# Compact Feature Representation for Image Classification Using ELMs

Dongshun Cui[1,2], Guanghao Zhang[2], Wei Han[2],
Liyanaarachchi Lekamalage Chamara Kasun[2], Kai Hu[3] and Guang-Bin Huang[2]
[1]Energy Research Institute @ NTU (ERI@N), Interdisciplinary Graduate School.
[2]School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore.
[3]College of Information Engineering, Xiangtan University.
{dcui002, gzhang009, hanwei, chamarakasun, egbhuang}@ntu.edu.sg, kaihu@xtu.edu.cn

## Abstract

*Feature representation/learning is an essential step for many computer vision tasks (like image classification) and is broadly categorized as 1) deep feature representation; 2) shallow feature representation. With the development of deep neural networks, many deep feature representation methods have been proposed and obtained many remarkable results. However, they are limited to real-world applications due to the high demand for storage space and computation ability. In our work, we focus on shallow feature representation (like PCANet) as these algorithms require less storage space and computational resources. In this paper, we have proposed a Compact Feature Representation algorithm (CFR-ELM) by using Extreme Learning Machine (ELM) under a shallow network framework. CFR-ELM consists of compact feature learning module and a post-processing module. Each feature learning module in CRF-ELM performs the following operations: 1) patch-based mean removal; 2) ELM auto-encoder (ELM-AE) to learn features; 3) Max pooling to make the features more compact. Post-processing module is inserted after the feature learning module and simplifies the features learn by the feature learning modules by hashing and block-wise histogram. We have tested CFR-ELM on four typical image classification databases, and the results demonstrate that our method outperforms the state-of-the-art methods.*

## 1. Introduction

Feature representation/learning is an critical part for many computer vision tasks, such as object detection [6, 22, 27], object recognition [10, 2], object segmentation [9, 12] and image classification [11, 16, 21]. With years of research, a lot of work has been done on how to extract efficient and discriminative features manually or automatically. But this is still a hot research field because of facing challenges from illumination, occlusion, deformations and so on. Nowadays, neural network-based feature representation methods have made remarkable achievements, and they can be broadly categorized as deep and shallow feature learning.

Many deep feature learning algorithms have been proposed for image classification problems. Ciregan *et al.* [4] proposed a multi-column deep neural network which is a wide and deep artificial neural network architecture, which was claimed could match human performance on tasks like traffic signs image classification. A pyramid convolutional neural network was proposed for face representation by adopting a greedy-filter-and-down-sample operation in [7]. Ouyang *et al.* [27] proposed deformable deep convolutional neural networks to learn features for object detection. Shojaeilangari *et al.* [28] used extreme sparse learning to represent facial images for robust facial emotions recognition and achieved a high accuracy. However deep feature learning algorithms require a lot of computational resources, large amount of training time and huge storage space [29, 1].

In contrast, shallow feature learning algorithms usually don't have these disadvantages and for some tasks achieve performance comparable to deep feature learning algorithms. For example, Coates *et al.* [5] analyzed the feature learning ability of single-layer networks and achieved a high performance when the number of hidden nodes and other parameters are pushed to their limits. A method of unsupervised representation learning based on ELM is proposed on in [31], in which, ELM-AE was adopted as the learning unit, and a transferred layer was introduced. Besides, local contrast normalization (LCD) and whitening were employed as pre-processing steps. Chan *et al.* [3] proposed a simple learning architecture named PCA network (PCANet for short, and PCA is the abbreviated form of 'Principle Component Analysis'), which is quite intuitive and can be efficiently estimated. PCANet has attained remarkable results, but computing PCA costs a lot of time and resources [17].

Our work is partially motivated by the recent results

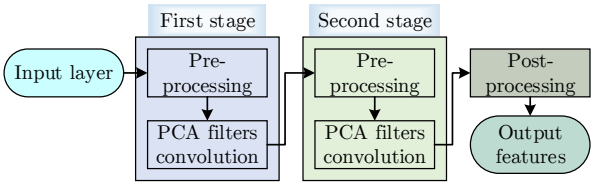



Figure 1. A schematic of two-stage PCANet.

in dimension reduction using ELM [17] which proved the ELM's generalization capability of learning local feature with fast speed. Besides, our work is inspired by PCANet architecture for its astonishing performance on image classification and pooling method adopted in many deep learning methods for its compact ability on feature extraction. All of these lead us to come up with a novel ELM-based shallow framework to learn compact features for image classification.

In the following sections, we introduce the related work on PCANet and ELM briefly. Then, we explain the whole compact feature representation method (CFR-ELM) we proposed, including the details of each feature learning module, max-pooling module and the overall framework for image classification which has a support vector machine (SVM) as a classifier. The proposed algorithm is tested on four typical image classification databases, and conclusions are given finally.

