Effective insect recognition using a stacked autoencoder with maximum correntropy criterion

International Joint Conference on Neural Network, 2015, Killarney.
http://www.producao.usp.br/handle/BDPI/49425

Downloaded from: Biblioteca Digital da Produção Intelectual - BDPI, Universidade de São Paulo
Effective Insect Recognition Using a Stacked Autoencoder with Maximum Correntropy Criterion

Yu Qi\textsuperscript{1}, Goktug T. Cinar\textsuperscript{2}, Vinicius M. A. Souza\textsuperscript{3}, Gustavo E. A. P. A. Batista\textsuperscript{3}, Yueming Wang\textsuperscript{1}, Jose C. Principe\textsuperscript{2}
\textsuperscript{1}Qiushi Academy for Advanced Studies, Zhejiang University, Hangzhou, China
\textsuperscript{2}Computational NeuroEngineering Laboratory, University of Florida, Gainesville, FL, USA
\textsuperscript{3}Institute of Mathematics and Computer Science, University of Sao Paulo, Sao Carlos, SP, Brazil

Abstract—Throughout the history, insects had been intimately connected to humanity, in both positive and negative ways. Insects are an important part in crop pollination, on the other hand, some of them spread diseases that kill millions of people every year. Effective control of harmful insects while having little impact to beneficial insects and environment is extremely important. Recently, an intelligent trap that uses laser sensors was presented to control the population of target insects. The device could record and analyze sensor signals when an insect passes through the trap and make quick decisions whether to catch it or not. The effectiveness of the trap relies on the correct choice of classification algorithm to perform the insect detection. In this paper, we propose to use a deep neural network with maximum correntropy criterion (MCC) for reliable classification of insects in real-time. Experimental results show that, deep networks are effective for learning stable features from brief insect passage signals. By replacing the mean square error cost with MCC, the robustness of autoencoders against noise is improved significantly and robust features could be learned. The method is tested on five species of insects and a total of 5325 passages. High classification accuracy of 92.1\% is achieved. Compared with previously applied methods, better classification performance is obtained using only 10\% of the computation time. Therefore, our method is efficient and reliable for online insect detection.

I. INTRODUCTION

Insects have had an intimate relationship with humankind in both positive and negative ways. Beneficial insects play an important role in pollinating the crops. On the other hand, insects also carry diseases that kill millions of people every year and leave tens of millions sickened. Pesticide is one of the popular solutions for harmful insect control. However, due to the increased pesticide resistance and the negative effects on human health and beneficial insects, the usage of pesticides should be cautious and controlled. New solutions for effective insect control with minimal impact to beneficial insects are still being studied.

Recently, Batista et al. have presented a new low-cost optical sensor that captures insect flight information using a laser light and an array of phototransistors [1]. The main purpose of this sensor is to classify the insects according to their species. This sensor is an important part of an environment friendly intelligent trap for harmful insect population control as presented in [1]. By analyzing the signals gathered by the sensor in real-time, the device quickly identifies and selectively trapsest target species, while releasing other insects back into the environment. In this way, the population of harmful insects as agricultural pests or disease vectors could be effectively controlled while the impact on beneficial insects as pollinators is minimized.

The effectiveness of intelligent traps relies on insect detection. In order to identify target insects precisely in real-time, the detection should be reliable and efficient. Since the insect signal is similar to audio signal, Silva et al. applied audio analysis methods such as Mel-Frequency Cepstral Coefficients (MFCC), Linear Predictive Coding coefficients (LPC) and Line Spectral Frequencies (LSF) to extract useful features [1]. For multi-class insect classification, different classifiers including Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Decision Trees, and Gaussian Mixture Model (GMM) have been applied [1].

