# Invariant Object Recognition using Circular Pairwise Convolutional Networks

Choon Hui Teo[1] and Yong Haur Tay[2]

[1] National ICT Australia Ltd. and
Research School of Information Sciences and Engineering
Australian National University
choonhui.teo@rsise.anu.edu.au
[2] Faculty of Information and Communication Technology
Universiti Tunku Abdul Rahman
tayyh@mail.utar.edu.my

**Abstract.** Invariant object recognition has been one of the most rewarding are of research in computer vision as there are many applications need the capability of recognizing objects of interest in various environments. However, there is no single technique which claims to achieve the goal in all possible conditions and domains. Out of many techniques, convolutional network has proved to be a good candidate in this area. Given large numbers of training samples of objects under various variation aspects such as lighting, pose, background, etc., convolutional network can learn to extract invariant features by itself. This comes with the price of lengthy training time. Hence, we propose a circular pairwise classification technique to shorten the training time. We compared the recognition accuracy and training time complexity between our approach and a benchmark generic object recognizer LeNet7 which is a monolithic convolutional network.

## 1 Introduction

*Invariant Object Recognition* is an object recognition approach which deals with recognizing an interested object in an image under some variation aspects. Possible variation aspects are translation, scaling, shearing, rotation, distortion, illuminations, angle of view (pose), occlusion, noisy (jittered and/or cluttered) background, changes in object colors and texture and etc.

An object can be defined as a set of geometrical, structural or pictorial features (or characteristics) which exhibit large inter-class variance while keeping intra-class variance small at the same time. Such features could be geometrical shape, number of interconnected cylinders with certain relationship, color histogram, texture, height-width ratio, etc.

In invariant object recognition, the first challenge would be defining a set of detectable invariant features for each class of interest such that objects can be recognized under many variation aspects. In order to devise such ideal object representation, many research rooted in statistics (e.g. K-Nearest Neighbor),

psychology (e.g. Recognition By Component) and biology (e.g. neural networks) have been done [1]. The end products of those researches are some pattern recognition techniques which either require large number of training sample or explicitly defined geometrical and structural descriptions of objects for accurate recognition.

Basically, there are two issues associated with invariant object recognition on features level. First, the system must be able to detect the object features under various variation aspects. Besides, the system must be robust to the noise in input so that a small faction of wrongly identified features does not affect the recognition accuracy much. Second, features defined for objects in one domain (e.g. digit) cannot be used directly for other objects in other domains (e.g. human face). So, human expert is needed to define features and handcraft the corresponding feature extractor for different problem domain every time.

With convolutional networks [2] (i.e., a type of neural network designed for two dimensional image inputs), the abovementioned issues can be tackled. First, neural network approach is known to be robust to noise and second, convolutional network facilitates its own feature extraction mechanism without the need of human intervention. However, the shortcoming of convolutional network is computation-intensive which, in turn, causes a lengthy training time.

In this paper, we investigate on the capability of convolutional network in shape-based object recognition invariance to lighting, pose, and background. Also, we propose a novel approach, Circular Pairwise Classification (CPC), to reduce the convolutional networks training time for that problem. By using NORB (NYU Object Recognition Benchmark) [3] as benchmark dataset, we compared the network training time and recognition accuracy between proposed Circular Pairwise Convolutional Network (CPCN) and LeNet7 developed by LeCun et al. at New York University [3].

## 2 Convolutional Networks

Convolutional Network is a kind of multi-layered neural network which facilitates the feature extraction and input-output mapping together with a global learning algorithm. The built-in trainable feature extractor of convolutional networks makes it a good candidate for end-to-end object recognition problem. In addition, the trainable feature extractor is adaptable to different problem domain. In a recognition problem from raw input (e.g. image pixels), convolutional networks usually perform better than the Multi-Layered Perceptron (MLP) because the former takes the topology of inputs into consideration while the latter does not. Furthermore, convolutional networks combine three architectural ideas, namely local receptive fields, shared weights, and spatial sub-sampling, which ensure some degree of invariance to shift, scale and distortion.

