Noname manuscript No.
(will be inserted by the editor)

An Efficient Multi-scale Convolutional Neural Network for Image Classification Based On PCA

Shaohua Li · Huimin Huang · Yue Zhang · Ming Liu

the date of receipt and acceptance should be inserted later

Abstract Object recognition has always been regarded as one of the most important capacities of an intelligent robot. Image classification, which aims to label an image with a category of the object it belongs to, is a lightweight form of object recognition. Traditional ways to achieve this task is to extract handcrafted features followed by a multi-label classifier. With the development of deep learning, it turns out that a deep architecture outperforms traditional machine learning models especially for large-scale data. Convolutional neural network(CNN), with its advantage of alleviating the calculation of engineered features, has been successfully used in computer vision tasks. However, most deep learning models have high computational cost and are difficult to be applied to robotic applications.

In this paper, we propose a multi-scale convolutional neural network for feature extraction, followed by some fully connected layers for feature representation. We show that there are data redundancy in these fully connected layers, and use principle component analysis(PCA) for data reduction, which reduces the computational cost. We name our model Multi-scale Convolution + PCA Network. We evaluate our model on several benchmark datasets for image classification, mainly considering recognition accuracy as well as runtime and show the efficiency of our model.

Shaohua Li
The Hong Kong University of Science and Technology
E-mail: slias@ust.hk

Huimin Huang
The Hong Kong University of Science and Technology
E-mail: hhuangaf@connect.ust.hk

Yue Zhang
The Hong Kong University of Science and Technology
E-mail: yzhangdj@connect.ust.hk

Ming Liu
The Hong Kong University of Science and Technology
E-mail: eelium@ust.hk

1 Introduction

1.1 Background and motivation

The development of artificial intelligence has granted the rise and fast progress of robotics. One significant capability for a mobile robot is environment understanding, which includes object recognition, scene recognition *etc.* Of the above mentioned aspects, scene recognition plays a significant role in environment understanding. Traditional ways to achieve this task are based on sparse feature representations [1,2]. In recent years, deep learning has been used for dense feature extraction [3, 4]. Meanwhile, the importance of robots' ability for object recognition could not be neglected since recognition of objects always helps recognition of scenes. Numbers of scene recognition approaches based on object recognition have appeared in the literature [5–7].

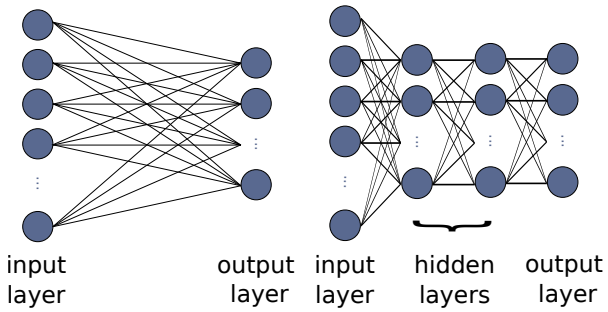
Compared to the ability of using geometrical primitives for environment understanding [8–10], which is the typical approach for simultaneous localization and mapping(SLAM), the ability of feature extractions from images is a type of high-level intelligence. Human beings are equipped with a complete and powerful visual system, which is of high sensitivity. Such a system helps them distinguish among even tiny differences between different kinds of objects with little effort, regardless of scale, rotation, luminance or even occlusion. However, this task is not seemingly easy for robots, who extract features from pixel information(sometimes also with depth information), since a variety of factors should be taken into consideration, such as the above

mentioned aspects. To achieve high recognition accuracy, a large amount of training data is usually required, leading to a time-consuming training process.

Typical processes for image classification consist of two steps: feature detection/extraction from raw images and classification of features. That is to say, robots recognise images using their features. A great many algorithms to extract features from images have arose in the literature. Most of these approaches took into consideration handcrafted features, such as SIFT [11], SURF [12], Spin Images [13] and so on. However, the utilization of handcrafted features has one significant limitation in that certain kind of features may work well with some specific categories of objects or images, but may not easily generalize to other types. Besides, some kinds of handcrafted features, with high computational cost, cannot easily be adapted to real-time applications.

