공학석사학위논문

Neural networks Ensemble for multi-dimensional classification

다중 클래스 분류 문제를 위한 인공신경망 앙상블

2016년 2월

서울대학교 대학원

컴퓨터공학부

박 상 철

Abstract

Neural networks Ensemble for multi-dimensional classification

Sangchul Park
Department of Computer Science and Engineering
College of Engineering
The Graduate School
Seoul National University

Artificial neural network (ANN) is devised from human neural system in the field of machine learning. Several techniques including ensemble have been proposed to improve ANN's performance. In this thesis, we show three types of ensembles for classification task which approaches in the style of divide and conquer: pairwise, hierarchy, helper. The pairwise ensemble is using binary classifiers to solve one classification problem. The hierarchical ensemble is composed of networks which are independently trained to solve each small part of the problem. And the other ensemble method uses a traditional network that is assisted by a helper, trained to solve easy classification that is modified from original classification. The experiments are conducted with the MNIST database and the results show the problem and possibility of the style of divide and conquer in neural network.

Keyword : Neural networks, Ensemble, Divide and conquer
Student Number : 2014-21790

Contents

List of tables

List of figures

Chapter 1

Introduction

Deep learning is the most attractive branch in machine learning. It is a set of algorithms to learn inherent features in data typically by using artificial neural networks. The neural networks in deep learning with multiple hidden layers have a lot of nodes in each layer. In general, learned features in multiple layers compose hierarchy, that is, higher layers extract high level features of hierarchy formed by the combination of lower level features of lower layers.

Training a deep neural network is not easy because of the computational cost of training multiple layers. But the development of computing power including GPU processing have reduced it to an acceptable level and thus deep learning became much practical. Consequently, it has interested many researchers and it has brought remarkable progressions.

Another reason that many computer scientists were skeptical about deep learning is that conventional training methods such as backpropagation with stochastic gradient descent are not effective enough to train a deep neural network. As the number of layers increases in a network, in particular, traditional stochastic gradient descent is not enough to optimize a number of model parameters. To

overcome this problem, effective training strategies for deep neural networks using unsupervised pre-training appeared (e.g., deep belief networks (DBN) and stacked auto-encoders) [1, 2]. Both are based on a similar approach to train networks, greedy layer-wise unsupervised pre-training, before supervised fine-tuning. This approach had improved the performance of networks. Because of its extra training time cost, it is considered unnecessary and other better ways to train networks have arose.

To increase the effectiveness of training in deep learning, several techniques such as regularizations and adaptive learning rate skills are widely used. Adding L1 or L2 norm to the loss function which gradient descent optimization methods aim to minimize is the typical regularization. In principle, adding a regularization term to the loss function forms a model having smooth solution by penalizing large values in parameters. Dropout is the useful regularization for the purpose of preventing a model from overfitting [3]. Also, ADADELTA is a method varying learning rate adaptively during training [4].

Unlike training methods described above, there is another approach called ensemble which trains multiple networks and produces a result by mixing outcomes from each network. Ensemble can bring stability and even better performance to the result.

In this thesis, we show three ensemble methods applying divide and conquer style and their performance on MNIST database. The first one is pairwise ensemble. That is using independent binary classifiers to solve one classification problem. The second one is hierarchical ensemble which solves classification problem stage by stage by using networks trained for each specific stage. The last one is ensemble that uses a helper. Helper is a network solving easy problem modified from original one. The helper assists the original network to predict labels

of inputs.

In chapter 2, we will briefly look into artificial neural networks, ensemble and several methods used in our experiments to improve performance. Chapter 3 describes ensemble methods which we tried. Chapter 4 shows the results of experiments on MNIST of those ensemble methods. Finally, we will discuss the limitations and possibilities of future work related to them.

Chapter 2

Preliminaries

In this chapter, we briefly look into artificial neural networks, ensemble and some of training methods which are applied to all networks in our experiment: rectified linear unit, dropout and ADADELTA.

2.1 Artificial neural networks

In machine learning, artificial neural networks are a family of models inspired by biological neural networks. A network is composed of an input layer, multiple hidden layers and an output layer. Figure 2.1 shows one example.

Commonly, a network is trained by supervised learning. That is, the model is trained to classify the class label of given input. To perform it, during training, the adjustable parameters of a network are modified by learning algorithm such as stochastic gradient descent. This algorithm calculate a gradient vector, for each weight vectors, in the direction of decreasing the measure of error, called objective function. The gradient vector then is adjusted to weight vector. This learning process keeps doing until the measured error is not decreasing anymore [8].

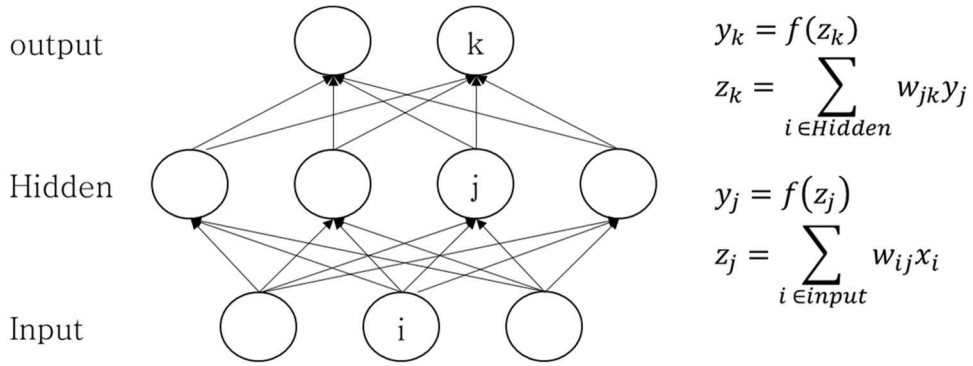

Figure 2.1: An example of artificial neural networks.

2.2 Ensemble

Ensemble is a learning paradigm where a collection of a finite number of neural networks is trained for the same task [9]. Hansen and Salamon showed that combined predictions of a number of networks gives improved generalization ability to a network system [10]. Two steps are needed to ensemble, i.e. training a number of networks and combining them.

The most prevailing approaches to combine networks are voting and majority voting [10], simple averaging [11] and weighted averaging [12]. For example, simple averaging method uses the average of outcomes of all networks composing one ensemble system. Let o_i be an outcome probability vector of i-th network where $1 \leq i \leq n$, then the final result vector O is $\frac{1}{n}\sum_1^n o_i$.

The style of ensemble methods in this thesis is different from prevailing approaches. Those methods are adopting the divide and conquer strategy. That is, the networks are trained for different sub-tasks instead of for the same task. This approach is performed in [13, 14, 15, 16, 17] as well.

2.3 Rectified linear unit

In 2011, Xavier Glorot has showed that rectified linear unit (ReLU) is more efficient to train deep networks [5]. The activation function of ReLU is defined as

$$f(x) = \max(0, x)$$

where x is the input to a unit. This means that a node accept only positive values as their input. When this activation function is compared to commonly used activation functions such as hyperbolic tangent and sigmoid, it has some advantages followed:

- Sparse activation
- Only comparison

With ReLU, only about 50% of units are activated, i.e. an input value of unit is over 0. Therefore sparse representations are possible. Also, computational cost from calculation of activation is lower than the one from common activation functions.

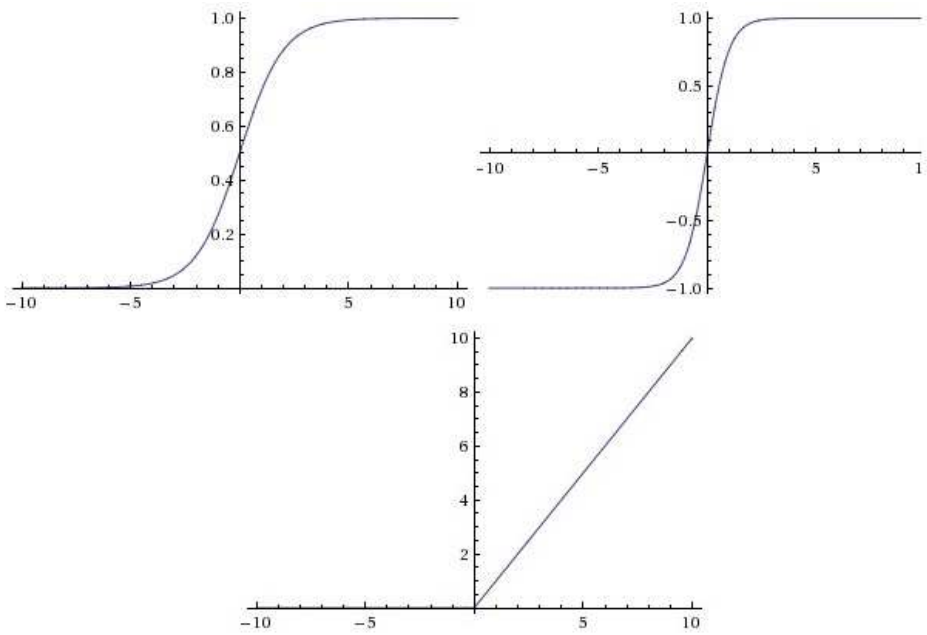

Figure 2.2: The activation functions of neural network. The top left and right are sigmoid function, hyperbolic tangent function. The lower is rectifier.

2.4 Dropout

Dropout, proposed in [6], is widely known as a powerful regularization. With dropout, the probabilities of retaining a hidden unit are assigned to all layers in a network while training (all the layers have their own probabilities). For example, if a layer has the probability $p = 0.5$, then each unit respectively has chance to update its own weight values with $p = 0.5$ in every training iteration. After training phase, whole hidden units are used without dropout. By doing this, a network acquires robustness to overfitting.

The experiments in [6] showed that dropout certainly improves performance by preventing overfitting. They tested networks with and without dropout on MNIST classification for comparison. The test

error of networks with dropout are better while the train error are similar.

