Bounds for the Average Generalization Error of the Mixture of Experts Neural Network

Luís A. Alexandre[1,⋆], Aurélio Campilho[2], and Mohamed Kamel[3]

[1] Networks and Multimedia Group, IT, Covilhã, Portugal
lfbaa@di.ubi.pt
[2] INEB - Instituto de Engenharia Biomédica, Portugal
[3] Dept. Systems Design Engineering, Univ. Waterloo, Canada

Abstract. In this paper we derive an upper bound for the average-case generalization error of the mixture of experts modular neural network, based on an average-case generalization error bound for an isolated neural network. By doing this we also generalize a previous bound for this architecture that was restricted to special problems.
We also present a correction factor for the original average generalization error, that was empirically obtained, that yields more accurate error bounds for the 6 data sets used in the experiments. These experiments illustrate the validity of the derived error bound for the mixture of experts modular neural network and show how it can be used in practice.

Keywords: modular neural networks, mixture of experts, generalization error bounds

1 Introduction

This paper addresses generalization error bounds for supervised learning. In [1], an average-case generalization error bound was introduced that is not as pessimistic as the error bounds derived using Vapnik-Chervonenkis theory [2] since the later are based on a worst case analysis. In this paper we derive an upper bound for the average-case generalization error of the mixture of experts (ME) [3] modular neural network (MNN), based on the average-case generalization error bound introduced in [1] for a single multi-layer perceptron (MLP). By doing this we also generalize a previous bound for this architecture [4] that was restricted to special problems (problems that could be completely separated into two subproblems without overlapping classes, by a hyperplane in the input space) and was derived assuming that the gate does not introduce errors and does not use adjustable parameters.

The bound proposed here assumes that the generalization error of the experts is higher than their training set error. In fact, this assumption was already made in the derivation of the bound for the isolated MLP in [1]. It also assumes independency between the errors of the gate and of the experts.

[⋆] We acknowledge the support of project number POSI/35590/SRI/2000 approved by the Portuguese 'Fundação para a Ciência e Tecnologia'(FCT) and POSI and partially financed by FEDER. Corresponding author: Luís Alexandre.

We tested the error bound on 6 publically available data sets corresponding to real classification problems, with distinct characteristics (number of features, number of patterns and number of classes). These experiments showed that the derived bound was not useful because it was quite loose. This was due to the original expression for the average generalization error (AvGE) in [1] and not to the derivation of the bound for this particular architecture. Thus, we found empirically a modified expression for the AvGE of the isolated MLP, that when used to derive the AvGE of the ME architecture, produces much tighter bounds.

The paper is organized as follows: the next section describes the ME MNN, section 3 introduces the average error bound for a single NN. In section 4, the AvGE error bound for the ME is derived. Three experiments are presented in section 5, with a discussion of their results. The final section presents the conclusions.

2 The Mixture of Experts MNN

The idea behind the MNN is the divide-and-conquer paradigm: the problem should be divided into smaller subproblems that are solved by experts and their partial solutions should be integrated to produce a final solution.

To use an MNN, three stages have to be considered: first, the task decomposition where the problem is divided into smaller problems, each one to be given to one of the modules or expert networks. Then each individual expert is trained until it learns to solve its particular subproblem. Finally, a decision integration strategy is used to combine the decisions of the experts to produce a final network output.

The decision integration can be obtained through different approaches: using a gating network [5], making the modules vote [6] or through hierarchical integration (which can also use voting and/or gating networks) [7, 8].

In this paper we consider the use of a gating network. The gate is trained to learn which region of the input space should be classified by which expert.

During the test phase, when the gate receives an input pattern it decides which is the expert that should classify the pattern and selects one of the experts to produce the final classification. This type of MNN is called a Mixture of Experts.

3 Average Error Bound

3.1 Problem Definition

The bounds discussed in this paper apply to the generalization error of a learning machine, and in particular, to an MLP and its generalization, an MNN.

We now discuss the general learning problem and define explicitly the generalization error and the empirical error (the one measured in the section 4).

Consider a learning problem in which a learning machine is given a set of data $\{x_1, \ldots, x_m\}$ and it is expected that, by adjusting a set of parameters, w, it can learn to associate the respective targets $\{y_1, \ldots, y_m\}$ to each input.

Typically, for a classification problem, which is the type of problem we are addressing in this paper, $x_i \in X$ and $X \subseteq \mathbb{R}^n$ and $y_i \in Y$ and $Y = \{1, 2, \ldots, L\}$, where L is the number of classes in the problem.