*Local receptive field* is a small two dimensional neighborhood on the input image. Each neighborhood on the input is connected by a unit on a feature map (i.e., a two dimensional plane of units) using a vector of weights that is the same size as its neighborhood. The feature maps which store elementary visual

features (e.g. edges, corners) extracted from those small neighborhoods in the input image will then be the input of the feature maps in the next layer etc., to compose more complex features - a process known as spatial convolution. Features are extracted locally from the input image because spatially nearby inputs are highly correlated [2], and these nearby input variables are normally elementary features of an object such as edges and corners which differentiate one object from another. For example, a circle has no corners but a rectangle has four. Due to the fact that the exact locations of those features are less important compared to their relative positions to each others, any slight distortion in the input image would then be further alleviated when it is transformed into feature maps.

*Weight sharing* is a technique of using a set of weight vector for each unit in a feature map to extract similar features across all possible local receptive fields on the input image[2]. Therefore, we could have many feature maps to extract many different features from the input image. Moreover, weight sharing also keeps the complexity of the convolutional network small; hence the problem of overfitting can be reduced.

In a complete convolutional network, each convolutional layer is followed by a *sub-sampling* layer to reduce the feature map resolution because the feature map outputs are sensitive to the translation in the input image [2]. By decreasing the feature map resolution, the amount of translation as well as the variance with respect to the translation would be reduced.

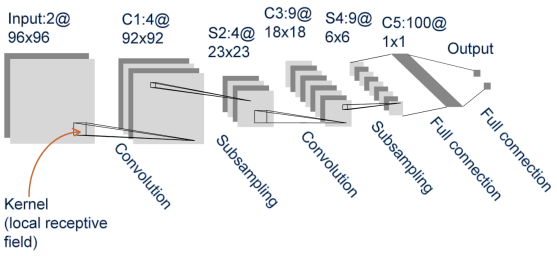



**Fig. 1.** Convolutional network for invariant object recognition

Figure 1 shows a typical convolutional network for invariant object recognition. The network consists of a series of alternating convolutional layers and sub-sampling layers. These layers are the core of the automatic feature extraction mechanism which extracts elementary visual features in lower layers and combine them at subsequent layers. The following are layers of perceptron units which act exactly the same as the Multi-Layered Perceptron (MLP), mapping a vector of features extracted by previous convolutional/sub-sampling layers to its desired output. The two major operations of convolutional networks that realize the three architectural ideas mentioned above are spatial convolution and spatial sub-sampling.

### 2.1 Spatial Convolution

In the convolutional network, the spatial convolution process works as such: A trainable two dimensional kernel (or weights) is overlaid on a small neighborhood or *local receptive field* (same size with the kernel) on the top right hand corner of an input image, and the pixels and their corresponding kernel cell values are multiplied. The summation of those products together with a trainable bias term is then passed on to a nonlinear activation function such as the hyperbolic tangent to get a feature value for that particular neighborhood with respect to the kernel. This continues by shifting the kernel 1 pixel to the left for each convolution until the horizontal end of the image, then, repeats the same process again from the start of second row and so on until the kernel covers the bottom left pixel. Figure 2(a) shows an input image of size 8x8 pixels is subjected to convolution by three feature maps in the first convolutional layer, each with a kernel of size 4x4 pixels. Three 5x5 feature maps in the first convolutional layer then form the input for the six 2x2 feature maps in the second convolutional layer. The output layer is fully connected to all feature maps in the second convolutional layer. Note that each feature map uses a different kernel or weights vector for different inputs (or preceding feature maps).

