Weed Seeds Classification Based on PCANet Deep Learning Baseline

Wang Xinshao, Cai Cheng*
1. College of Information Engineering, Northwest A&F University, Yangling,712100.
cheney.chengcai@gmail.com

Abstract— **There are a large number of various kinds of weeds in agriculture. Weeds have a great impact on the development of agricultural production and agricultural economy. The reproduction and spread of weeds are mainly dependent on weed seeds. So we want to find an efficient algorithm with robust and accurate classification of weed seeds, which has an important practical value and economic significance. PCA Network has been applied to image feature extraction and achieved fantastic effects. Here we propose a variant of PCA Network and apply it to the classification of weed seeds in agriculture. The difference between the proposed method and the original PCA Network lies in that we get L_1 families of orthogonal filters rather than one family of orthogonal filters in the second stage of PCA. After using the PCA Network variant method to extract image features, we conduct an experiment to test the classification accuracy of weed seeds. The data sets contain 91 types of weed seeds. In the experiment we use a large margin classifier to construct a linear classifier, which is based on affine hulls. Next we use the features extracted from the test samples to examine the recognition accuracy rate. Experiment results show that the PCA Network variant method obtains good classification results and improves the recognition accuracy. In the data sets composed of 91 types of weed seeds, 45 arrives at 100% recognition rate of classification, and 90.96% average recognition rate. At the same time, our algorithm shows relatively higher robustness than the recognition of weed seeds images does. This algorithm improves the classification accuracy rate of weed seeds greatly and thus can be applied to the agricultural production practice.**

I. INTRODUCTION

Deep Learning is a new area in the machine learning research. Currently, a lot of regression and classification learning methods rely on shallow structure algorithm. These methods have great limitations in performing complex functions with limited samples and calculating units. Besides, when it comes to complex classification, the generalization ability of shallow structure algorithms is also limited in some degree. However, deep learning learns by the deep structure of nonlinear network, so it can achieve fewer parameters to represent the complex function with the benefits of multi-layers. The most attractive aspect of deep learning lies in that it can obtain essential features of the data from a small number of training samples. Therefore, deep learning sets off a second wave of machine learning.

We can learn from deep learning of PCANet (Principal Component Analysis Network) in [1] that it is a terrific way to minimize the limitations of manual feature extraction by learning features from the data of interest.

The purpose of deep learning is "feature learning". We can improve the accuracy of classification and prediction by learning inherent abundant and useful features of the data set. Deep learning can build neural network of analysis and learning by simulating the human brain. It also interprets the data obtained by leaning through simulating the mechanism of the human brain. As a kind of unsupervised learning, deep learning can overcome the difficulties of the training of deep neural network by means of initialization step by step. The main applications of the deep learning are the convolutional architecture in [2], [3], [4], [5], [6], [7], [8], [9], convolutional deep neural network (ConvNet) in [2], [3], [4], [7], [8], as well as supervised classification accompanying it.

In this paper, we use the deep learning method to solve the problem of weed seeds classification efficiently. The weed has a great impact on agricultural production. In natural population competition, weed has its incomparable advantages. It can restrain the growth of crops and impose a suppressed effect on the efficiency of agricultural production. We know that the breeding and spread of the weed seeds are mainly dependent on mixing in crop seeds, grain seeds and some other plant seeds. It is very challenging and full of practical significance for us to find an algorithm which can classify weed seeds efficiently, making use of the feature differences among different types of weed seeds.

In the research area of weed seeds classification, there are many people who have done a lot of work and achieved some significant results. The methods they have employed are mainly the low-level image feature extraction with the manual design. Most feature extraction and classification are based on seed size, shape, color and texture features. There are also some famous algorithms such as linear discriminant analysis in [10], Naïve Bayes algorithm in [11], neural network classification algorithm in [12] and Boosting algorithm in [13].

Previous researches simply extract the local features of images. The features extracted in this way will lose some important information for identifying noise and blocks. So some seeds with mildew and much noise cannot be recognized effectively. However, the high-level feature information of images can be extracted by deep learning. We can get the essential features of each seed through multilayer learning in the weed seeds training, which provides a large number of features for classification. So we can solve the problems of

occlusion and noise in the weed seeds classification and improve the classification efficiency.

The PCANet is a simple deep learning baseline for image classification. We classify the weed seeds by applying some changes to the PCANet in [1]. After extracting the features by ameliorative PCANet, we use the large margin classifier (LMC) in [14] based on affine hulls to construct a linear classifier. The large margin classifier is then applied to weed seeds classification. Fig. 1 shows how the PCANet of two layers extracts different kinds of features from an input image. In the first stage, suppose that we have one filter bank with L_1 filters, then we get L_1 output images. In the second stage, suppose that we have L_1 filter banks, and each bank including L_2 filters, then we get L_1L_2 output images. That is to say, the output layer is L_1 image banks, each including L_2 images. Next we can binarize the output images and convert the L_2 images into one image. Every pixel in this image is a decimal value. Then we partition each of the L_1 decimal images into m blocks. Then we count the histogram of every block. We will get the feature vector of one input image features by concatenating all the histograms of the blocks into one column vector.

In this paper, Part II introduces how to extract image features by means of a changed PCANet. Part III introduces how to construct a multi-class classifier among different kinds of weed seed features by applying the method of large margin classifiers based on the affine hulls. In Part IV, we conduct an experiment of the classification of weed seeds and then analyze the experiment results. We make a brief conclusion in Part V.

II. FEATURES EXTRACTION BASED ON PCANET

As is shown in the Fig. 1, when we extract the image features, the method PCANet is adopted. We get the filters banks after all the images have been trained after the two layers PCANet. Then we get the high-layer outputs of each image by the filters banks. We get the feature vector of each image by binarization and counting the histograms in the high layer.

The following is the design of the PCANet.

Suppose we have N images for training. The size of every image is $h \times w$. We set the patch size of each image as $k_1 \times k_2$. We get the filters banks by the training of these N images. Fig. 2 is the detailed structure diagram of the two layers PCANet.