Let $Z = X \times Y$. Each $z_i = (x_i, y_i)$ is called a training sample. The joint distribution $P(x, y)$ is represented by $P(z)$.

For a particular choice of the parameters w, the learning machine produces a hypothesis. Each hypothesis is represented by $f(x, w)$. When the prediction $f(x_i, w)$ is different from the respective target y_i, a loss occurs. This loss is measured by the loss function $l(y, f(x, w))$.

The expected value of the loss is called generalization error and is given by

$$R(w) = \int_Z l(y, f(x, w)) dP(z) \tag{1}$$

For the majority of real problems, $R(w)$ is not zero. Usually it is not possible to find $R(w)$ since the distribution $P(z)$ is not known. Instead, the empirical error is found,

$$E_m(w) = \frac{1}{m} \sum_{i=1}^{m} l(y_i, f(x_i, w)) \tag{2}$$

This is an estimate of $R(w)$.

Usually the data set is divided into two disjoint sets, one is used for training, or the adjustment of the weights w, and is called the training set. The other is used to estimate $R(w)$ and is called a test set. The empirical error measured in the training set is usually called the training set error.

In this paper we are concerned with an average error bound for $R(w)$. We call it the Average Generalization Error (AvGE).

3.2 AvGE for the MLP

The AvGE for an MLP was introduced in [1]. It has the form of

$$AvGE_{MLP} \leq \alpha + \frac{1}{2}\sqrt{\frac{d}{m}} \tag{3}$$

where α is the training set error, d represents the number of weights and m is the number of training samples. Note that it is an upper bound for the generalization error based on the training set error and a penalty for the number of weights that is reduced by the number of training samples.

3.3 AvGE for the ME MNN

To produce an AvGE for the ME MNN we first introduce the complement of the AvGE, the AvGC, which we define as

$$AvGC = 1 - AvGE \tag{4}$$

It is the 'Average Generalization Correctness'.

Using this definition it is possible to write, assuming independency between the errors made by the gate and by the experts,

$$AvGC_{ME} = AvGC_g \sum_{i=1}^{N} P(mod_i) AvGC_i \tag{5}$$

where $AvGC_g$ is the $AvGC$ of the gate, $P(mod_i)$ is the probability that the module i is chosen by the gate and $AvGC_i$ is the $AvGC$ of module i.

The assumption of the independency of the errors will not be true only in points on the frontiers between distinct regions of expertize of different experts.

To obtain the AvGE for the ME, we replace in expression (5) the relation in expression (4) for the gate and the individual modules, we get

$$AvGE_{ME} = 1 - AvGC_{ME} = 1 - \left[(1 - AvGE_g) \sum_{i=1}^{N} P(mod_i)(1 - AvGE_i) \right] \tag{6}$$

After some simple manipulation, we get

$$AvGE_{ME} = \beta + AvGE_g(1 - \beta) \tag{7}$$

with

$$\beta = \sum_{i=1}^{N} P(mod_i) AvGE_i \tag{8}$$

To obtain the upper bound for the $AvGE_{ME}$ we have to find a lower bound for β. A trivial lower bound is zero, but we can assume that the generalization error will be higher than the training set error, and thus β has the following upper and lower bounds

$$\sum_{i=1}^{N} P(mod_i)\alpha_i \leq \beta \leq \sum_{i=1}^{N} P(mod_i) \left(\alpha_i + \frac{1}{2}\sqrt{\frac{d_i}{m_i}} \right) \tag{9}$$

The upper bound is obtained directly from expression (8) by simply replacing $AvGE_i$ by expression (3), with each variable using a subscript i that corresponds to the expert i.

Now it is possible to write the final expression for the $AvGE_{ME}$

$$\begin{aligned} AvGE_{ME} \leq & \sum_{i=1}^{N} P(mod_i) \left(\alpha_i + \frac{1}{2}\sqrt{\frac{d_i}{m_i}} \right) \\ & + \left(\alpha_g + \frac{1}{2}\sqrt{\frac{d_g}{m_g}} \right) \left(1 - \sum_{i=1}^{N} P(mod_i)\alpha_i \right) \end{aligned} \tag{10}$$

We will now use this bound to predict the generalization error of the ME MNN. Note that the parameters d_i, d_g, m_i and m_g are set a priori. The $P(mod_i)$ can be estimated from the proportion of points in the training set that belong to the region assigned for expert i. The α_i (the training set errors) are estimated and the number of modules used, N, is defined by the task decomposition algorithm or set a priori.