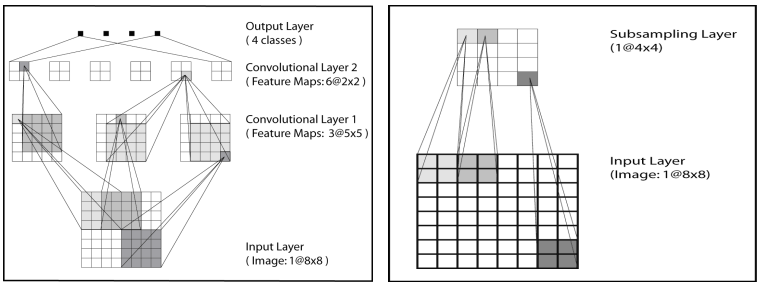
### 2.2 Spatial Sub-sampling

Spatial Sub-sampling is a technique of reducing the input or feature map resolution. Since feature maps are sensitive to translation in the input; down-sampling the resolution will help reduce the precision of the translation effect. Spatial sub-sampling is done by averaging a small neighborhood of the image window and then multiplying the average values with a trainable weight. The product is then passed on to a nonlinear activation function together with a trainable bias. The output values generated from the neighborhoods of the image window are organized in the same order and position as in the input. In figure 2(b), an 8x8 input image is down-sampled into a 4x4 image (or feature map) using a trainable weight and a bias term.

## 3 Circular Pairwise Classification

We proposed a novel pairwise classification framework, *Circular Pairwise Classification*, in which a $k$-class classification problem is decomposed into $k$ two-class classification sub-problems. Each of the classes lies in exactly two different sub-problems, with each sub-problem being handled by a pairwise classifier. One can imagine that the $k$ classes are arranged in a circle whereby each class is only paired with its adjacent left and right neighboring classes. This is illustrated in figure 3.

Since each pairwise classifier is trained on an adjacent pair of classes, there is no direct competition between two non-adjacent classes. Consequently, if an unknown input, $\mathbf{x}$, is given, we cannot justify that $\mathbf{x}$ belongs to one class but




(a) (b)

**Fig. 2.** Two major operations in convolutional networks. 2(a) and 2(b) are the examples of spatial convolution and spatial sub-sampling, respectively

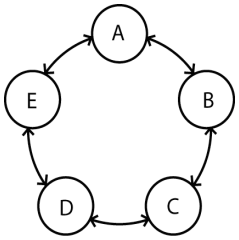



**Fig. 3.** Class pairing in Circular Pairwise Classification. Each small circle here represents a class. This 5-class classification example is decomposed into 5 sub-problems represented by 5 ordered pairs, namely (A, B), (B, C), (C, D), (D, E), and (E, A)

not others. Moreover, due to the fact that a pairwise classifier can give erroneous output if $\mathbf{x}$ does not belong to the pair of classes the pairwise classifier is trained on, chances are high that more than one class will get the maximum votes (i.e., two), especially when the number of classes is more than three (i.e. $k > 3$). Obviously, this conflict can not be solved because there is no direct competition between the two conflicting classes (i.e. two adjacent classes will never get maximum votes at the same time). From the above mentioned difficulties, we believe that a naïve configuration of CPC would not work well because of the lack of 'knowledge' between the non-adjacent classes. Hence, we propose that an estimation technique should be used to compute these missing 'knowledge' from the $k$-classifiers' probabilistic output values.

Given a pairwise classifier (say, MLP), $\mathbf{C}_{\mathrm{ij}}$, which is trained on the class pair $(i, j)$ to produce two probabilistic output, $\mathbf{r}_{\mathrm{ij}}$ and $\mathbf{r}_{\mathrm{ji}}$ $(= 1 - \mathbf{r}_{\mathrm{ij}})$ where $\mathbf{r}_{\mathrm{ij}}$ and $\mathbf{r}_{\mathrm{ji}}$ are the ratios of probability densities [4], the probability of an unknown input, $\mathbf{x}$, to belong to class $i$ and class $j$ may be computed as such, viz.