A. The first stage of PCA

In the ith image, we select a $k_1 \times k_2$ patch around each pixel. In the jth pixel of the ith image, we convert the patch into a column vector $\boldsymbol{y}_{i,j}$. So we get $h \times w$ column vectors of the ith image.

$$\boldsymbol{Y}_i = [\boldsymbol{y}_{i,1}, \boldsymbol{y}_{i,2}, \cdots, \boldsymbol{y}_{i,hw}] \quad (1)$$

Using each column vector to subtract its own mean value, we get:

$$\bar{\boldsymbol{Y}}_{\boldsymbol{i}} = [\bar{\boldsymbol{y}}_{i,1}, \bar{\boldsymbol{y}}_{i,2}, \cdots, \bar{\boldsymbol{y}}_{i,hw}] \quad (2)$$

$\bar{\boldsymbol{y}}_{i,j}$ represents a patch vector after the removal of its mean value.

We apply the same operation to every image, and then get a matrix.

$$\boldsymbol{Y} = [\bar{\boldsymbol{Y}}_1, \bar{\boldsymbol{Y}}_2, \ldots, \bar{\boldsymbol{Y}}_N] \in \boldsymbol{R}^{k_1 k_2 \times Nhw} \quad (3)$$

Presume that the ith layer has L_i filters. We can minimize the reconstruction errors by the orthonormal filters banks. (4)

$$\begin{cases} \min\limits_{\boldsymbol{v} \in \boldsymbol{R}^{k_1 k_2 \times L_1}} \| \boldsymbol{Y} - \boldsymbol{U}\boldsymbol{U}^{\boldsymbol{T}}\boldsymbol{Y} \|_{\boldsymbol{F}}^{\boldsymbol{2}} \\ s.t.\ \boldsymbol{U}^{\boldsymbol{T}}\boldsymbol{U} = \boldsymbol{E}_{\boldsymbol{L_1}} \end{cases}$$

$\boldsymbol{E}_{\boldsymbol{L_1}}$ represents a $L_1 \times L_1$ matrix. $\boldsymbol{U}$ is the filters bank that we want to get. In our method, we just obtain the L_1 primary eigenvectors of $\boldsymbol{Y}\boldsymbol{Y}^{\mathrm{T}}$. So the PCA filters can be stated as:

$$\boldsymbol{W}_l^1 = matrics_{k_1,k_2}(ql(\boldsymbol{Y}\boldsymbol{Y}^{\mathrm{T}})) \in \boldsymbol{R}^{k_1 \times k_2} \quad (5)$$

$l = 1,2, \ldots L_1$. The function of $matrics_{k_1,k_2}(\boldsymbol{colvector})$ is to convert the column vector, such as $\boldsymbol{colvector} \in \boldsymbol{R}^{k_1 \times k_2}$, to a matrix $\boldsymbol{W} \in \boldsymbol{R}^{k_1 \times k_2}$, $ql(\boldsymbol{Y}\boldsymbol{Y}^{\mathrm{T}})$ represents the lth primary eigenvector of $\boldsymbol{Y}\boldsymbol{Y}^{\mathrm{T}}$. The primary eigenvectors witness the principal changes of all the training patches after the removal of the mean value. We can extract higher level image features by applying multi-level PCA filters like the DNN(Deep Neural Network) in [15] and ScatNet(Scattering Networks) in [16], [17].

B. The second stage of PCA

Similar to what we did in the first stage of PCA, we apply the following operation to the N images by using the output filters of the first stage.

$$\boldsymbol{Z}_i^l = \boldsymbol{Z}_i * \boldsymbol{W}_l^1, \quad i = 1,2, \ldots N \quad (6)$$

Here $*$ represents the convolution of two dimensions. $\boldsymbol{Z}_i$ represents the ith training image. In order not to change the size of each image, we need to make each image zero-padded in the boundary before it convolves with $\boldsymbol{W}_l^1$. Similar to the first stage, the column vector of each patch should subtract its mean value.

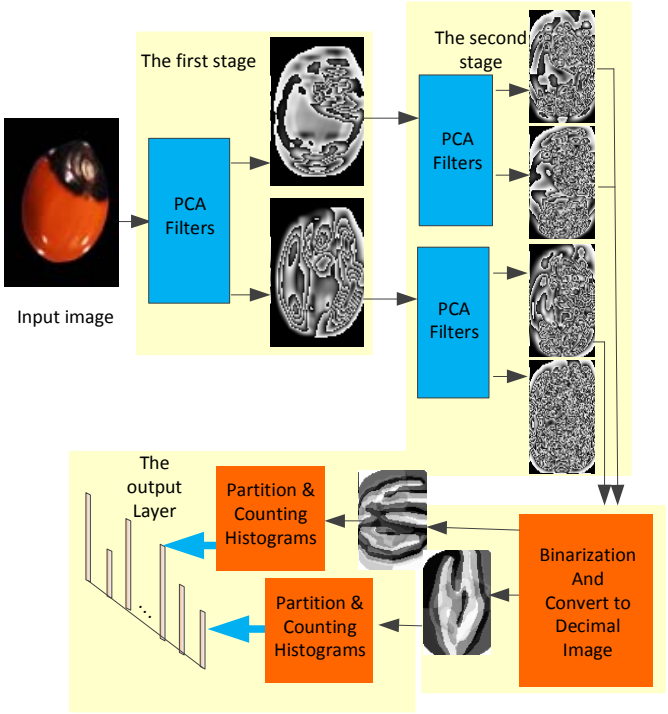


Fig. 1. PCANet deep learning method extracts the features of one image mainly through three steps: PCA filters banks, binarization, and counting the histograms.