4 Experiments

4.1 Introduction

In these experiments we intend to show how the expression (10) can be used to bound the generalization error.

We use six data sets that illustrate different conditions: the number of classes ranges from 2 to 10; the number of data points ranges from 106 to 2126; the number of features ranges from 2 to 60.

Table 1 contains information about these data sets. They are all publically available, and the table shows references to their sources. It also shows the name, number of data points, number of features and the number of classes.

Table 1. The data sets used in the experiments.

Data set	Name	N. points	N. features	N. classes	Source
1	Breast cancer	106	9	6	[9]
2	Cleveland	296	13	5	[10]
3	CTG	2126	22	10	[9]
4	Diabetes	768	8	2	[10]
5	Speech recognition	608	2	4	[5]
6	Sonar	208	60	2	[10]

4.2 Classification

As we made the experiments, we noticed that the original error bound in expression (3) was quite loose. This behavior was also observed by Gu and Takahashi in [11] when they applied another average error bound to MLPs.

We empirically found that if instead of using $d = (number\ of\ weights)$ we used $d = (number\ of\ weights)/18$, the error bound becomes much tighter. A justification for this replacement is that in fact, although d was considered the number of weights for an MLP in [1], it is more recently considered to be an upper bound on the number of adjustable weights of the MLP (Lemma 1 in [11]). Hence, it can be replaced by a smaller value. Of course, a theoretical guideline for the value of d to use is desirable.

In what follows we present measures of the two versions of the error bound for the ME: $AvGE_{ME}$ and $AvGE_{ME2}$. They both correspond to expression (10), but the former uses $d = (number\ of\ weights)$ and the later uses $d = (number\ of\ weights)/18$.

The experiments were made with the holdout method: half the data set was used for training and the other half for testing. Then the data sets were used with inverted roles (the original training set became the test set and the original test set became the training set) and the error and error bounds were averaged between the two repetitions.

The task decomposition was made with the fuzzy c-means algorithm [12]. The experts and the gate were one hidden-layer multi-layer perceptrons (MLPs) trained with resilient backpropagation [13], for 100 epochs. The number of hidden layer neurons used for each expert and for the gate are listed in tables 2 and 3. These numbers were obtained empirically and gave acceptable classification errors for all data sets. We were

Table 2. The topology of the experts and gate along with the error and error bounds with standard deviations when using two experts.

Data set	Topology	$AvGE_{ME}$ (std)	$AvGE_{ME2}$(std)	Error (std)
1	10,10,4	163.0 (2.3)	40.6 (3.6)	39.6 (10.7)
2	5,5,2	89.6 (4.7)	29.6 (5.9)	47.6 (0.5)
3	21,21,4	76.1 (0.2)	21.4 (0.1)	19.3 (1.1)
4	1,1,6	50.2 (0.2)	30.2 (0.2)	27.9(0.4)
5	2,2,2	29.2 (0.6)	11.3 (0.7)	9.6 (1.9)
6	2,2,6	174.0 (0)	41.0 (0)	20.2 (1.6)

Table 3. The topology of the experts and gate along with the error and error bounds with standard deviations when using three experts.

Data set	Topology	$AvGE_{ME}$ (std)	$AvGE_{ME2}$ (std)	Error (std)
1	10,10,10,4	186.3 (3.4)	48.2 (1.2)	32.1 (8.0)
2	5,5,5,2	108.9 (2.3)	33.2 (0.2)	45.3 (3.8)
3	21,21,21,4	88.1 (0.9)	24.1 (1.1)	20.0 (1.0)
4	1,1,1,6	51.1 (0.6)	25.2 (0.7)	23.1 (2.8)
5	2,2,2,2	32.9 (0.2)	11.9 (0.2)	7.9 (0.9)
6	2,2,2,6	191.1 (1.0)	45.0 (0.2)	17.3 (6.8)

not concerned in finding the optimal value of the number of neurons, we intended only to illustrate the behavior of the error bound derived in section 3.2.

Table 2 shows the values obtained in the experiments using two experts in the mixture of experts. The second column (topology) gives the number of hidden layer neurons used with each expert and gate, respectively. For instance, 18,18,2 means that both the first expert and the second had 18 neurons in the hidden layer and that the gate used 2 neurons in the hidden-layer.

Table 3 shows the values obtained in the experiments using three experts in the mixture of experts.

