$\frac{\partial \sum_{t=1}^{T} \ln P(v^t)}{\partial a_i} = \sum_{t=1}^{T} [v_i^t - \sum_{v} P(v) v_i] \tag{7}$$

$$\frac{\partial \sum_{t=1}^{T} \ln P(v^t)}{\partial b_j} = \sum_{t=1}^{T} [P(h_i = 1) \mid v^t) - \sum_{v} P(v) P(h_i = 1 \mid v)] \tag{8}$$

In equations (6) (7) (8), computational complexity regarding $\sum$ is $O(2^{n+m})$. Therefore, Hinton proposed a fast RBM learning algorithm in the literature [14], i.e. contrastive divergence (CD) algorithm. The procedure of K-step CD algorithm can be described as follows: for $\forall v \in S$, set the initial value $v^{(0)} := v$; then perform the k-step Gibbs sampling, where t-th step is ($t = (1, 2, \ldots, k)$; after Gibbs sampling, successively perform: 1) sampling out of $h^{(t-1)}$ using $P(h \mid v^{(t-1)})$; 2) sampling out of $v^{(t)}$ using $P(v \mid h^{(t-1)})$. The objective of k-step CD algorithm is to obtain the approximations of partial derivatives $\Delta w, \Delta a, \Delta b$ during gradient ascent.

2.1.3 RBM Training

Given that the training sample set is $S = \{v^1, v^2, \ldots v^T\}$, where T is the number of training samples, then the CD-based RBM algorithm can be described as:

Inputs of algorithm: training sample $v^{(i)}$; learning rate η ; CD-k algorithm parameter k; number of visible layer units n; and number of hidden layer units m.

Outputs of algorithm: connection weight between visible and hidden layers w; bias vector of visible layer a; and bias vector of hidden layer b;

Initialization of algorithm:

(1)Set the training cycle I, learning rate η and CD-k algorithm parameter k;

(2)Specify the numbers of visible and hidden layer units n, m;

(3)Initialize the state of visible layer unit $v_1 = v^{(i)}$;

(4)Initialize the bias vector of visible layer a, bias vector of hidden layer b and weight matrix w;

RBM training procedure:

For $i = 1, 2, \ldots, I$ Do

(1)Obtain the approximations of $\Delta w, \Delta a, \Delta b$ using CD-k algorithm, specifically:

For $k = 1, 2, \ldots K$ (CD algorithm parameter K)

For $j = 1, 2, \ldots m$ (hidden unit j)

Calculate $P(h_{1j} = 1 \mid v_1) = \sigma(b_j + \sum_i v_{1i} w_{ij})$

End

For $i = 1, 2, \ldots n$ (visible layer unit i)

Calculate $P(v_{2i} = 1 \mid h_1) = \sigma(a_i + \sum_j h_{1j} w_{ij})$

End

For j 1, 2, m (hidden unit j)

Calculate $P(h_{2j} = 1 \mid v_2) = \sigma(b_j + \sum_i v_{2i} w_{ij})$

End

End

(2)Update the parameters

$w = w + \eta(\frac{1}{T}\Delta w), a = a + \eta(\frac{1}{T}\Delta a), b = b + \eta(\frac{1}{T}\Delta b)$

End

An already learned or being learned RBM can be evaluated through reconstruction error. Reconstruction error refers to the difference from the original data after completion of a Gibbs transfer through RBM distribution with training sample as the initial state. Reconstruction error reflects the RBM likelihood of training samples to some extent, which is computationally simple as well.

2.1.4 DBN Training

DBN is a type of deep neural network consisting of multi-layer RBMs, which is a generative model that allows the deep neural network to generate training data according to the maximum probability by training the weights between neurons. In DBN, the bottom layer RBM receives the original feature vector, and the output of bottom layer RBM is taken as the input of one upper layer RBM. During the bottom-up transmission process, the concrete feature vector is gradually transformed into abstract feature vector.

DBN training comprises two major steps: in step 1, each layer of RBM network is trained separately without supervision; in step 2, the entire DBN network is fine-tuned by back-propagation. RBM network training is the process of initializing DBN network's weight parameters, which allows DBN to overcome the shortcomings of easy entrapment into local optimum and long training time resulted from random weight initialization in BP networks.