## 2. Related work

PCANet was introduced as a shallow feature learning algorithm in [3] and can be used as a simple baseline for deep feature learning algorithms. It contains three different types of modules: pre-processing, PCA filter convolution, post-processing. In general, the first two types of modules are combined together to form one feature learning layer. The overall architecture of PCANet is created by connecting the feature learning layers and the post-processing, and a two-stage PCANet architecture is shown in Fig.1.

In the pre-processing state, the sliding window with no stride is used to extract patches and remove the mean as:

$$\bar{\boldsymbol{x}}_k = \boldsymbol{x}_k - \frac{\sum_{i=1}^{n} \boldsymbol{x}_{k_i}}{n}\mathbf{1} \tag{1}$$

, where $\boldsymbol{x}_k$ denotes the $k$-th patch which contains $n$ pixel values, and $\mathbf{1}$ is an all one vector whose size is the same with $\boldsymbol{x}_k$. All patches are combined together to form a normalized matrix $\boldsymbol{X}$. After that, a PCA filter convolution operation is done. The objective function of PCANet is to minimize the reconstruction error with a set of filters, which is expressed as

$$\underset{\boldsymbol{V}}{\operatorname{argmin}} \|\boldsymbol{X} - \boldsymbol{V}\boldsymbol{V}^T\boldsymbol{X}\|_{\mathrm{F}}^2 \tag{2}$$

, where $\boldsymbol{V}$ is an orthogonal matrix, and $\|\cdot\|_{\mathrm{F}}^2$ is the Frobenius norm. The solution of the objective function is known as the principal eigenvectors of $\boldsymbol{X}$'s covariance matrix. First $l$ principal vectors (called PCA filters) are chosen as a convolution core to extract the features of the original images.

The second stage is similar to the first stage, and the main difference is that the total number of patches to be processed is $L$ (the number of filters in the first stage) times of the first stage. With the increase of stages, the amount of patches to be processed becomes larger and larger.

From the above analysis, we know that PCA learns a linear transformation, and it is not enough for representing the complex real-world features. ELM can make up this by its inherent abilities of linear and nonlinear representation. Extreme learning machines (ELM) was proposed along with the interdisciplinary development of effective machine learning theories and techniques in 2006 [15]. ELM has been attracting the increasing attentions from more and more researchers due to the amazing abilities of classification, regression, feature learning, sparse coding, and compression [14]. It has successfully been applied in more and more real industrial applications and usually achieves comparable or better results than many conventional learning algorithms at much fast learning speed.The basic ELM is expressed as:

$$\boldsymbol{\beta} = \boldsymbol{Y}(g(\boldsymbol{W}\boldsymbol{X} + \boldsymbol{B}))^{\dagger}. \tag{3}$$

In Equ.3, $\boldsymbol{X}$ is a $D \times N$ matrix that denotes the input data, where $D$ is the number of features, and $N$ is the number of samples. $\boldsymbol{W}$ is a $N_h \times D$ weight matrix between the input layer and the hidden layer, and the elements in $W$ is randomly initialized. $\boldsymbol{B}$ is a $N_h \times N$ bias matrix of the hidden neurons, where the first column is randomly initialized, and other columns are the duplicates of the first column. $\boldsymbol{Y}$ is the target data, and it is replaced by $X$ for the auto-encoder, that is, $\boldsymbol{Y} = \boldsymbol{X}$. $\boldsymbol{\beta}$ is a $D \times N_h$ weight matrix between output layer and hidden layer. $g(\cdot)$ is an activation function, which can be a Sigmoid function (default), a Sine function, a Radbas function, etc. The $(\cdot)^{\dagger}$ is a Moore-Penrose pseudoinverse operator.

Feature learning with ELM based on auto-encoder (ELM-AE) for big data is introduced in [23]. Recently, dimension reduction with extreme learning machine is analyzed in [17]. In this paper, ELM-AE is adopted as the core function of learning the features of input patches.

Most closely to our work is the hierarchical extreme learning machine (H-ELM) proposed in [31], which also uses ELM-AE to extract local receptive features and the training patches are randomly selected. In our proposed method, we drop out no patches and adopt a full connection network, which simplifies the network and need fewer parameters. Besides, we have a max-pooling layer for each feature learning state for learning a compact feature representation and the network architecture of our method is far

simpler since we have no whitening pre-processing and the connections between the first layer and last layer.

## 3. Compact Feature Representation using ELM

### 3.1. Overall Framework

To learning compact features efficiently, we proposed a shallow feature-learning architecture using ELM and the framework is shown in Fig.2. Each central processing unit of this framework consists of three parts: patch-based ELM auto encoder (AE), patch-based ELM decoder, and max-pooling operation. After the main process, two post-processing modules are added, and they are binary hashing and block-wise histogram respectively.