2.5 ADADELTA

Learning rate controls the step-size to take in the direction of the negative gradient. Choosing an efficient learning rate is important for training a network because it is directly related to the speed of training. For instance, using a large learning rate would be better than using small one in a training point where a model can dramatically advance. Also, the step-size is critical to escape from a poor local minima. In those reasons, many studies have been done to set a good learning rate.

ADADELTA is one of the flexible learning rates which keep changing in training phase. A model calculates its learning rate adaptively in every iterations so that its step-size efficiency can be maximized. Basically, ADADELTA accumulates updated data of parameters and uses it to calculate the current learning rate. That is, as looking into the past learning history, we can compute the adaptive learning rate in current iteration. The whole algorithm of ADADELTA and details are in [9].

Those methods in section 2.3, 2.4, 2.5 are all for efficient training. And they are applied over all networks in our experiments.

Chapter 3

Ensemble

In this chapter, we will show three different ensemble methods: Pairwise, Hierarchy, Helper. Similar works with pairwise have done in [15, 16]. And [13, 14, 17] are using the hierarchy in their machine learning algorithm. Helper is proposed and implemented at the first time in this thesis.

The following words means the network which is trained to classify all $\mathtt{N}$ classes at a time: traditional big network, big network.

3.1 Pairwise

Pairwise ensemble is using $N(N-1)/2$ binary classifiers of every class pair of N class classification, also called One-vs-One (OvO) strategy (there is a similar approach called One-vs-All (OvA). A network of this strategy classifies that an input is in a specific class or in the others). Commonly, this is implemented with support vector machine (SVM) because SVM is originally designed to solve two dimensional classification problem. However, in our experiments, we have used modern neural networks, i.e. networks with modern training techniques explained in section 2.3, 2.4, 2.5.

To implement, we first train $N(N-1)/2$ binary classifiers for each pair and then we combine them by methods such as majority voting and weighted voting. In majority voting, each of the binary classifier votes one for its favored class, and finally the class with maximum votes wins. In weighted voting, each of the binary classifier votes the probability value, instead of one, of its favored class (the result output vector is composed of possibilities for each class, i.e. the sum of all values of result vector equals one).

Majority voting can cause a problem that multiple classes have the equivalent number of votes (weighted voting can also cause this problem, however it barely happens). To handle this problem, there exist a method that we select one which has the maximum sum of the probability values from all networks among those classes (i.e. we use same method with weighted voting only in those cases).

Each class is involved to $N-1$ networks as a possible output of them. And all they are experts when they are compared to traditional big network. Hence, intuitively, we can expect high accuracy from this ensemble method if the $N-1$ networks related to a specific class would vote for the right class strongly.

Additionally, to overcome trivial problem of pairwise ensemble, that one network still votes even if the class of the given input is not related to them, we tried pairwise ensemble of networks that each has three output nodes where two are for classes and one is for abstention. This is possible because neural network can have output nodes over two.

3.2 Hierarchical ensemble

In 2009, [17] proposed support vector machine classifiers utilizing binary decision tree (SVM-BDT). In the tree (the hierarchy), each internal node is a trained SVM to solve the subtask assigned to itself, and leaves are classes. The subtask assigned to an internal SVM is different from the other subtasks assigned to the other SVMs, and all subtasks together are composing the given classification problem. To make the tree, they used Euclidean distance between centers that is calculated for the N different classes. In this section, we propose a new method to make the shape of the tree. And then we show three shapes of the hierarchy composed of neural networks instead of SVMs.

Before explain the method to make the hierarchy, to understand this approach, let's take an example of the hierarchy described in Figure 3.1. A set of all possible classes of all samples are $\{A, B, C, D\}$. If an input is given into the root network, that is classified whether the class label of the given input is involved in subset $\{A, C\}$ or $\{B, D\}$. And then that is classified again by the node of the previously selected subset. And finally, the given input is classified when that reaches one of the leaves.

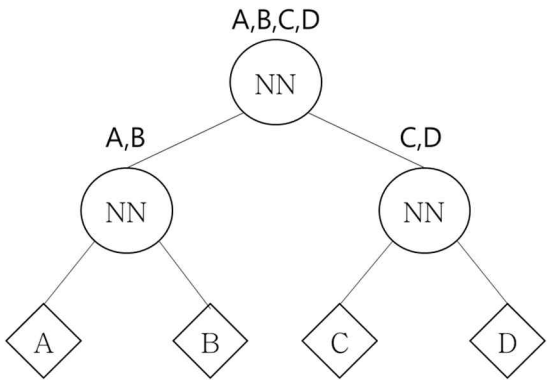

Figure 3.1: An illustration of the hierarchy.

Grouping classes of upper node to two disjoint subsets is critical for good performance of this ensemble model. To achieve this, we use the error information from the traditional big network trained earlier. We draw an undirected graph with error data from the big network. In the graph, each node is a specific class, and a value of an edge between connected two nodes is connectivity by error data. The weight of edge between two classes A and B is the sum of the number of misclassified inputs of A as B and the number of misclassified inputs of B as A by the big model. After we draw the graph, divide the graph by min-cut algorithm, because we want to reduce the amount of error occurred between two groups by a trained expert network. Until all classes remains alone, repeatedly cut the subgraphs divided from previous step. Then we train networks corresponding to all the cuts.

In this architecture, reducing the height of the hierarchy is of advantage for good performance. Because the probability of that an input is correctly classified is calculated as the product of all probabilities of networks in the path where the input pass through (i.e. the probability of that the given input of class A is correctly classified equals $\prod_{i \in path} p_i$ where p_i is the probability of that network i classifies A correctly). Therefore, we tried two balanced form of the hierarchy.

Figure 3.2 shows the hierarchy of MNIST handwritten digit recognition database made by the method explained above: hierarchy divided by min-cut (MCH). To make it balanced, the earlier divided five classes $0,6,1,2,7$ are merged into one subset of the root node. And then we recursively have done same work including the dividing method explained above and merging the classes divided earlier, until all classes remain alone. The top of the Figure 3.3 shows the result: balanced hierarchy (BH).

In addition to the balanced hierarchy, we made the hierarchy which has lower height: 2 level hierarchy (2LH). This architecture only uses the root network of the balanced network, and samples passed the root network are classified by the networks, which is trained to classify 5 classes of each subset divided by the root network at a time. This is possible because a neural network can have the length of the output over two. The bottom of the Figure 3.3 shows 2LH.

This ensemble architecture has the expected advantage, i.e. the tasks for each network easier than classifying all classes at a time. On the other hand, disadvantages are also exist such that lower networks never get a chance to classify samples misclassified by the upper networks.

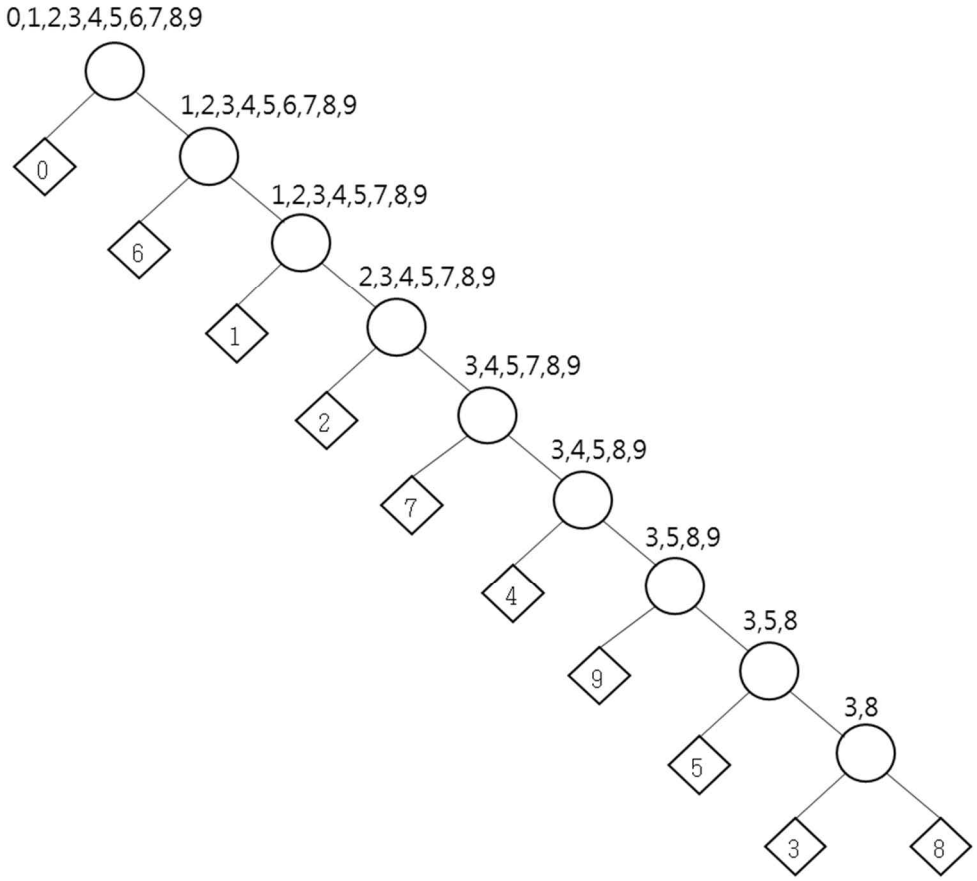

Figure 3.2: The hierarchy of MNIST database.