Recently, the mainstream for image classification and object recognition is to use deep learning models. Deep learning, also called hierarchical learning, is based on a set of algorithms that attempt to model high-level abstractions in data by using a hierarchical structure. Fig.1 shows the difference between a traditional neural network and a deep neural network. This hierarchical structure is inspired by the way how human brain process with information, with a deep structure and process with information hierarchically. Among the algorithms of deep learning, convolutional neural network(CNN) is a type of feed-forward artificial neural network that attempts to extract features from inputs, especially from images. One characteristic of CNN is that the convolution operation learns to extract features hierarchically from a receptive field, where the pixels are usually highly correlated and represents some kinds of patterns. The advantage of CNN over traditional feature extractors lies in that CNN avoids the calculation of handcrafted features. CNN learns the features of input data by back propagation that aims to minimize a certain cost function, thus it could theoretically extract useful features that helps the classification. CNN also differs from traditional neural networks. The kernels in the convolution layers act as shared weights, together with the pooling operation that decreases the data dimension, largely reduce the memory size.

Although deep learning has been widely used in computer vision tasks, quite a proportion of researchers spend their effort increasing the recognition accuracy on some datasets, regardless of the execution time for test. Therefore, some models, using complicated architectures, achieved high recognition accuracy but is difficult to be put into real time applications. Motivated by this, we propose a method using PCA for removing data redundancy of the fully connected layer for fast execution for testing, which is potential for further integration with real-time robotic applications.



(a) A Traditional Shallow Neural Network (b) A Deep Neural Network

Fig. 1: Difference between traditional neural network model and deep learning model

1.2 Contributions

For this paper, we summarize our contributions as follows:

- We propose a novel Multi-scale Convolution + PCA Network for feature extraction from different size of receptive fields, and data redundancy reduction of the fully connected layer, which reduces the computational cost and energy cost for testing.
- We make insight views of a deep neural network by analysing some of the hidden layers and reveal certain pattern learned by the network. We show that there is sparsity and correlation of the hidden layer nodes.
- We evaluate our approach on several benchmark datasets of object classification, and showed our results, mainly considering recognition accuracy and test time.

1.3 Organization

The rest of this paper is organized as follows: In Section 2, we make literature review of the work that is related to ours. Section 3 introduces the structure and theoretical basis our Multi-scale Convolution + PCA Network, followed by experiments and results of some benchmark datasets described in Section 4. We make discussion in Section 5 and conclude our work in Section 6.

2 Related Work

2.1 Deep learning and CNN applications

Typical deep learning algorithms include convolutional neural network [14,15], deep belief network(DBN) [16, 17], stack (denoising) auto-encoders [18,19] etc. Up till now, various deep learning algorithms have been applied to different research fields, like computer vision, natural language processing, speech recognition and so on and have achieved state-of-the-art results.

First proposed in 1980 by K. Fukushima [14] and further improved in 1998 by Yann LeCun *et. al* [15], CNN has now been successfully applied to various fields, including computer vision and speech recognition. For computer vision applications, CNN has been applied for hand-written digits recognition [20], object recognition [21], face recognition [22] and scene recognition [3]. [23,24] show examples of CNN applied to speech recognition.

2.2 CNN algorithms and object recognition

The basic structure of CNN for object recognition is as follows: A bank of filters makes up the convolution layer. Then each node is activated by a non-linear activation function. What we usually use are `tanh()` function and sigmoid function, followed by a pooling layer for downsampling. We use these set of layers one after another for feature extraction. Then the features maps are forward propagated to a fully connected layer. Finally a classifier is used for classification.