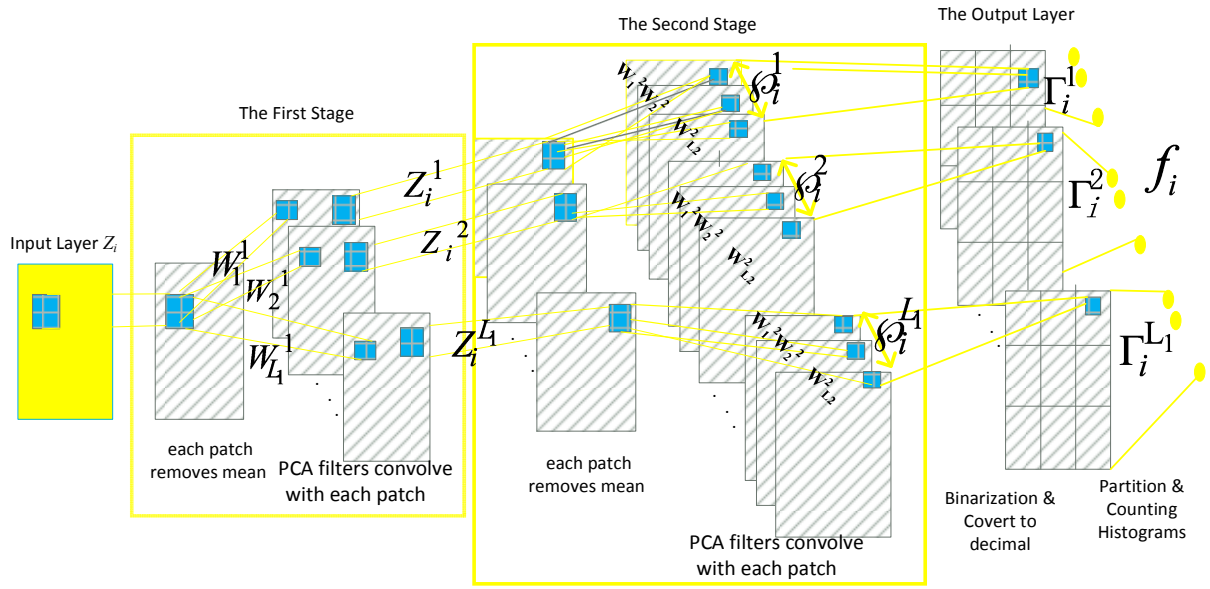


Fig. 2. Detailed structure diagram of the two layers PCANet

$$\bar{\boldsymbol{X}}_i^l = [\bar{\boldsymbol{x}}_{i,l,1}, \bar{\boldsymbol{x}}_{i,l,2}, \dots, \bar{\boldsymbol{x}}_{i,l,hw}] \quad (7)$$

$\bar{\boldsymbol{x}}_{i,l,j}$ represents the jth patch after the removal of its mean value in $\boldsymbol{Z}_i^l$.

$$\boldsymbol{X}^l = [\bar{\boldsymbol{Y}}_1^l, \bar{\boldsymbol{Y}}_2^l, \dots, \bar{\boldsymbol{Y}}_N^l] \in \boldsymbol{R}^{k_1 k_2 \times Nhw} \quad (8)$$

$\boldsymbol{X}^l$ represents the outputs of all the images after convolving with $\boldsymbol{W}_l^1$.

The same as the first stage, we get:

$$\boldsymbol{W}_{l,\ell}^2 = matrics_{k_1 k_2}(ql(\boldsymbol{X}^l \boldsymbol{X}^{l^T})) \epsilon \boldsymbol{R}^{k_1 k_2} \quad (9)$$

$\ell = 1,2, \dots, L_2$, $\boldsymbol{W}_{l,\ell}^2$ represents a family of orthonormal filters responding to the outputs of all the images after convolving with $\boldsymbol{W}_l^1$. So the L_1 filters banks of the second stage can be represented as follows.

$$\boldsymbol{W}_\ell^2 = \{\boldsymbol{W}_{l,\ell}^2\}_{l=1}^{L_1} \quad (10)$$

For one input Z_i^l of the second stage, we can get its L_2 outputs.

$$\wp_i^l = \{\boldsymbol{Z}_i^l * \boldsymbol{W}_{l,\ell}^2\}_{\ell=1}^{L_2} \quad (11)$$

$\ell = 1,2, \dots, L_2$. So the number of outputs of one image after the two layers training is $L_1 L_2$.

If we find more layers training of PCANet more efficient, we can easily establish more stages similar to the process aforementioned.

C. *The output layer of PCANet*

For one input Z_i^l of the second stage, we can get L_2 outputs after convolving with $\boldsymbol{W}_{l,\ell}^2$. Then we binarize the pixel values of these outputs.

$$\{\text{Binarify}(\boldsymbol{Z}_i^l * \boldsymbol{W}_{l,\ell}^2)\}_{\ell=1}^{L_2} \quad (12)$$

Binarify(x) is a binarization function. If the x is a positive value, the function value is 1. Otherwise, the function value is 0.

In the same pixel position of the L_2 outputs, we convert the L_2 binary bits to a decimal number. So the L_2 outputs can be converted into one decimal value map:

$$\boldsymbol{\Gamma}_i^l = \sum_{\ell=1}^{L_2} 2^{\ell-1} \text{Binarify}(\boldsymbol{Z}_i^l * \boldsymbol{W}_{l,\ell}^2) \quad (13)$$

Each pixel value of this map is in the range $[0, 2^{L_2} - 1]$.

For every of the L_1 maps after binarization, we divide it into m blocks. In each block, we count the histograms of the pixel values. Then the histogram has 2^{L_2} bins. We can get m histograms of each map. Then we connect the m histograms to a vector, i.e. the image feature column vector $\mathbf{H}(\boldsymbol{\Gamma}_i^l)$. So the input Z_i can be extracted to L_1 feature column vectors. Then the L_1 feature column vector is concatenated to one feature column vector.

$$\boldsymbol{f}_i = [\mathbf{H}(\boldsymbol{\Gamma}_i^1), \mathbf{H}(\boldsymbol{\Gamma}_i^2), \dots, \mathbf{H}(\boldsymbol{\Gamma}_i^{L_1})]^T \epsilon \boldsymbol{R}^{(2^{L_2}) L_1 m} \quad (14)$$

When we select the local blocks, the blocks can be either overlapping or not. The experiment has showed that when we are extracting the features of faces, the non-overlapping local blocks have better performance. When we are extracting the features of textures, hand-written digits, object images, it is preferable to select overlapping local blocks.