$$\mathbf{r}_{\mathrm{ij}} = \mathbf{P}(\mathrm{i}|\,\mathbf{x}) \;/\; (\mathbf{P}(\mathrm{i}|\,\mathbf{x}) + \mathbf{P}(\mathrm{j}|\,\mathbf{x})) \tag{1}$$

$$\mathbf{r}_{\mathrm{ji}} = \mathbf{P}(\mathrm{j}|\,\mathbf{x}) \;/\; (\mathbf{P}(\mathrm{i}|\,\mathbf{x}) + \mathbf{P}(\mathrm{j}|\,\mathbf{x})) \tag{2}$$

This is based on Cutzu's *vote-against* scheme [4]. With another pairwise classifier, $\mathbf{C}_{\mathrm{jk}}$, and its probabilistic output, $\mathbf{r}_{\mathrm{jk}}$ and $\mathbf{r}_{\mathrm{kj}}$, we can also estimate $\mathbf{r}_{\mathrm{ik}}$ and $\mathbf{r}_{\mathrm{ki}}$ even though none of the pairwise classifiers are trained with a class pair

$(i, k)$. The calculation of $\mathbf{r}_{ik}$ and $\mathbf{r}_{ki}$ are shown follow:

$$\mathbf{r}_{ik} \approx \mathbf{r}_{ij} \cdot \mathbf{r}_{jk} \,/(\mathbf{r}_{ij} \cdot \mathbf{r}_{jk} + \mathbf{r}_{kj} \cdot \mathbf{r}_{ji}) \tag{3}$$

$$\mathbf{r}_{ki} \approx \mathbf{r}_{kj} \cdot \mathbf{r}_{ji} \,/(\mathbf{r}_{ij} \cdot \mathbf{r}_{jk} + \mathbf{r}_{kj} \cdot \mathbf{r}_{ji}) \tag{4}$$

Similarly, we can also provide an estimate for $\mathbf{r}_{im}$ and $\mathbf{r}_{mi}$ if $\mathbf{r}_{km}$ and $\mathbf{r}_{mk}$ are given as such,

$$\mathbf{r}_{im} \approx \mathbf{r}_{ik} \cdot \mathbf{r}_{km} \,/(\mathbf{r}_{ik} \cdot \mathbf{r}_{km} + \mathbf{r}_{mk} \cdot \mathbf{r}_{ki}) \tag{5}$$

$$\mathbf{r}_{mi} \approx \mathbf{r}_{mk} \cdot \mathbf{r}_{ki} \,/(\mathbf{r}_{ik} \cdot \mathbf{r}_{km} + \mathbf{r}_{mk} \cdot \mathbf{r}_{ki}) \tag{6}$$

The similar estimation steps are applied for all possible pair of classes. In the estimations shown above, we normalized those pair of ratios so that their sum is equal to one; otherwise, the ratios will be very small if the estimation step is lengthy. As the classes are arranged in a circle, we may estimate the ratios in a clockwise or anti-clockwise direction and produce a full set of pairwise ratios. To combine these pairwise ratios into a final decision for the multi-class problem, a meta-classifier such as MLP can be trained to map them into their corresponding desired outputs.

## 4 Experimental Setup

Since this paper is centered at invariant object recognition, we use NORB jittered-cluttered dataset [3] as the benchmark. The original dataset is subdivided into six sub-datasets in the manner which suit CPC. The pairing of classes for pairwise convolutional networks is shown in table 1.

Each pairwise network was trained on a sub-dataset assigned to them for a maximum of 35 epochs with no specific stopping criteria applied. The parameters such as learning rate, momentum coefficient and network complexity are kept fixed for each network. Input images for the network were transformed so that pixel brightness intensity (i.e., 0 to 255) is converted into continuous value between -1 and 1. The following subsections describe NORB dataset [5] and the architecture of pairwise convolutional network in detail.

| **CPCN** | **Sub-dataset (Pairs of classes)** |
|---|---|
| $\mathbf{N}_{A,H}$ | 'Animal' & 'Human' |
| $\mathbf{N}_{H,P}$ | 'Human' & 'Plane' |
| $\mathbf{N}_{P,T}$ | 'Plane' & 'Truck' |
| $\mathbf{N}_{T,C}$ | 'Truck' & 'Car' |
| $\mathbf{N}_{C,J}$ | 'Car' & 'Junk' |
| $\mathbf{N}_{J,A}$ | 'Junk' & 'Animal' |