Specific procedure of DBN training:

(1) As a first step, fully train the first RBM;

(2)Fix the weight and offset of the first RBM, and then use its hidden neuron state as the input vector of the second RBM;

(3)After fully training the second RBM, stack the second RBM on top of the first RBM;

(4) Repeat the above three steps any number of times;

(5) If the data in the training set are labeled, during the training of top layer RBM, there should be neurons representing categorical labels in the visible layer of top layer RBM apart from visible neurons, which should be trained together.

2.2 Transfer Learning

Transfer learning refers to setting the corresponding learning task T_S of source domain D_S and the corresponding learning task T_T of target domain D_T, whose objective is to use the knowledge in D_S and T_S to improve the performance of target prediction function $f_T(\bullet)$ in D_T and T_T. Transfer learning aims to transfer the knowledge learned from one domain to a new domain to facilitate the leaning of tasks in the new domain. Parameter transfer method generally assumes that the source and target task models share some parameters, and that the parameter w of each source and target problem domain task can be divided into two parts, of which one part is common among tasks, and the other part is unique to each task, that is:

$$P(h_{1j}=1\,|\,v_1)=\sigma(b_j+\sum_i v_{1i}w_{ij}) \quad (9)$$

$$P(h_{1j}=1\,|\,v_1)=\sigma(b_j+\sum_i v_{1i}w_{ij}) \quad (10)$$

w_S and w_T in equations (9) and (10) are the parameters of learning source and target tasks, respectively. w_c is the common parameter, whereas v_s and v_T are the source task- and target task-specific parameters. Parameter transfer establishes the connection between target and source tasks by mining the model parameters shared between the source and target domains.

3. Deep Learning- and Transfer Learning-based Defect Detection

3.1 Image Preprocessing

Main purposes of image preprocessing are to enhance the image contrast, and remove noise in images.

Image preprocessing done in this paper includes image normalization and image contrast enhancement: (1) image gray scaling and normalization. Firstly, the sample images are grayed to leave only the gray scale information; then, the sizes of images are normalized to 64*64, so as to reduce the amount of image data. (2) Contrast of images is enhanced, and the images are gray scale transformed to allow large difference between product gray scale and background.

3.2 Construction and Training of Source Problem Domain DBN

Structure of DBN network is determined based on the features of source problem domain samples. In this paper, solar cell is used as the source problem domain. The DBN network adopted has four hidden layers, and the training samples contain both defective and non-defective images. Input layer of the DBN network is 4,096 dimensional, besides, the first hidden layer is 3,000 dimensional, the second hidden layer is 1,500 dimensional, the third hidden layer is 750 dimensional, and the fourth hidden layer is 90 dimensional. The structure of DBN is shown in Figure 1.

After determining the structure of source problem domain DBN, samples of source problem domain are input for training and learning DBN. The specific procedure is as follows:

Firstly, the first RBM of DBN is trained; probabilities of RBM hidden and visible units are calculated according to equations (3) and (4); parameters are adjusted and updated using gradient descent with equation (5) as the objective function of RBM; and a 4096*3000 weight matrix and offset are obtained through training.

(2)Weight and offset of the first RBM are fixed, and its hidden neuron state is used as the input vector of the second RBM to obtain a 1500*750 weight matrix and offset through training and optimization. The third and fourth RBMs are trained and optimized in the same way.

(3)The above four RBMs are expanded and connected into a new network, whose initial values are assigned with the weights and offsets obtained by steps 1 and 2.

(4)The entire network is fine-tuned using BP algorithm to obtain optimized network parameters.

Through DBN training, the reconstructed images of source problem domain samples are obtained, and

the mapping relationship from training samples into defect-free template is established.