During recent years, varieties of variants of CNN have been proposed. First, some response normalization methods are proposed, also known as normalization layers. Such layers are designed to encourage competition for neurons with high response. Local response normalization [25,21] takes into consideration a subset of neurons at the same topographic position. While local contrast normalization [26] is a kind of normalization within a channel. There are also many approches to prevent deep neural networks from overfitting. A simple trick for that is training set augmentation, as used by many researchers for implementation. G. Hinton *et. al* [25] proposed an approach for preventing coadaptations of feature detectors, known as "dropout". In the proposed algorithm, hidden nodes are randomly dropped out during the training process, which leads to training a sub-network at each iteration. While at test time, all the nodes are enabled. Such a structure trains a more robust model and decreases the degree of overfitting. M. Zeiler *et. al* [27] proposed an approach using stochastic pooling and achieved state of the art results on many benchmark datasets. During ImageNet competition of 2012, A. Krizhevsky *et. al* [21] used a type of activation function $f(x) = \max(0, x)$, and neurons with this nonlinearity is called Rectified Linear Units(ReLUs). Such a structure highly improves the convergence time of training and is now widely used. Besides the above mentioned approaches, most of which are local modifications of the traditional CNN model, there are also some researchers who proposed a new deep architecture. M. Lin *et. al* [28] proposed a deep structure called "Network In Network"(NIN). Instead of using a convolution kernel to scan local receptive fields to get feature maps, NIN introduces multi layer perceptron after the convolution operation. This operation aims to enhance the model discriminability. Maxout networks [29] takes the maximum value of a set of inputs and takes the advantage of dropout to facilitate optimization.

3 Multi-scale Convolution + PCA Network

In this section we are going to give a detailed description of our proposed model. The whole structure is shown in Fig.2. We will introduce our model from three aspects: multi-scale feature extraction, data redundancy reduction using PCA and feature classification.

3.1 Multi-scale Feature Extraction

Inspired by the way human beings process information, CNN uses a hierarchical structure for feature extraction for images, in which the bottom layers represent low-level abstraction while the top levels represent high-level abstraction. Similar to [3], which used a multi-scale CNN for scene labelling, we use a multi-scale CNN for object feature extraction, which also is inspired by the way human beings perceive feature information from visual inputs. Human beings are able to recognise objects from different ranges, which roots from our ability to get object features from long range distances as well as from nearby. Our model aims to imitate this process and extract features from different scales of inputs.

As shown in Fig.2, our proposed network takes inputs of different scales. The original scale image should be part of the inputs. The rest part of the inputs are the upsampled or downsampled images, depending on the size of the original images(in this figure we show both). It is natural that one may think of using multiple separate CNNs for different scales of inputs, and then concatenate the features from different networks. However such a model have several drawbacks. First

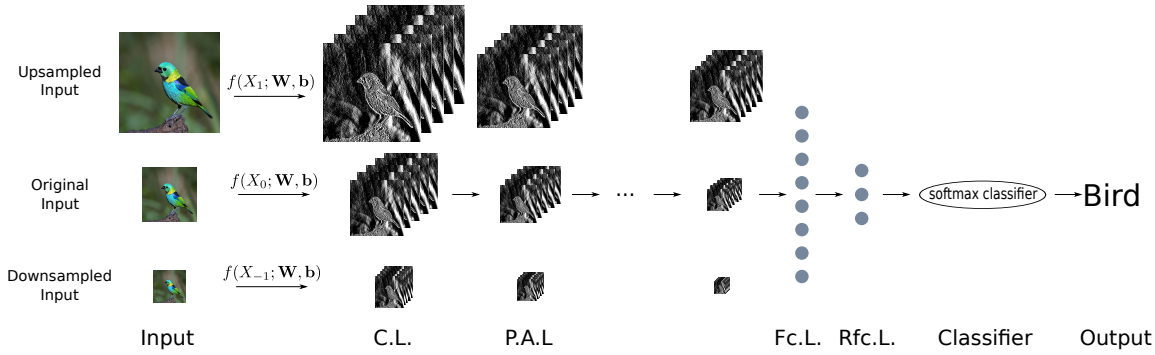


Fig. 2: Proposed Model of Multi-scale Conv. + PCA Network. C.L.: Convolution Layer, P.A.L.: Pooling and Activation Layer, Fc.L.: Fully connected Layer, Rfc.L.: Reduced fully connected Layer.

of all, this operation could sometimes introduce serious overfitting. Second, for a certain number of feature maps we wish to achieve from the convolution layers, this model leads to higher dimension of the parameter space and slower convergence during the training process, compared to our proposed model, which uses shared kernels between different scales of inputs. Another reason that we use shared convolution kernels between different scales of input is that we aim to extract scale-invariant features from the input and get a more robust model.