4.3 Discussion

The Errors. The true errors are smaller when the experiment is done with three experts than when it is done with only two. This is not surprising since in the first case the number of weights used in the MNN is larger and it can adjust itself better to the data. There is one exception, that is data set 3. This indicates that for this particular problem a division in three sub-problems does not improve its learnability when compared to a division in only two subproblems.

The Bounds. In all cases, the $AvGE_{ME2}$ is tighter than $AvGE_{ME}$, indicating that d should not be set to the number of weights but to a smaller value. This is due to the fact that the $AvGE_{ME2}$ gives a smaller penalty for the increase in the number of adjustable parameters in the model (the network weights) than $AvGE_{ME}$.

In all cases, both the $AvGE_{ME}$ and the $AvGE_{ME2}$ increase their values when going from an MNN with two experts to one with three experts. This is due to an increase in the total number of weights for the MNNs with 3 experts when compared to the MNN used for the same problem but with only 2 experts.

The bound $AvGE_{ME}$ is always larger than the measured error. But for the bound $AvGE_{ME2}$ this does not happen with data set 2. Both with the 2 expert MNN and with the 3 expert MNN, the bound $AvGE_{ME2}$ is considerably smaller than the measured error. The bounds have an important component that is the measured training error. We checked the training and test set errors for an isolated MLP for this data set and there is a big difference between their values. Training set error is about 15% while the test set error is over 40%. We think this explains the poor performance of the bound for this particular case. The reason for the difference between the training and test errors might be due to a training overfitting.

The predictions of the $AvGE_{ME2}$ bound are very close to the measured errors. The exceptions are data set 3, which was already mentioned, and data set 6. This data set represents a problem in a high dimensional space (60-dimensional). This makes the number of weights of an MLP vary a great deal with the introduction of a single neuron in the hidden layer (each new hidden layer neuron adds 61+(number of classes) weights to the MLP). So it is easy to go from one situation where the number of weights is insufficient to learn the problem to one where the network overfits the problem. We argue that the difference between the bound and the errors measured for this data set is due to the possibly larger than necessary number of weights used in the network.

5 Conclusions

In this paper we developed an average error bound for the mixture of experts MNN. It is an extension of the bound presented in [1] for an isolated NN. It also generalizes the bound in [4] since there are no restrictions on the data sets or on the type of gate used. This way it can be applied to any classification problem.

We also propose an empirical correction factor to the original expression in [1] that produces much tighter bounds. A theoretical justification for the particular value of this adjustment is still lacking.

These bounds can be used to estimate the generalization error of the ME MNN, as illustrated in the experiments section and they behave well for most of the tests made.

References

1. Gu, H., Takahashi, H.: Towards more practical average bounds on supervised learning. IEEE Trans. Neural Networks **7** (1996) 953–968
2. Vapnik, V.: The Nature of Statistical Learning Theory. Springer (1999)
3. Haykin, S.: Neural Networks: A Comprehensive Foundation, 2nd edition. Prentice Hall (1999)
4. Alexandre, L., Campilho, A., Kamel, M.: Average error bound for the mixture of experts MNN architecture. In: Proceedings of the 12th Portuguese Conference on Pattern Recognition, Aveiro, Portugal (2002)

5. Jacobs, R., Jordan, M., Nowlan, S., Hinton, G.: Adaptive mixtures of local experts. Neural Computation (1991) 79–87
6. Auda, G., Kamel, M.: Modular neural network classifiers: A comparative study. J. Intel. Robotic Systems (1998) 117–129
7. Jordan, M., Jacobs, R.: Hierarchical mixture of experts and the EM algorithm. Neural Computation (1994) 181–214
8. Jacobs, R., Peng, F., Tanner, M.: A bayesian approach to model selection in hierarchical mixtures-of-experts architectures. Neural Networks **10** (1997) 231–241
9. Marques de Sá, J.: Pattern Recognition: Concepts, Methods and Applications. Springer (2001)
10. Blake, C., Keogh, E., Merz, C.: UCI repository of machine learning databases (1998) http://www.ics.uci.edu/˜mlearn/MLRepository.html.
11. Gu, H., Takahashi, H.: Estimating learning curves of concept learning. Neural Networks **10** (1997) 1089–1102
12. Bishop, C.: Neural Networks for Pattern Recognition. Oxford University Press (1995)
13. Riedmiller, M., Braun, H.: A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In: IEEE International Conference on Neural Networks, San Francisco (1993) 586–591