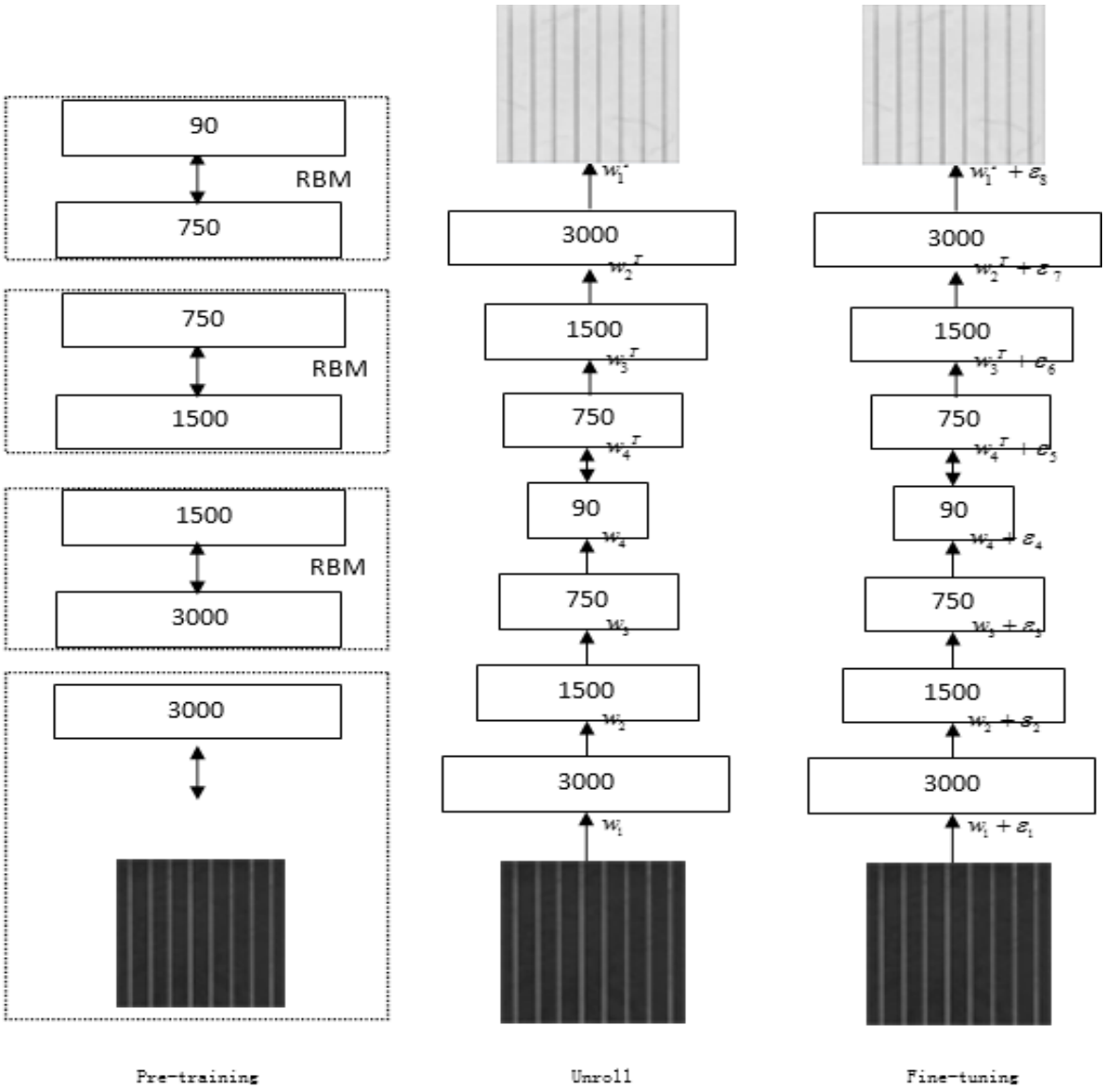


Figure 1. DBN pre-training, unrolling and fine-tuning stages in defect detection

3.3 DBN Parameter Transfer

Product defect detection using DBN faces the following problem: when the training samples of product are small in number, DBN can hardly obtain optimized network parameters through training, which is prone to over-fitting. On the other hand, when faced with a new defect detection problem, DBN network is still built and trained using limited samples generally. But such approach ignores the correlation between problem domains. There are some similar visual defects between different problem domains; for example, solar cells and capsules both have scratches or other similar defects. If transfer training is performed exploiting such correlation, it will greatly reduce the number of samples required for training DBN, thereby improving the accuracy of defect detection.

