A Novel Approach to Improve the Training Time of Convolutional Networks for Object Recognition

Choon Hui TEO
Faculty of Information and
Communication Technology,
Universiti Tunku Abdul Rahman,
46200 Petaling Jaya
Selangor, MALAYSIA
email : *teochoonhui@gmail.com*

Yong Haur TAY
Faculty of Information and
Communication Technology,
Universiti Tunku Abdul Rahman,
46200 Petaling Jaya
Selangor, MALAYSIA
email : *tayyh@mail.utar.edu.my*

Weng Kin LAI
Advanced Informatics Lab.
MIMOS Bhd.,
Technology Park Malaysia,
57000 Kuala Lumpur,
MALAYSIA
email : *lai@mimos.my*

Abstract — Convolutional neural 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. Even though the recognition accuracy is good, the lengthy training time involved can be a major set back. However, in some applications of object recognition, the computational accuracy of the network is not as important as its computational speed.

In this paper we describe how a new approach that combines a convolutional network with circular pairwise classification can significantly shorten the network's training time. In addition, we compare the training time as well as the proposed new approach's ability to expand the list of objects of interest with that of a common monolithic neural network. We will also show the results of recognition accuracy for both networks.

1.0 INTRODUCTION

A typical convolutional network for shape-based generic object recognition with invariance to pose, lighting and surrounding clutter, *LeNet7* [4], has been shown to be superior, in terms of recognition accuracy, to other learning methods such as linear classifier, Support Vector Machine (SVM) and K-Nearest Neighbor (KNN) [1]. The invariance properties of LeNet7 in the end-to-end (i.e., from image pixels to class label) object recognition problem is achieved by training with a large number of samples per class under different aspect view, lighting, geometrical distortion, and superimposed background images. Naturally, the dataset is huge in size and caused a long training time for convolutional network which is known to be computation intensive. Although the stochastic *Levenberg-Marquardt* learning algorithm used to train *LeNet7* is fast, the computation steps required per weight update are still large.

Instead of developing fast learning algorithm, we tackle the long training time problem by adopting a divide and conquer approach. The large dataset is divided into a set of sub-datasets with each consisting of the samples from two classes. The samples of each class exist exactly in two different sub-datasets so that the data can now be processed by the minimal pairwise classification framework that we propose in this paper – *Circular Pairwise Classification (CPC).*

In this paper we describe how convolutional network and circular pairwise classification can be combined to shorten the network's training time. In addition, we contrast the training time and the capability to expand the list of objects of interest between *LeNet7* and Circular Pairwise Convolutional Networks (CPCNs). We also carried out an experiment that addresses the recognition accuracy of CPCNs and compare it with that of *LeNet7*.

2.0 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 important architectural ideas, namely local receptive fields, shared weights, and spatial sub-sampling, that ensure some degree of invariance to shift, scale and distortion.

Local receptive field is defined as 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.

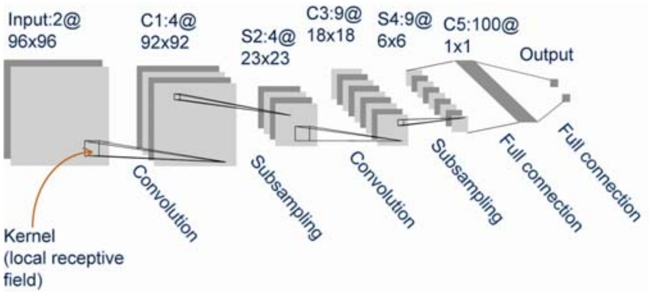


Figure 1: Convolutional network for generic object recognition.

Figure 1 shows a typical convolutional network for generic 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-Layer 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 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).