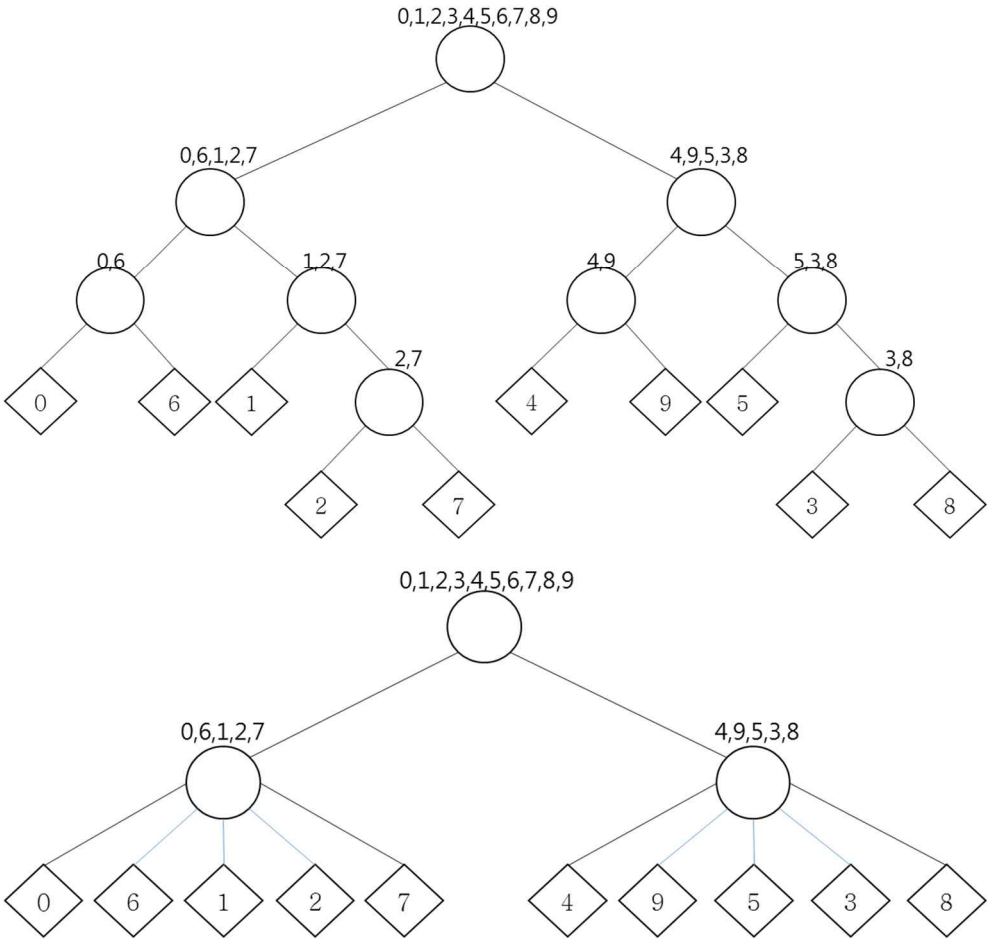

Figure 3.3: Two types of balanced hierarchies.

3.3 Helper

In the above ensemble methods, individual networks are trained to solve the small tasks which are composing one classification problem. That is, no one network can solve the given classification problem alone. However, this ensemble method is composed of a big network and the root network (the helper) of the balanced hierarchy.

Conceptually, the root network solves the classification problem that is modified from original problem, and both problems involve all samples of all classes. If we success to modify the problem to easier one, the root network can result in better accuracy. Purpose of this ensemble is to give the information from the root network to a big network so that the big network can classify better.

We tried following method to implement this. Let o is the output vector from a big model where each element is the probability value of one class. And let p is the output vector from a helper. Then we multiply a constant α to the probability value of the group of classes selected by the helper. And then add the value to the each of elements of o except elements which are not in the selected group. For example, assume the result of a big model is $o = [0.0, 0.1, 0.0, 0.0, 0.0, 0.0, 0.0, 0.6, 0.0, 0.4]$ where each element of this vector is the probability value for classes 0,1,2,3,4,5,6,7,8,9 in order. And assume that we use the root network in Figure 3.3 and that produced the output vector $p = [0.0, 1.0]$. If α is set as 0.9, the final result vector becomes $r = [0.0, 0.1, 0.0, 0.9, 0.9, 0.9, 0.0, 0.6, 0.9, 1.3]$. Then the given input is classified as class 9 which has maximum value among all classes. If the class of the given input in this example is 9, the helper contributes to the big network classifying correctly.

Likewise, by doing this, the number of misclassification crossing the groups could decrease if following conditions are satisfied. First, those cases exist. Second, the helper classifies correctly in those cases. Third, the big network can correctly classify among classes in the group selected by the helper.

Chapter 4

Experiments

4.1 MNIST database

MNIST handwritten digit recognition database is the one of the popular image data set for testing neural networks. Each example of the database is classified as one of digits from 0 to 9. It has a training set of 60,000 examples and a test set of 10,000 examples, an image in it has 28x28 size. Traditionally, 10,000 examples of the training set are used to validate a model and is called validation set. The validation set is not used to train a network. Instead, the validation set gives a chance to stop training at the right point where one can prevent a model from overfitting. Details of MNIST database are in [7] and this site provides the performance record as well.

Standardization is always used in every network in our experiment. By doing this, we can obtain the values of parameters in balance.

4.2 Experimental Setup

We have used the Theano library which is the most preferred artificial neural network library. Training techniques mentioned in chapter 2 are added to the library. Thus, our networks adopt ReLU, dropout,

ADADELTA additionally with other options such as L2 norm which theano basically provides.

Each network has three hidden-layers and each layer has 2,000 hidden units. In the hierarchical ensemble, the helper ensemble, training iteration is limited to 2,000 such that one training iteration means that a model looks into all 50,000 training examples at once. And that of the pairwise ensemble is limited to 500. Because the number of required networks for this ensemble is larger than others (also, they converge faster because they handle only two classes). Early-stop condition is used to prevent overfitting by using validation set.

4.3 The result of pairwise ensemble

Table 4.1 shows the result of pairwise ensemble. OVO-SVM is the result of pairwise ensemble of SVMs in [16]. OvO-NN is pairwise ensemble of neural networks, and OvOvA-NN is ensemble of networks which has one more output node for abstention. In OvO-NN, we used weighted voting to avoid the equivalent vote results problem occurred by majority voting. Two pairwise classifications gives similar results, however the performance of OvOvA is clearly better than both of them.

Table 4.1: The result of pairwise ensemble.

Ensemble model	Test error (%)
OvO-SVM	1.70
OvO-NN	1.69
OvOvA-NN	1.31

As discussed about pairwise ensemble in section 3.1, one class can be strongly voted by $N-1$ networks which has the class as a possible output, because those $N-1$ networks are all specialists to classify the class. However, it often has the opposite effect. For example, there exist a misclassified input of class 7 as class 9. According to the voting data, all the nine networks related to class 9 voted for class 9 and only eight networks related to class 7 voted for class 7. That is, the network, which has two classes $7,9$ as its possible output classes, misclassified the given input of class 7 as class 9. This means that the network was critical to classify the given input correctly. In short, if an input of class A is given, that has very similar features with another class B, the network having A,B as its outputs takes a critical role to classify correctly the given input. And actually, many misclassified samples of our experiment are in this case. For example, in one specific OvO-NN ensemble system of our experiment, whose test error is 1.70%, 94 of 170 are in this case.

On the other hand, because all networks of OvOvA-NN are trained to avoid those cases, the performance is significantly improved. A specific OvOvA-NN ensemble system of our experiment correctly classifies 41 samples of those 94 examples.

Consequently, normal pairwise ensemble with binary classifiers have limitation from the serious disadvantage that networks, which are not for the class of the given input, disturb the right networks. And that can be considerably improved by training networks which have one more output node for abstention.

4.4 The result of hierarchical ensemble

Table 4.2: The result of hierarchical ensemble.

Ensemble	Test error (%)
MCH	2.13
BH	1.56
2LH	1.35
SVM-BDT	2.45

Table 4.2 shows the result of this ensemble method with the performance of previous result from [17] with SVM-BDT. The performance of MCH is clearly better than that of SVM-BDT.

For the sake of fairness, we reproduced SVM-BDT with neural networks which are same with those of our ensemble models (because better performance of MCH is contribution from environment). We made the tree of MNIST database by algorithm explained in [17] and then trained networks needed. The test error of reproduced SVM-BDT with neural networks is 2.25%, this is still lower accuracy than MCH.

The result also implies that as the height of the hierarchy decreases, the performance becomes better. The performance of BH is better than MCH, and that of 2LH is the best. This is because of that one given input is correctly classified only when all the networks in the path of the input correctly classify. Therefore, it is better to keep the height of the hierarchy low.

To improve this ensemble more, there exist some issues followed:

- The form of hierarchy might not be the best: it is hard to find the best form of hierarchy.
- Optimizing all hyper parameters of several networks is burden.

There exist many ways to group classes and to make the form of hierarchy. We used the algorithm explained in section 3.1 to make the form of hierarchy, however, there could be better method than this.

Setting hyper parameters such as number of training iterations, network size, batch-size, and others is very important to train a network efficiently. We have to optimize all the networks used in hierarchy respectively. In short, we have to try various setting of all networks and it causes heavy computational cost.

For those reasons, using this ensemble method are more difficult than just training the big model. But it is still worth to work with this ensemble because this has possibility to improve.

4.5 The result of helper

In the hierarchical ensemble, only the root node use all training samples which is the same amount with the big network. Interesting point is the performance of the root network. That is, because the root network solves the problem to divide all samples into two disjoint groups of classes, accuracy is higher than the big network which is used to make groups. Although the performances of them have different meanings, the knowledge of the root network may be useful. Table 4.3 shows the performances of the big network and the root network of hierarchy.

Table 4.3: The big network and the root network.

Classifier	Test error	Classifiy
The root network	0.47	[0,6,1,2,7],[4,9,3,5,8]
The big network	1.12	[0],[1],[2],[3],[4],[5],[6],[7],[8],[9]

There exist many ways to transmit the information from the root to the big network. We have used the method explained section 3.3. The experiment is conducted while increasing the constant α which is multiplied to the probability value of the group selected from the root network. Figure 4.1 shows the result of experiment.

