Generic Object Recognition with Local Receptive Fields Based Extreme Learning Machine

Zuo Bai, Liyanaarachchi Lekamalage Chamara Kasun and Guang-Bin Huang

School of Electrical and Electronic Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 639798

Abstract

Generic object recognition is the classification of an individual object to a generic category. Intra-class variabilities, such as different objects of the same category, different poses and lighting conditions, cause big troubles for this task. Traditional methods involve plenty of pre-processing steps, such as shape model construction, extraction of hand-crafted features, etc. Moreover, these methods are only effective for some specific dataset. In this paper, we propose to use local receptive fields based extreme learning machine (ELM-LRF) as a general framework for object recognition. It is operated directly on the raw images without any pre-processing, and thus suitable for all different datasets. Additionally, the architecture is simple and only requires few computations, as most connection weights are randomly generated. We have compared the performance of ELM-LRF with state-of-the-art methods on several generic object recognition datasets, NORB, ETH-80 and COIL. It is on par with the best result on ETH-80 and sets the new records for NORB and COIL.

Keywords:
Generic object recognition, local receptive fields, Extreme Learning Machine (ELM)

1. Introduction

Generic object recognition is the task to classify an unknown object to a certain generic category [1, 2]. It remains as a challenging task due to its large amount of intra-class variabilities, such as different objects of one category, poses, lighting conditions, positions in the image, scales and camera settings. The existing methods are: 1) in [3], shape- and appearance-based methods are used and compared; 2) in [4], local features are extracted and subsequently classified with support vector machines (SVMs); 3) in [5], discriminant analysis is performed on the canonical correlations amongst pairs of images. However, these methods require plenty of human intervention, such as shape model construction [6], careful design of suitable features [2], bounding box detection of the object of interest [7], etc. Furthermore, these methods are only applicable for some specific tasks since local features or shape models that are used cannot be guaranteed to extend to new tasks [8, 9].

Convolutional neural network (CNN), which is operated on the raw pixels, is introduced in [10]. It presents state-of-the-art results on many different image processing tasks [11, 12]. Back-propagation (BP) algorithm [13] is adopted to adjust the connection weights in CNN. Thus, CNN faces trivial issues inherited
from BP algorithm, such as local minima, time consuming, etc. Furthermore, CNN requires huge computations and numerous training samples as vast connection weights need to be iteratively tuned.

Later, local receptive fields based extreme learning machine (ELM-LRF) is proposed in [14], which also handles the raw images directly. ELM-LRF generates the input weights randomly and calculates the output weights analytically. It provides a simple and deterministic solution, and thus solves the trivial issues associated with BP. Additionally, the requirement for computational capability and number of training samples are also largely reduced since most connection weights (the input weights) are simply generated randomly.

In this paper, we propose to use ELM-LRF as a general framework for generic object recognition. ELM-LRF has several advantages: 1) it does not use task specific information such as local features or global shapes for learning; 2) it is a simple learning algorithm; 3) it is computationally efficient as most connection weights are generated randomly.

In the experiments, we evaluate the performance of ELM-LRF on different generic object recognition datasets, NORB [10], ETH-80 [3], COIL [15]. It shows better results than current state-of-the-art for NORB, COIL and is comparable with the best result on ETH-80.

2. Reviews of related works

2.1. Generic object recognition

In this subsection, we undertake a close investigation into the methods on generic object recognition. Generic object recognition is to classify an individual object to a certain category and is also called object categorization [1]. It has been explored for several decades and various method are proposed [16].

1. Shape-based methods: Shape models are constructed explicitly, for subsequent recognition, while other attributes, like color or texture, are ignored [3, 6]. Additionally, parts-based shape models are used and achieves satisfactory performance in some applications [17, 18].

2. Appearance-based methods: Objects belonging to the same category may have variant appearance due to different poses, lighting conditions, multiple instances of one category, etc. Low-level information, such as texture and color histograms, may be useful. However, the images are highly-correlated, requiring PCA or other compression methods to generate compact representations. Subsequently, each object will be classified based on the similarity between itself and the model images [19, 20, 21].

