



Building a Hierarchical Network of Perceptrons with a PAC-Bayes Bound

M. T. Elliott

Sentient Technologies, LLC

4924 Balboa Blvd, Suite #386, Encino, CA 91316, U.S.A.

mike_elliott@sentienttechnologies.com

Abstract

Much current research has focused on methods of improving model selection for the purpose of minimizing the generalization error of classifiers. This complex problem often involves choosing a kernel, or the appropriate size for a neural network, and a loss function. Frequently, this process is sufficiently solved via trial and error methods such as cross validation. In this paper we specify an algorithm to perform model selection by directly minimizing a PAC-Bayes bound on the generalization error. We prove an extension of current bounds to the setting of hierarchical classifiers and then apply our algorithm to construct a network of perceptrons that minimizes this bound. We apply our results to the USPS dataset and show how our bound is an accurate predictor of the empirically observed test set error. While this particular algorithm does not achieve record setting performance on this dataset, we believe that we are just scratching the surface of algorithms possible with this hierarchical PAC-Bayes bound.

Keywords: *PAC-Bayes, PAC, Bayesian, Generalization Error bounds, Hierarchical classifier, Neural network, Perceptron Network, Perceptron, Learning theory*

1. Introduction

The primary objective in building a classifier is to produce one that generalizes with low error. Often, before training, it is unclear what kind of model, Support Vector Machine (SVM), Neural Network, or otherwise, and what model parameters, kernel, size of network, etc., will produce optimal or even satisfactory performance on a given dataset. The process of choosing the desired classifier is known in machine learning as “model selection” and has been the subject of much research [4].

If one selects a model that is too complex, there is a risk of over-fitting the data. Too simple a model may result in the inability of the classifier to effectively learn the patterns in the dataset. Even with maximal margin classifiers, while some protection is provided against over-fitting, there are no guarantees, and over-fitting is still observed in some situations. Also, there

are instances where minimizing the error on a training sample does not result in minimizing generalization error [20].

One type of frequently used model is an SVM [3]. In order to implement an SVM one must first select, prior to training, a kernel, for example, radial-basis function [16] or polynomial [17]. Second, parameters for the kernel must be selected, for example, the order of polynomial, or the width of the Gaussian in the case of Radial Basis Function (RBF) kernels. Third, an appropriate loss function and its parameters must be set, including, for instance, the 2-norm loss parameter which controls the trade-off between training set error and margin maximization. Given these choices it is unclear what combination will produce a classifier with the lowest generalization error.

A common method for performing model selection with SVM’s is cross validation [2]. With this trial-and-error technique, one trains a number of possible classifier models on subsets of the training set, always leaving out a few examples. Each classifier is trained on one subset, and it is tested on the remainder of the training set to give an idea of the generalization error of the classifier. Generally, the classifier that produces the lowest error on the test subset is the one chosen.

A more formalized SVM model selection method was recently presented addressing “learning the kernel” where the kernel parameters, including the loss parameter, are calculated using semi-definite programming in a transductive setting [10, also note 5]. This is an environment in which we know the data points in a test set, but not the labels. This approach has the distinct advantage of maintaining the SVM’s elegance of achieving a global optimum over the parameter space. However, we frequently are not working in a transductive environment.

Another type of model is a neural network. Here one selects a network structure appropriate to the classification task at hand. The specifiable parameters include, for example, the number of layers in the network, the number of neurons in each hidden layer, and the number of connections to each neuron [6]. Also, one must choose a regularization or loss function similar to the loss parameter in SVM’s.