Although existing methods have shown some strengths on insect recognition, capturing reliable features from laser signals proves to be a challenging task. First, since the moment of insect passage through the laser sensor is usually brief, i.e. less than 100 milliseconds, the patterns are unstable in both time and frequency domain; thus stable features are difficult to extract using conventional features. Second, signals recorded by the sensor are formed by occasional events (insect’s passages) and background noise due to the physical characteristics of low-cost lasers; thus achieving reliable detection which is robust to noise is a challenging task.

In order to learn effective features from insect data, we propose to apply deep networks [2], [3] for reliable feature learning and robust insect classification. The main contributions of our work can be summarized as follows:

- We propose to use a deep network called robust stacked autoencoder (R-SAE) to reduce the effect of noise in the sensor signals, employing a maximum correntropy criterion (MCC). Compared with standard stacked autoencoder (S-SAE) model which uses mean square error (MSE), MCC is more robust to noise and outliers;
- The R-SAE model is integrated with a classifier to be fine tuned in a supervised manner. Thus, the features could be optimized together with the classifier to obtain a better classification performance.

Our method is evaluated on a dataset including five species of insects and 5325 very short signals examples. Experimental results show that the features learned by R-SAE model are stable within classes and robust to noise. Using R-SAE features, high classification accuracy of 92.1\% is achieved. Besides, our
method is computationally more efficient compared to currently deployed solutions and suitable for real-time detection, which is especially important for the intelligent trap due to the need in making quick decisions of capturing or releasing an insect.

The rest of this paper is organized as follows. Section 2 presents the detail of data collection and the R-SAE model. Section 3 illustrates the insect classification results by R-SAE in comparison with other methods. Finally, we draw the conclusions in Section 4.

II. METHOD

The framework of our method is shown in Figure 1. First the signals are recorded by the sensor when insects pass the laser. Then we preprocess the signal by removing sections with no activity and convert the signals into mel-spectrum domain. After that, R-SAE model is trained to learn stable features from mel-spectrum. Finally, the features are fed into a classifier for insect recognition.

A. Data Collection

In our study, we use the insect dataset previously evaluated in [4]. This dataset has insect passage signals from two species of flies and three species of mosquitoes. The flies species are the *Drosophila melanogaster* (popularly known as fruit flies) and the *Musca domestica* (flies). The mosquito species are the *Culex quinquefasciatus* (a vector of lymphatic filariasis), *Culex tarsalis* (a vector of St. Louis Encephalitis and Western Equine Encephalitis) and the *Aedes aegypti* (a vector of filariasis, dengue fever, yellow fever, and West Nile virus). The signal sample rate is 16000 Hz and the amplitude of signal has been normalized to [0, 1]. Table I presents a description of the dataset and the training/testing split sizes used in our experiments.

<table>
<thead>
<tr>
<th>Species</th>
<th>Total</th>
<th>Train</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aedes aegypti</td>
<td>904</td>
<td>800</td>
<td>104</td>
</tr>
<tr>
<td>Musca domestica</td>
<td>917</td>
<td>800</td>
<td>117</td>
</tr>
<tr>
<td>Drosophila melanogaster</td>
<td>954</td>
<td>800</td>
<td>154</td>
</tr>
<tr>
<td>Culex quinquefasciatus</td>
<td>1285</td>
<td>800</td>
<td>485</td>
</tr>
<tr>
<td>Culex tarsalis</td>
<td>1265</td>
<td>800</td>
<td>465</td>
</tr>
<tr>
<td>Sum</td>
<td>5325</td>
<td>4000</td>
<td>1325</td>
</tr>
</tbody>
</table>

The insect signals were collected by a laser sensor which is part of the intelligent trap as illustrated in Figure 2. The trap includes an attractant to bait the insects, some open-close doors which are responsible for trapping or releasing the insect, a fan to drive the insect through the trap, and a laser sensor. When an insect flies in front of the trap entrance, it is gently pushed by an airflow toward the laser sensor. As the insect crosses the laser, its wings partially occlude the light, causing slight light fluctuations captured by the phototransistor. This signal is recorded by the sensor and analyzed in real-time for insect classification. If a target insect is identified, the airflow would push the insect into the chamber to trap it. Otherwise, the airflow would be reversed to release the insect.