3. Local feature-based methods: These methods usually detect some points of interest and then extract local features in the neighborhood of detected points. Different local features have been proposed, including HOG features [22], SIFT features [23], scale invariant descriptors (SIDs) [24], etc. Classifiers are followed to assign all images to respective categories.

2.2. Convolutional neural network (CNN)

CNN [25] is a variant of multilayer feedforward neural networks (or called multi-layer perceptrons) inspired from biology [26]. Different from aforementioned methods, CNN is operated directly on the raw pixels, eliminating the shape model construction or local features extraction. Additionally, CNN is more general than traditional methods, as the features are implicitly learned by the network itself instead of designed by users for some specific tasks. The common approach to train a CNN is the back-propagation (BP) algorithm [13], which involves gradient-descent learning steps.

In some recent variants of CNN, such as Deep CNN [11], GoogLeNet [12], it shows superior performance on super-large image datasets, such as ImageNet [27]. However, with so many parameters to be tuned, huge computational capability is required. In addition, the training set has to be large enough in order to train the network properly. Therefore, it comes to a question: is there any simple network that can handle raw images directly, while does not require learning with back-propagation algorithm?
3. Local receptive fields based extreme learning machine (ELM-LRF)

3.1. Fully connected ELM

ELM is a generalized single-hidden layer feedforward neural networks (SLFNs) with many different types of hidden nodes [28, 29, 30, 31]. Input and hidden nodes are in full connection. It theoretically proves that hidden nodes can be generated randomly, as long as the activation functions of the hidden nodes are nonlinear piecewise continuous [32]. Although ELM has some relationship with previous works such as QuickNet [33] and random vector functional link (RVFL) [34], there exists significant differences between them. The detailed relationship and differences can be found in [35].

Unlike traditional learning methods, such as BP algorithm, ELM does not require any iterative tunings. It presents better accuracy and high efficiency, in various applications such as system modelling, biomedical analysis, power systems, etc. [35].

Given a set of training data \((x_i, t_i), i = 1, \cdots, N, x_i \in \mathbb{R}^{1 \times d}, t_i \in \mathbb{R}^{1 \times m}\), state ELM implementation in matrix form:

\[
x_i \rightarrow h(x_i) = [h_1(x_i), \cdots, h_L(x_i)], \ i = 1, \cdots, N
\]

\[
H\beta = T
\]

where \(H\) and \(T\) are:

\[
H = \begin{bmatrix}
    h(x_1) \\
    \vdots \\
    h(x_N)
\end{bmatrix}_{N \times L}, \quad T = \begin{bmatrix}
    t_1 \\
    \vdots \\
    t_N
\end{bmatrix}_{N \times m}
\]

Various methods have been introduced to calculate \(\beta\), such as iterative methods [36, 37, 38], singular value decomposition (SVD) [39]. A popular and efficient closed-form solution is suggested in [31]:

\[
\beta = \begin{cases}
    H^T (\frac{1}{\epsilon} + HH^T)^{-1} T, & \text{if } N \leq L \\
    (\frac{1}{\epsilon} + H^T H)^{-1} H^T T, & \text{if } N > L
\end{cases}
\]

3.2. Locally connected ELM

When facing applications with strong local correlations, like image processing and speech recognition, local receptive fields based extreme learning machine (ELM-LRF) is proposed to handle the local structures. It was shown in [14] that different shapes of local receptive fields may be suited for different applications. For instance, McDonnell et al. utilize random sampling method to generate the receptive fields and produce...
superior accuracy on the MNIST, NORB and SVHN datasets [40, 41]. Subsequently, combinatorial node can be formulated to generate even more abstract representations of the raw inputs by combining several sub-nodes together, as shown in Fig. 1.

3.3. One feasible network of ELM-LRF

ELM-LRF is a two-stage network: (1) **tuning-free** nodes; (2) least-squares solution \( \beta \). Although many types of local receptive fields and combinatorial nodes are applicable, for simplicity, we use convolution operation and square/square-root pooling to construct one feasible network of ELM-LRF as in Fig. 2. A car image of ETH-80 dataset is chosen as an example, where the input layer includes 3 RGB maps. And there are \( K \) maps in the feature and pooling layers to obtain comprehensive representations for the raw input.