Another way to interpret our model is that using the shared kernels for downsampled images is the same as using another convolution kernel of larger size to extract larger scale features, and similarly we could explain for upsampled images. Still our model has the advantage that a comparatively smaller number of parameters is to be trained.

Denote the input feature maps as $\mathbf{X}$, where $\mathbf{X} = \{X_{-k}, ..., X_{-1}, X_0, X_1, ..., X_l\}$. The negative subscripts denote downsampled images and positive ones denote upsampled images, X_0 stands for the original image. Denote the convolutional kernels by $\mathbf{W}$, where each kernel W_i is a 3D kernel. The bias term is denoted as $\mathbf{b}$, where each element b_i corresponds to the i-th element in $\mathbf{W}$, i.e. W_i. The same set of convolutional kernels $\mathbf{W}$ and bias $\mathbf{b}$ is applied to each scale of the inputs, producing a set of feature maps for each scale.

$$f(X_i; W_j, b_j) = X_i * W_j + b_j \tag{1}$$

where "*" denotes 2d convolution. Afterwards a non-linear activation function is applied to each element in the feature maps. In our model, we choose the Rectified Linear Unit(ReLU)[21], which is easy to compute and proved to achieve quick convergence.

$$f(x) = \max(0, x) \tag{2}$$

The output of the convolution layers is a set of feature maps of different sizes. These feature maps are then vectorized and concatenated to form a vector representation of the features, which are take as the inputs of the fully connected layers.

3.2 Feature Compression Using PCA

Deep learning models are usually regarded as a black-box model. What we are most cared about is the inputs and outputs, and such a black box tries to learn the specified relationships between inputs and outputs. However, it usually makes no explicit sense what a certain hidden layer implies. So delving the relationship between hidden nodes is a first step for us to understand the implications of hidden layers. Besides, test time and computational cost are crucial for robotic applications, which inspires us to simplify the model for real-time computing. Motivated by these ideas, we search for the relationships of the hidden nodes and propose a feature compression method using PCA.

A fully connected layer is a layer that for each hidden node, it takes the sum of all input nodes, followed by a non-linear activation function. Experiments show that there are several characteristics about the hidden layers. First, some hidden nodes varies little for different inputs. Second, some certain hidden nodes are highly correlated with some other nodes. These findings implies that there exists information redundancy over the hidden layer. Therefore it is possible if we use some compression algorithms so that the computational cost would be reduced. In our model, we use PCA for the states of hidden nodes, which reduces the dimension of the feature vectors and reserves the principle components.

Denoting the states of the hidden layer nodes by $\mathbf{s}$, an n dimensional vector where n is the number of hidden nodes. The covariance matrix $\mathbf{C}$ is calculated as

$$\mathbf{C} = \mathbb{E}[(\mathbf{s} - \mathbb{E}(\mathbf{s}))(\mathbf{s} - \mathbb{E}(\mathbf{s}))^T] \tag{3}$$

It could be proved that $\mathbf{C}$ is a semi-positive definite matrix, which means that none of its eigenvalues are less than 0. Next use eigenvalue decomposition to calculate the eigenvalues and eigenvectors of $\mathbf{C}$. Reserve a certain amount of eigenvectors corresponding to the largest m eigenvalues to form the transformation matrix $\mathbf{A}_{n\times m}$.

$$\mathbf{T} = [\mathbf{t}_1, \mathbf{t}_2, ..., \mathbf{t}_m] \tag{4}$$

where m is chosen manually. Then transform the original hidden nodes' states to a reduced space.