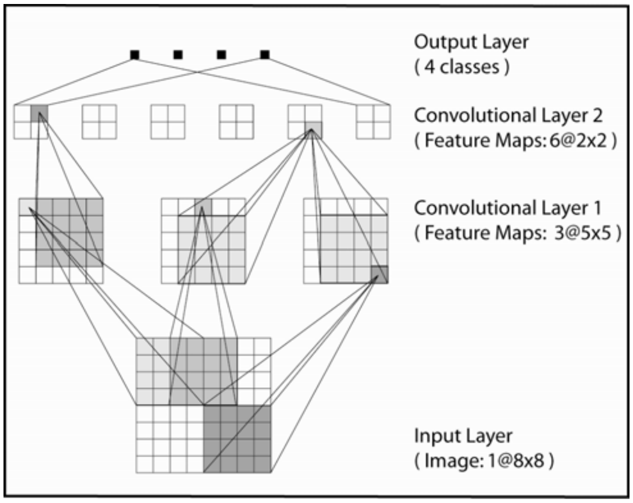


Figure 2: Spatial Convolution.

2.2 SPATIAL SUB-SAMPLING

Spatial Subsampling 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 3, an 8x8 input image is down-sampled into a 4x4 image (or feature map) using a trainable weight and a bias term.

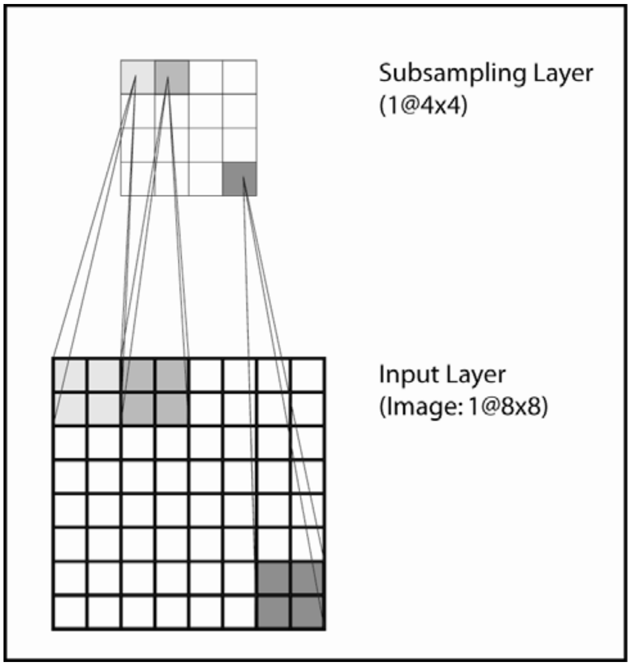


Figure 3: Example of spatial sub-sampling**.**

3.0 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 4.

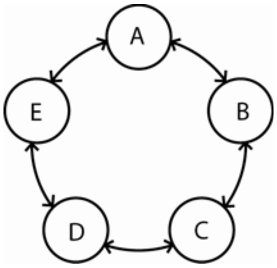


Figure 4: Classes pairing in Circular Pairwise Classification. Each small circle here represents a class. This 5-class classification example is decomposed into 5 sub-problems represented by 5 ordered pairs, namely (A, B), (B, C), (C, D), (D, E), and (E, A).

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, $\boldsymbol{x}$, is given, we cannot justify that $\boldsymbol{x}$ belongs to one class but not others. Moreover, due to the fact that a pairwise classifier can give erroneous output if $\boldsymbol{x}$ 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), C_{ij}, which is trained on the class pair (i, j) to produce two probabilistic output, r_{ij} and r_{ji} (= 1- r_{ij}) where r_{ij} and r_{ji} are the *ratios* of probability densities [3], the probability of an unknown input, $\boldsymbol{x}$, to belong to class j and class i may be computed as such, viz.

$$r_{ij} = \frac{P(i\,|\,\mathbf{x})}{P(i\,|\,\mathbf{x}) + P(j\,|\,\mathbf{x})} \tag{1}$$

$$r_{ji} = \frac{P(j\,|\,\mathbf{x})}{P(i\,|\,\mathbf{x}) + P(j\,|\,\mathbf{x})} \tag{2}$$

This is based on *Cutzu's* vote-against scheme [3]. With another pairwise classifier, C_{jk}, and its probabilistic output, r_{jk} and r_{kj}, we can also estimate r_{ik} and r_{ki} even though none of the pairwise classifiers are trained with a class pair (i, k). The calculation of r_{ik} and r_{ki} are shown in equations (3) and (4) on the next page.