B. Data Preparation

As shown in Figure 3, the insect passages only last for less than 100 milliseconds for most of the signals. Thus, we preprocess the data segments to remove the blocks of signal with no activities at each end.
In silence truncation, a sliding window of 50 samples moves sample by sample across the signal. For each time window, the maximum absolute magnitude of the amplitude is calculated. We set a threshold of \( TH = 0.1 \). The first and last time windows with their maximum absolute values in amplitude bigger than \( TH \) are considered to be the start and end of insect passage, respectively. Examples of silence truncation results are illustrated in Figure 4.

C. Mel-spectrum Computation

After truncation, the signal segments are windowed using a Hanning window. Inspired by the success of audio analysis in insect feature extraction [1], we first convert the signal into mel-spectrum to reveal useful features. The mel-spectrum is a frequency spectrum calculated on the mel scale, which approximates the human auditory system’s response. Equation 1 shows the conversion from frequency \( f \) to mel scale.

\[
\text{mel} = 2595 \times \log_{10} \left(1 + \frac{f}{700}\right).
\]  

In our experiment, a total of 100 triangular mel weighting filters are employed in mel-spectrum computation.

D. Robust Autoencoder

Due to the short duration of insect signals, the mel-spectrum feature is unstable within classes and sensitive to noise. Therefore, robust stacked autoencoder (R-SAE) is employed to extract stable and robust features from mel-spectrum. By virtue of good outlier suppression ability of maximum correntropy criterion (MCC), the R-SAE model is robust to noise and has shown high feature learning performance in some applications [5].

In this section, the formulation of robust autoencoder is described. We first begin with the basic autoencoder. Then the robust autoencoder with MCC is presented to learn robust features under noise. Finally, we stack the robust autoencoders together with a classifier to build a deep network for global optimization.

1) Basic Autoencoder: The basic autoencoder model is a three-layer artificial neural network including an encoder and a decoder. The encoder takes an input vector \( x \), and maps it to a hidden representation \( x' \) through a non-linear function:

\[
x' = s(W^{(1)}x + b^{(1)}),
\]  

where \( s(.) \) is the sigmoid function. Suppose \( x \) and \( x' \) are \( d \)-dimensional and \( d' \)-dimensional vectors respectively, then \( W^{(1)} \) is a \( d' \times d \) weight matrix and \( b^{(1)} \) is a \( d' \)-dimensional bias vector.

The decoder then converts the vector \( x' \) back to vector \( y \) for input reconstruction:

\[
y = s(W^{(2)}x' + b^{(2)}),
\]  

where the output vector is \( d \)-dimensional, \( W^{(2)} \) is \( d \times d' \) and \( b^{(2)} \) is a \( d \)-dimensional bias vector.

The objective for autoencoder is to minimize the reconstruction error between input \( x \) and output \( y \), with respect to a loss function \( L \):

\[
\theta = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} L(x_i, y_i),
\]  

where \( L \) is the loss function, and the parameter set \( \theta = \{W^{(1)}, b^{(1)}, W^{(2)}, b^{(2)}\} \). For traditional autoencoder, mean square error (MSE) is used as:

\[
J_{MSE}(\theta) = \frac{1}{n} \sum_{i=1}^{n} L_{MSE}(x_i, y_i) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{2}||y_i - x_i||^2.
\]  

2) Correntropy and MCC: The drawback of the traditional autoencoder is that, the feature learning performance could be highly weakened when there are non-Gaussian noises and outliers in the training samples because the MSE loss is sensitive to outliers [6], [7]. In order to improve the robustness to noise of the autoencoder, we replace the MSE loss by a correntropy based criterion.