**Table 1.** Six circular pairwise convolutional networks trained on six sub-datasets of original NORB jittered-cluttered dataset

**Fig. 4.** Samples (left camera image) of NORB jittered-clutter dataset. From the leftmost column to the rightmost are samples of classes **Animal**, **Human**, **Plane**, **Truck**, **Car**, and **Junk**

### 4.1 NORB dataset

The NORB jittered-cluttered dataset consists of stereo image pairs (captured by two cameras with 7.5cm in between) of 50 uniform-colored toys under thirty-six angles, nine different azimuths, and six lighting conditions. These fifty toys may be classified into 5 generic categories, namely four-legged animals, human figures, airplanes, trucks and cars. Each category has ten different instances - five for training and the remaining five for testing. The toys are painted with a uniform color to keep the object texture variance fixed. Since the object images are gray-scale, the object color will not provide any useful information. There are a total of 291,600 training examples and 58,320 testing examples from 6 classes (5 object classes and an junk or background class) (see figure 4). The background class is used for detecting false-positives when a system is trained for detection/segmentation/recognition task. The images were subjected to random perturbation (translation, scaling, rotation, changes in brightness and contrast), cluttered background, and surrounding distracter objects [3].

### 4.2 Pairwise Convolutional Network Architecture

Each of the pairwise convolutional networks reported in this paper consist of six layers, namely, C1, S2, C3, S4, C5, and Output. The letter C indicates a convolutional layer while letter S a sub-sampling layer. For ease of presentation, the input image is shown as a layer named Input. The Input layer holds two stereo (left and right camera) object images of size 96x96 pixels each .The C1 layer has four feature maps and uses six 5x5 convolution kernels. The first two feature maps in C1 take their inputs from the left and right images of the layer Input respectively. The other two feature maps take their inputs from both images (see table 2(a)). S2 is a 4x4 sub-sampling layer which takes its inputs from C1. C3 has nine feature maps that use twenty-four 6x6 convolution kernels. Each C3 feature map takes a different combination of monocular inputs and binocular inputs. These connections are shown in table 2(b). S4 is a 3x3 sub-sampling layer. C5 consists of one hundred feature maps that combine all the inputs in S4 through 6x6 kernels. The output layer has two units and is fully connected to C5.

The networks are trained with stochastic backpropagation algorithm with Cross-Entropy criterion to give probabilistic outputs or ratios. All CPCNs were trained in parallel on computers of similar specification for a maximum of 35 iterations. The list of CPCNs trained with different pairs of classes is shown in

| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| **0** | X | | X | X |
| **1** | | X | X | X |

(a)

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| **0** | X | X | X | | | | X | X | X |
| **1** | | | | X | X | X | X | X | X |
| **2** | X | | X | X | | X | X | | X |
| **3** | | X | X | | X | X | | X | X |

(b)

**Table 2.** Connection tables between 2(a) layers Input (rows) and C1 (columns), and 2(b) layers S2 (rows) and C3 (columns) of PCNs trained on NORB jittered-cluttered sub-datasets

table 1. Table 3 lists the specification of a pairwise convolutional network including the number of feature maps, feature map dimension, kernel size, number of trainable weights and connections.

| **Layer** | **Number of feature maps/units** | **Dimension** | **Kernel size** | **Number of weights** | **Number of connections** |
|---|---|---|---|---|---|
| **Input** | 2 (binocular images) | 96x96 | - | - | - |
| **C1** | 4 | 92x92 | 5x5 | 154 | 1320384 |
| **S2** | 4 | 23x23 | 4x4 | 8 | 35970 |
| **C3** | 9 | 18x18 | 6x6 | 873 | 287712 |
| **S4** | 9 | 6x6 | 3x3 | 18 | 3240 |
| **C5** | 100 | 1x1 | 6x6 | 32500 | 33300 |
| **Output** | 2 | 1 | - | 202 | 202 |
| | | | **Total** | **33,760** | **1,680,808** |