$$\mathbf{r} = \mathbf{T}^T\mathbf{s} \tag{5}$$

We must point out here that reducing the dimension of the hidden nodes using PCA is not the same as training a different model with smaller number of hidden nodes directly. Suppose that in our model there are n_o hidden nodes, and after PCA n_r principle components are reserved, i.e. the feature vectors are transformed into another space with dimensionality n_r. However we can seldom(or almost never) result in the same model if we train a network with n_r hidden nodes. The reason is simple: for this type of classification problem, we aim to get an optimal solution minimizing the objective function of the classifier, however such a problem is usually highly non-convex. This means that the optimal solution depends largely on the initial values of those parameters to be trained. It is possible that the states of the hidden nodes are still highly correlated even if we try to train a model with n_r hidden nodes. Besides, the reduction in the number of hidden nodes will generate a different model since the number of parameters is changed. These factors together make it almost impossible to get a same model only by reducing the number of hidden nodes.

There are also differences between our model and dropout, although both have the procedure of "dropping out" nodes. Dropout is used for overcoming overfitting, which trains a different sub-network for each iteration of the training process, and uses all nodes when testing. Our model reduces the number of hidden nodes during testing, which reduces the test time.

3.3 Feature Classification

For feature classification, we use a linear classifier for multi-category classification, softmax classifier. The classifier takes the features after PCA as inputs and computes the probability distribution over classes. The probability of the i-th input to be predicted as the j-th class is computed as

$$\hat{p}_{ij} = \frac{\exp(\mathbf{w}_j^T\mathbf{r}_i)}{\sum_{k=1}^{n}\exp(\mathbf{w}_k^T\mathbf{r}_i)} \tag{6}$$

where n is the number of classes. The objective function is defined as

$$J(\mathbf{W}) = -\frac{1}{n}\sum_{i=1}^{m}\sum_{j=1}^{n} p_{ij}\ln p_{ij} \tag{7}$$

In (7), n is the number of input examples, which means that the objective function is the average cost over the input examples. $\mathbf{W} = [\mathbf{w}_1, ...\mathbf{w}_m]$ is the parameters of the classifier model.

4 Experiments

We evaluated our approach on several benchmark datasets for object recognition, CIFAR-10 [30], CIFAR-100 [30] and STL-10 datasets [31]. We show the experiment results on these datasets in this section.

4.1 Test on CIFAR-10

CIFAR-10 dataset is a set of tiny images that consists of 60,000 images of size 32×32. The dataset consists of images from 10 classes, 50,000 images for training and 10,000 for testing.

We use a 2-scale CNN for feature extraction in our model, with inputs the original images and downsampled images. We have to point out that our downsampling does not work the same way as data augmentation. Data augmentation is a technique for reducing overfitting. It usually uses image translation or cropping to increase the number of training set. While in our model the downsampling operation does not increase the number of training examples, but aims to extract features of different scales. In our model, we use 3 steps of convolution-pooling-normalization(C-P-N) for feature extraction. The first convolution layer consists of 32 convolution kernels with size 5×5, followed by 2×2 non-overlapping max pooling and local contrast normalization. The second and third C-P-N layers takes similar structure except for the pooling area of 1×1, i.e. no pooling is used for these layers, and the third convolution layer uses 64 feature maps. The fully connected layer consists of 64 hidden nodes originally, and utilizing PCA we transform the 64-dimensional feature representation into 10-dimensional. Finally a 10-class softmax classifier is used for classification. In order to ensure quick convergence for training and prevent overfitting, we take in to consideration of the momentum of gradient and add a weight decay term for each update in the training, the same way as [21].

First, we analyse the states of the hidden nodes. By plotting the histogram of all the hidden nodes, we

find that there are certain hidden nodes that have quite low variance and are most time with value 0. Fig. 3 shows these type of hidden nodes. Although the plot just shows the histogram of one training process. We make analysis for each training process and find that there are always hidden nodes like this.