When we are extracting the image features, the parameters we need to set are the size of the filters $k_1 \times k_2$, the filter number of each stage L_1, L_2, and the block number m of the output layer. $k_1 k_2 \geq L_1$, L_2 is required in the PCA filters banks. We know that the two layers of PCA can meet our needs of conducting the experiments. The performance is good when the $L_1 = L_2 = 8$. Besides, the larger blocks can make the features $\boldsymbol{f}_i$ more rotation invariant.

III. THE LMC CLASSIFICATION METHOD BASED ON AFFINE HULLS

When we are conducting weed seeds classification, the use of the supported vector machine is limited by the number of training samples. However, the use of LMC classifiers is different from that of the supported vector machine. The former makes use of the affine hulls, rather than the convex hulls among different racial samples, for the convex hulls are far from effective in the high dimension spaces.

So in this paper, after obtaining all the features of different racial weed seeds by means of PCANet, we build multi-class linear classifiers by applying the method of LMC classifiers.

A. How to build a linear classifier between two classes

Suppose that we have two classes of training samples, A and B. We use the two layers PCANet to extract the features of the samples. So each sample image can be extracted to one feature column vector. We can get two feature data sets after all the image features have been extracted by means of PCANet. Each feature data set contains the features of all the images of one class. The feature data set of each class can be presented as:

$$\{\boldsymbol{f}_i, y_i\}, i = 1, \dots, n, y_i \epsilon \{-1, +1\} \tag{15}$$

$\boldsymbol{f_i} \in \boldsymbol{R}^{(2^{L_2})L_1 m}$ represents the feature column vector of each sample.

The affine hull of each class contains the smallest linear affine subspace. So the two feature data sets of class A and class B can be represented in the form of affine hulls.

$$\boldsymbol{H}^{aff} = \left\{ \boldsymbol{f} = \sum_{i=1}^{n} \alpha_i \boldsymbol{f}_i \,\middle|\, \sum_{i=1}^{n} \alpha_i = 1 \right\} \tag{16}$$

Here n is the number of the training samples in the class.

We get two affine hulls after the deep learning of PCANet. Assume that class A is the positive class, and class B is the negative class. That is to say, in (15), all the $\boldsymbol{y}_i$ values are 1 in the affine hull of class A. But in the affine hull of class B, all the $\boldsymbol{y}_i$ values are -1. We need to find the best linear separating hyperplane. Every point lying on this separating hyperplane meets $< \boldsymbol{w}, \boldsymbol{f} > +b = 0$, the $\boldsymbol{w}$ is the standard of the separating hyperplane, $|\boldsymbol{b}|/\|\boldsymbol{w}\|$ represents the distance from the origin to the separating hyperplane, and $\|\boldsymbol{w}\|$ represents the Euclidean norm of $\boldsymbol{w}$. So we can make use of this linear separating hyperplane to judge which class a point belongs to. Because all the points in the positive class meet $< \boldsymbol{w}, \boldsymbol{f} > +b > 0$, all the points in the negative class meet $< \boldsymbol{w}, \boldsymbol{f} > +b < 0$. That is to say, all the points belonging to the positive class or negative class meet:

$$y_i(< \boldsymbol{w}, \boldsymbol{f}_i > +b) > 0 \tag{17}$$

After we found the best separating hyperplane, a new input image can be identified by using the function:

$$\mathrm{g}(f) =< \boldsymbol{w}, \boldsymbol{f} > +b \tag{18}$$

If $\boldsymbol{g}(\boldsymbol{f})$ is larger than zero, the input image is judged as the positive class. If $\boldsymbol{g}(\boldsymbol{f})$ is smaller than zero, the input image is judged as the negative class.

If the affine hulls of the two classes are separable linearly, these two affine hulls do not intersect. For more convenience, the affine hull of one class can be written as:

$$\boldsymbol{H}^{aff} = \{\boldsymbol{f} = \boldsymbol{U}\boldsymbol{v} + \boldsymbol{\mu} | \boldsymbol{v} \epsilon \boldsymbol{IR}^t\} \tag{19}$$

Here $\boldsymbol{\mu} = (1/n)\sum_{i=1}^{n} \boldsymbol{f}_i$ represents the mean of all the training samples within each class, and $\boldsymbol{U}$ is an orthotropic basis for the orientations stepped over by the affine subspace. The $\boldsymbol{v}$ contains the lessened coordinates of every point in the subspace, responding to the basis $\boldsymbol{U}$. We can get the basis $\boldsymbol{U}$ by the Singular Value Decomposition of the matrix $[\boldsymbol{f}_1 - \boldsymbol{\mu}, \dots, \boldsymbol{f}_i - \boldsymbol{\mu}]$. Each column vector of $\boldsymbol{U}$ is one vector respect to one non-zero singular value of the SVD of the matrix $[\boldsymbol{f}_1 - \boldsymbol{\mu}, \dots, \boldsymbol{f}_i - \boldsymbol{\mu}]$. t is the number of non-zero singular value of the matrix.

The following is the whole process of computing and problem solving in detail.

Now we have two affine hulls, A and B:

$$\begin{cases} \boldsymbol{A} = \{\boldsymbol{U}_+ \boldsymbol{v}_+ + \boldsymbol{\mu}_+\} \\ \boldsymbol{B} = \{\boldsymbol{U}_- \boldsymbol{v}_- + \boldsymbol{\mu}_-\} \end{cases} \tag{20}$$

We can find the closest pair of points within these two affine hulls by function:

$$\min_{\boldsymbol{v}_+ \boldsymbol{v}_-} \|(\boldsymbol{U}_+ \boldsymbol{v}_+ + \boldsymbol{\mu}_+) - (\boldsymbol{U}_- \boldsymbol{v}_- + \boldsymbol{\mu}_-)\|^2 \tag{21}$$

Assume that $\mathbf{U} \equiv [\mathbf{U}_+ \quad -\mathbf{U}_-]$, $\boldsymbol{v} \equiv [\boldsymbol{v}_+ \quad \boldsymbol{v}_-]^{\mathrm{T}}$. So the function can be transformed as

$$\min_{\boldsymbol{v}} \|\boldsymbol{U}\boldsymbol{v} - (\boldsymbol{\mu}_- - \boldsymbol{\mu}_+)\|^2 \tag{22}$$