3.3.1. Tuning-free nodes

The nodes in the pooling layer are tuning-free, connecting with the input layer by random convolutional weights and pooling structure.

i Random convolutional weights: The convolutional weights between input and feature layer are random. Assume that the input image is \( d \times d \) and the local receptive field is \( r \times r \). Thus, the feature map is \( (d - r + 1) \times (d - r + 1) \): 1) randomly generate \( \hat{A}^{\text{init}} \in \mathbb{R}^{r \times K} \) based on standard Gaussian distribution; 2) orthogonalize into \( \hat{A} \in \mathbb{R}^{r \times K} \) with SVD method; 3) reshape the columns of \( \hat{A}, \hat{a}_k \), into \( a_k \in \mathbb{R}^{r \times r} \), \( k = 1, \cdots, K \). Thus, node \((i, j)\) in the \( k\)-th feature map, \( c_{i,j,k}(x) \) is calculated as:

\[
c_{i,j,k}(x) = \sum_{m=1}^{r} \sum_{n=1}^{r} x_{i+m-1,j+n-1} \cdot a_{m,n,k}, \quad i, j = 1, \cdots, (d - r + 1) \quad (4)
\]

ii Square/square-root pooling: As shown in Fig. 2, nodes in the feature layer are grouped within each pooling area, formulating subsequent pooling layer. Thus, node \((p, q)\) in the \( k\)-th pooling map, \( h_{p,q,k} \) is:

\[
h_{p,q,k} = \sqrt{\sum_{i=p-e}^{p+e} \sum_{j=q-e}^{q+e} c_{i,j,k}^2}, \quad p, q = 1, \cdots, (d - r + 1) \quad c_{i,j,k} = 0 \text{ if } (i, j) \text{ out of bound} \quad (5)
\]

3.3.2. Regularized least-squares solution

The output weight \( \beta \) is the only weight vector to be calculated. All nodes in the pooling layer are obtained by calculating (4) and (5) sequentially. Concatenating all nodes in the pooling layer into a row vector and putting all rows of \( N \) training samples together, the matrix \( H \in \mathbb{R}^{N \times (d-r+1)^2} \) is generated:
Table 1. Datasets descriptions

<table>
<thead>
<tr>
<th>Dataset</th>
<th># of categories</th>
<th># of training data</th>
<th># of testing data</th>
<th># of input channels</th>
</tr>
</thead>
<tbody>
<tr>
<td>NORB</td>
<td>5</td>
<td>24300</td>
<td>24300</td>
<td>2</td>
</tr>
<tr>
<td>ETH-80</td>
<td>8</td>
<td>1640</td>
<td>1640</td>
<td>3</td>
</tr>
<tr>
<td>COIL</td>
<td>100</td>
<td>1800</td>
<td>5400</td>
<td>3</td>
</tr>
</tbody>
</table>

\[
\beta = \begin{cases} 
  H^T \left( \frac{1}{c} + HH^T \right)^{-1} T, & \text{if } N \leq K \cdot (d - r + 1)^2 \\
  \left( \frac{1}{c} + H^T H \right)^{-1} H^T T, & \text{if } N > K \cdot (d - r + 1)^2 
\end{cases}
\]  

It should be noted that ELM-LRF is different from conventional CNNs: 1) ELM-LRF is more flexible as it could use different types of local receptive fields given that they are randomly generated according to any continuous probability distribution; 2) hidden nodes in ELM-LRF are tuning-free and the output weight vector \( \beta \) is calculated analytically. Thus, ELM-LRF provides an efficient and deterministic solution. Additionally, it requires further investigation for more feasible networks of ELM-LRF with different types of local receptive fields. It should also be noted that ELM-LRF is different from [42] in the sense that conventional neurons can be naturally used in ELM while keeping the rest of ELM architectures and solutions unchanged.

4. Experiments

In this section, we conduct thorough investigations of ELM-LRF on different generic object recognition tasks, NORB [10], ETH-80 [3] and COIL [15]. All experiments are conducted in MATLAB 2013a, running on a Windows Server 2012, with Intel Xeon E5-2650, 2GHz CPU, 256G RAM.