For image classification, assume the database has $N$ images $I_i$ ($i \in [1, N]$, and the resolution of each image is $W \times H$) and the corresponding labels are $Y_i$. The target of learning a more efficient representation $R(\cdot)$of $I_i$ is to minimize the loss function $\sum_{i=1}^{N} \|C(R(I_i) - Y_i)\|$, where $C(\cdot)$ denotes a classifier. In our work, support vector machine (SVM, [8]) is adopted to find the patterns of the learned features.

### 3.2. Unit of CFR-ELM

To remove the influence of illumination, we implement a patch operation of subtracting the mean. Assume the width and the height of the patch are $w_p$ and $h_p$ respectively, then we convert each sliding $w_p \times h_p$ window of the image $I_i$ into a column. All the columns construct a matrix $X_i$ by subtracting their means one by one. The width of $X_i$ is $(W - w_p + 1) \times (H - h_p + 1)$ and the height is $w_p \times h_p$.

An auto-encoder based on ELM is implemented to learn the features from all the $X_i$, which are denoted as X. We have introduced the details of ELM in Section 2. Three differences between ELM-AE and the basic ELM are listed as below:

- The input weight $\boldsymbol{W}$ and the bias $\boldsymbol{B}$ are both orthogonal matrixes. That is, we have $\boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}$ and $\boldsymbol{B}^T\boldsymbol{B} = \boldsymbol{I}$ with regard to ELM-AE.
- The output weight $\boldsymbol{\beta}$ is an orthogonal matrix. That is, $\boldsymbol{\beta}^T\boldsymbol{\beta} = \boldsymbol{I}$.
- The output of the ELM-AE network equals to the input.

Here, the number of hidden nodes $N_{F_k}$ corresponds to the number of convolution filters $F_k$ that we use in the $k$-th stage of the network. The filters are named as "channels" that are presented in Fig. 2. $N_{F_k}$ can be learned by optimization, and empirically it can be assigned a number less than 10 due to the limitation of the resources.

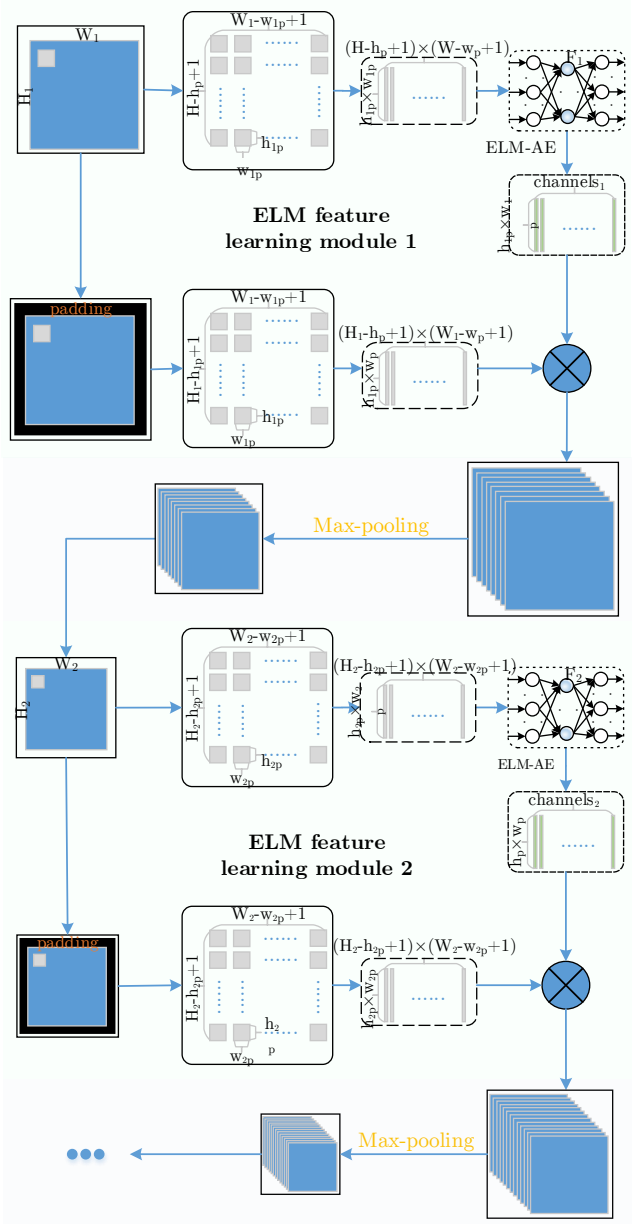



Figure 2. The framework of CFR-ELM

With the filters we get from ELM-AE, we can compute the outputs of the patches (overall denoted as $X$) from the raw samples. The outputs are used to represent the patterns of the raw samples. To keep the information on the edges, we pad each sample with zeros in its surroundings. In our experiments, we pad $w-1$ columns and $h-1$ rows of zeros before the first element and after the last element along the horizontal and vertical directions respectively. After that, the same process of mean-removal patch operation is performed. Finally, we can achieve all the patches $X$ of one sample, whose output $O$ in the $k$-th stage of the network is