Next, we make analysis to the correlation of the hidden nodes. Fig. 4 shows a 2D plot of some highly correlated nodes(the values are distributed in a narrow ellipse) compared to the less correlated nodes shown in Fig. 5. The highly correlated nodes are mainly distributed along some certain direction, while the less correlated ones have no direction preference.

We summarise the findings about the hidden layer as follows:

- There is sparsity of the hidden layer nodes' states, which do not contribute much to the classification.
- There are nodes that are highly correlated, which means that there is data redundancy within the hidden layer.

Inspired by these findings, and noticing that PCA is an efficient algorithm for decorrelation and compression, we propose to apply PCA in CNN to reduce the computational cost.

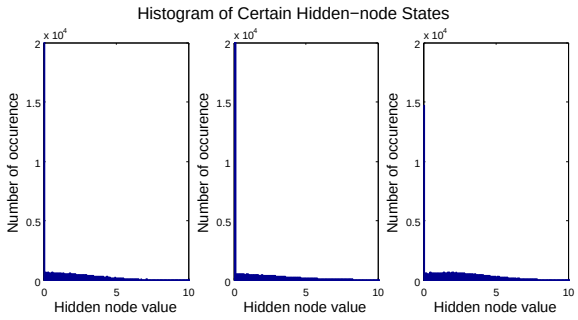


Fig. 3: Histogram of Low-Variance Hidden Nodes. Note there is a quite large value at 0.

Table.1 shows the results of our proposed structure. We compare the results of Multi-scale CNN, Multi-scale Conv. + PCA Network and some of the other approaches without using training data augmentation.

Method	Best Test Accuracy	Test Time
PCANet [32]	78.67%	> 80 ms
Stochastic Pooling CNN [27]	84.87%	36.8 ± 0.2 ms
Multi-scale CNN	76.17%	37.5 ± 0.3 ms
Multi-scale Conv. + PCA Network	73.20%	35.3 ± 0.2 ms

Table 1: Test Results on CIFAR-10 Dataset.

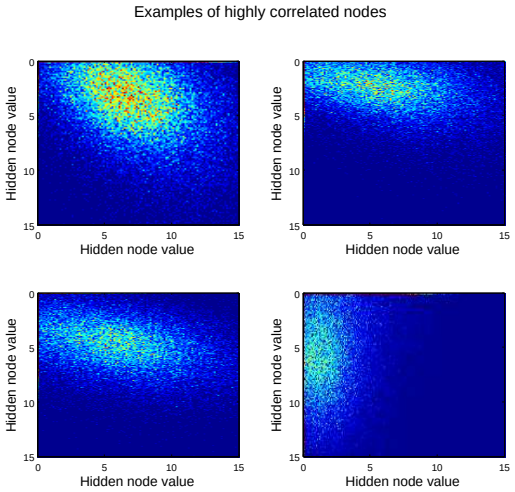


Fig. 4: 2D plot of Highly Correlated Nodes. X and y axes represent the value of different hidden nodes. Each point in the plot represents the density of points in corresponding coordinate.

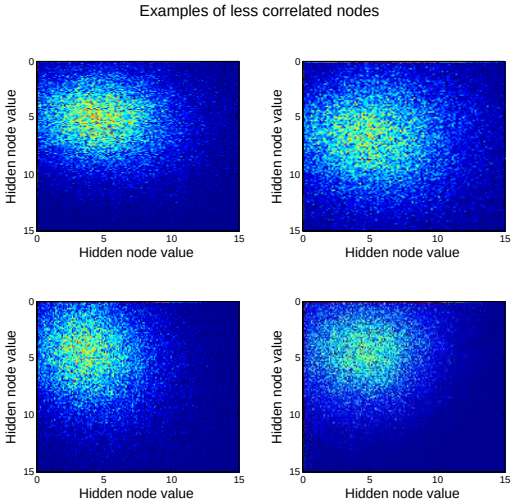


Fig. 5: 2D Plot of Less Correlated Nodes. X and y axes represent the value of different hidden nodes. Each point in the plot represents the density of points in corresponding coordinate.