4.1. Datasets descriptions

These generic object recognition datasets are subject to different variations: poses, lighting conditions, scales, positions in the image and camera settings (white balance, contrast). Raw images are resized into 32×32 for all datasets and used directly without any preprocessing. NORB contains stereo images, thus 2 channels. Other datasets contain RGB images, thus 3 channels.

NORB includes 48600 pairs of binocular images of 50 objects from 5 generic categories under different angles, lightings and azimuths. Half is used for training and the other half for testing based on the standard partition provided in [10]. ETH-80 [3] contains 8 generic categories, each of which has 10 objects, under 41 viewing angles. In each category, 10 objects are randomly split into training and testing sets (5 objects respectively) as done in [5]. COIL [15] includes 100 objects under 72 rotated views (5° increment). The testing set includes images every 20° rotation (0°, 20°, . . . ) and thus is consisted of 18 views. The training set includes the remaining images. We reserve a hold-out validation set for each problem, consisting of 20% samples of the training set.

4.2. Influence of the number of feature maps \( K \)

In essence, the purpose of multiple feature maps is to obtain thorough representations for the raw images. In general, the more feature maps, the more exhaustive representations. However, after the number passes a certain threshold, more feature maps may easily lead to overfitting and hurt the performance.

At here, we fix other parameters and vary the number of feature maps \( K \) from 10 to 100, to examine the influence of \( K \). Other parameters are fixed: receptive field 4 × 4, pooling size 5 and \( C = 0.01 \). As observed from Fig. 3(a), the validation accuracy generally increases with more feature maps, indicating that the threshold has not yet been reached. Additionally, more feature maps require more training time as illustrated in Fig. 3(b). Thus, a compromise between better accuracy and fewer feature maps should be made. In this paper, we choose to fix \( K = 50 \), though not optimal, for all problems to reduce computations.
4.3. Parameter selection

After fixing $K = 50$, parameters to be chosen are: 1) size of receptive field; 2) pooling size; 3) value of $C$. And the optimal parameters will be selected by grid search based on the validation accuracy. Receptive field is tried with 5 values: $3 \times 3$, $4 \times 4$, $5 \times 5$, $6 \times 6$ and $7 \times 7$. Pooling size is also tried with 5 values: $3$, $4$, $5$, $6$ and $7$. And $C$ is tried with 3 values: 0.01, 1 and 100. Table 2 specifies the parameters for all these datasets.

4.4. Performance on NORB

The test error rates and training time of different methods on the NORB dataset are compared in Table 3. ELM-LRF achieves the best accuracy with much faster training speed, up to 200 times compared with deep belief network (DBN) [43] and CNN [10].

4.5. Performance on ETH-80

The results of ELM-LRF and some leading methods are listed in Table 4. The test error rate of ELM-LRF is comparable with state-of-the-art result achieved by DCC [5] method. And ELM-LRF is also exceptionally efficient that it only requires 48.64 seconds for training and 15.35 seconds for testing.

4.6. Performance on COIL

As seen from Table 5, ELM-LRF also sets the new record for COIL dataset. Additionally, there are some works using CNN methods to handle the COIL dataset [48]. It can be observed that ELM-LRF is
Table 3. Test error rates and training time on the NORB dataset

<table>
<thead>
<tr>
<th>Methods</th>
<th>Test error rates</th>
<th>Training time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ELM-LRF</td>
<td>2.76%</td>
<td>400.78</td>
</tr>
<tr>
<td>Random weights [42]</td>
<td>4.8%</td>
<td>1764.28</td>
</tr>
<tr>
<td>K-means + soft activation [44]</td>
<td>2.8%</td>
<td>6920.47</td>
</tr>
<tr>
<td>Tiled CNN [45]</td>
<td>3.9%</td>
<td>15104.55</td>
</tr>
<tr>
<td>CNN [10]</td>
<td>6.6%</td>
<td>53378.16</td>
</tr>
<tr>
<td>DBN [43]</td>
<td>6.5%</td>
<td>85717.14</td>
</tr>
</tbody>
</table>