expressed as

$$O = F_k * X, \tag{4}$$

where $F_k$ is the filter banks that we have achieved from the ELM-AE, and $X$ indicates the input samples that have been processed with zeros-padding and mean-removal operations. The output $O$ have $N_{F_k}$ layers for each $X$ in the $k$-th stage.

Noted that the size of each channel after the first ELM feature learning module is still $W_1 \times H_1$ with the cropping operation has been performed to countervail the impact of padding. So, the dimension of the output $O$ of $X$ increases to $W_1 \times H_1 \times N_{F_k}$, which requests lots of storage space and a large amount of computation. To learn a compact feature representation, $O$ is fed to a max pooling layer (shown in Fig.2 which is highlighted with golden poppy font).

Max pooling (MP) can significantly reduce the dimension of the input representation with losing limited valid information by the assumption that features are contained in the subregions. Each non-overlap subregion of $O$ is processed with a max filter, and the maximum values are retained to represent the corresponding subregions. Assume the size of the subregion of max pooling is $W_{MP} \times H_{MP}$, then after passing the max pooling layer, the dimension of $O$ will be $(W_1/W_{MP}) \times (H_1/H_{MP} * N_{F_K})$. Performances on image classification are provided and compared in Section 4.

Units of CFR-ELM are connected in sequence, and each unit is similar to the first unit but can have different parameters. Two units are adopted in our proposed shallow and compact feature representation architecture (denoted as $S$).

### 3.3. Post-processing Modules and Classification

CFR-ELM contains two post-processing modules: binary hashing and block wise histogram. They play an important role to transfer the output of the central processing units into a more efficient form. From the explanation of ELM, we can achieve $N_F$ output images for each sample, and all the output images are reshaped into the same size with the input images. Here, we have $N_F = \prod_{k=1}^{S} N_{F_k}$.

There are $N_{F_{S-1}}$ outputs $\Psi$ for one raw sample in the stage $S-1$, and each $\psi_\sharp \in \Psi$ ($\sharp \in [1, N_{F_{S-1}}]$) is processed by $F_S$ filters in the stage $S$ and generates $F_S$ corresponding outputs $\Theta$. Each $\theta_\hbar \in \Theta$ ($\hbar \in [1, F_S]$) is converted to a binary data matrix $\theta'_\hbar$ with a zero threshold. A hashing processing is performed on all the $\theta'$, and the operation is expressed as

$$T_\sharp = \sum_{\hbar=1}^{F_S} 2^{F_S - \hbar} \theta'_\hbar. \tag{5}$$

The block-wise histograms are computed for all the blocks we achieved with the sliding window technique. Assume the size of the block is represented as $w_b \times h_b$ and the overlap rate of the blocks is denoted as $\epsilon$, then the stride size of the sliding windows is $\langle (1-\epsilon)w_b \times (1-\epsilon)h_b \rangle$ ($\langle \cdot \rangle$ is a rounding operator). We go through the entire $T_\sharp$ with this stride and compute the block-wise histogram for each block and combine them together to form the final features.

SVM is adopted to make a classification decision based on the obtained compact features. It constructs an optimal hyperplane for the independent samples in feature space to maximize the functional margin and minimizing the misclassification cost (approximately equivalent to the number of misclassification samples) simultaneously. The advantage of SVM is the sparseness of the solution and good generalization ability even in a high dimension space.

## 4. Experiments

With the development of techniques for feature extraction and image classification, the accuracy has been very high for many datasets. To better exhibit the effectiveness of our proposed method, we make the test more difficult by decreasing the amount of training samples significantly and take more samples for testing. In our experiments, we test our algorithms on four classical image databases: a variant of MNIST (from Mixed National Institute of Standards and Technology [19]), Coil-20/100 (from Columbia Object Image Library [26, 25]), ETH-80 (from Eidgenssische Technische Hochschule Zrich [20]), and CIFAR-10 (from Canadian Institute for Advanced Research [18]). The first dataset is for digits classification, the second and third datasets are for controlled-environment objects classification, and the last one is for real-world objects classification.