Correntropy is defined as a localized similarity measure [8]. Since it is less sensitive to outliers compared with traditional second-order statistics such as MSE, it has been applied for robust algorithm design [9], [10]. The correntropy induces a new metric such that, as the distance between \( X \) and \( Y \) gets larger, the equivalent distance evolves from 2-norm to 1-norm and eventually to zero-norm when \( X \) and \( Y \) are far apart [8]. Therefore, the correntropy measure is particularly effective in outlier suppression.

For two random variables \( X \) and \( Y \), the formulation of correntropy is defined by:

\[
V_\sigma(X, Y) = E[\kappa_\sigma(X - Y)],
\]
where $E[.]$ is the mathematical expectation and $\kappa_\sigma(.)$ is the Gaussian kernel with kernel size of $\sigma$:

$$\kappa_\sigma(.) = \frac{1}{\sqrt{2\pi} \sigma} \exp\left(-\frac{(.)^2}{2\sigma^2}\right).$$  

(7)

In practice, the joint probability density function is usually unknown and only a finite set of samples of $\{(x_i, y_i)\}_{i=1}^N$ is available for both $X$ and $Y$. Then the estimated correntropy can be calculated by:

$$\hat{V}_\sigma(X, Y) = \frac{1}{N} \sum_{i=1}^N \kappa_\sigma(x_i - y_i).$$  

(8)

The maximum of correntropy error in Equation (8) is called the maximum correntropy criterion (MCC) [8]. Due to the good outlier rejection property of correntropy, MCC is robust to non-Gaussian noises.

3) Robust Sparse Autoencoders with MCC: By replacing the MSE with MCC, the anti-noise ability of autoencoders could be highly improved. In R-SAE model, we measure the reconstruction loss between the input vector $x$ and the output vector $y$ by MCC instead of MSE:

$$J_{MCC}(\theta) = \frac{1}{n} \sum_{i=1}^n L_{MCC}(x_i, y_i)$$

$$= \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^m \kappa_\sigma(x_i^j - y_i^j),$$  

(9)

where $n$ is the number of training samples and $m$ is the length of each training samples. The optimal parameter $\theta$ is obtained when $J_{MCC}(\theta)$ is maximized.

In the cost function formulation, a sparsity-inducing term is adopted to encourage the deep model to capture more patterns. The sparsity-inducing term is defined as in [11]:

$$J_{\text{sparse}}(\theta) = \beta \sum_{i=1}^{s_2} KL(\rho||\hat{\rho}_i),$$  

(10)

where $\beta$ is the weight adjustment parameter, $s_2$ is the number of units in the second layer, $\hat{\rho}_i$ is the activation value for the $i$th hidden layer unit and $\rho$ should be small. The sparsity-inducing term constrains that the value of $\hat{\rho}_i$ should be near $\rho$ under Kullback-Leibler divergence.

In order to avoid over-fitting problem, a weight decay term $J_{\text{weight}}(\theta)$ is added:

$$J_{\text{weight}}(\theta) = \frac{\lambda}{2} \sum_{l=1}^L \sum_{i=1}^{s_l} \sum_{j=1}^{s_{l+1}} (w_{lj}^{(l)})^2,$$  

(11)

where $w_{lj}^{(l)}$ represents an element in $W^{(l)}$, $\lambda$ is the parameter to adjust the weight of $J_{\text{weight}}(\theta)$ and $s_l$ denotes number of units in layer $l$. Therefore, the cost function of the proposed robust autoencoder is defined as:

$$J_{R-SAEx}(\theta) = -J_{MCC}(\theta) + J_{\text{weight}}(\theta) + J_{\text{sparse}}(\theta).$$  

(12)

By minimizing the cost of $J_{R-SAEx}(\theta)$, the parameter set $\theta$ could be optimized.

4) Stacked Robust Sparse Autoencoders: Finally, we stack the robust autoencoders into a deep neural network, which is similar to stacking the ordinary autoencoders [12].