From Table.1 we see that without using complicated structure, our model achieved satisfactory results. The test time is for a single image without using GPU. We show that by using PCA, we reduce the computational cost and improve the test time. The test time for PCANet [32] is based on the estimation of complexity of algorithm.

The confusion matrix is shown in Fig. 6. Large proportion of misclassification comes from mistakes among animals, which could be understood easily. It is quite understandable to mistake a cat for a dog.

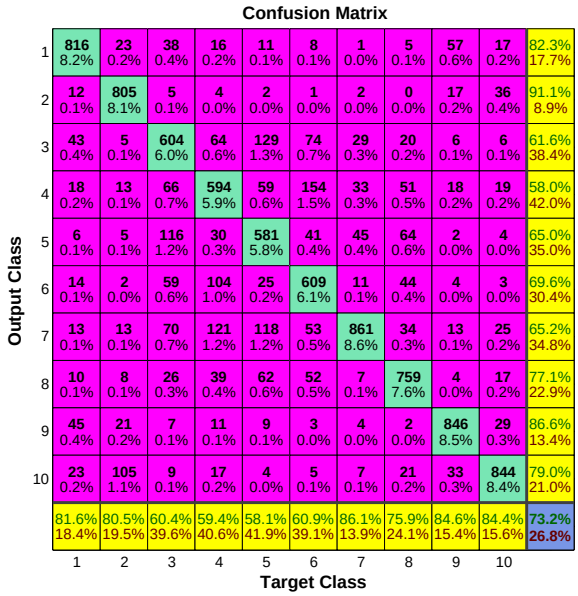


Fig. 6: Confusion Matrix of Recognition Results. Class number 1: Airplane, 2: Automobile, 3: Bird, 4: Cat, 5: Deer, 6: Dog, 7: Frog, 8: Horse, 9: Ship, 10: Truck.

4.2 Test on CIFAR-100

CIFAR-100 dataset is just like CIFAR-10, except that it has 100 class containing 600 images each. The increase in the number of classes leads a to more difficult recognition task. We use a similar structure like what we use for CIFAR-10 dataset. The only difference is here we use 2 fully connected layers instead of just 1. And the number of hidden nodes in each layer is 512 and 256 respectively. Meanwhile, since the number of classes changes, the softmax classifier should be modified correspondingly. Table. 2 shows the result of our proposed model. The introduction of PCA improves the test time.

Method	Best Test Accuracy	Test Time
Stochastic Pooling CNN [27]	57.49%	44.2 ± 0.7 ms
Maxout Nets [29]	61.43%	52.6 ± 1.3 ms
Multi-scale CNN	47.74%	45.9 ± 1.0 ms
Multi-scale Conv. + PCA Net	44.23%	43.4 ± 0.9 ms

Table 2: Test Results on CIFAR-10 Dataset

4.3 Tests on STL-10 Dataset

Similar to CIFAR-10, STL-10 dataset contains 10 categories but has a comparatively smaller training set and larger test set. The images have size 96×96, acquired from examples on ImageNet. The challenge for STL-10 relies in the small capacity of training set. We get best test accuracy of 58% for multi-scale CNN. The proposed multi-scale PCA Net achieves test accuracy of 54%. Fig. 7 shows some recognition results of our model on STL-10.

Fig. 7: Sample results of STL-10 dataset. Length of the bar represents the predicted probability. Red corresponds to the correct category. Blue corresponds to false categories.

5 Discussion

In our experiments, we show that there exists data redundancy in the fully connected layers, mainly from two types: nodes with low variance and correlation between nodes. Thus PCA could work effectively for redundancy reduction. In the experiment results showed in Section 4, with the use of PCA, the execution time for test improves. Theoretically, our proposed architecture has potential for larger scale networks with more hidden

nodes and is supposed to improve largely the execution time.

6 Conclusion and Outlook

In this paper, we propose a new structure called Multi-scale Convolution + PCA Network. We use shared CNN for different scales of input for feature extraction. We show by experiments that there is information redundancy in fully connected layers, which inspire us to use PCA for compression. We test our model on several benchmark datasets. There is also some future work to be done.