The main idea of parameter transfer is to transfer the structure and optimized parameters of source problem domain DBN network into the DBN network of target problem domain, and to fine-tune the relevant weight parameters of DBN network using the training samples of target problem domain. Through parameter transfer, model parameters are shared between models of source and target problem domains; a common feature space is found between source and target problem domains; data distribution distance between the two items on the common feature space is minimized; insufficient amount of data problem is resolved; and the efficiency of the algorithm is improved.

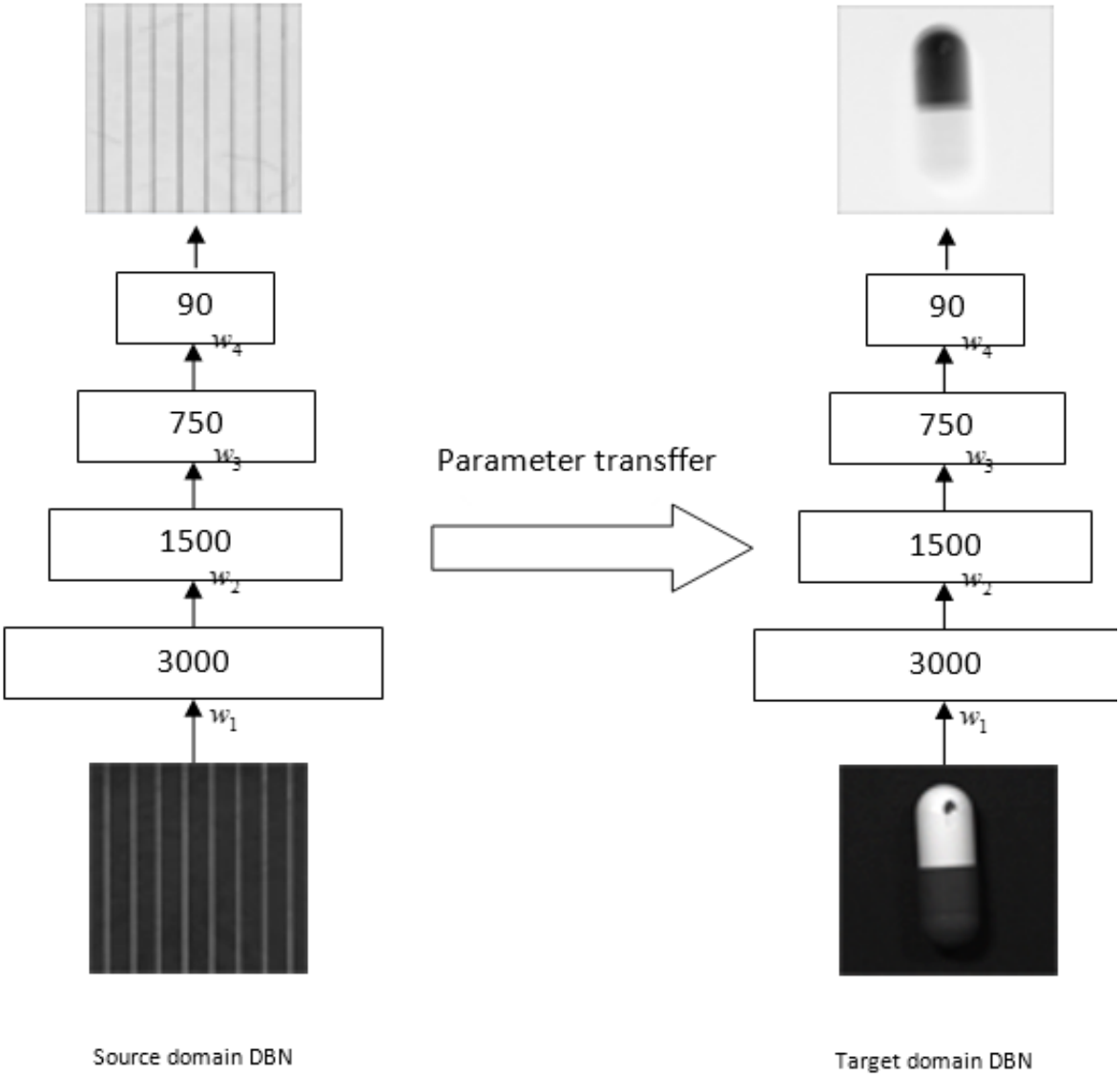


Figure 2. Transfer from source problem domain to target domain

3.4 Defect Recognition and Detection

Comprehensively considering the high accuracy and high efficiency required by capsule defect detection and the advantages of deep learning algorithm, a deep learning-based capsule defect detection algorithm is proposed. Specific steps of the detection algorithm are as follows:

Step1: Acquire the source and target domain sample images.