The training process of the deep network includes two stages: unsupervised pretraining and supervised fine tuning. In the pretraining stage, the network is trained layer-wisely by the proposed robust autoencoder model to learn useful features from the data. A well pretrained network yields a good starting point for fine tuning [13], [14]. In the fine tuning stage, a softmax classifier is cascaded to the highest layer of the stack, and the whole system is tuned to minimize the classification error in a supervised way. The network is globally tuned through back-propagation and all the parameters of both feature extraction and classification are jointly optimized. After fine tuning, the deep network is well configured to obtain optimal overall classification performance.

III. RESULTS AND DISCUSSION

In this section we present the evaluation of our method to the classification task of insect species. First we compare the R-SAE model with S-SAE to demonstrate the noise robustness of MCC. Then, we compare the accuracies of our method against other methods previously used in the insect classification. Finally, we compare the computational costs to show the efficiency of our method.

A. Feature Learning Performance

Examples of features learned by the R-SAE model are illustrated in Figure 5 in comparison with mel-spectrum. The mel-spectrum has been normalized for plotting. Since the segments of signal are short and unstable, the feature in mel-spectrum domain shows high intra-class variation. The intra-class variation is most significant in the 3rd species of insects for which the peaks of mel-spectrum vary in a wide range for different individuals. The noises also lead to fluctuations in mel-spectrum as shown in the 43rd and 74th segments in Figure 5 (b). Compared with mel-spectrum, the features of R-SAE model are much more stable and robust to noise. Figure 5 (c) and (d) show the features learned by R-SAE in 50 and 3 dimensions, respectively. Due to the noise reduction ability of correntropy, the features could stay stable and robust even in three dimensions.

B. Comparison with S-SAE

In order to evaluate the noise suppression ability of R-SAE model, experiments are carried out to compare the feature learning performance of S-SAE and R-SAE model. In order to evaluate the ability of the features quantitatively, we utilize the classification performance as the criterion.

For fair comparison, the cost function for S-SAE model is as follows:

$$J_{S-SAEx}(\theta) = J_{MSE}(\theta) + J_{\text{weight}}(\theta) + J_{\text{sparse}}(\theta),$$  

(13)

where the loss function $J_{MSE}(\theta)$ is formulated with MSE-based loss function as in Equation (5), $J_{\text{weight}}(\theta)$ and $J_{\text{sparse}}(\theta)$ are formulated the same as R-SAE.

In this experiment, we stack two autoencoders to constitute a three-layer network with 100 input units, 80 hidden units,
and we vary the output dimension from 50 to 5. Same stacked architectures are applied for both S-SAE and R-SAE. The networks are initialized randomly and back-propagation is used for layer-wise training. The parameters are set as $\lambda = 0.003, \beta = 3$ and $\rho = 0.1$ for both methods and $\sigma = 0.1$ for R-SAE.

The insect classification results for both S-SAE and R-SAE are shown in Table II and Figure 6. In unsupervised feature learning, R-SAE model with MCC cost obtains 89.6% of classification accuracy when using 50 output units, which is 12.7% higher than S-SAE with MSE cost. As the dimension of output reduces to 5, the performance of S-SAE is only 44.3%, while R-SAE reaches 77.9%. Due to good outlier suppression property of MCC, the R-SAE model could learn features from data effectively under noise.

After unsupervised feature learning, the R-SAE is optimized with softmax classifier to obtain optimal features for classification purpose. After the supervised fine tuning, high classification accuracies of 90.8%-91.3% are achieved with R-SAE when output dimension varies from 50 to 5. Therefore, supervised feature tuning could help learn more distinct and reliable features.