- We will integrate our algorithms with robotic platforms, e.g. Robotic Operating System(ROS) for real systems and test the efficiency and acceptability of our proposed model.
- We will go further inside deep neural networks and delve the rules that might be helpful for quick convergence and real-time execution.

Acknowledgements This work was partially supported by the HKUST Project IGN13EG03; partially sponsored by the Research Grant Council of Hong Kong SAR Government, China, under project No. 16206014 and National Natural Science Foundation of China No. 6140021318.

References

1. M. Liu and R. Siegwart, "Dp-fact: Towards topological mapping and scene recognition with color for omnidirectional camera," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 3503–3508.
2. M. Liu, C. Pradalier, F. Pomerleau, and R. Siegwart, "Scale-only visual homing from an omnidirectional camera," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 3944–3949.
3. C. Farabet, C. Couprie, L. Najman, and Y. LeCun, "Learning hierarchical features for scene labeling," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 8, pp. 1915–1929, 2013.
4. J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," *arXiv preprint arXiv:1411.4038*, 2014.
5. P. Espinace, T. Kollar, A. Soto, and N. Roy, "Indoor scene recognition through object detection," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 1406–1413.
6. A. Torralba, K. P. Murphy, W. T. Freeman, and M. A. Rubin, "Context-based vision system for place and object recognition," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*. IEEE, 2003, pp. 273–280.
7. A. Bosch, A. Zisserman, and X. Muñoz, "Scene classification via plsa," in *Computer Vision–ECCV 2006*. Springer, 2006, pp. 517–530.
8. M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit *et al.*, "Fastslam: A factored solution to the simultaneous localization and mapping problem," in *AAAI/IAAI*, 2002, pp. 593–598.
9. J. A. Castellanos, J. Montiel, J. Neira, and J. D. Tardós, "The spmap: A probabilistic framework for simultaneous localization and map building," *Robotics and Automation, IEEE Transactions on*, vol. 15, no. 5, pp. 948–952, 1999.
10. M. Liu, C. Pradalier, F. Pomerleau, and R. Siegwart, "The role of homing in visual topological navigation," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 567–572.
11. D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.
12. H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Computer vision–ECCV 2006*. Springer, 2006, pp. 404–417.
13. A. E. Johnson, "Spin-images: a representation for 3-d surface matching," Ph.D. dissertation, Citeseer, 1997.
14. K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.
15. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
16. G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.
17. G. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.
18. Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle *et al.*, "Greedy layer-wise training of deep networks," *Advances in neural information processing systems*, vol. 19, p. 153, 2007.
19. P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 1096–1103.
20. D. Ciresan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 3642–3649.
21. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
22. S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face recognition: A convolutional neural-network approach," *Neural Networks, IEEE Transactions on*, vol. 8, no. 1, pp. 98–113, 1997.
23. O. Abdel-Hamid, A.-r. Mohamed, H. Jiang, and G. Penn, "Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, pp. 4277–4280.
24. G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, 2012.
25. G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.
26. K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "What is the best multi-stage architecture for object recognition?" in *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 2146–2153.

27. M. D. Zeiler and R. Fergus, "Stochastic pooling for regularization of deep convolutional neural networks," *arXiv preprint arXiv:1301.3557*, 2013.
28. M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.
29. I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio, "Maxout networks," *arXiv preprint arXiv:1302.4389*, 2013.
30. A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," *Computer Science Department, University of Toronto, Tech. Rep*, vol. 1, no. 4, p. 7, 2009.
31. A. Coates, A. Y. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *International Conference on Artificial Intelligence and Statistics*, 2011, pp. 215–223.
32. T.-H. Chan, K. Jia, S. Gao, J. Lu, Z. Zeng, and Y. Ma, "Pcanet: A simple deep learning baseline for image classification?" *arXiv preprint arXiv:1404.3606*, 2014.