Then we take derivation for $\boldsymbol{v}$ and make the derivation function equal to zero, we get

$$\boldsymbol{U}^{\boldsymbol{T}}\boldsymbol{U}\boldsymbol{v} - \boldsymbol{U}^{\boldsymbol{T}}(\boldsymbol{\mu}_- - \boldsymbol{\mu}_+) = \boldsymbol{0} \tag{23}$$

The solution of this function is

$$\boldsymbol{v} = (\boldsymbol{U}^{\boldsymbol{T}}\boldsymbol{U})^{-\boldsymbol{1}}\boldsymbol{U}^{\boldsymbol{T}}(\boldsymbol{\mu}_- - \boldsymbol{\mu}_+) \tag{24}$$

Combine this with the decision function $\mathbf{g}(\boldsymbol{f}) =< \boldsymbol{w}, \boldsymbol{f} > +b$.We get

$$\boldsymbol{w} = \frac{1}{2}(\boldsymbol{f}_+ - \boldsymbol{f}_-) = \frac{1}{2}(\boldsymbol{I} - \boldsymbol{Q})(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-) \tag{25}$$

Here $\boldsymbol{Q} = \boldsymbol{U}(\boldsymbol{U}^{\boldsymbol{T}}\boldsymbol{U})^{-\boldsymbol{1}}\boldsymbol{U}^{\boldsymbol{T}}$, $\boldsymbol{f}_+$, $\boldsymbol{f}_-$ represents the closest pair of points between the two affine hulls. In the decision function, the standard separating hyperplane $\boldsymbol{w}$ lies along the closest pair of points between these two affine hulls. We can also obtain the offset b

$$b = -\boldsymbol{W}^{\boldsymbol{T}}(\boldsymbol{f}_+ + \boldsymbol{f}_-)/2 \tag{26}$$

We can find the best separating hyperplane between the two classes A and B by following the process aforementioned. So we have built a linear classifier between Class A and Class B. We can judge which class an input image belongs to by classifier.

B. The multi-class LMC classifiers based on two classes of LMC

For multi-class classification problems, the supported vector machine (SVM) algorithm includes one-vs-one SVM in [20], one-vs-all SVM in [21], in BT-SVM [22] and DAGS-SVM in [23].

In this paper, when we conduct weed seeds classification, we build multi-class classifiers of the weed seeds by applying one-vs-one LMC, which is similar to one-vs-one SVM.

Specifically, we apply the algorithm of building a linear classifier between two classes to build a classifier between every two classes, just as one-vs-one SVM. Assume that we have different weed seeds training samples of k different

classes. Then we need to build $k(k\text{-}1)/2$ classifiers by using the one-vs-one method. Then we can convert the multi-class classification problem into many classification problems between two classes.

The detailed description of building one classifier between every two classes is as follows.

First, we select two classes randomly from the k classes samples, and assume they are class A and class B. Class A is a positive class, while class B is a negative one. Then use (24) and (25) to build a classifier between these two classes. Later we employ the same method in building a classifier between every two classes of all, so as to get $k(k\text{-}1)/2$ classifiers needed for the identification and classification of k classes weed seeds.

C. How to predict the class of an unknown test sample

After obtaining the decision functions of the training samples, we can predict which class an input image belongs to by the $k(k\text{-}1)/2$ decision functions easily.

We can also predict the class of an unknown input by means of voting. Suppose we have a sample q to be tested, and we need to judge which class q belongs to. The voting mechanism requires us to consider how all the $k(k\text{-}1)/2$ decision functions judge the class of the sample q. If one decision function judges the input sample q as the sth class ($1 \le s \le k$), then the sth class will get one ticket. Finally, we count the tickets each class has obtained, and then make the prediction simply by figuring out which class receives the largest number of tickets.

Applying the one-vs-one method to building many separating hyperplanes has many advantages. Building each decision function is very fast because the training samples are relatively fewer, and each time we only use the training data of two classes. Furthermore, the process of calculating decision functions is easy to be parallelized.

IV. EXPERIMENT

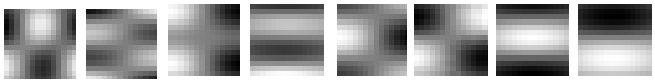
A. The PCANet training of weed seeds

We first set the PCANet parameters. In our experiment, the patch size is set with $k_1 \times k_2 = 15 \times 15$ based on experimental results. With the normal setting of Gabor filters [24] with eight direction, we set the filter number of each stage with $L_1 = L_2 = 8$. The block size is set as 16x16 based on experimental results; furthermore, the larger blocks can make the features $\boldsymbol{f_i}$ more rotation invariant. There are two stages of PCA network in our experiment. The PCANet filter banks we have obtained by deep learning are showed in Fig. 3. Because the filter number of each stage is $L_1 = L_2 = 8$, the filter number of the first stage is eight. In the second stage, we get eight filter banks with eight filters in each.

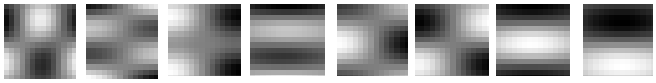
Next we can apply these filter banks to extracting the features of all the training samples by convolution operation.

B. Feature extraction of weed seeds images and building the multi-class classifiers

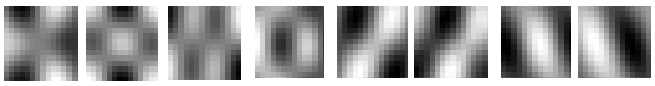
After obtaining the PCANet filter banks, we need to extract the features of each image and convert it to one feature column vector $\boldsymbol{f}$ of $(2^{L_2})L_1 m$ dimensions. All the feature vectors of each class form an affine hull. In our experiment, we have weed seeds of 91 different classes, so we get 91 affine hulls. We then obtain the multi-class classifiers by applying the method mentioned in Part III.