**Table 3.** Specifications of PCN trained on NORB jittered-cluttered sub-dataset

For LeNet7, in order to model the six categories (i.e., five object classes and one junk/background image class) well, a certain number of feature maps and trainable weights are necessary. Logically, we set the capacity of CPCNs reported in this paper to be smaller than that of LeNet7 because each CPCN handles only a two-class classification problem whereas LeNet7 handles a six-class problem. Hence, the CPCNs used here are smaller in terms of the number of feature maps as well as the number of trainable weights. An additional benefit from this new architecture is that there would now be a lesser number of connections that connect the layers in CPCN together. Effectively this means that the amount of computation required is now smaller.

After all the CPCNs were trained, a meta-dataset was constructed based on the original one in two steps. Firstly, all the CPCNs, regardless of which pair of classes they were trained on, are fed with all training and testing samples from all the object classes. Next, the pairwise ratios produced by all CPCNs were then

used to estimate the rest of the ratios that were not trained to be produced by those CPCNs. For each input image, there will be twelve ratios produced by six CPCNs and another eighteen ratios were estimated using the technique described in previous section. Hence, each original sample from both training and testing set was represented by these thirty values in the meta-dataset. A MLP would be trained on this meta-dataset to map the ratios to its desired class label. In this paper, we have used a configuration of a 2-layer MLP where there would be thirty input units, two hundred hidden units, and six output units. In short, the whole recognition process would start by feeding every CPCN an image of the object of interest. Next an estimate the remaining ratios is computed, and finally, the ratios are fed into the meta-classifier to get the end results. See figure 5 for an example of CPCNs for a 5-class classification problem.

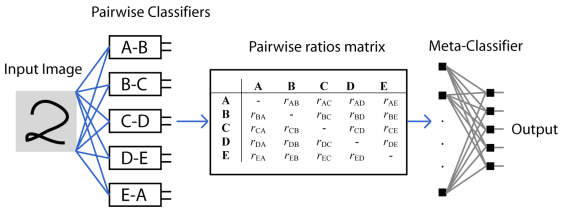



**Fig. 5.** Example of a circular pairwise convolutional network for a 5-class classification

## 5 Results and Discussion

Before the discussion of recognition accuracy and training time complexity of CPCNs, there is a need to investigate the capability of convolutional network in imaged-based end-to-end (i.e., from image pixels to class labels) invariant object recognition. The internal states of a CPCN when dealing with an input image is shown in figure 6. It can be observed on figure 6 that the feature maps are the blurred version of input images. This actually eliminates the irrelevant variability of the inputs. From the point of view of digital image processing, this change is similar to edge detection. The detected edges of objects (i.e., shape) are actually the key features in invariant object recognition. The subsequent resolution reduction steps further reduce the translational variance in the feature maps. A series of convolution and sub-sampling derive a set of high level features. These features are then used to train a classifier to associate input images to desired class labels.

### 5.1 Recognition accuracy

The results of the experiments are shown in table 4. We can see that the test error rates of CPCNs for the pairs (Car, Junk) and (Plane, Truck) are significantly lower than the rest. The reason is that, images related to animal, human, plane, and junk are generally more complex and "branchy" whereas trucks and cars are not. These two categories of classes share less common visual features and

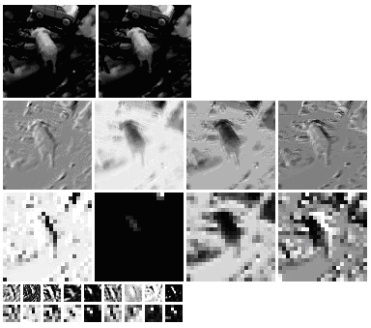


**Fig. 6.** Internal state of a CPCN trained on the pair (Animal, Human) for a pair of stereo images from the NORB jittered-cluttered dataset. The two images in the first row are the input stereo image pair. Second row consists of the feature maps in layer C1. Rows following by are feature maps in layers S2, C3 and S4, respectively