Table 4. Test error rates on the ETH-80 dataset

<table>
<thead>
<tr>
<th>Methods</th>
<th>Test error rates</th>
</tr>
</thead>
<tbody>
<tr>
<td>ELM-LRF</td>
<td>10.0%</td>
</tr>
<tr>
<td>Discriminant Analysis of Canonical Correlations (DCC) [5]</td>
<td>8.3%</td>
</tr>
<tr>
<td>Orthogonal Subspace Method (OSM) [5]</td>
<td>9.5%</td>
</tr>
<tr>
<td>Constrained Mutual Subspace Method (CMSM) [46]</td>
<td>10.3%</td>
</tr>
<tr>
<td>kNN-LDA [47]</td>
<td>24.8%</td>
</tr>
<tr>
<td>kNN-PCA</td>
<td>23.8%</td>
</tr>
</tbody>
</table>

Quite advantageous over CNN when dealing with COIL. Even if CNN uses unlabeled test images or COIL-like images as additional information for further training and achieves significant improvements, ELM-LRF still outperforms CNN by a big gap for this task. The authors believe that it is caused by the relatively too few training samples. In CNN, numerous parameters need to be tuned. And when there are not enough training samples, the parameters (connection weights) cannot be well trained, which degenerates the generalization capability of the network.

4.7. High efficiency of ELM-LRF

Let us inspect the efficiency of ELM-LRF as a general framework. The training and testing time are summarized in Table 6. As easily observed from the table, ELM-LRF is highly efficient that it requires less than 0.03 seconds per image for training and less than 0.01 seconds per image for testing. In addition, ELM-LRF can be easily extended to real-time applications, since it is able to test more than 100 images per second after properly trained.

Table 5. Test error rates on the COIL dataset

<table>
<thead>
<tr>
<th>Methods</th>
<th>Test error rates</th>
</tr>
</thead>
<tbody>
<tr>
<td>ELM-LRF</td>
<td>0.02%</td>
</tr>
<tr>
<td>Local Affine Frames (LAFs) [49]</td>
<td>0.1%</td>
</tr>
<tr>
<td>Linear SVM [50]</td>
<td>8.7%</td>
</tr>
<tr>
<td>Spin-Glass Markov Random Field (MRF) [6]</td>
<td>3.2%</td>
</tr>
<tr>
<td>Standard CNN [48]</td>
<td>28.51%</td>
</tr>
<tr>
<td>CNN+video (test images of COIL) [48]</td>
<td>7.75%</td>
</tr>
<tr>
<td>CNN+video (COIL-like images) [48]</td>
<td>20.23%</td>
</tr>
</tbody>
</table>

1 The current state-of-the-art result.
2 Use the unlabeled test images as additional learning source. It is a semi-supervised method together with the labeled training images.
3 Use COIL-like images as additional learning source.
Table 6. Training and testing time (seconds) on different datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Training stage</th>
<th>Testing stage</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total training time</td>
<td>Per image</td>
</tr>
<tr>
<td>NORB</td>
<td>400.78</td>
<td>0.0165</td>
</tr>
<tr>
<td>ETH-80</td>
<td>48.64</td>
<td>0.0297</td>
</tr>
<tr>
<td>COIL</td>
<td>33.23</td>
<td>0.0185</td>
</tr>
</tbody>
</table>

5. Conclusions

In this paper, we propose to use ELM-LRF as a general framework for generic object recognition. Distinct merits exist for ELM-LRF compared with traditional methods: 1) task non-specific for not utilizing any task-specific information; 2) simple to use that it requires no pre-processing steps, such as design of suitable features, shape model construction or anything else; 3) highly efficient as only a small portion of connection weights need to be calculated. Additionally, unlike the newly-emerging CNN, where connection weights are iteratively tuned, most weights in ELM-LRF are simply generated randomly and only the vector of output weights is calculated deterministically. Comparing to CNN, it significantly reduces: 1) computational complexity; 2) requirement for huge training set. In the experiments, the general framework of ELM-LRF is evaluated on several generic object recognition tasks. And it presents superior accuracy with exceptionally high speed.

References