$$r_{ik} \approx \frac{r_{ij} \cdot r_{jk}}{r_{ij} \cdot r_{jk} + r_{kj} \cdot r_{ji}} \quad (3)$$

$$r_{ki} \approx \frac{r_{kj} \cdot r_{ji}}{r_{ij} \cdot r_{jk} + r_{kj} \cdot r_{ji}} \quad (4)$$

Similarly, we can also provide an estimate for r_{im} and r_{mi} if r_{km} and r_{mk} are given as such,

$$r_{im} \approx \frac{r_{ik} \cdot r_{km}}{r_{ik} \cdot r_{km} + r_{mk} \cdot r_{ki}} \quad (5)$$

$$r_{mi} \approx \frac{r_{mk} \cdot r_{ki}}{r_{ik} \cdot r_{km} + r_{mk} \cdot r_{ki}} \quad (6)$$

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.

4.0 EXPERIMENTAL SETUP

4.1 NORB DATASET

In the context of generic object recognition, we used the NORB (*NYU Object Recognition Benchmark*) jittered-clutter dataset as benchmark dataset in order to compare the performance between circular pairwise convolutional networks and *LeNet7*. This was developed by *LeCun et al.* at New York University for the same purpose [4]. 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 fix. 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 5). 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 *distractor* objects [1].

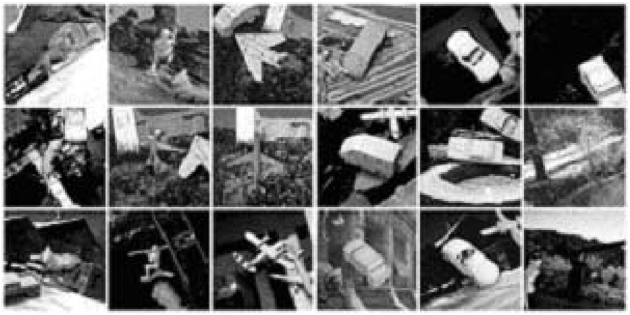


Figure 5: Samples (left camera image) of NORB jittered-clutter dataset. From the leftmost column to the rightmost are samples of **classes Animal, Human, Plane, Truck, Car**, and **Junk.**

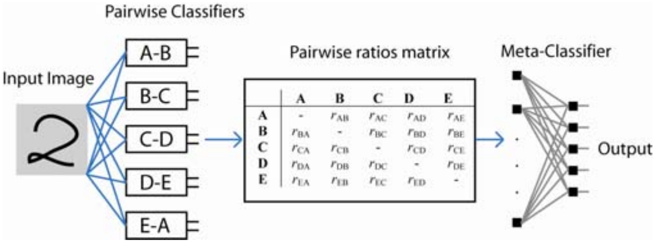
4.2 PAIRWISE CONVOLUTIONAL NETWORK ARCHITECTURE

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). 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 3. 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 a stochastic backpropagation algorithm with Cross-Entropy criterion to give probabilistic outputs or ratios. All the CPCNs were trained in parallel on similar computers for a maximum of 35 iterations. The list of CPCNs trained with different pairs of classes is shown in table 1. The overall specification of a pairwise convolutional network including the number of feature maps, feature map dimension, kernel size, number of trainable weights and connections are listed in table 4.

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 section 3. 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. This is shown in see figure 6.



	A	B	C	D	E
A	-	r_{AB}	r_{AC}	r_{AD}	r_{AE}
B	r_{BA}	-	r_{BC}	r_{BD}	r_{BE}
C	r_{CA}	r_{CB}	-	r_{CD}	r_{CE}
D	r_{DA}	r_{DB}	r_{DC}	-	r_{DE}
E	r_{EA}	r_{EB}	r_{EC}	r_{ED}	-

Figure 6: CPCNs based object recognizer. The input image is fed to all CPCNs to produce a pairwise ratio matrix that, in turn, is fed into a meta classifier to generate the final output.