To have a fair comparison with PCANet, the parameters assigned in our method are the same with the optimized PCANet (refer to [3] for more details on how to find the optimized parameters). We adopt two-stage($S = 2$) network, and the number of filters $N_{F_1}$ and $N_{F_2}$ are assigned to eight. The size of patches is $7 \times 7$, and max-pooling size is $2 \times 2$. The block size $w_b$ and $h_b$ are both assigned as seven. Error Rate is adopted to evaluate the performance of the experiments, and ER equals to $\frac{\sum_{i=1}^{n} ([P_i \neq T_i])}{N}$, where the value of $[\star]$ 1 when $\star$ is true and 0 if $\star$ is false. So, we know that the smaller ER is, the better the performance is. We have evaluated all the possible overlap rate $\epsilon$ from 0 to 0.9 by a step of 0.1, so for each dataset, we can obtain ten error rates.

The size of all the samples used in our experiments is consistent with the original resolution of these databases. We consider that obtaining higher accuracy by resizing the original images is unfair and not reasonable due to the curse of dimension. Additionally, we use less training samples and more test samples for the above three databases for a better illustration of the effectiveness of our algorithms. Even though our algorithm has the ability of generalization, some few parameters still should be noted. In the following subsections, we evaluate the performances of our method

Figure 3. Several sample images from MNIST dataset.

and PCANet on these databases separately. Few of the related state-of-the-art results are also be compared briefly.

### 4.1. Digits Image Classification

We first evaluate our proposed method on a variant of the widely used handwritten digits (0 - 9) image database MNIST, whose original version includes 60k training samples and 10k samples, and the state-of-the-art error rate is $0.21\%$ [30]. The variant of MNIST has much less training samples and more test samples, which means that it is much more challenging. Specifically, 10k, 2k, and 50k samples are used for training, validation, and test respectively. Classification of Handwritten Digits is not an easy task since *thickness* and *rotation angles* may differ a lot even for the same digit. The resolution of each image is $28 \times 28$, and the amounts of these ten digits have an approximate uniform distribution. We have shown some samples from MNIST in Fig.3 and the total number of each digit for training and test in Fig.4.

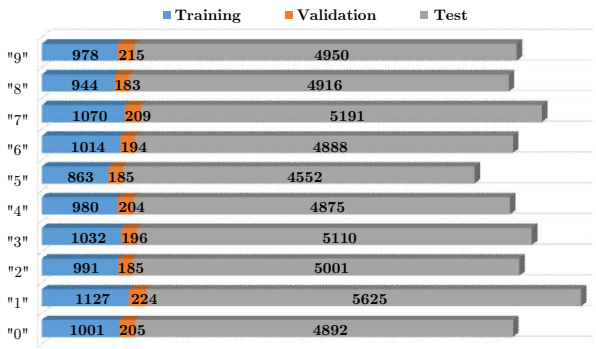



Figure 4. Amounts of the 10 digits in the MNIST variant.

For a better visualization of the mean-removal process, we have randomly selected ten samples for each digit from the database and show their mean-removal results in Fig.5.

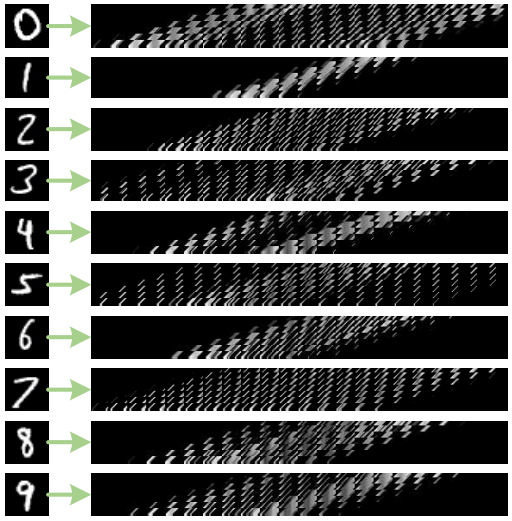


Figure 5. Results of the mean-removal patch operation. Each $28 \times 28$ image after the $7 \times 7$ patch mean-removal process.

Table 1. Results of PCANet and CFR-ELM with and without max pooling on the MNIST variant dataset.

| Overlap Rate ($\epsilon$) | ER & ER+MP ($[1, 1]$) (%) | |
|---|---|---|
| | PCANet | CFR-ELM |
| 0.0 | 1.150 | **1.082** |
| 0.1 | 1.118 | **1.052** |
| 0.2 | 1.118 | **1.022** |
| 0.3 | **1.066** | 1.094 |
| 0.4 | 1.072 | **1.010** |
| 0.5 | 1.072 | **1.034** |
| 0.6 | 1.020 | **0.990** |
| 0.7 | 1.050 | **1.040** |
| 0.8 | 1.058 | **1.022** |
| 0.9 | 1.058 | **0.980** |