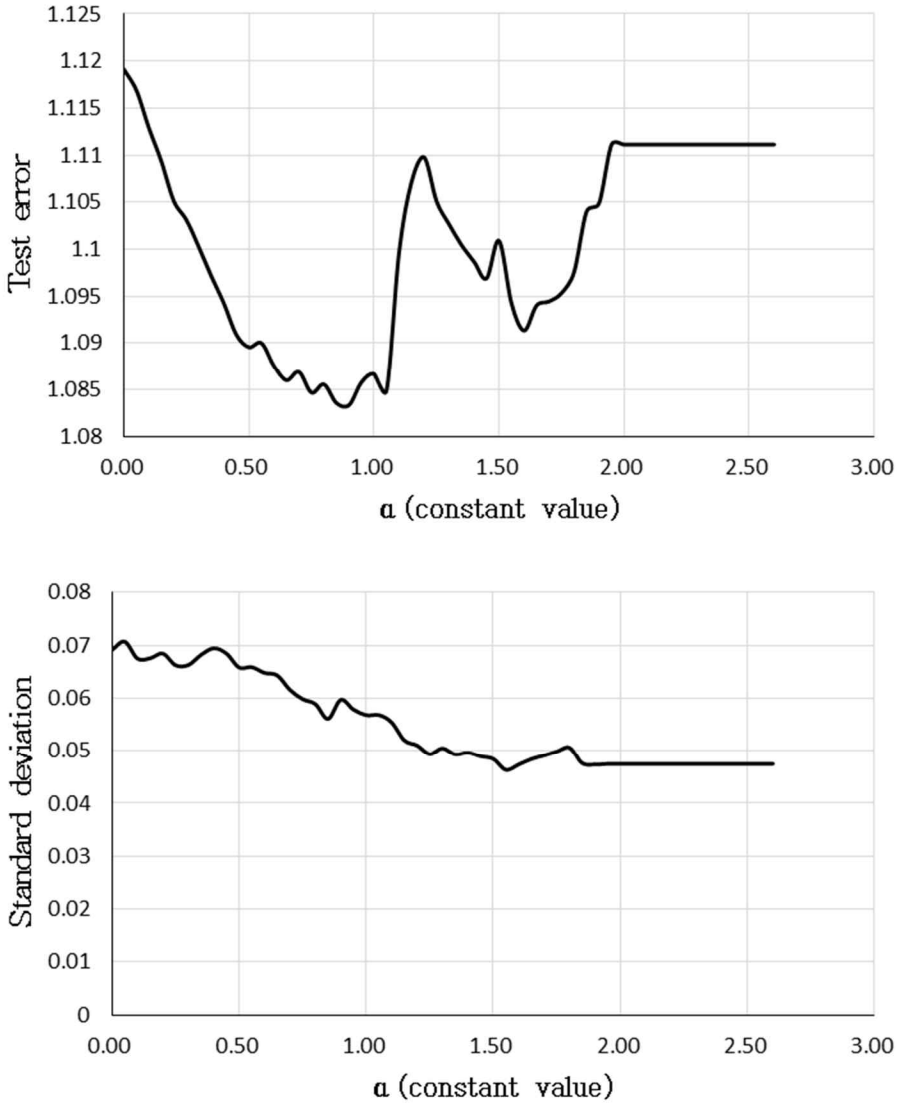

Figure 4.1: The result of helper ensemble.

In the figure, the top graph is the test error of this ensemble model and the bottom graph is the standard deviation (we used 50 randomly initialized models). The result reveals that the accuracy of the ensemble increases as α increases until that becomes 0.9 which is the best case (the value 0.9 of best case can vary depending on the environment). In the best case, the big network classified about four more samples on average. And since α is over 2.0, the performance doesn't change anymore. That is because, after the helper dominates to select the half of classes, the big network select one specific class only among those 5 classes.

In the bottom graph, we also can see that the standard deviation decreases as the constant α increases. Contrary to the accuracy, this keeps to decreases until the helper dominates to select the half. This means that the helper ensemble is more stable than using the big network alone while the accuracy is maintained (the performance of the ensemble is always better than when $\alpha = 0.0$).

To reveal the reason of this result, we analyzed the change in the misclassification samples between the big network and the helper. For example, in one specific example, the big network misclassifies 107 test samples and the helper misclassifies 99. Therefore, the total benefit of the helper ensemble is 8. This is the difference between the benefits and the losses occurred by using this ensemble, i.e. the benefits from the helper are 13 and the losses are 5.

The benefits occur when the big network alone selects the class among wrong half of all classes. The helper makes the big network look into the right 5 classes, and then the big network selects a specific class which has maximum probability value among those 5 classes. Nevertheless this happens, the big network can misclassify when the right class doesn't have the maximum value among those 5 classes.

However this is not the loss because the big network cannot classify correctly alone as well. On the other hand, the losses happen when the helper leads the big network, which can classify correctly alone, to the wrong group.

In short, the change on the result only occurs when the big network changes the group which that selects because of the principle of the combining method in this ensemble. And according to the result, the changes for advantage are over those for disadvantage on average. That is, because the ability to select the right group of the helper is better than the one of the big network, the benefits are over the losses with a high probability.

Consequently, when we consider that all the networks are given enough amount of the training time for themselves, this is meaningful. Because this result implies that we can make the performance better with a high probability, if we can train a network which solves one modified from original classification problem better than traditional networks and then use the information of the network in the right direction.

Table 4.4: All results of the experiments.

Model	Test error (%)	STDEV
Traditional network	1.12	0.069
OvO-SVM	1.70	none
OvO-NN	1.69	0.027
OvOvA-NN	1.31	0.027
MCH	2.13	0.140
BH	1.56	0.070
2LH	1.35	0.092
SVM-BDT	2.45	none
Helper	1.08	0.059

Chapter 5

Conclusion

Table 4.4 shows all the results from our experiments. We showed the limitation of pairwise ensemble (OvO) through analysis of misclassified samples and then proposed additional abstention node as one way to overcome the problem in itself. And we proposed a new method to form the tree (the hierarchy) in hierarchical strategy of classification. The result reveals that the performance of the hierarchy made by the method is better than one proposed in [17]. Also, we showed the possibility of combining traditional networks with the network which solves modified problem for better performance.

These three ensembles still have room for improvement. There could exist better methods to divide or modify the given problem. And also ensembles of the style of divide and conquer could increase as the number of all classes increase. Testing three ensembles in this thesis on other databases is one of the possible future works.

Bibliography

[1] Hinton, Geoffrey E., et al. "A fast learning algorithm for deep belief nets." *Neural computation* 18.7 (2006): 1527-1554.

[2] Bengio, Yoshua, et al. "Greedy layer-wise training of deep networks."*Advances in neural information processing systems* 19 (2007): 153.

[3] Hinton, Geoffrey E., et al. "Improving neural networks by preventing co-adaptation of feature detectors." *arXiv preprint arXiv:1207.0580* (2012).

[4] Zeiler, Matthew D. "ADADELTA: An adaptive learning rate method." *arXiv preprint arXiv:1212.5701* (2012).

[5] Glorot, Xavier,, et al. "Deep sparse rectifier neural networks." *International Conference on Artificial Intelligence and Statistics*. (2011).

[6] Srivastava, Nitish, et al. "Dropout: A simple way to prevent neural networks from overfitting." *The Journal of Machine Learning Research* 15.1 (2014): 1929-1958.

[7] LeCun, Yann, et al. "The MNIST database of handwritten digits." *http://yann.lecun.com/exdb/mnist/* (1998). (accessed November 28, 2015)

[8] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." *Nature*521.7553 (2015): 436-444.

[9] Cannon, Alex J., and Paul H. Whitfield. "Downscaling recent streamflow conditions in British Columbia, Canada using ensemble neural network models." *Journal of Hydrology* 259.1 (2002): 136-151.

[10] Hansen, Lars Kai, and Peter Salamon. "Neural network ensembles." *IEEE Transactions on Pattern Analysis & Machine Intelligence* 10 (1990): 993-1001.

[11] Opitz, David W., and Jude W. Shavlik. "Actively searching for an effective neural network ensemble." *Connection Science* 8.3-4 (1996): 337-354.

[12] Perrone, M. P., and L. N. Cooper. "When networks disagree: Ensemble methods for neural networks for speech and image processing." (1993).

[13] Jacobs, Robert A., et al. "Adaptive mixtures of local experts." *Neural computation* 3.1 (1991): 79-87.

[14] Jordan, Michael I., and Robert A. Jacobs. "Hierarchical mixtures of experts and the EM algorithm." *Neural computation* 6.2 (1994): 181-214.

[15] Kreßel, Ulrich H-G. "Pairwise classification and support vector machines."*Advances in kernel methods*. MIT Press, (1999).

[16] Lei, Hansheng, and Venu Govindaraju. "Speeding up multi-class SVM evaluation by PCA and feature selection." *Feature Selection for Data Mining*(2005): 72.

[17] Madzarov, Gjorgji, Dejan Gjorgjevikj, and Ivan Chorbev. "A multi-class SVM classifier utilizing binary decision tree." *Informatica* 33.2 (2009).

요약

인간의 신경망으로부터 고안된 인공신경망 은 다중 클래스 분류 문제 등을 풀기 위해 적합한 기계학습 기술이다. 앙상블을 포함한 여러 가지 기술들이 인공신경망의 기능을 향상시키기 위하여 제안되어오고 있다. 이 논문에서는 분할정복 방법이 접목된 세 가지의 앙상블을 다룬다. 첫 번째는 이진 분류기들을 이용한 다중 클래스 분해 기법이다. 그리고 두 번째는 문제를 작은 문제들로 나누고, 계층적으로 풀어내는 계층적 앙상블 방법이다. 세 번째는 기존 문제를 변경해 난이도를 낮추고, 그것을 풀도록 학습한 인공신경망이 기존의 문제를 푸는 일반적인 인공신경망을 돕도록 하는 형태의 앙상블 방법이다. MNIST 데이타베이스에 실험한 결과 및 분석을 통해 인공신경망을 이용한 다중 클래스 분류 문제 해결에서 분할정복이 가진 문제점과 가능성에 대해 보인다.

