On the Capability of Accommodating New Classes Within Probabilistic Neural Networks

Tetsuya Hoya, *Member, IEEE*
Laboratory for Advanced Brain Signal Processing,
Brain Science Institute, RIKEN,
2-1, Hirosawa, Wakoh-City, Saitama, 351-0198, JAPAN

Abstract

To date probabilistic neural networks (PNNs) have been widely used in various pattern classification tasks due to their robustness. In this letter, it is shown that, by exploiting the flexible network configuration property, the PNN classifiers also exhibit the capability in accommodating new classes. This is verified by extensive simulation studies on using four different domain data sets for pattern classification tasks.

accepted for
IEEE Transactions On Neural Networks
(Topic Area: Pattern Classification and Clustering)

Index Terms:— **Probabilistic neural networks, incremental learning, accommodating new classes, pattern classification.**

[1]Please address correspondence to: Tetsuya Hoya, Laboratory for Advanced Brain Signal Processing, BSI-RIKEN, 2-1, Hirosawa, Wakoh-City, Saitama, 351-0198, JAPAN Electronic mail address: hoya@bsp.brain.riken.go.jp

1 Introduction

Probabilistic neural networks (PNNs) [1]/generalized regression neural networks (GRNNs) [2] belong to the family of radial basis function neural networks (RBF-NNs) [3, 4] which, due to their robustness, are now widely used in various pattern classification tasks. Although the roots of these two networks are strictly and statistically not the same, they exhibit similar characteristics to each other and share the attractive property, namely, that they require no iterative training of the weight vector between the RBFs and the output units [5]. By exploiting this property, the author has proposed an instance-based incremental training scheme using a GRNN [6] in which pattern correction of misclassification data was performed by an on-line batch correction mechanism.

In a recent study [7], a new guideline for the incremental learning paradigm in pattern classification has been given in accordance with the four criteria:

1) The pattern classifier(s) should be able to learn additional information from the new data,
2) They should not require access to the original data, used to train the existing classifier,
3) They should preserve previously acquired knowledge (that is, they should not suffer from catastrophic forgetting),
4) They should be able to accommodate new classes that may be introduced within the new data.

In a previous study [6], it is clearly shown that the incremental scheme based upon PNNs/GRNNs suffices the criteria 1) and 3) above, however the remaining two criteria are yet to be met. Thus, the purpose of this letter is to give an analytical and empirical basis of the fact that PNN classifiers inherently satisfy also Criterion 4) above as well as ARTMAP algorithm [8, 7], namely the capacity of accommodating new classes, which, in essence, is an important and desirable feature for any pattern classifiers.

2 The PNN Classifier

The configuration of a three-layered PNN with N_i input neurons, N_h RBFs, and N_o output neurons is shown in Fig. 1 and is summarized as follows [9]:

$$\begin{aligned} o_k &= \frac{1}{\delta}\sum_{j=1}^{N_h} w_{j,k} h_j \\ h_j &= \exp(-\frac{\|\boldsymbol{x}-\boldsymbol{c}_j\|_2^2}{\sigma_j^2}), \\ \delta &= \sum_{k=1}^{N_o}\sum_{j=1}^{N_h} w_{j,k} h_j \end{aligned} \tag{1}$$

where the outputs o_k $(k = 1, 2, \ldots, N_o)$ are the respective sums of Gaussian mixtures (or kernels), $\boldsymbol{x} = [x_1, x_2, \ldots, x_{N_i}]$ is the input vector, $\boldsymbol{c}_j$ and σ_j are respectively referred to as the centroid

vector of the j-th RBF and the radius. Since, in pattern classification, the value o_k indicates the probability that the input vector $\boldsymbol{x}$ belongs to Class k, the weights between the RBF h_j and the output neurons o_k, namely $w_{j,1}, w_{j,2}, \ldots, w_{j,N_o}$, can then be fixed as

$$w_{j,k} = \begin{cases} 1, & \text{if } \boldsymbol{x} \text{ belongs to Class } k \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

Accordingly, the free parameters within the PNN are $\boldsymbol{c}_j$ and σ_j. Then, an instance-based adaptation scheme for both the network growing and shrinking of a PNN is given as follows:

Network Growing: Set $\boldsymbol{c}_j = \boldsymbol{x}$ and fix σ_j, then add the term $w_{j,k}h_j$ in (1). The weights $w_{j,k}$ are given as binary values in (2).