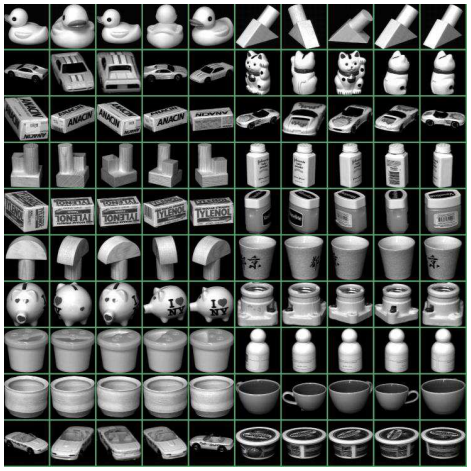
With the parameters we have introduced above, we have achieved ten error rates corresponding the ten overlap rates $\epsilon$. The results of PCANet and our method are indicated in Table.1 (the lower error rate has been set to bold). After the comparison, we know that our method has got a $0.980\%$ error rate and achieved lower error rates for the nine out of ten overlap rates and it performs $4.18\%$ (the average of the improvement rates) better than PCANet.

We have also performed our algorithm on the original version of MNIST dataset and obtains $0.49\%$ error rate which outperforms PCANet's $0.62\%$.

### 4.2. Coil-20 & Coil-100 Databases

Coil-20 is a gray-level black-background image database of 20 objects (shown in Fig.6 (a)). Objects are placed in a 360° turntable, and the table rotates 5° each time. So 72 images are captured for each object. Each image is cropped with a rectangle along the boundary of the object, and all

images are resized to 128×128 by maintaining the shape of the objects (shown in Fig.6 (b)). Coil-100 database is collected in the same controlled environment, but with more objects and all the images are colored. In the previous, they selected $1/3$ or $1/2$ of the samples for training. To make the classification problems more challenging, we split the samples into six groups and randomly select one of them for training, and the other groups are used for the test.



(a)

(b)

Figure 6. Several sample images from Coil-20 (a) and Coil-100 datasets (b).

The parameters setting for our method and PCANet are same with to the parameters assigned on MNIST except that the max pooling size is $2 \times 2$. Error rates are list in Table.2 and Table.3

### 4.3. ETH-80 shape classes Database

Even though the images of both ETH-80 database and Coil-100 database are colored and same size, the former is more challenging than the latter due to the following reasons:

Table 2. Results of PCANet and CFR-ELM with and without max pooling on the Coil-20 dataset.

| Overlap Rate ($\epsilon$) | ER (%) | | ER + MP (%) | |
|---|---|---|---|---|
| | PCANet | CFR-ELM | PCANet | CFR-ELM |
| 0.0 | 9.500 | **8.000** | 6.667 | **5.500** |
| 0.1 | 9.500 | **8.000** | 6.417 | **4.750** |
| 0.2 | 9.500 | **8.083** | 6.417 | **5.167** |
| 0.3 | 9.667 | **8.167** | 6.417 | **5.333** |
| 0.4 | 9.500 | **7.417** | 6.500 | **4.750** |
| 0.5 | 9.500 | **8.250** | 6.500 | **4.500** |
| 0.6 | 9.667 | **7.417** | 6.083 | **4.417** |
| 0.7 | 9.833 | **7.583** | 5.833 | **3.917** |
| 0.8 | 9.750 | **7.750** | 6.167 | **5.000** |
| 0.9 | 9.750 | **7.833** | 6.167 | **4.667** |

Table 3. Results of PCANet and CFR-ELM with and without max pooling on the Coil-100 dataset.

| Overlap Rate ($\epsilon$) | ER (%) | | ER + MP (%) | |
|---|---|---|---|---|
| | PCANet | CFR-ELM | PCANet | CFR-ELM |
| 0.0 | 11.830 | **9.050** | 6.183 | **5.600** |
| 0.1 | 11.680 | **10.300** | 6.083 | **5.683** |
| 0.2 | 11.680 | **8.550** | 6.083 | **5.267** |
| 0.3 | 11.780 | **8.920** | 6.033 | **5.850** |
| 0.4 | 11.770 | **10.400** | 5.967 | **5.133** |
| 0.5 | 11.770 | **10.270** | 5.967 | **5.533** |
| 0.6 | 11.850 | **9.150** | 5.617 | **5.400** |
| 0.7 | 11.870 | **10.470** | 5.700 | **5.367** |

- Color Background. Images of ETH-80 have a non-uniform blue chromakeying background while images of Coil-100 have a black background. Non-uniform background increases the useless information and affects the classifier's performance in further.