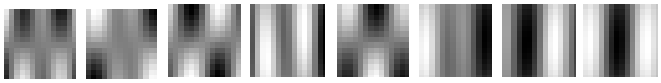
(a) PCANet filter banks of the first stage（1-8）



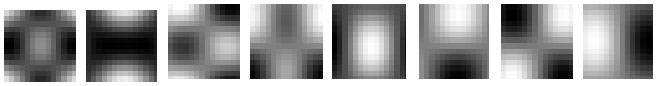
（b） PCANet filter banks of the second stage with respect to filter 1 of the first stage（1-8）



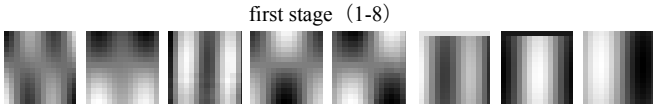
（b） PCANet filter banks of the second stage with respect to filter 2 of the first stage（1-8）



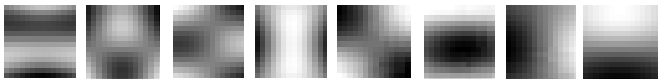
（c） PCANet filter banks of the second stage with respect to filter 3 of the first stage（1-8）



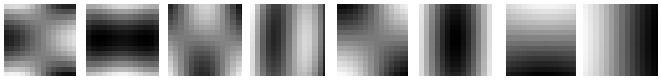
（e） PCANet filter banks of the second stage with respect to filter 4 of the first stage（1-8）



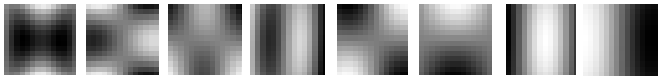
（f） PCANet filter banks of the second stage with respect to filter 5 of the first stage（1-8）



（g） PCANet filter banks of the second stage with respect to filter 6 of the first stage（1-8）



（h） PCANet filter banks of the second stage with respect to filter 7 of the first stage（1-8）



（i） PCANet filter banks of the second stage with respect to filter 8 of the first stage（1-8）

Fig. 3. PCANet filters

C. The test of the identification rate and analysis of the test results.

From all the training samples of each class, we select 80% to build our multi-class classifiers, with the rest 20% reserved to test the classification rate.

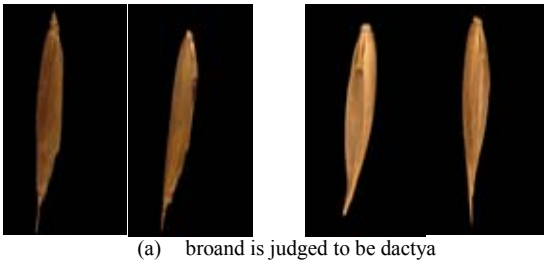
TABLE I

RESULT OF THE EXPERIMENT OF WEED SEEDS RECOGNITION

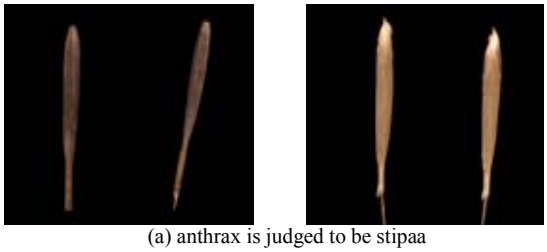
PCANet training samples number	Recognition training samples number	Recognition testing samples number	Correct Recognition number	Class number (rate= 100%)	Average accuracy rate
3980	3155	825	744	45	90.96%

The test result is displayed in Table I. We can see that our method achieves good performance. The efficiency of the weed seeds classification is very good, where the accuracy rate of 45 classes is 100%, and the average accuracy rate of each class is 90.96%.

After tracking the seeds with wrong classification and respective classes, we find the similarity between the two classes is very high. Because the difference between their shape, texture, and color is very slight, the image of a class is easier to be identified than the one with incorrect identification. As showed in Fig. 4, the broand is classified to dactya, and the anthra is classified to stipaa.



(a) broand is judged to be dactya



(a) anthrax is judged to be stipaa

Fig. 4. Seeds samples being identified incorrectly. Seeds in left is judged to be the seeds in the right.

From the above figure, we can see that the similarity between two classes is high, while the difference of the extracted features is quite small. As a result, it is hard to classify them well, which results in the incorrect identification of these seeds.

D. The classification of weed seeds without using PCANet.

We use the row features of weed seeds to build classifiers, rather than those extracted by PCA filter banks, binarization, and counting the histograms after the training of PCANet. To make it comparable, we still use the LMC to build the classifiers between the weed seeds. The average accuracy rate of our experiments is 64.80%.

Comparing this result with that of the experiments with PCANet, we can find that the PCANet helps us to obtain more useful features of weed seeds, which is indispensable to building effective classifiers in achieving high classification accuracy rate.

E. The impact of translation on weed seeds identification rate

In this experiment, we translate the weed seeds images with different distances in different directions.

a. Translation in x direction. We translate the test images in x direction with several different distances (the positive distance means that we translate the images to the right side, while the negative distance means that we translate the images to the left side).Then we test the impact of different distances of translation on the identification rate of test images. In our experiments, the pixel distances we translate the images into are 0, 2, -2, 5,-5, 8, -8, 10, -10. So we have done 9 experiments in total.

Fig. 5 displays the changes of the test images after translation in x direction. The results of our experiments are recorded in Table II. Fig. 6(a) shows the changing trend of identification rate.

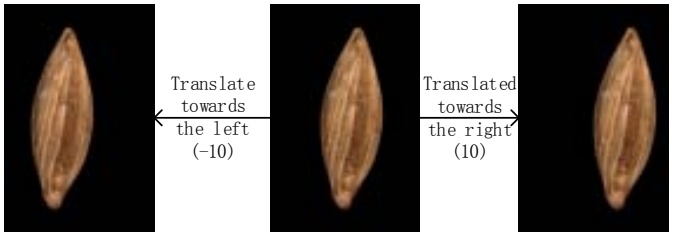


Fig. 5. Images is translated in x direction

b. Translation in y direction. We translate the test images in y direction with several different distances (the positive distance means that we translate the images downwards, while the negative distance means that we translate the images upwards).Then we test the impact of different distances of translation on the identification rate of test images. In our experiments, the pixel distances we translate the images into are 0, 2, -2, 5,-5, 8, -8, 10, -10. So we have done 9 experiments in total.