Step2: Gray scale and normalize the images first, and then perform gray scale transformation of images to enhance the contrast of images.

Step3: Construction and training of source domain DBN. Train and optimize the preprocessed training samples of source domain using DBN to obtain the weights and offsets of source domain DBN.

Step4: DBN parameter transfer. Transfer the DBN structure and parameters of source domain into target domain, and fine-tune the DBN network parameters by inputting the target domain training samples to obtain the mapping between target samples and defect-free images.

Step5: Compare the target domain test samples with the resulting reconstructed images, and obtain the binary images of difference images by thresholding for discrimination and classification between normal and defected samples.

It can be seen from the implementation principle of the algorithm that the deep learning- and transfer learning-based defect detection method proposed herein has the following advantages compared with the single-source data defect detection method:

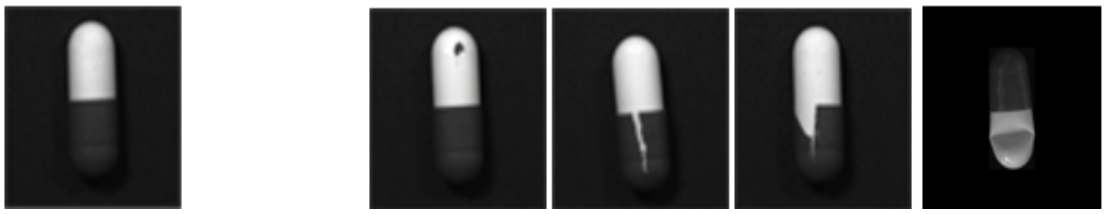
(1) It can effectively resolve the insufficient amount of data problem. As the proportion of defective samples is relatively low in industrial production, the product defect detection data are often insuffi-

cient, while the training of DBN network often requires large amounts of data. The method proposed herein overcomes this issue by exploiting the already existing training data, which finds relevant data in the rich historical data. In theory, its detection efficiency will be significantly improved.

(2) Weights of training target domain samples can further be used to assist the defect detection of source domain, and to achieve the transfer learning and cross learning between source and target problem domains, so as to greatly improve the detection efficiencies for both.

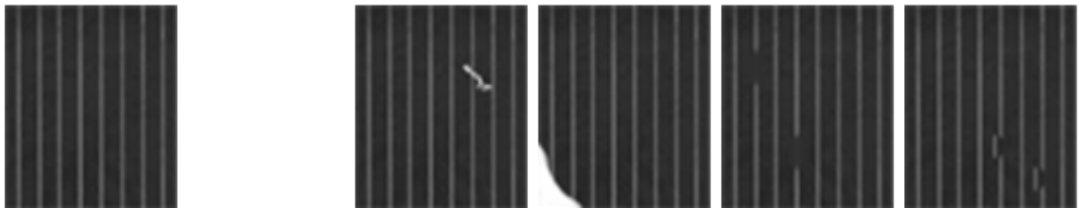
4. Defect Recognition and Detection

In this paper, experiments are completed on an Intel Core i7-440K 3.5GHz, 8G memory CPU platform using Matlab R2012b software. There are solar cell samples and capsule samples in the experiments, where the solar cells are the source problem domain and capsule samples are the target problem domain. In the experiments, the source and target problem domains each has 100 images as the training samples and 20 images as test samples, which are normalized to 64*64 gray scale images. Training samples consist of defective and defect-free images. Defects of capsule samples mainly include cracks, black spots and deformation, while the defects of solar cells include scratches, chips, creases and ridges. Figures 3 and 4 illustrate the defect-free and defective images of the capsules and solar cells, respectively.