- Subcategories. ETH-80 contains 80 objects (shown in Fig.7) from 8 chosen categories (or super categories), that is, each category has ten subcategories. Different subcategories are adopted for training and test, which requires that the feature learning algorithms be sufficiently capable to learn the common characteristics of various objects of same super categories.

In the previous work [20, 13], they use not less than half of the dataset for training. While in our experiments, eight out of ten subcategories are used as test samples for each category.

We have compared our method with PCANet and the error rates are shown in Table.4 and the max pooling size is assigned as $2 \times 2$. It is clear that our method outperforms PCANet under all overlap rate.

Figure 7. Several sample images from ETH-80 dataset.

Table 4. Results of PCANet and CFR-ELM with and without max pooling on the ETH-80 dataset.

| Overlap Rate ($\epsilon$) | ER (%) | | ER + MP (%) | |
|---|---|---|---|---|
| | PCANet | CFR-ELM | PCANet | CFR-ELM |
| 0.0 | 20.351 | **19.512** | 18.052 | **16.374** |
| 0.1 | **20.236** | 21.113 | 16.054 | **15.815** |
| 0.2 | 20.236 | **19.817** | 17.744 | **15.815** |
| 0.3 | 20.122 | **19.970** | 17.687 | **15.100** |
| 0.4 | 20.160 | **18.674** | 16.924 | **15.772** |
| 0.5 | 20.160 | **18.483** | 17.565 | **15.772** |
| 0.6 | 19.817 | **19.627** | **16.128** | 16.301 |
| 0.7 | **19.588** | 19.855 | **16.859** | 16.872 |
| 0.8 | 19.360 | **18.293** | 17.219 | **16.638** |
| 0.9 | **19.360** | 20.922 | 17.500 | **16.638** |

### 4.4. CIFAR-10

The CIFAR-10 dataset is more challenging than all the above databases. It contains 60k $32 \times 32$ color images in 10 classes, and each class has the same amount. All the images are collected from the real-world scene, and the images are cropped without removing the various background. Besides, objects are difficult to identify due to the different conditions of illumination, occlusion, and non-alignment. Some samples are randomly selected from CIFAR-10 database and shown in Fig.8.

In our experiment, the experimental setup here is similar to [18, 24, 3]. However, due to the limited memories, we are 30k samples in the training set, and the remains are used for the test. And we only go through the overlap rate from 0 to 0.7 by the step of 0.1. For the experiments with max pooling layers, $W_{MP}$ and $H_{MP}$ are both set to 4. The results of our proposed method on CIFAR-10 are given in Table.5.

From the results we can see, our method outperforms PCANet in the majority of cases with or without map-pooling operation.

## 5. Conclusion

In this paper, we propose a novel compact feature representation method named CFR-ELM, which has a shallow network architecture. A detailed introduction is given to its framework, which contains critical units based on ELM, post-processing modules, and the classification using SVM. We have explained the reason why CFR-ELM can learn efficient and compact features by adopting ELM auto encoder and the max pooling operation. The experiments on four typical variant databases prove the effectiveness of our proposed method since CFR-ELM achieves impressive results by learning a better compact representation of the input data. Different overlap rates have been analyzed, and experiments with and without max pooling layers have been implemented and compared. The error rate of CFR-ELM is generally lower than PCANet.

Figure 8. Several sample images from CIFAR-10 dataset.

Table 5. Results of PCANet and CFR-ELM with and without max pooling on the CIFAR-10 dataset.

| Overlap Rate ($\epsilon$) | ER (%) | | ER + MP (%) | |
|---|---|---|---|---|
| | PCANet | CFR-ELM | PCANet | CFR-ELM |
| 0.0 | 27.606 | **26.697** | 26.130 | **24.447** |
| 0.1 | 27.720 | **25.865** | **24.698** | 24.954 |
| 0.2 | 27.710 | **26.290** | 25.909 | **24.324** |
| 0.3 | 27.447 | **26.121** | 26.094 | **25.267** |
| 0.4 | 26.440 | **25.951** | 25.260 | **23.760** |
| 0.5 | **26.194** | 26.689 | **24.520** | 25.157 |
| 0.6 | **26.110** | 26.376 | 24.930 | **24.117** |
| 0.7 | 26.550 | **26.136** | 26.058 | **24.854** |

## 6. Acknowledgements

This work is supported by EEE-Delta Joint Group Laboratory on Internet of Things under Project M4061567.045.