Network Shrinking: Delete the term, $w_{j,k}h_j$, from (1).

Note that a collection of centroids (RBFs) for Class k can be seen as the corresponding subnet (that is, SubNet k, in Fig. 1) describing the pattern space [6][1]. Each subnet then consists of the centroids which reside within the same class.

As described in [6], the setting of radii values σ_j $(j = 1, 2, \ldots, N_h)$ is crucial for the generalization capability of RBF-NNs and still remains an open issue (e.g., see [10]). However, the author has empirically found that for the radii values σ_j the following unique choice according to the simplified version in [3] still yields reasonable generalization capability:

$$\sigma_j = \sigma = \theta d_{max}, \tag{3}$$

where d_{max} is maximum Euclidean distance between all the centroid vectors within a PNN, i.e., $d_{max} = \max(\|\boldsymbol{c}_l - \boldsymbol{c}_m\|_2^2), (l \neq m)$, and θ is a suitably chosen constant.

Thus, from the structural point of view, accommodating new classes is equivalent to simply adding new subnets within the PNN.

2.1 Accommodation of New Classes Within a PNN

By virtue of the flexible network configuration property as described above, adding new classes can be straightforwardly performed, under the assumption that one pattern space spanned by a subnet is *reasonably* separated from the others. This principle is particularly applicable to PNNs and GRNNs; the training data for other widely used layered networks such as multilayered perceptron neural networks (MLP-NNs) [3] trained by a backpropagation algorithm [11, 3] or ordinary RBF-NNs is encoded and stored within the network after the iterative learning. In MLP-NNs, the encoded data is then distributed over the weight vectors between the input and hidden and those between hidden and output layers (and hence not directly accessible). Therefore, it is generally considered that, not to mention the accommodation of new classes, to achieve a

[1] This is illustrated on the right hand side of Fig. 1.

flexible network configuration by an MLP-NN similar to that by a PNN/GRNN (that is, the quick network growing and shrinking) is very hard. This is because even a small adjustment of the weight parameters causes a dramatic change in the pattern space constructed, which may eventually lead to a catastrophic corruption of the pattern space [7]. For the network reconfiguration of MLP-NNs, it is thus normally necessary for the iterative training from scratch. From another point of view, by MLP-NNs, the separation of the pattern space is represented in terms of the hyperplanes, while that performed by PNNs and GRNNs is based upon the location and spread of the centroids in the pattern space. In PNNs/GRNNs, it is therefore considered that, since a single class is essentially represented by a cluster of centroids, a small change in a particular cluster does not have any serious impact upon other classes, unless the spread of the centroids pervades the neighbor clusters.

2.2 Necessity of Re-accessing the Stored Data

In [7] the authors pointed out that supervised networks such as ARTMAP suffer from poor generalization capability due to over-fitting, at the expense of no access to the previously seen data. To overcome this drawback, it is necessary to involve either the *a priori* knowledge (e.g., data distributions), or a sort of *ad hoc* parameter adjustment scheme. A similar principle also applies to the case of PNNs; in order to maintain the good generalization capability, the internal access to the stored data is necessary in order to update the radii values. However, one of the advantages using PNNs may be that, since a PNN represents a memory-based architecture, it does not require storage of the original data besides the memory space for the PNN itself. In other words, the original data is directly accessible via the internal data, i.e., the centroid vectors $\boldsymbol{c}_j$. In practice, Criterion 2) is therefore too strict and hence re-accessing the original data is still unavoidable.

3 Simulation Study

To show the capability of PNNs in accommodating new classes, four benchmark data sets for pattern classification were used; the SFS [12] for digit voice recognition (/ZERO/, /ONE/, ..., /NINE/, in English) and the three UCI data sets, which are chosen from "UCI Machine Learning Repository" of the University of California, namely the OptDigit, PenDigit, and ISOLET data set. For the UCI data sets, the first two are used for digit character recognition tasks, while the latter is for isolated letter speech recognition tasks. The description of the data sets used is summarized in Table 1.