C. Comparison with Other Methods

In this section, the classification performance is compared against different methods. For each species of insects, 800 randomly selected segments are used for training the method while the rest is used for test. For each method, the classification accuracy has been averaged over 10 train-test trials. The methods included in the comparison are configured as follows:

- **MFCC+SVM**: For each segment, the MFCC vector is calculated as feature set and fed into a Support Vector Machine (SVM) classifier with Radial Basis Function (RBF) kernel. The parameters of SVM model are selected using 3-fold cross-validation ($C = 10^i, i = [-7, 5]; \gamma = 10^i, i = [-4, 0]$);

- **Mel+SVM**: The mel-spectrum vector is fed into SVM with RBF kernel. The parameters of SVM model are selected using 3-fold cross-validation ($C = 10^i, i = [-7, 5]; \gamma = 10^i, i = [-4, 0]$);

- **Mel+KNN**: The mel-spectrum vector is used in a KNN classifier. The number of neighbors $K$ is set to 15;

- **R-SAE+SVM(p)**: The R-SAE model is configured with 100 input units, 80 hidden units and $p$ output units. The parameters are set as $\lambda = 0.003, \beta = 3, \rho = 0.1$ and $\sigma = 0.1$. The outputs of the highest level are used as features for SVM training. SVM with RBF kernel is employed and the parameters are selected using 3-fold cross-validation ($C = 10^i, i = [-7, 5]; \gamma = 10^i, i = [-4, 0]$);

- **R-SAE+Softmax(q)**: The R-SAE model is configured with 100 input units, 80 hidden units and $q$ output units. The parameters of R-SAE model are set the same as in R-SAE+SVM(p). Softmax classifier is used for classification.

The classification accuracies are shown in Table III. For each insect species, the top two classification accuracies are highlighted. Overall, R-SAE+SVM obtains highest insect detection performance for each class and in total. R-SAE+SVM(50) achieves the best result of 92.1%. In the second place is R-SAE+Softmax method that, with 50-dimension R-SAE features, high classification accuracy of 91.3% is obtained. For Mel+SVM and Mel+KNN, the overall classification accuracies are 90.7% and 89.1%, respectively. With the MFCC feature set and SVM classifier, the classification accuracy is 87.4%. Compared with Mel+SVM method, the accuracies of R-SAE+SVM in single classes increase by 0.7%-6.0%. The most significant improvement is achieved with the *Drosophila melanogaster*, where the mel-spectrum shows large intra-class variation. Therefore, R-SAE feature shows better stability so that higher insect recognition performance could be achieved.

D. Comparison of Computational Cost

In this section, the computational costs of different methods are compared. Since the intelligent trap should classify insects in real-time, quick detection and low computational cost is required. In this experiment, we test the time cost of training stage and testing stage separately for each method. The train/test sets have already been divided randomly and the features have been calculated for each segments. The
Fig. 7. Comparison of time cost in a single trial. In subfigure (a), the blue (light) parts of bars are the time costs for R-SAE computation. In subfigure (b), the blue (light) parts of bars are the time costs for R-SAE training. In subfigure (a), the blue (light) parts of bars are the time costs for R-SAE computation.

<table>
<thead>
<tr>
<th>Method</th>
<th>50</th>
<th>40</th>
<th>30</th>
<th>20</th>
<th>10</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-SAE</td>
<td>76.9% ± 2.5%</td>
<td>76.0% ± 2.3%</td>
<td>71.0% ± 7.1%</td>
<td>64.1% ± 5.3%</td>
<td>45.9% ± 3.9%</td>
<td>44.3% ± 7.6%</td>
</tr>
<tr>
<td>R-SAE</td>
<td>89.6% ± 0.7%</td>
<td>88.4% ± 1.0%</td>
<td>87.9% ± 0.7%</td>
<td>86.6% ± 1.0%</td>
<td>86.8% ± 0.9%</td>
<td>77.9% ± 1.4%</td>
</tr>
<tr>
<td>R-SAE+FT</td>
<td>91.3% ± 0.9%</td>
<td>90.9% ± 0.7%</td>
<td>90.8% ± 0.8%</td>
<td>91.3% ± 0.8%</td>
<td>91.0% ± 0.6%</td>
<td>90.1% ± 0.5%</td>
</tr>
</tbody>
</table>