Keyword : 인공신경망, 앙상블, 분할 정복
Student Number : 2014-21790

공학석사학위논문

Neural networks Ensemble for multi-dimensional classification

다중 클래스 분류 문제를 위한 인공신경망 앙상블

2016년 2월

서울대학교 대학원

컴퓨터공학부

박 상 철

Abstract

Neural networks Ensemble for multi-dimensional classification

Sangchul Park
Department of Computer Science and Engineering
College of Engineering
The Graduate School
Seoul National University

Artificial neural network (ANN) is devised from human neural system in the field of machine learning. Several techniques including ensemble have been proposed to improve ANN's performance. In this thesis, we show three types of ensembles for classification task which approaches in the style of divide and conquer: pairwise, hierarchy, helper. The pairwise ensemble is using binary classifiers to solve one classification problem. The hierarchical ensemble is composed of networks which are independently trained to solve each small part of the problem. And the other ensemble method uses a traditional network that is assisted by a helper, trained to solve easy classification that is modified from original classification. The experiments are conducted with the MNIST database and the results show the problem and possibility of the style of divide and conquer in neural network.

Keyword : Neural networks, Ensemble, Divide and conquer
Student Number : 2014-21790

Contents

List of tables

List of figures

Chapter 1

Introduction

Deep learning is the most attractive branch in machine learning. It is a set of algorithms to learn inherent features in data typically by using artificial neural networks. The neural networks in deep learning with multiple hidden layers have a lot of nodes in each layer. In general, learned features in multiple layers compose hierarchy, that is, higher layers extract high level features of hierarchy formed by the combination of lower level features of lower layers.

Training a deep neural network is not easy because of the computational cost of training multiple layers. But the development of computing power including GPU processing have reduced it to an acceptable level and thus deep learning became much practical. Consequently, it has interested many researchers and it has brought remarkable progressions.

Another reason that many computer scientists were skeptical about deep learning is that conventional training methods such as backpropagation with stochastic gradient descent are not effective enough to train a deep neural network. As the number of layers increases in a network, in particular, traditional stochastic gradient descent is not enough to optimize a number of model parameters. To

overcome this problem, effective training strategies for deep neural networks using unsupervised pre-training appeared (e.g., deep belief networks (DBN) and stacked auto-encoders) [1, 2]. Both are based on a similar approach to train networks, greedy layer-wise unsupervised pre-training, before supervised fine-tuning. This approach had improved the performance of networks. Because of its extra training time cost, it is considered unnecessary and other better ways to train networks have arose.

To increase the effectiveness of training in deep learning, several techniques such as regularizations and adaptive learning rate skills are widely used. Adding L1 or L2 norm to the loss function which gradient descent optimization methods aim to minimize is the typical regularization. In principle, adding a regularization term to the loss function forms a model having smooth solution by penalizing large values in parameters. Dropout is the useful regularization for the purpose of preventing a model from overfitting [3]. Also, ADADELTA is a method varying learning rate adaptively during training [4].

Unlike training methods described above, there is another approach called ensemble which trains multiple networks and produces a result by mixing outcomes from each network. Ensemble can bring stability and even better performance to the result.

In this thesis, we show three ensemble methods applying divide and conquer style and their performance on MNIST database. The first one is pairwise ensemble. That is using independent binary classifiers to solve one classification problem. The second one is hierarchical ensemble which solves classification problem stage by stage by using networks trained for each specific stage. The last one is ensemble that uses a helper. Helper is a network solving easy problem modified from original one. The helper assists the original network to predict labels

of inputs.

In chapter 2, we will briefly look into artificial neural networks, ensemble and several methods used in our experiments to improve performance. Chapter 3 describes ensemble methods which we tried. Chapter 4 shows the results of experiments on MNIST of those ensemble methods. Finally, we will discuss the limitations and possibilities of future work related to them.

Chapter 2

Preliminaries

In this chapter, we briefly look into artificial neural networks, ensemble and some of training methods which are applied to all networks in our experiment: rectified linear unit, dropout and ADADELTA.

2.1 Artificial neural networks

In machine learning, artificial neural networks are a family of models inspired by biological neural networks. A network is composed of an input layer, multiple hidden layers and an output layer. Figure 2.1 shows one example.

Commonly, a network is trained by supervised learning. That is, the model is trained to classify the class label of given input. To perform it, during training, the adjustable parameters of a network are modified by learning algorithm such as stochastic gradient descent. This algorithm calculate a gradient vector, for each weight vectors, in the direction of decreasing the measure of error, called objective function. The gradient vector then is adjusted to weight vector. This learning process keeps doing until the measured error is not decreasing anymore [8].

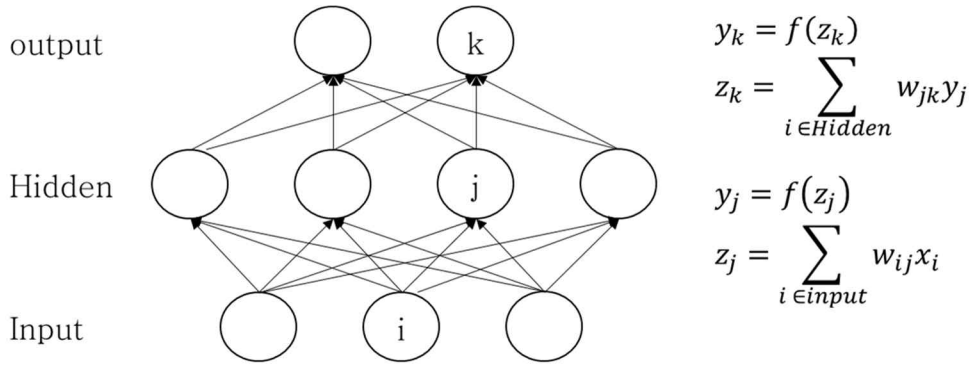

Figure 2.1: An example of artificial neural networks.

2.2 Ensemble

Ensemble is a learning paradigm where a collection of a finite number of neural networks is trained for the same task [9]. Hansen and Salamon showed that combined predictions of a number of networks gives improved generalization ability to a network system [10]. Two steps are needed to ensemble, i.e. training a number of networks and combining them.

The most prevailing approaches to combine networks are voting and majority voting [10], simple averaging [11] and weighted averaging [12]. For example, simple averaging method uses the average of outcomes of all networks composing one ensemble system. Let o_i be an outcome probability vector of i-th network where $1 \leq i \leq n$, then the final result vector O is $\frac{1}{n}\sum_1^n o_i$.

The style of ensemble methods in this thesis is different from prevailing approaches. Those methods are adopting the divide and conquer strategy. That is, the networks are trained for different sub-tasks instead of for the same task. This approach is performed in [13, 14, 15, 16, 17] as well.

2.3 Rectified linear unit

In 2011, Xavier Glorot has showed that rectified linear unit (ReLU) is more efficient to train deep networks [5]. The activation function of ReLU is defined as

$$f(x) = \max(0, x)$$

where x is the input to a unit. This means that a node accept only positive values as their input. When this activation function is compared to commonly used activation functions such as hyperbolic tangent and sigmoid, it has some advantages followed:

- Sparse activation
- Only comparison

With ReLU, only about 50% of units are activated, i.e. an input value of unit is over 0. Therefore sparse representations are possible. Also, computational cost from calculation of activation is lower than the one from common activation functions.

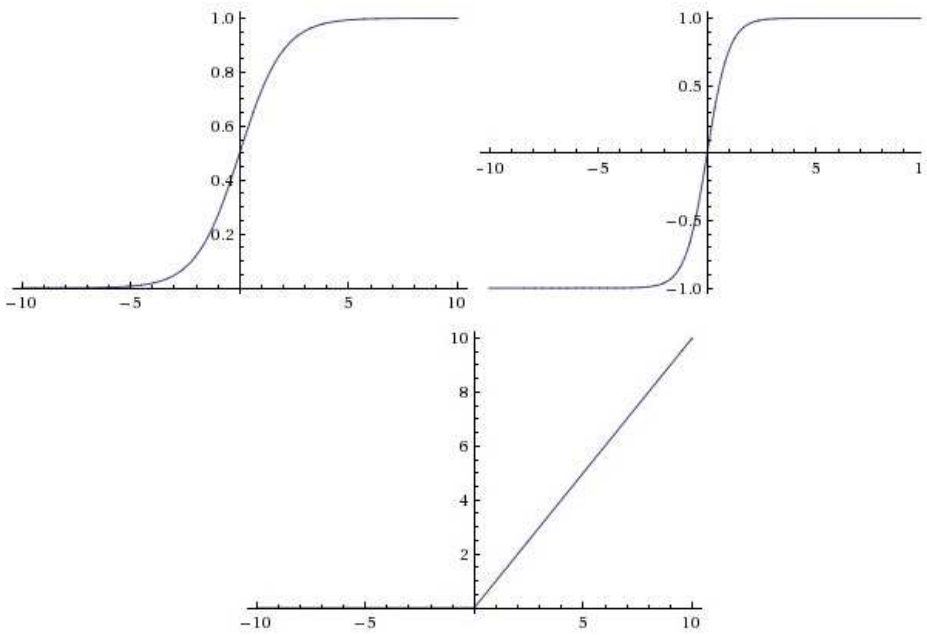

Figure 2.2: The activation functions of neural network. The top left and right are sigmoid function, hyperbolic tangent function. The lower is rectifier.

2.4 Dropout

Dropout, proposed in [6], is widely known as a powerful regularization. With dropout, the probabilities of retaining a hidden unit are assigned to all layers in a network while training (all the layers have their own probabilities). For example, if a layer has the probability $p = 0.5$, then each unit respectively has chance to update its own weight values with $p = 0.5$ in every training iteration. After training phase, whole hidden units are used without dropout. By doing this, a network acquires robustness to overfitting.

The experiments in [6] showed that dropout certainly improves performance by preventing overfitting. They tested networks with and without dropout on MNIST classification for comparison. The test

error of networks with dropout are better while the train error are similar.