The changes of the test images after translation in y direction are displayed in Fig. 7. The results of our experiments are recorded in Table III. What's more, Fig. 6(b) shows the changing trend of identification rate.

As the experiments *a* and *b* demonstrate, even though the test weed seeds are translated in x or y direction, the PCANet still maintains good identification rate as long as the translation distance is not very large. That is to say, PCANet has translation invariance to weed seeds classification to some extent.

F. The impact of rotation on weed seeds identification rate

In this experiment, we deform the test images with in-plane rotation artificially. There are several different rotation angles (the positive angle means that we rotate the images in a clockwise direction, and the negative angle means that we rotate them counterclockwise). In our experiment, the angles from which we rotate the images are 0^o, 2^o, -2^o, 5^o, -5^o, 8^o, -8^o, 10^o, 10^o. We have done 9 experiments in total.

Fig.8 displays the changes of the test images after in-plane rotation. The results of our experiments are recorded in Table IV. Fig. 6(c) shows the changing trend of identification rate.

As the results indicate, PCANet can still keep relatively high identification rate to the test weed seeds in condition that the in-plane rotation angle is within a certain range. In our experiments, when the rotation angle ranges from -10° to 10°, the recognition rate can reach above 80%. This implies that

TABLE II
THE EXPERIMENT RESULTS OF WEED SEEDS CLASSIFICATION WITH TEST IMAGES TRANSLATED IN X DIRECTIONS

Translation distance (pixels)	0	2	-2	5	-5	8	-8	10	-10
Recognition rate (%)	90.96	88.69	88.65	81.25	81.18	60.78	61.56	42.32	43.39

TABLE III
THE EXPERIMENT RESULTS OF WEED SEEDS CLASSIFICATION WITH TEST IMAGES TRANSLATED IN Y DIRECTIONS

Translation distance (pixels)	0	2	-2	5	-5	8	-8	10	-10
Recognition rate (%)	90.96	89.06	89.58	82.40	81.86	66.27	67.89	50.59	52.90

TABLE IV
THE EXPERIMENT RESULTS OF WEED SEEDS CLASSIFICATION WITH TEST IMAGES ROTATED IN-PLANE

Rotation angle (o)	0	2	-2	5	-5	8	-8	10	-10
Recognition rate (%)	90.96	88.71	89.30	87.32	85.97	85.25	83.58	81.41	80.54

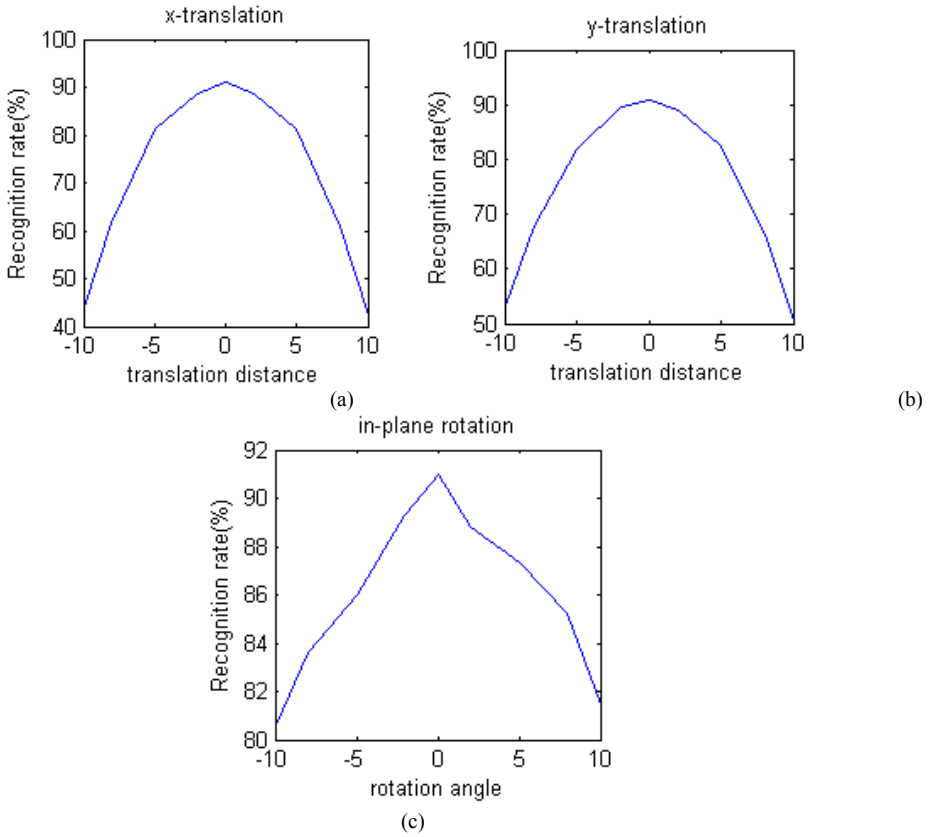


Fig. 6. Deform the test images and test the recognition rate of seeds images after deformation. (a) Translation in x direction. (b) Translation in y direction. (c) In-plane rotation

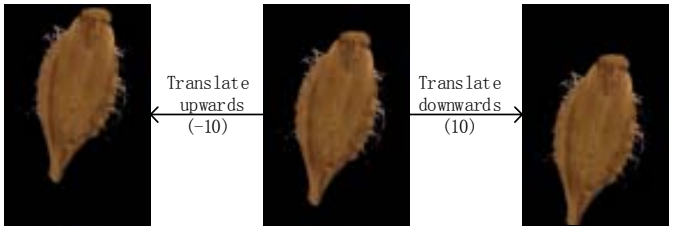


Fig. 7. Images is translated in y direction

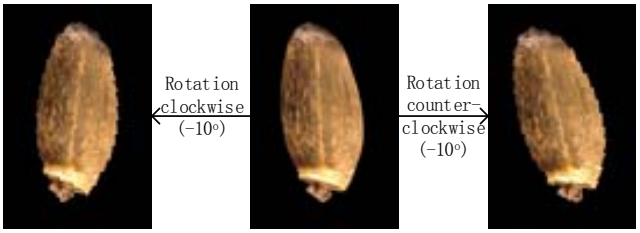


Fig. 8. Images is rotated in-plane

PCANet has relatively high rotation invariance to weed seeds classification.