## References

[1] M. Blot, M. Cord, and N. Thome. Max-min convolutional neural networks for image classification. In *Image Process-*

*ing (ICIP), 2016 IEEE International Conference on*, pages 3678–3682. IEEE, 2016. 1

[2] M. Blum, J. T. Springenberg, J. Wülfing, and M. Riedmiller. A learned feature descriptor for object recognition in rgb-d data. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1298–1303. IEEE, 2012. 1

[3] T.-H. Chan, K. Jia, S. Gao, J. Lu, Z. Zeng, and Y. Ma. Pcanet: A simple deep learning baseline for image classification? *IEEE Transactions on Image Processing*, 24(12):5017–5032, 2015. 1, 2, 4, 7

[4] D. Ciregan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3642–3649. IEEE, 2012. 1

[5] A. Coates, H. Lee, and A. Y. Ng. An analysis of single-layer networks in unsupervised feature learning. *Journal of Machine Learning Research*, 1001(48109):2, 2011. 1

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005. 1

[7] H. Fan, Z. Cao, Y. Jiang, Q. Yin, and C. Doudou. Learning deep face representation. *arXiv preprint arXiv:1403.2802*, 2014. 1

[8] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *Journal of machine learning research*, 9(Aug):1871–1874, 2008. 3

[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 580–587, 2014. 1

[10] M. Guillaumin, J. Verbeek, and C. Schmid. Is that you? metric learning approaches for face identification. In *Computer Vision (ICCV), 2009 IEEE 12th international conference on*, pages 498–505. IEEE, 2009. 1

[11] Z. Guo, L. Zhang, and D. Zhang. A completed modeling of local binary pattern operator for texture classification. *IEEE Transactions on Image Processing*, 19(6):1657–1663, 2010. 1

[12] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik. Learning rich features from rgb-d images for object detection and segmentation. In *European Conference on Computer Vision*, pages 345–360. Springer, 2014. 1

[13] M. Hayat, M. Bennamoun, and S. An. Deep reconstruction models for image set classification. *IEEE transactions on pattern analysis and machine intelligence*, 37(4):713–727, 2015. 6

[14] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang. Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 42(2):513–529, 2012. 2

[15] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew. Extreme learning machine: theory and applications. *Neurocomputing*, 70(1), 2006. 2

[16] Y. Huang, Z. Wu, L. Wang, and T. Tan. Feature coding in image classification: A comprehensive study. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(3):493–506, 2014. 1

[17] L. L. C. Kasun, Y. Yang, G.-B. Huang, and Z. Zhang. Dimension reduction with extreme learning machine. *IEEE Transactions on Image Processing*, 25(8):3906–3918, 2016. 1, 2

[18] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. 2009. 4, 7

[19] H. Larochelle, D. Erhan, A. Courville, J. Bergstra, and Y. Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *Proceedings of the 24th international conference on Machine learning*, pages 473–480. ACM, 2007. 4

[20] B. Leibe and B. Schiele. Analyzing appearance and contour based methods for object categorization. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–409. IEEE, 2003. 4, 6

[21] F. Liu, G. Lin, and C. Shen. Crf learning with cnn features for image segmentation. *Pattern Recognition*, 48(10):2983–2992, 2015. 1

[22] T. Liu, Z. Yuan, J. Sun, J. Wang, N. Zheng, X. Tang, and H.-Y. Shum. Learning to detect a salient object. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(2):353–367, 2011. 1

[23] G.-B. H. Liyanaarachchi Lekamalage Chamara Kasun, Hongming Zhou and C. M. Vong. Representation learning with extreme learning machine for big data. *IEEE Intelligent Systems*, 28(6):31–34, 2013. 2

[24] R. Memisevic. Gradient-based learning of higher-order image features. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1591–1598. IEEE, 2011. 7

[25] S. Nayar, S. Nene, and H. Murase. Columbia object image library (coil 100). *Department of Comp. Science, Columbia University, Tech. Rep. CUCS-006-96*, 1996. 4

[26] S. A. Nene, S. K. Nayar, H. Murase, et al. Columbia object image library (coil-20). 1996. 4

[27] W. Ouyang, X. Wang, X. Zeng, S. Qiu, P. Luo, Y. Tian, H. Li, S. Yang, Z. Wang, C.-C. Loy, et al. Deepid-net: Deformable deep convolutional neural networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2403–2412, 2015. 1

[28] S. Shojaeilangari, W.-Y. Yau, K. Nandakumar, J. Li, and E. K. Teoh. Robust representation and recognition of facial emotions using extreme sparse learning. *IEEE Transactions on Image Processing*, 24(7):2140–2152, 2015. 1

[29] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1

[30] L. Wan, M. Zeiler, S. Zhang, Y. L. Cun, and R. Fergus. Regularization of neural networks using dropconnect. In *Proceedings of the 30th International Conference on Machine Learning*, pages 1058–1066, 2013. 5

[31] W. Zhu, J. Miao, L. Qing, and G.-B. Huang. Hierarchical extreme learning machine for unsupervised representation learning. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2015. 1, 2