2.5 ADADELTA

Learning rate controls the step-size to take in the direction of the negative gradient. Choosing an efficient learning rate is important for training a network because it is directly related to the speed of training. For instance, using a large learning rate would be better than using small one in a training point where a model can dramatically advance. Also, the step-size is critical to escape from a poor local minima. In those reasons, many studies have been done to set a good learning rate.

ADADELTA is one of the flexible learning rates which keep changing in training phase. A model calculates its learning rate adaptively in every iterations so that its step-size efficiency can be maximized. Basically, ADADELTA accumulates updated data of parameters and uses it to calculate the current learning rate. That is, as looking into the past learning history, we can compute the adaptive learning rate in current iteration. The whole algorithm of ADADELTA and details are in [9].

Those methods in section 2.3, 2.4, 2.5 are all for efficient training. And they are applied over all networks in our experiments.

Chapter 3

Ensemble

In this chapter, we will show three different ensemble methods: Pairwise, Hierarchy, Helper. Similar works with pairwise have done in [15, 16]. And [13, 14, 17] are using the hierarchy in their machine learning algorithm. Helper is proposed and implemented at the first time in this thesis.

The following words means the network which is trained to classify all $\mathtt{N}$ classes at a time: traditional big network, big network.

3.1 Pairwise

Pairwise ensemble is using $N(N-1)/2$ binary classifiers of every class pair of N class classification, also called One-vs-One (OvO) strategy (there is a similar approach called One-vs-All (OvA). A network of this strategy classifies that an input is in a specific class or in the others). Commonly, this is implemented with support vector machine (SVM) because SVM is originally designed to solve two dimensional classification problem. However, in our experiments, we have used modern neural networks, i.e. networks with modern training techniques explained in section 2.3, 2.4, 2.5.

To implement, we first train $N(N-1)/2$ binary classifiers for each pair and then we combine them by methods such as majority voting and weighted voting. In majority voting, each of the binary classifier votes one for its favored class, and finally the class with maximum votes wins. In weighted voting, each of the binary classifier votes the probability value, instead of one, of its favored class (the result output vector is composed of possibilities for each class, i.e. the sum of all values of result vector equals one).

Majority voting can cause a problem that multiple classes have the equivalent number of votes (weighted voting can also cause this problem, however it barely happens). To handle this problem, there exist a method that we select one which has the maximum sum of the probability values from all networks among those classes (i.e. we use same method with weighted voting only in those cases).

Each class is involved to $N-1$ networks as a possible output of them. And all they are experts when they are compared to traditional big network. Hence, intuitively, we can expect high accuracy from this ensemble method if the $N-1$ networks related to a specific class would vote for the right class strongly.

Additionally, to overcome trivial problem of pairwise ensemble, that one network still votes even if the class of the given input is not related to them, we tried pairwise ensemble of networks that each has three output nodes where two are for classes and one is for abstention. This is possible because neural network can have output nodes over two.

3.2 Hierarchical ensemble

In 2009, [17] proposed support vector machine classifiers utilizing binary decision tree (SVM-BDT). In the tree (the hierarchy), each internal node is a trained SVM to solve the subtask assigned to itself, and leaves are classes. The subtask assigned to an internal SVM is different from the other subtasks assigned to the other SVMs, and all subtasks together are composing the given classification problem. To make the tree, they used Euclidean distance between centers that is calculated for the N different classes. In this section, we propose a new method to make the shape of the tree. And then we show three shapes of the hierarchy composed of neural networks instead of SVMs.

Before explain the method to make the hierarchy, to understand this approach, let's take an example of the hierarchy described in Figure 3.1. A set of all possible classes of all samples are $\{A, B, C, D\}$. If an input is given into the root network, that is classified whether the class label of the given input is involved in subset $\{A, C\}$ or $\{B, D\}$. And then that is classified again by the node of the previously selected subset. And finally, the given input is classified when that reaches one of the leaves.

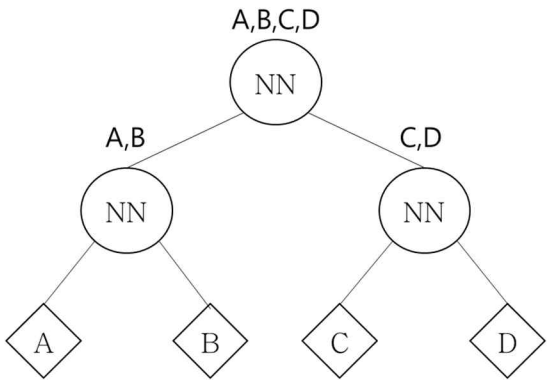

Figure 3.1: An illustration of the hierarchy.

Grouping classes of upper node to two disjoint subsets is critical for good performance of this ensemble model. To achieve this, we use the error information from the traditional big network trained earlier. We draw an undirected graph with error data from the big network. In the graph, each node is a specific class, and a value of an edge between connected two nodes is connectivity by error data. The weight of edge between two classes A and B is the sum of the number of misclassified inputs of A as B and the number of misclassified inputs of B as A by the big model. After we draw the graph, divide the graph by min-cut algorithm, because we want to reduce the amount of error occurred between two groups by a trained expert network. Until all classes remains alone, repeatedly cut the subgraphs divided from previous step. Then we train networks corresponding to all the cuts.

In this architecture, reducing the height of the hierarchy is of advantage for good performance. Because the probability of that an input is correctly classified is calculated as the product of all probabilities of networks in the path where the input pass through (i.e. the probability of that the given input of class A is correctly classified equals $\prod_{i \in path} p_i$ where p_i is the probability of that network i classifies A correctly). Therefore, we tried two balanced form of the hierarchy.

Figure 3.2 shows the hierarchy of MNIST handwritten digit recognition database made by the method explained above: hierarchy divided by min-cut (MCH). To make it balanced, the earlier divided five classes $0,6,1,2,7$ are merged into one subset of the root node. And then we recursively have done same work including the dividing method explained above and merging the classes divided earlier, until all classes remain alone. The top of the Figure 3.3 shows the result: balanced hierarchy (BH).

In addition to the balanced hierarchy, we made the hierarchy which has lower height: 2 level hierarchy (2LH). This architecture only uses the root network of the balanced network, and samples passed the root network are classified by the networks, which is trained to classify 5 classes of each subset divided by the root network at a time. This is possible because a neural network can have the length of the output over two. The bottom of the Figure 3.3 shows 2LH.

This ensemble architecture has the expected advantage, i.e. the tasks for each network easier than classifying all classes at a time. On the other hand, disadvantages are also exist such that lower networks never get a chance to classify samples misclassified by the upper networks.

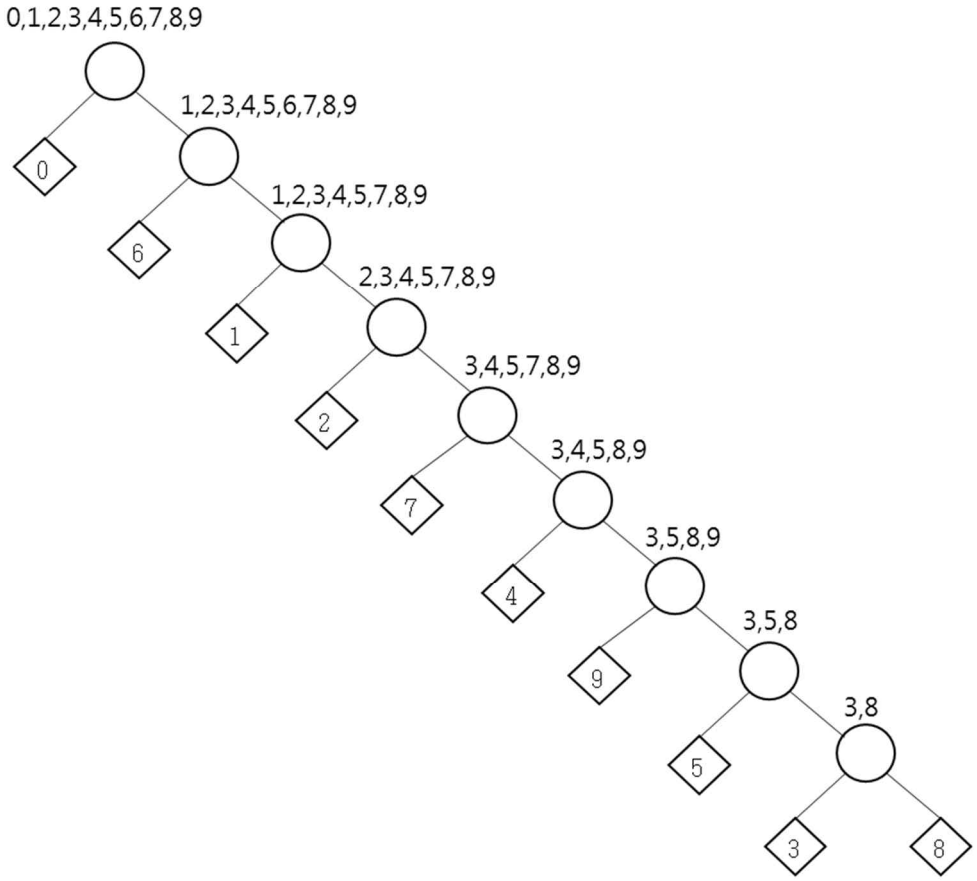

Figure 3.2: The hierarchy of MNIST database.