hence easier to be separated from each other. The test error rate for the network trained on the pair (Truck, Car) is the worst because trucks and cars share many common and important features such as outline. The overall test error rate of CPCNs when combined with a meta-classifier as shown here is higher than the monolithic convolutional network, LeNet7. This poorer recognition accuracy may be due to insufficient training applied to those CPCNs. This reasoning is supported by the fact that LeNet7 actually achieved a lower test error rate of 7.8% [5] (from 16.7% [3]) after it was trained for more iteration and with a set of slightly different learning parameters.

### 5.2 Training time

Since different learning algorithms were applied in training both LeNet7 and CPCNs, comparing their training time in terms of time units may not be fair. Instead, we have adopted the number of training samples needed to train each network and the number of multiply-add computations per full propagation through the network as the performance metric. Table 5 shows the comparison between LeNet7 and CPCNs based on these two criteria. As the number of samples per class is the same (i.e., 48600) for all classes [3], it may be clear that the CPCNs only need to be trained with one-third the numbers of training samples as that of LeNet7. In addition, CPCNs takes only about 36% of the total numbers of multiply-add computations in LeNet7 (i.e., 1.68 million : 4.66 million) in a full propagation. Approximately, a CPCN is at least 8 times faster than LeNet7 for a fixed number of training iterations. Generally speaking, the two-class problem is easier to learn than a six-class problem. So, CPCNs can achieve convergence more than 8 times faster than LeNet7 using the same learning algorithm. In general, as the number of classes in a classification problem grows, CPCNs will maintain the constant training time unlike a monolithic convolutional network like LeNet7.

## 6 Conclusion and Future works

We have presented how convolutional network is used as an invariant object recognizer. The lengthy training time for a monolithic convolutional network can

| Pairs of classes | Test error rate (%) | Ref. |
|---|---|---|
| (Car, Junk) | 5.56 | - |
| (Plane, Truck) | 6.96 | - |
| (Junk, Animal) | 10.61 | - |
| (Human, Plane) | 10.85 | - |
| (Animal, Human) | 13.77 | - |
| (Truck, Car) | 15.92 | - |
| OVERALL (meta classifier) | **36.06** | - |
| LeNet7 (250k online updates) | **16.70** | [3] |
| LeNet7 (> 250k online updates) | **7.80** | [5] |

**Table 4.** Test error rates of individual CPCN, meta-classifier, and LeNet7

| Criteria | LeNet7 | CPCN |
|---|---|---|
| **Number of class** | 6 | 2 |
| **Number of sample per class** | 48600 | 48600 |
| **Number of multiply-add** | 4.66 Million | 1.68 Million |
| **Reference** | [5] | - |

**Table 5.** Comparison between LeNet7 and CPCN on the NORB jittered-cluttered dataset

be shortened by a novel pairwise convolutional network coupled with a circular pairwise framework proposed in this paper. Even though there is a drop in the recognition accuracy, however the accuracy may not be the most important success factor for a recognizer in most cases. In some real world applications it is common that the main priority for the recognizer is to learn quickly.

As pairwise classifiers are modular in nature, prior information or desired behavior such as low false positives could be explicitly build into a detection/recognition system by putting more emphasis on the background class. In the invariant object recognition task described above, we can now train the CPCNs in a way such that each of them learn to recognise a junk class along with another two classes of different objects. The feasibility of this idea is left for future work.

## References


1. Edelman, S.: Computational theories of object recognition (1997)
2. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition (1998)
3. LeCun, Y., Bottou, L., Huang, F.J.: Learning methods for generic object recognition with invariance to pose and lighting (2004)
4. Cutzu, F.: Polychotomous classification with pairwise classifiers: a new voting principle. Technical Report TR573, Indiana University (2003)
5. Computational, Biological Learning Lab, N.Y.U.: (Norb: Generic object recognition in images) [Accessed Feb 2005].