(a) Defect-free image (b) Defective capsule images

Figure 3. Defect-free and defective images of capsules



(a) Defect-free image (b) Defective solar cell images

Figure 4. Defect-free and defective images of solar cells

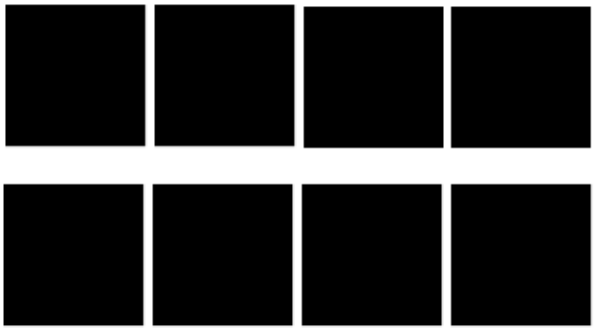
This paper designs two different sets of experiments. The first set of experiment performs training and learning of DBN directly using the capsule samples to give reconstructed images of capsules; the second set of experiment firstly trains the DBN with solar cells to obtain the initial weights of DBN, and then directly transfers the weight parameters of solar DBN into the capsule sample training to obtain the reconstructed images of capsules. The weights of capsule sample training are again used as the initial weights of solar cell training, thereby achieving the transfer learning and cross learning between solar cell and capsule problem domains.

4.1 Experimental Analysis of DBN Training with Capsule Samples Alone

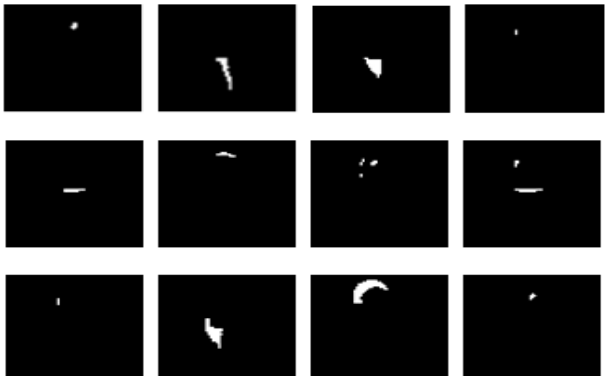
In the experiment, the ratio of defect-free capsule images in training samples is 20%, while the remaining 80% is defective capsule images. Training time is 2,597 seconds for the entire experiment, of which the time consumed for deep learning is 39.62 seconds. Time required for 200 iterations of BP algorithm is preset as 2,557 seconds. The experimental results are shown in Figure 5, where figure (a) is the reconstructed image of capsule obtained through training; figure (b) is the test image obtained through gray scaling, complement and binarization; and figure (c) is the defect image obtained through difference comparison between figures (a) and (b). Time elapsed for testing 20 images is 2.46 seconds, which makes the average per image detection time to be 0.123 seconds.

(a) Reconstructed image of capsule (b) Test image (c) Defect image

Figure 5. Reconstructed image, test image and defect image of capsule images



(a) Defect-free images of test images



(b) Defective images among test images

Figure 6. Experimental results of test images

4.2 Experimental Analysis of Parameter Transfer Learning with Solar Cell and Capsule Samples

The purpose of this experiment is to achieve parameter transfer between solar cell and capsule problem domains. Solar cells have relatively regular shapes, which are easy to train, while capsules are in a three-dimensional shape, whose training is relatively difficult. During the experiment, DBN is trained with solar cells firstly to give weight and offset of each layer of RBM, and then these weights and offsets of DBN are used as the initial values of training capsule sample DBN to perform training and optimization. The experimental results show that the parameter transfer from solar cell DBN to capsule DBN can improve the training efficiency of capsule DBN while reducing the training time and the number of iterations. Weights of capsule DBN can again further optimize the learning of solar cell DBN. The two problem domains can mutually promote each other, learn crosswise from each other, and jointly improve the detection efficiency.

In this set of experiments, we constructed a DBN network consisting of a 4,096-dimensional input layer and four hidden layers 3,000, 1,500, 750 and 90 in dimensions, respectively. As a first step, solar cell samples are used as the input of DBN network, which include defective and defect-free samples. By inputting the solar cell sample features, optimized parameters (weight and offset) of DBN are obtained. All parameters of the DBN are transferred into the DBN network of capsule problem domain, and then fine-tuning is performed by inputting capsule sample features to obtain the defect-free template of capsules. The experimental results indicate that the parameter transfer between solar cell and capsule problem domains enables improvement of efficiency by around 20%.