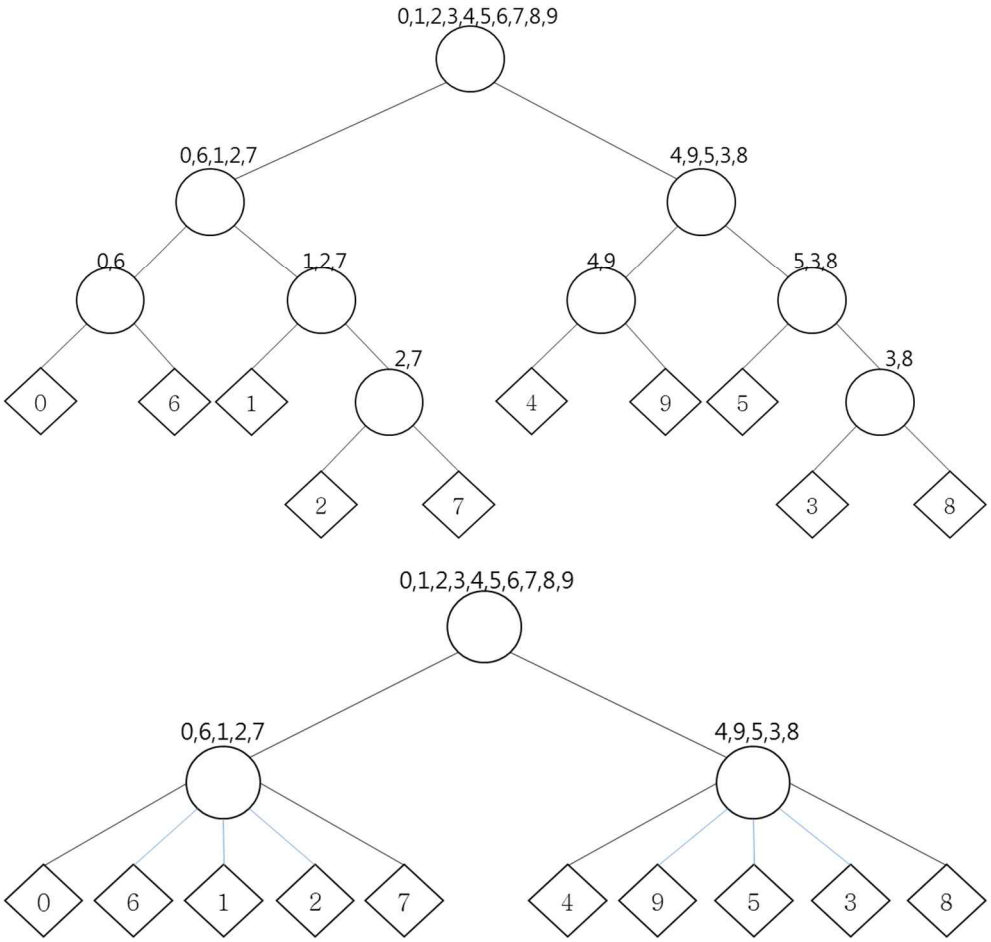

Figure 3.3: Two types of balanced hierarchies.

3.3 Helper

In the above ensemble methods, individual networks are trained to solve the small tasks which are composing one classification problem. That is, no one network can solve the given classification problem alone. However, this ensemble method is composed of a big network and the root network (the helper) of the balanced hierarchy.

Conceptually, the root network solves the classification problem that is modified from original problem, and both problems involve all samples of all classes. If we success to modify the problem to easier one, the root network can result in better accuracy. Purpose of this ensemble is to give the information from the root network to a big network so that the big network can classify better.

We tried following method to implement this. Let o is the output vector from a big model where each element is the probability value of one class. And let p is the output vector from a helper. Then we multiply a constant α to the probability value of the group of classes selected by the helper. And then add the value to the each of elements of o except elements which are not in the selected group. For example, assume the result of a big model is $o = [0.0, 0.1, 0.0, 0.0, 0.0, 0.0, 0.0, 0.6, 0.0, 0.4]$ where each element of this vector is the probability value for classes 0,1,2,3,4,5,6,7,8,9 in order. And assume that we use the root network in Figure 3.3 and that produced the output vector $p = [0.0, 1.0]$. If α is set as 0.9, the final result vector becomes $r = [0.0, 0.1, 0.0, 0.9, 0.9, 0.9, 0.0, 0.6, 0.9, 1.3]$. Then the given input is classified as class 9 which has maximum value among all classes. If the class of the given input in this example is 9, the helper contributes to the big network classifying correctly.

Likewise, by doing this, the number of misclassification crossing the groups could decrease if following conditions are satisfied. First, those cases exist. Second, the helper classifies correctly in those cases. Third, the big network can correctly classify among classes in the group selected by the helper.

Chapter 4

Experiments

4.1 MNIST database

MNIST handwritten digit recognition database is the one of the popular image data set for testing neural networks. Each example of the database is classified as one of digits from 0 to 9. It has a training set of 60,000 examples and a test set of 10,000 examples, an image in it has 28x28 size. Traditionally, 10,000 examples of the training set are used to validate a model and is called validation set. The validation set is not used to train a network. Instead, the validation set gives a chance to stop training at the right point where one can prevent a model from overfitting. Details of MNIST database are in [7] and this site provides the performance record as well.

Standardization is always used in every network in our experiment. By doing this, we can obtain the values of parameters in balance.

4.2 Experimental Setup

We have used the Theano library which is the most preferred artificial neural network library. Training techniques mentioned in chapter 2 are added to the library. Thus, our networks adopt ReLU, dropout,

ADADELTA additionally with other options such as L2 norm which theano basically provides.

Each network has three hidden-layers and each layer has 2,000 hidden units. In the hierarchical ensemble, the helper ensemble, training iteration is limited to 2,000 such that one training iteration means that a model looks into all 50,000 training examples at once. And that of the pairwise ensemble is limited to 500. Because the number of required networks for this ensemble is larger than others (also, they converge faster because they handle only two classes). Early-stop condition is used to prevent overfitting by using validation set.

4.3 The result of pairwise ensemble

Table 4.1 shows the result of pairwise ensemble. OVO-SVM is the result of pairwise ensemble of SVMs in [16]. OvO-NN is pairwise ensemble of neural networks, and OvOvA-NN is ensemble of networks which has one more output node for abstention. In OvO-NN, we used weighted voting to avoid the equivalent vote results problem occurred by majority voting. Two pairwise classifications gives similar results, however the performance of OvOvA is clearly better than both of them.

Table 4.1: The result of pairwise ensemble.

Ensemble model	Test error (%)
OvO-SVM	1.70
OvO-NN	1.69
OvOvA-NN	1.31

As discussed about pairwise ensemble in section 3.1, one class can be strongly voted by $N-1$ networks which has the class as a possible output, because those $N-1$ networks are all specialists to classify the class. However, it often has the opposite effect. For example, there exist a misclassified input of class 7 as class 9. According to the voting data, all the nine networks related to class 9 voted for class 9 and only eight networks related to class 7 voted for class 7. That is, the network, which has two classes $7,9$ as its possible output classes, misclassified the given input of class 7 as class 9. This means that the network was critical to classify the given input correctly. In short, if an input of class A is given, that has very similar features with another class B, the network having A,B as its outputs takes a critical role to classify correctly the given input. And actually, many misclassified samples of our experiment are in this case. For example, in one specific OvO-NN ensemble system of our experiment, whose test error is 1.70%, 94 of 170 are in this case.

On the other hand, because all networks of OvOvA-NN are trained to avoid those cases, the performance is significantly improved. A specific OvOvA-NN ensemble system of our experiment correctly classifies 41 samples of those 94 examples.

Consequently, normal pairwise ensemble with binary classifiers have limitation from the serious disadvantage that networks, which are not for the class of the given input, disturb the right networks. And that can be considerably improved by training networks which have one more output node for abstention.

4.4 The result of hierarchical ensemble

Table 4.2: The result of hierarchical ensemble.

Ensemble	Test error (%)
MCH	2.13
BH	1.56
2LH	1.35
SVM-BDT	2.45

Table 4.2 shows the result of this ensemble method with the performance of previous result from [17] with SVM-BDT. The performance of MCH is clearly better than that of SVM-BDT.

For the sake of fairness, we reproduced SVM-BDT with neural networks which are same with those of our ensemble models (because better performance of MCH is contribution from environment). We made the tree of MNIST database by algorithm explained in [17] and then trained networks needed. The test error of reproduced SVM-BDT with neural networks is 2.25%, this is still lower accuracy than MCH.

The result also implies that as the height of the hierarchy decreases, the performance becomes better. The performance of BH is better than MCH, and that of 2LH is the best. This is because of that one given input is correctly classified only when all the networks in the path of the input correctly classify. Therefore, it is better to keep the height of the hierarchy low.

To improve this ensemble more, there exist some issues followed:

- The form of hierarchy might not be the best: it is hard to find the best form of hierarchy.
- Optimizing all hyper parameters of several networks is burden.

There exist many ways to group classes and to make the form of hierarchy. We used the algorithm explained in section 3.1 to make the form of hierarchy, however, there could be better method than this.

Setting hyper parameters such as number of training iterations, network size, batch-size, and others is very important to train a network efficiently. We have to optimize all the networks used in hierarchy respectively. In short, we have to try various setting of all networks and it causes heavy computational cost.

For those reasons, using this ensemble method are more difficult than just training the big model. But it is still worth to work with this ensemble because this has possibility to improve.

4.5 The result of helper

In the hierarchical ensemble, only the root node use all training samples which is the same amount with the big network. Interesting point is the performance of the root network. That is, because the root network solves the problem to divide all samples into two disjoint groups of classes, accuracy is higher than the big network which is used to make groups. Although the performances of them have different meanings, the knowledge of the root network may be useful. Table 4.3 shows the performances of the big network and the root network of hierarchy.

Table 4.3: The big network and the root network.

Classifier	Test error	Classifiy
The root network	0.47	[0,6,1,2,7],[4,9,3,5,8]
The big network	1.12	[0],[1],[2],[3],[4],[5],[6],[7],[8],[9]

There exist many ways to transmit the information from the root to the big network. We have used the method explained section 3.3. The experiment is conducted while increasing the constant α which is multiplied to the probability value of the group selected from the root network. Figure 4.1 shows the result of experiment.