As shown in Table 1, the SFS data set contains a total of 999 patterns, with 100 patterns available for each digits (except 99 for digit /NINE/). Then, for the simulation study, the data set was divided into two subsets; one for training, consisting of 599 patterns (60 patterns for each

digit, except 59 for Digit /NINE/), and the other for testing (unknown) data set (40 patterns for each digit). In contrast, the OptDigit contains a total of 3823 for training and 1797 patterns for testing available, respectively, whereas 7494 for training and 3498 patterns for testing in the PenDigit data set. Unlike the SFS/ISOLET cases, in both the OptDigit and PenDigit data sets, the number of available patterns of training/testing set for each digit was not fixed, i.e., for the OptDigit, the number of patterns in each digit varied between 376 and 379 for training, while for testing it was 174 and 183. For the PenDigit, these varied between 719 and 780 for training, while 335 and 364 for testing. (For both the data sets, the varying numbers were just the maximum numbers of the patterns available.) For the ISOLET, unlike the three other data sets, there are a total of 26 classes to classify and the training and testing sets contain 40 and 20 patterns per each letter, respectively. In the simulations, the constant $\theta = 0.1$ in (3) was used for all the four data sets.

3.1 Performance Measurement

To investigate the capability of accommodating new classes within a PNN, the measurement in terms of deterioration rate d, which is given as the difference in the number of correctly classified patterns between the initially configured PNN and its grown version with new classes, is introduced as

$$d = \frac{c_i - c_g}{N} \tag{4}$$

where

c_i: number of correctly classified patterns with the initial configuration;

c_g: number of correctly classified patterns with the grown network;

N: total number of testing patterns.

Note that, for the computation of (4), to give a fair comparison, the total number of testing patterns N was also varied according to the number of *initially* accommodated classes (digits/letters).

3.2 Simulation Results & Discussion

In Figs. 2 (a)-(d), each of the four lines shows the transition of the deterioration rate (defined in (4)) obtained by varying the number of new classes (digits/letters) accommodated within the original PNN. In each figure, the label 'Digit i-j' (or 'Letter i-j' for ISOLET) indicates that the PNN was initially configured with the pattern vectors for only the classes from Digit/Letter i to j. For all the data sets, the overall generalization performance (using the testing set) with the initial configuration remained satisfactory, i.e., within the range from 90.4% to 100.0%.

For all the cases, a similar tendency was observed; as the number of new classes is increased, the generalization performance deteriorates. This naturally follows, since the number of degrees

of freedom is also increased by adding new classes to be classified. However, as in Figs. 2 (a)-(d), this is also dependent upon the length of the pattern vectors and was confirmed by another set of simulations; with identical numbers of the patterns in both the training and testing sets (i.e., 200 for training and 300 for testing) for the three data sets, the SFS, OptDigit, and PenDigit (of which the number of classes is also identical), the overall deterioration rate of PenDigit was less than that of the SFS. In other words, this observation indicates that the coverage of pattern space by the centroids is accordingly broadened as the dimensionality is decreased.

In addition, for the PenDigit case (i.e., using the original large data set of PenDigit), a deterioration rate of around 14% was observed for 'Digit 1-2', by increasing the number of classes to three (though this was only such a case, and the result is not shown here). In this case, it can be said that 'over-training' may have occurred, due to the excessive amount of the training data, taking into account the length of each pattern vector (i.e., 16). This indicates that, as for other neural based pattern classifiers, pruning of the training data in advance is important for the training (or constructing) of PNNs (for a further discussion of this, see e.g., [13]).

Then, as shown in solid lines in Fig. 2, the deterioration rate of the initial configuration with the smaller number of classes (i.e., trained with either one or two classes only) was the highest for the three data sets, i.e., SFS, PenDigit, and ISOLET. By following the principle in Section 2.1, this can be interpreted such that the separation of the pattern space with a smaller number of classes is rather broad and thus is easily eroded by adding new classes. This erosion was noticeable in the case of ISOLET. However, it can also be said that the degree of erosion is bounded. In other words, the spread of the centroids is limited, since, as shown in Fig. 2, the deterioration rate remained the same when the number of classes was increased. In this regard, it is considered that the structure of the training data set for OptDigit is most well-balanced among the four, since the deterioration rate was low (which was less than 0.3%), whereas the generalization performance was relatively high, i.e., around 99.0% for all the initial conditions. In contrast, for SFS, the deterioration rate was rather steadily increased as the number of new classes for all the initial configurations (except the case 'Digit 1 only') was increased, in comparison with the other three data sets. This is perhaps due to the insufficient number of pattern vectors and thereby the weak coverage of the pattern space.