Table 1. Comparison of results between parameter transfer learning experiment and non-transfer defect detection experiment

Sample	DBN	Training time	Required number of iterations
Capsule	Without parameter transfer	3,187 seconds	50 times
Capsule	With parameter transfer	2,874 seconds	34 times
Solar cell	Without parameter transfer	1,644 seconds	20 times
Solar cell	With parameter transfer	1,378 seconds	12 times

5. Summary and Outlook

This paper constructs DBN based on the features of samples, effectively extracts product sample features through DBN, addresses the over-fitting problem in the training process by transfer of DBN structure and parameters, and obtains the reconstructed image of target problem domain. For test samples, defects are identified simply by binarization of test samples and differential comparison with the reconstructed images. The advantages of the proposed method are fast detection, universality, suitability for general defect detection, generalizability and applicability to industrial areas. However, the method also has some shortcomings, which can only detect the product defects at the same locations, and does not have the translational and rotational invariance. Our next work is to enable defect detection of product samples with certain translational and rotational properties.

Acknowledgements

This work was supported by Zhejiang Provincial Natural Science Foundation of China.(Grant No: Z14F030004)

References

1. Wei Lin, Hu Rongqiang (2002) Automatic Inspection System of Capsule Product Based on BP Neural Network. *Application Research of Computers*, 19(4), p.p.52-53.
2. Zuo Qi, Shi Zhongke (2002) General Design for Capsule Integrality Detection System Based on Machine Vision. *Journal of Xi'an Jiaotong University*, 36(12), 1262-1265.
3. Feng Shanshan, Chen Shuyue (2008) Research on identification method for real and false capsule-grains based on image analysis. *Transducer and Microsystem Technologies,* 27(8), p.p.54-56.
4. Lai Dahu, Huang Yanwei (2012) Capsule defect detection based on extreme learning machine. *Journal of Fuzhou University (Natural Science Edition)*, 40(4), p.p.489-494.
5. Hinton G, Salakhutdinov R. (2006) Reducing the dimensionality of data with neural networks. *Science*, 313(504), Doi:10.1121/Science. 1127647.
6. Dahl G E, Yu Dong, Deng Li, Acero A. (2012) Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transaction on Audio, Speech and Language Processing*, 20(1), p.p.30-42.
7. Hinton G E, Deng Li, Yu Dong, Dahl G E, Mohamed A, Jaitly N, Senior A, Vanhoucke V, Nguyen P, Sainath T N, Kingsbury B (2012) Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal Processing Magazine*, 29(6), p.p.82-97.
8. Choi Sungjoon, Kim Eunwoo, Oh Songhwai (2013) Human behavior prediction for smart homes using deep learning. *Proc. Conf. on the 22nd IEEE International Symposium on Robot and Human Interactive Communication*, p.p.173-179.
9. Hinton G E, Osindero S (2006) The Y W.A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7), p.p.1527-1554.
10. Hinton G E, Zemel R S.(1994) Autoencoder, minimum description length, and Helmholtz free energy. *Proc. Conf. on Advances in Neural Information Processing System*, p.p.3-10.
11. Vincent P, Larochelle H, Bengio Y, et al. (2008) Extracting and composing robust features with denoising autoencoders. *Proc. Conf. on the 25th International Conference on Machine Learning*, p.p.1096-1103.
12. Smolensky P. (1986) *Information processing in dynamical systems: Foundations of harmony theory*. Cambridge, MA, USA: MIT Press, p.p.194-281.
13. Hinton G E. (2002) Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8), p.p.1771-1800.
14. Y. Bengio (2011) Deep learning of representations for unsupervised and transfer learning. *Proc. Conf. on Unsupervised and Transfer Learning*.
15. G. Mesnil, Y. Dauphin, X. Glorot, et al. (2011) Unsupervised and transfer learning challenge: a deep learning approach, *Proc. Conf. on NIPS Workshop on Challenges in Learning Hierarchical Models*.
16. Wang Xianbao, Li Jie, Yao Minghai, et al. (2014) Solar Cell Surface Defects Detection Based on Deep Learning. *Pattern Recognition and Artificial Intelligence*, 27(6), p.p.517-523.