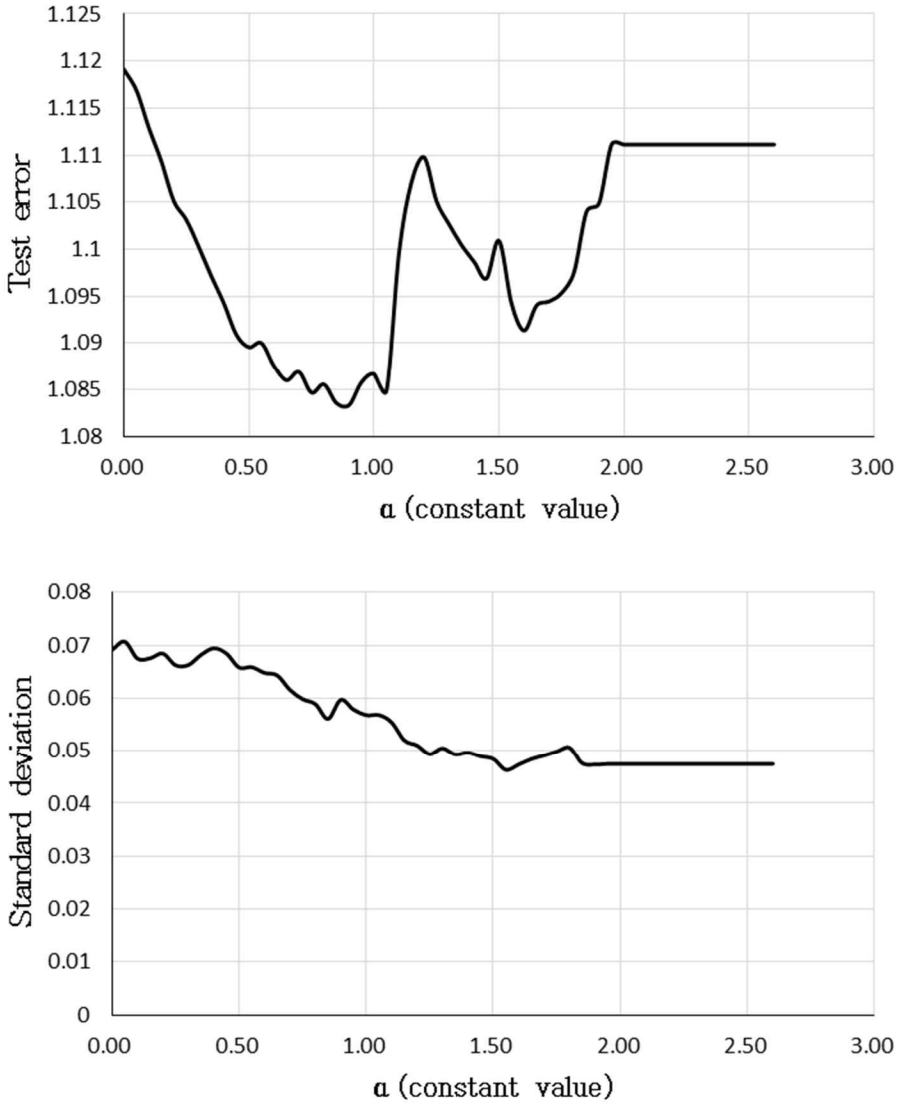

Figure 4.1: The result of helper ensemble.

In the figure, the top graph is the test error of this ensemble model and the bottom graph is the standard deviation (we used 50 randomly initialized models). The result reveals that the accuracy of the ensemble increases as α increases until that becomes 0.9 which is the best case (the value 0.9 of best case can vary depending on the environment). In the best case, the big network classified about four more samples on average. And since α is over 2.0, the performance doesn't change anymore. That is because, after the helper dominates to select the half of classes, the big network select one specific class only among those 5 classes.

In the bottom graph, we also can see that the standard deviation decreases as the constant α increases. Contrary to the accuracy, this keeps to decreases until the helper dominates to select the half. This means that the helper ensemble is more stable than using the big network alone while the accuracy is maintained (the performance of the ensemble is always better than when $\alpha = 0.0$).

To reveal the reason of this result, we analyzed the change in the misclassification samples between the big network and the helper. For example, in one specific example, the big network misclassifies 107 test samples and the helper misclassifies 99. Therefore, the total benefit of the helper ensemble is 8. This is the difference between the benefits and the losses occurred by using this ensemble, i.e. the benefits from the helper are 13 and the losses are 5.

The benefits occur when the big network alone selects the class among wrong half of all classes. The helper makes the big network look into the right 5 classes, and then the big network selects a specific class which has maximum probability value among those 5 classes. Nevertheless this happens, the big network can misclassify when the right class doesn't have the maximum value among those 5 classes.

However this is not the loss because the big network cannot classify correctly alone as well. On the other hand, the losses happen when the helper leads the big network, which can classify correctly alone, to the wrong group.

In short, the change on the result only occurs when the big network changes the group which that selects because of the principle of the combining method in this ensemble. And according to the result, the changes for advantage are over those for disadvantage on average. That is, because the ability to select the right group of the helper is better than the one of the big network, the benefits are over the losses with a high probability.

Consequently, when we consider that all the networks are given enough amount of the training time for themselves, this is meaningful. Because this result implies that we can make the performance better with a high probability, if we can train a network which solves one modified from original classification problem better than traditional networks and then use the information of the network in the right direction.

Table 4.4: All results of the experiments.

Model	Test error (%)	STDEV
Traditional network	1.12	0.069
OvO-SVM	1.70	none
OvO-NN	1.69	0.027
OvOvA-NN	1.31	0.027
MCH	2.13	0.140
BH	1.56	0.070
2LH	1.35	0.092
SVM-BDT	2.45	none
Helper	1.08	0.059

Chapter 5

Conclusion

Table 4.4 shows all the results from our experiments. We showed the limitation of pairwise ensemble (OvO) through analysis of misclassified samples and then proposed additional abstention node as one way to overcome the problem in itself. And we proposed a new method to form the tree (the hierarchy) in hierarchical strategy of classification. The result reveals that the performance of the hierarchy made by the method is better than one proposed in [17]. Also, we showed the possibility of combining traditional networks with the network which solves modified problem for better performance.

These three ensembles still have room for improvement. There could exist better methods to divide or modify the given problem. And also ensembles of the style of divide and conquer could increase as the number of all classes increase. Testing three ensembles in this thesis on other databases is one of the possible future works.

Bibliography

[1] Hinton, Geoffrey E., et al. "A fast learning algorithm for deep belief nets." *Neural computation* 18.7 (2006): 1527-1554.

[2] Bengio, Yoshua, et al. "Greedy layer-wise training of deep networks."*Advances in neural information processing systems* 19 (2007): 153.

[3] Hinton, Geoffrey E., et al. "Improving neural networks by preventing co-adaptation of feature detectors." *arXiv preprint arXiv:1207.0580* (2012).

[4] Zeiler, Matthew D. "ADADELTA: An adaptive learning rate method." *arXiv preprint arXiv:1212.5701* (2012).

[5] Glorot, Xavier,, et al. "Deep sparse rectifier neural networks." *International Conference on Artificial Intelligence and Statistics*. (2011).

[6] Srivastava, Nitish, et al. "Dropout: A simple way to prevent neural networks from overfitting." *The Journal of Machine Learning Research* 15.1 (2014): 1929-1958.

[7] LeCun, Yann, et al. "The MNIST database of handwritten digits." *http://yann.lecun.com/exdb/mnist/* (1998). (accessed November 28, 2015)

[8] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." *Nature*521.7553 (2015): 436-444.

[9] Cannon, Alex J., and Paul H. Whitfield. "Downscaling recent streamflow conditions in British Columbia, Canada using ensemble neural network models." *Journal of Hydrology* 259.1 (2002): 136-151.

[10] Hansen, Lars Kai, and Peter Salamon. "Neural network ensembles." *IEEE Transactions on Pattern Analysis & Machine Intelligence* 10 (1990): 993-1001.

[11] Opitz, David W., and Jude W. Shavlik. "Actively searching for an effective neural network ensemble." *Connection Science* 8.3-4 (1996): 337-354.

[12] Perrone, M. P., and L. N. Cooper. "When networks disagree: Ensemble methods for neural networks for speech and image processing." (1993).

[13] Jacobs, Robert A., et al. "Adaptive mixtures of local experts." *Neural computation* 3.1 (1991): 79-87.

[14] Jordan, Michael I., and Robert A. Jacobs. "Hierarchical mixtures of experts and the EM algorithm." *Neural computation* 6.2 (1994): 181-214.

[15] Kreßel, Ulrich H-G. "Pairwise classification and support vector machines."*Advances in kernel methods*. MIT Press, (1999).

[16] Lei, Hansheng, and Venu Govindaraju. "Speeding up multi-class SVM evaluation by PCA and feature selection." *Feature Selection for Data Mining*(2005): 72.

[17] Madzarov, Gjorgji, Dejan Gjorgjevikj, and Ivan Chorbev. "A multi-class SVM classifier utilizing binary decision tree." *Informatica* 33.2 (2009).

요약

인간의 신경망으로부터 고안된 인공신경망 은 다중 클래스 분류 문제 등을 풀기 위해 적합한 기계학습 기술이다. 앙상블을 포함한 여러 가지 기술들이 인공신경망의 기능을 향상시키기 위하여 제안되어오고 있다. 이 논문에서는 분할정복 방법이 접목된 세 가지의 앙상블을 다룬다. 첫 번째는 이진 분류기들을 이용한 다중 클래스 분해 기법이다. 그리고 두 번째는 문제를 작은 문제들로 나누고, 계층적으로 풀어내는 계층적 앙상블 방법이다. 세 번째는 기존 문제를 변경해 난이도를 낮추고, 그것을 풀도록 학습한 인공신경망이 기존의 문제를 푸는 일반적인 인공신경망을 돕도록 하는 형태의 앙상블 방법이다. MNIST 데이타베이스에 실험한 결과 및 분석을 통해 인공신경망을 이용한 다중 클래스 분류 문제 해결에서 분할정복이 가진 문제점과 가능성에 대해 보인다.

Keyword : 인공신경망, 앙상블, 분할 정복
Student Number : 2014-21790