**TABLE IV. COMPARISON OF TIME COST.**

<table>
<thead>
<tr>
<th>Method</th>
<th>Train cost (s)</th>
<th>Test cost (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MFCC+SVM</td>
<td>106.04 ± 4.45</td>
<td>0.10 ± 0.01</td>
</tr>
<tr>
<td>MEl+SVM</td>
<td>216.94 ± 2.90</td>
<td>0.48 ± 0.03</td>
</tr>
<tr>
<td>MEl+KNN</td>
<td>0.57 ± 0.64</td>
<td>0.76 ± 0.15</td>
</tr>
<tr>
<td>R-SAE+SVM(50)</td>
<td>197.47 ± 0.97</td>
<td>0.30 ± 0.02</td>
</tr>
<tr>
<td>R-SAE+SVM(10)</td>
<td>150.75 ± 6.28</td>
<td>0.26 ± 0.07</td>
</tr>
<tr>
<td>R-SAE+Softmax(50)</td>
<td>120.28 ± 0.54</td>
<td>0.01 ± 0.00</td>
</tr>
<tr>
<td>R-SAE+Softmax(10)</td>
<td>99.49 ± 3.15</td>
<td>0.01 ± 0.00</td>
</tr>
</tbody>
</table>

The parameter optimization of SVM also takes 65 runs. Since the dimension of feature vectors fed into SVM is smaller, each run only takes 1.69 seconds. Compared with SVM, Softmax classifier is more efficient. In R-SAE+Softmax(50), only 31.9 seconds is needed for classifier training. The lowest overall cost in training stages belongs to MEl+KNN, which only takes 0.77 seconds in average.

On the other hand, the MEl+KNN spends the longest time in the test phase. Similar time cost is required for both MEl+SVM and R-SAE+SVM. Although deep neural networks are slow to train, the feedforward computing is rather efficient given the reduced feature dimension at the output of R-SAE. In R-SAE+SVM(50), only 0.01 second is required for R-SAE computing and 96.5% time cost is due to SVM classification. When we replace the SVM with Softmax, low time cost of 0.01 second is achieved, which is about only 10% of MFCC+SVM, 3% of R-SAE+SVM(50) and 2% of MEl+SVM. Since the training could be done offline, the key issue for online classification is low detection time. Therefore, the R-SAE+Softmax might be a good choice as classification algorithm to be embedded on intelligent traps.

**IV. CONCLUSION**

In this paper we present a new method for insect recognition using deep networks with maximum correntropy criterion (MCC). The R-SAE network extracts stable and compact features from very short signals from insects gathered by laser sensors. By replacing the MSE cost with MCC, the feature learning ability of deep networks increases by 12%-34%. Further, the classification accuracies improved 2%-12% after supervised global fine tuning. Compared with other methods such as the SVM classifier with mel-spectrum feature set, our method achieves higher classification accuracy with only 2% time cost in the test phase. Therefore, our method is robust and efficient, which is ideal for real-time insect identification in intelligent traps using laser sensors.

**ACKNOWLEDGMENT**

The authors would like to thank Fundação de Amparo a Pesquisa e Desenvolvimento do Estado de São Paulo (FAPESP) for financial support in the award numbers.