Nevertheless, it is stated that, by exploiting the flexible configuration property of a PNN, the separation of pattern space can be kept sufficiently well for each class even when adding new classes, as long as the amount of the training data is not excessive for each class. Then, as discussed above, this is supported by the empirical fact that the generalization performance was not seriously deteriorated for almost all the cases. It can therefore be concluded that any 'catastrophic' forgetting of the previously stored data due to accommodation of new classes did not occur.

4 Conclusion

In this letter, the capability of PNNs in accommodating new classes has been addressed. Through the simulation study using four different domain data sets for pattern classification tasks, it has been shown that accommodating new classes within a PNN does not cause serious corruption of the previously constructed pattern space. Future work includes the investigation of smart algorithms to update the radii values to meet the hardest Criterion 2) in [7].

Acknowledgment

The author would like to thank both the reviewer and associate editor for their encouraging comments. The author would also like to thank Dr. Danilo Mandic of Imperial College London for the helpful comments.

References

[1] D. F. Specht, "Probabilistic neural networks," *Neural Networks*, vol. 3, pp. 109-118, 1990.

[2] D. F. Specht, "A general regression neural network," *IEEE Trans. Neural Networks*, vol. 2, no. 6, pp.568-576, Nov, 1991.

[3] S. Haykin, *Neural Networks: A Comprehensive Foundation*. New York:Macmillan, 1994.

[4] M. J. L. Orr, (1996, Apr.) *Introduction to radial basis function networks* [Online] Available www.cns.ed.ac.uk.

[5] P. D. Wasserman, *Advanced Methods In Neural Computing*, in Chapter 8, "Radial Basis-Function Networks" pp. 147-176, New York:Van Nostrand Reinhold, 1993.

[6] T. Hoya and J. A. Chambers, "Heuristic pattern correction scheme using adaptively trained generalized regression neural networks," *IEEE Trans. Neural Networks*, vol. 12, no. 1, pp.91-100, Jan. 2001.

[7] R. Polikar, L. Udpa, S. S. Udpa, and V. Honavar, "Learn++: an incremental learning algorithm for supervised neural networks," *IEEE Trans. Systems, Man, and Cybernetics - Part C: Applications and Reviews*, vol. 31, no. 4, Nov. 2001.

[8] G. A. Carpenter, S. Grossberg, and J. H. Reynolds, "ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network," *Neural Networks*, vol. 4, no. 5, pp. 565-588, 1991.

[9] T. Hoya, "Interpreting two psychological functions by a hierarchically structured neural memory model," *Proc. 6th Int. Symp. Distributed Autonomous Robotic Systems (DARS02)*, pp. 405-414, Fukuoka, Japan, June 2002.

[10] C. M. Bishop, *Neural Networks For Pattern Recognition*, Oxford, U.K.: Oxford Univ. Press, 1996.

[11] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," In D. E. Rumelhart and J. L. McClelland (Eds.), *Parallel distributed processing: explorations in the microstructure of cognition*, 1, Chapter 8, Cambridge, MA:MIT Press, 1986.

[12] M. Huckvale, *Speech Filing System Vs3.0 – Computer Tools For Speech Research*, London, U.K.:University College, Mar. 1996.

[13] T. Hoya, "Graph theoretic techniques for pruning data and their applications," *IEEE Trans. on Sig. Proc.*, vol.46-9, pp. 2574-2579, Sept. 1998.

Data Set	Length of Each Pattern Vector	Total Num. of Patterns in the Training Set	Total Num. of Patterns in the Testing Sets	Num. of Classes
SFS	256	599	400	10
OptDigit	64	3823	1797	10
PenDigit	16	7494	3498	10
ISOLET	617	1040	520	26

Table 1: Data sets used in the simulation study.