Classifier	Sub-Training dataset (Pair of classes)
$N_{A,H}$	'Animal' & 'Human'
$N_{H,P}$	'Human' & 'Plane'
$N_{P,T}$	'Plane' & 'Truck'
$N_{T,C}$	'Truck' & 'Car'
$N_{C,J}$	'Car' & 'Junk'
$N_{J,A}$	'Junk' & 'Animal'

Table 1: Six PCNs trained on different pairs of classes of NORB jittered-cluttered dataset.

	0	1	2	3
0	X		X	X
1		X	X	X

Table 2: Connection table between layer Input and layer C1 of PCN trained on NORB jittered-cluttered sub-dataset.

	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

Table 3: Connection table between layer S2 and layer C3 of PCN trained on NORB jittered-cluttered sub-dataset.

Layer	Number of feature maps or units	Dim.	Kernel size	Number of trainable weights	Number of connections
In	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
Out	2	1	-	202	202
			Total	33,760	1,680,808

Table 4: Specifications of PCN trained on NORB jittered-cluttered sub-dataset.

5.0 RESULTS AND DISCUSSION

The results of the experiments are shown in table 5. We can see that the test error rates for (Car, Junk) and (Plane, Truck) are significantly better than the rest. As may be observed from figure 5, images related to animal, human, and plane are generally more complex, with many branches.

Moreover, cars and trucks share many common features and as a result, separating these two classes have proved to be challenging. This may explain why the CPCN trained with pairs of classes such as (Plane, Truck) and (Car, Junk) achieved lower test error rate compared to the others.

Pair 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 (250,000 online updates)	***16.70***	**[1]**
LeNet7 (more than 250,000 online updates)	***7.8***	**[4]**

Table 5: Test error rates of individual CPCNs, meta-classifier of CPCNs and *LeNet7*.

The overall test error rate of CPCNs when combined with a meta-classifier as shown here is higher than those from the monolithic convolutional network, *LeNet7*. This poorer performance may be due to insufficient training applied to those CPCNs as the test error rate of *LeNet7* had also dropped from 16.7% [1] to 7.8% [4]. This had happened even though it had been further trained with a different set of learning parameters [5].

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 6 shows the comparison between *LeNet7* and *CPCN* based on these two criteria. As the number of samples per class is the same (i.e., 48600) for all classes [1], it may be clear that the CPCNs only need to be trained with one-third the number of training samples as that of *LeNet7*. In addition, CPCN takes only about 36% of the total number of multiply-add computations in *LeNet7*. Approximately, a CPCN is at least 8 times faster than *LeNet7* for a fixed number of training iterations. Even though the two-class problem is easier to learn than a six-class problem, CPCNs have shown that they can achieve convergence 8 times faster than *LeNet7* on 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*.

Another advantage of CPCNs is that an object recognition system built with the proposed CPCN architecture can easily extend its list of objects of interest by just training two new CPCNs, unlike a monolithic network whereby it would need a new session of training on all the object samples.

Criteria	**LeNet7**	**CPCN**
Number of classes	6	2
Number of samples per class	48600	48600
Number of multiply-add	4.66 Million	1.68 Million
Reference	[4]	-

Table 6: Comparisons between *LeNet7* and CPCN

6.0 CONCLUSIONS AND FUTURE WORK

In this paper, we have presented how a novel pairwise convolutional network coupled with a circular pairwise framework has significantly shortened the training time when it was used as a generic object recognizer. 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. Nevertheless, we would like to increase the overall recognition rate by improving the performance of each individual CPCN in the future.

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 generic 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.

REFRENCES

[1] Y. LeCun, L. Bottou, and F. J. Huang, "Learning Methods for Generic Object Recognition with Invariance to Pose and Lighting," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* 2004.

[2] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” in *Proceedings of the IEEE,* 1998, (86)11:2278-2324.

[3] F. Cutzu, “Polychotomous classification with pairwise classifiers: a new voting principle,” Tech. Rep. TR573, Indiana University, 2003.

[4] Computational and Biological Learning Lab, New York University, “NORB: Generic Object Recognition in Images,” February 2005, *http://cs.nyu.edu/~yann/research/norb/index.html.*

[5] Y. LeCun, Personal Communication, 2005.