**TABLE III. COMPARISON OF METHODS FOR INSECT SPECIES CLASSIFICATION.**

<table>
<thead>
<tr>
<th>Species</th>
<th>MFCC+SVM</th>
<th>MEl+SVM</th>
<th>MEl+KNN</th>
<th>R-SAE+SVM(50)</th>
<th>R-SAE+SVM(10)</th>
<th>R-SAE+Softmax(50)</th>
<th>R-SAE+Softmax(10)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aedes aegypti</td>
<td>90.0% ± 2.8%</td>
<td>90.5% ± 2.2%</td>
<td>82.7% ± 2.2%</td>
<td>91.0% ± 1.3%</td>
<td>83.9% ± 0.8%</td>
<td>87.4% ± 0.5%</td>
<td></td>
</tr>
<tr>
<td>Musca domestica</td>
<td>84.6% ± 1.6%</td>
<td>93.6% ± 1.6%</td>
<td>84.9% ± 4.0%</td>
<td>91.8% ± 1.7%</td>
<td>91.4% ± 1.3%</td>
<td>90.7% ± 0.9%</td>
<td></td>
</tr>
<tr>
<td>Drosophila melanogaster</td>
<td>92.4% ± 4.7%</td>
<td>98.7% ± 1.0%</td>
<td>80.5% ± 1.6%</td>
<td>90.1% ± 1.9%</td>
<td>89.9% ± 1.1%</td>
<td>89.1% ± 0.8%</td>
<td></td>
</tr>
<tr>
<td>Culex quinquefasciatus</td>
<td>89.6% ± 3.7%</td>
<td>95.2% ± 1.6%</td>
<td>90.0% ± 2.2%</td>
<td>92.2% ± 1.3%</td>
<td>92.2% ± 1.6%</td>
<td>92.1% ± 0.7%</td>
<td></td>
</tr>
<tr>
<td>Culex quinquefasciatus</td>
<td>88.9% ± 2.5%</td>
<td>94.3% ± 1.7%</td>
<td>88.2% ± 2.5%</td>
<td>91.4% ± 0.9%</td>
<td>92.3% ± 0.8%</td>
<td>91.4% ± 0.5%</td>
<td></td>
</tr>
<tr>
<td>Overall</td>
<td>88.6% ± 2.9%</td>
<td>93.4% ± 3.0%</td>
<td>88.4% ± 2.6%</td>
<td>91.4% ± 1.4%</td>
<td>92.2% ± 1.9%</td>
<td>91.3% ± 0.9%</td>
<td></td>
</tr>
</tbody>
</table>

**TABLE II. ACCURACIES ACHIEVED BY S-SAE AND R-SAE VARYING THE OUTPUT DIMENSION.**

<table>
<thead>
<tr>
<th>Method</th>
<th>50</th>
<th>40</th>
<th>30</th>
<th>20</th>
<th>10</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-SAE</td>
<td>76.9% ± 2.5%</td>
<td>76.0% ± 2.3%</td>
<td>71.0% ± 7.1%</td>
<td>64.1% ± 5.3%</td>
<td>45.9% ± 3.9%</td>
<td>44.3% ± 7.6%</td>
</tr>
<tr>
<td>R-SAE</td>
<td>89.6% ± 0.7%</td>
<td>88.4% ± 1.0%</td>
<td>87.9% ± 0.7%</td>
<td>86.6% ± 1.0%</td>
<td>86.8% ± 0.9%</td>
<td>77.9% ± 1.4%</td>
</tr>
<tr>
<td>R-SAE+FT</td>
<td>91.3% ± 0.9%</td>
<td>90.9% ± 0.7%</td>
<td>90.8% ± 0.8%</td>
<td>91.3% ± 0.8%</td>
<td>91.0% ± 0.6%</td>
<td>90.1% ± 0.5%</td>
</tr>
</tbody>
</table>

The experiment is carried out on a computer with 2.5GHz Intel Core i5 CPU and 4 GB RAM. The computer runs 64-bit Windows 8 operating system. All the methods are implemented in Matlab, and LibSVM is used for SVM classifier.

Table IV shows the time cost for each method. The results are averaged over five trials. Figure 7 illustrates details of time costs for each part of the methods. Experimental results show that, MEl+SVM and R-SAE+SVM cost the most in the training stage. For MEl+SVM, high time cost is mostly due to parameter optimization. Since we search parameters of C and γ using cross-validation, there are a total of 65 runs in SVM training. The average time cost for each run is about 3.29 seconds. As illustrated in Figure 7 (a), for R-SAE+SVM(50), the training of R-SAE costs 93.7 seconds while SVM takes 109.6 seconds.
REFERENCES