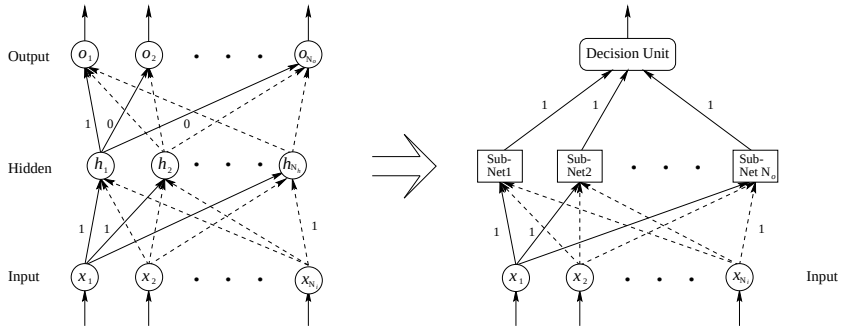
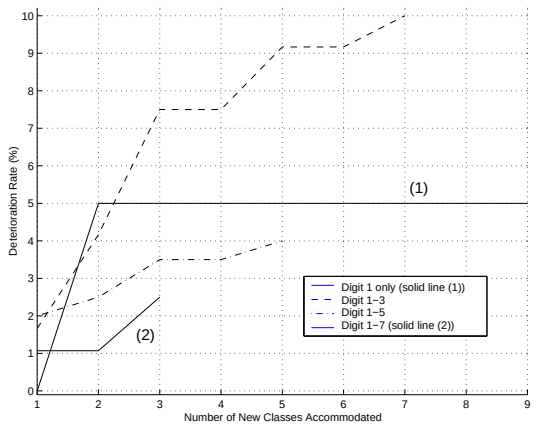
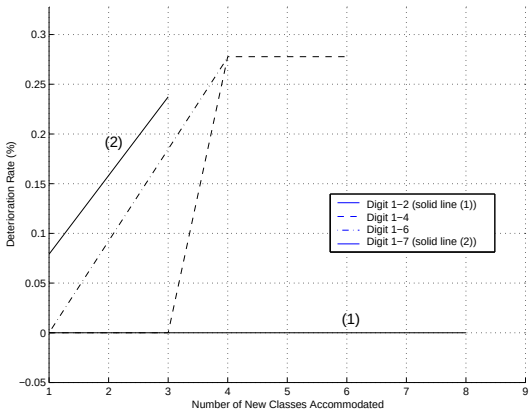


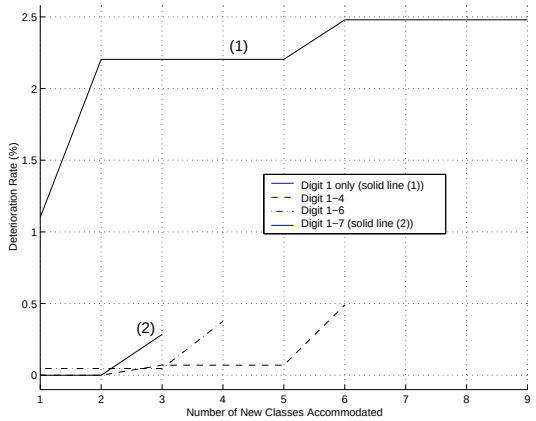
Figure 1: Illustration of topological equivalence between the three-layered PNN with N_h hidden and N_o output units and the assembly of the N_o distinct sub-networks.



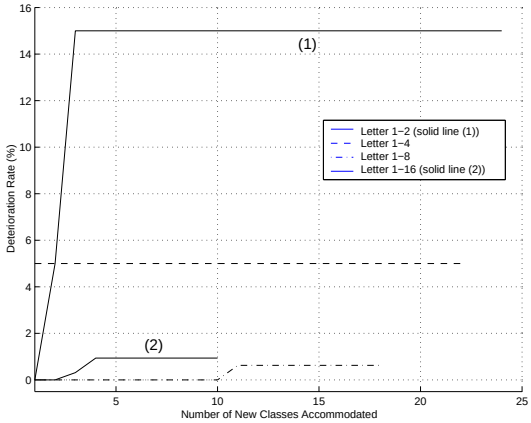
(a) SFS Data Set



(b) OptDigit Data Set



(c) PenDigit Data Set



(d) ISOLET Data Set

Figure 2: Transition of the deterioration rate with varying the number of new classes accommodated.