V. CONCLUSION

In this paper, we put forward a variation method of the PCANet and apply it to the feature extraction of the weed seeds. In the identification and classification of the weed seeds, we make use of the LMC method based on affine hulls.In our experiment, we build multi-class linear classifiers among the weed seeds of 91 different classes. We conduct the experiment and analyze its results. We find that this method achieves good performance in our experiment and improves the accuracy rate of the weed seeds classification.

At the same time, we find the algorithm has good robustness to the recognition of weed seeds. When we make relatively small deformation to seeds images, such as translation and rotation, our algorithm still maintains a relatively high recognition rate to the test images, especially for rotation. However, if two classes are very similar to each other in terms of shape, texture, and color, the accuracy rate is not satisfactory. Therefore, we need to make more improvement of this method. Nevertheless, this method is a good application of the deep leaning network to weed seeds classification and thus provides a new approach to improving classification efficiency

ACKNOWLEDGMENT

This work is supported by National Natural Science Foundation of China (Project 61202188).

REFERENCES

[1] Chan, T.H., Jia, K., Gao, S., Lu, J., Zeng, Z. and Ma, Y., "PCANet: A Simple Deep Learning Baseline for Image Classification," unpublished.

[2] Goodfellow, I.J., Warde-Farley, D., Mirza, M., Courville, A. and Bengio, Y., "Maxout networks," Journal of Machine Learning Research Workshop and Conference Proceedings, 2013, 28(3), pp. 1319-1327.

[3] Lecun, Y., Bottou, L., Bengio,Y. and Haffner, P., "Gradient-based learning applied to document recognition," Proceedings of the IEEE, 1998, 86(11), pp. 2278–2324.

[4] Jarrett, K., Kavukcuoglu, K., Ranzato, M. and LeCun, Y., "What is the best multi-stage architecture for object recognition," Computer Vision, 2009 IEEE 12th International Conference, 2013, pp. 2146-2153.

[5] Bruna, J. and Mallat, S., "Invariant scattering convolution networks," IEEE Transactions on Pattern Analysis and Machine Intelligence, 2013, 35(8), pp. 1872–1886.

[6] Lee, H., Grosse, R., Rananth, R. and Ng, A., "Convolutional deep belief networks for scalable unsupervised learnig of hierarchical representation," International Conference on Machine Learning, 2009, pp. 77-616.

[7] Krizhevsky, A., Sutskever, I. and Hinton, G., "Imagenet classification with deep convolutional neural network," Neural Information Processing Systems, 2012, pp. 1097-1105.

[8] Kavukcuoglu, K., Sermanet, P., Boureau, Y., Gregor, K., Mathieu, M. and LeCun, Y., "Learning convolutional feature hierarchies for visual recognition," Neural Information Processing Systems, 2010, pp. 1090-1098.

[9] Sifre, L. and Mallat, S., "Rotation, scaling and deformation invariant scattering for texture discrimination," IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 1233-1240.

[10] Chtioui, Y., Bertrand, D., Dattee, Y. and Devaux, M.F. "Identification of Seeds by Colour Imaging: Comparison of Discriminant Analysis and Artificial Neural Network," Journal of the Science of Food and Agriculture, 1996, 71(4), pp. 433-441.

[11] Granitto, P.M., Verdes, P.F., Ceccatto, H.A., "Large-scale investigation of weed seed identification by machine vision," Computers and Electronics in Agriculture, 2005, 47(1), pp. 15–24.

[12] Granitto, P.M., Navone, H.D., Verdes, P.F. and Ceccatto, H.A., "Weed seeds identification by machine vision," Computers and Electronics in Agriculture, 2002, 33(2), pp. 91–103.

[13] Granitto, P.M., Garralda, P.A., Verdes, P.F. and Ceccatto, H.A., "Boosting Classifiers for Weed Seeds Identification," Journal of Computer Science & Technology, 2003, 3(1), pp. 34-39.

[14] Cevikalp, H., Triggs, B., Yavuz, H.S., Küçük, Y., Küçük, M. and Barkana, A., "Large margin classifiers based on affine hulls," Neural Computation, 2010, 73(16), pp. 3160-3168.

[15] Hinton, G., Osindero, S. and Teh, Y.-W., "A fast learning algorithm for deep belief nets," Neural Computation, 2006, 18(7), pp. 1527–1554.

[16] Bruna, J. and Mallat, S., "Invariant scattering convolution networks," IEEE Transactions on Pattern Analysis and Machine Intelligence, 35(8), 2013, pp. 1872–1886.

[17] Sifre, L. and Mallat, S., "Rotation, scaling and deformation invariant scattering for texture discrimination," IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 1233-1240.

[18] Burges, C.J.C., "A tutorial on support vector machines for pattern recognition," Data Mining and Knowledge Discovery, 1998, 2(2), pp. 121-167.

[19] Cortes, C. and Vapnik, V., "Support vector networks," Machine Learning, 1995, 20(3), pp. 273-297.

[20] Chang, C.-C. and Lin, C.-J., "LIBSVM: a library for support vector machines," ACM Transactions on Intelligent Systems and Technology, 2011, 2(27), pp. 1-27.

[21] Burges, C.J.C., "A tutorial on Support Vector Machines for Pattern Recognition." Knowledge Discovery and Data Mining, 1998, 2(2), pp. 121-167.

[22] Weston, J. and Watkins, C., "Support vector machines for multi-class pattern recognition," Proceedings of 7th European Symposium on Artificial Neural Networks, 1999, pp. 219-224.

[23] Platt, J., Cristianini, N. and Shawe-Taylor, J., "Large margin DAGs for multiclass classification," Proceeding of Advances in Neural Information Processing Systems 12, 1999, pp. 547-553.

[24] Liu, C. and Wechsler H., "Gabor feature based classification using the enhanced fisher linear discriminant model for face recognition, " IEEE Transactions on Image Processing, 2002, 11(4), pp. 467-476.