Finite Size Effects in Neural Network Analyses

David Barber

Doctor of Philosophy
University of Edinburgh
1996

Declaration

This thesis has been composed by myself and it has not been submitted in any previous application for a degree. The work reported within was executed by me, unless otherwise stated.

July 1996

Acknowledgments

Academically, there have been many people from whom I've greatly benefitted throughout my PhD. In particular, the following have my sincerest thanks: Nick Handy, David Wallace, David Willshaw and David Saad. There are many others from whom I've learned that there is still much to be learned, and these include Alastair Bruce, John Hertz, Jonathan Shapiro and Chris Bishop.
Personally, this has been a wonderful three years. Sharing offices with Pete, Glenn and Ansgar has provided much fun and friendship. Goodness knows what I would have done without Pete, whose magical insight into physics and calm personality has been an inspiration to us all.
Edinburgh is a fine city, and I've greatly enjoyed my time with my flatmates and friends, amongst them Sara, Fran, Anja, Kai Kim and Liz. I reserve my biggest hug of all for Katja - a wonderful person, delightfully wrapped with a neat ribbon of fun.

Finally, I thank my family for their constant support throughout what has been a long academic journey.

Publications

- D. Barber, D. Saad. Does extra knowledge necessarily improve generalisation? *Neural Computation*, 1996.
- D. Barber, P. Sollich, and D. Saad. Finite size effects in on-line learning in multilayer neural networks. 1996. *Annals of Mathematics and Artificial Intelligence.*
- D. Barber and D. Saad. Knowledge and generalisation in simple learning systems. In *ESANN'95*, pages 161–166, Brussels, 1995. D Facto.
- D. Barber, D. Saad and P. Sollich. Finite-size effects and optimal test set size in linear perceptrons. *Journal of Physics A*, 1995.
- D. Barber, D. Saad and P. Sollich. Test error fluctuations in finite linear perceptrons. *Neural Computation*, 1995.
- D. Barber, D. Saad and P. Sollich. Finite size effects in on-line learning of multi-layer neural networks. 1995. *Europhysics Letters*, 1996.

Contents

Chapter 1

Introduction

1.1 Background

Artificial neural networks constitute biologically inspired techniques to perform data modelling, widely used in such diverse areas as pattern classification, control, and financial forecasting. One of the features that make neural networks so attractive is their perceived ability to assimilate or 'learn' the underlying rule producing the data. The framework within which this thesis is cast is that of *supervised learning*: we imagine that there is a unique rule producing the data, which we identify with a *teacher*.

As a demonstration of some of the central ideas involved in the theory of neural networks, let us consider the following scenario. A teacher sits in a room, and upon being given an input x generates outputs $y(x)$ according to the rule $y(x) = y^0(x) + \eta$, where η is some noise process corrupting the clean teacher output $y^0(x)$ - some fixed deterministic function. A set of P inputs $\left\{x^{(1)}....x^{(P)}\right\}$ is drawn independently and identically from an input distribution, and to each input, x, the teacher associates an output, y, so that we have a set of noisy *training examples* $\mathcal{P} = \left\{(x^{(1)}, y^{(1)}), ..., (x^{(P)}, y^{(P)})\right\}$. These are given to the student who is asked to infer the (uncorrupted) teacher rule, y^0. Without some clue as to what kind of function the teacher is using, the student's task is hopeless. The possible rules that the student could conceive that fit the training data are endless - she could fit a P dimensional polynomial, or a $P + 1$ dimensional polynomial, or a P dimensional polynomial with a large amplitude oscillation that interpolates the training points, and so on... Hence, the task of *fitting* the data is relatively easy, but when the student tests the model against previously unseen training examples, without some *a priori* knowledge about the teacher, the student's predictions would be no better than random guesses[Wol95, WL92](see figure(1.1)). What we are really interested in is the student's *generalisation* ability- how will the student perform on unseen inputs? If the student is told beforehand that the uncorrupted teacher rule comes from the class of polynomials with degree less than 10, we might hope that eventually the student would infer the rule with some accuracy, and have a low expected test error, (termed the *generalisation error*). Even restricting the set of possible functions to polynomials with degree less than 10 may lead to over-fitting of the noisy data points. Some degree of *regularisation* is therefore often required in the presence of noisy training examples, which typically takes the form of a penalty for the student complexity - *e.g.,* low degree polynomials should be favoured over high degree polynomials.

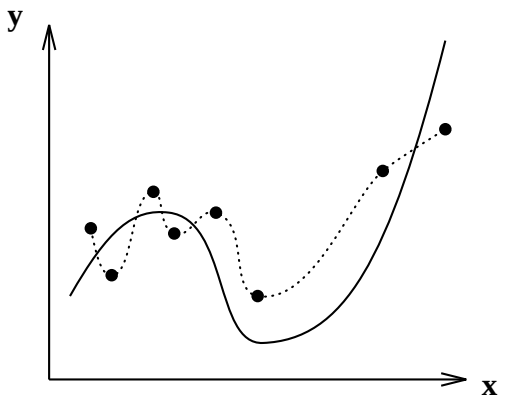

Figure 1.1. Curve fitting. The dots represent noisy training points generated by the uncorrupted teacher output given by the solid line. Fitting the training points may lead to the problem of over-over-fitting in which the student fits the noise. There are infinitely many possible functions that we could fit through the training points, but fitting the points does not guarantee good generalisation.

Having ascertained that, in order to frame the question of generalisation in a sensible manner, we need to restrict the space of possible teachers, we will naturally want to quantify the generalisation performance of the student, given a certain amount of data. In order to do this, we assume that the space of possible teacher functions and the space of students (the class of models) are defined. Often, the assumption is made that these two spaces are the same. It may be, however, that we choose a student much less sophisticated than the teacher for reasons of computational complexity, in which case the problem is *unrealisable*, and the task is to quantify the performance of the best student available. Throughout this thesis, however, we shall generally examine *learnable* problems and set the student and teacher function spaces to be the same.

One approach to evaluating generalisation performance is to bound the error that a student will make, given that it has been trained on P examples, and that we know the complexity of the class of possible functions. This approach is called the *probably approximately correct*, or PAC approach, and is typically practised within the computational learning theory school[VC71, Hau94, Val84]. More formally, an algorithm is PAC if there exists a number of training examples such that for more examples than this, with some specified confidence the model will make an error no greater than some specified accuracy.

There are advantages and disadvantages of the PAC approach. An elegant feature of PAC is that the results are distribution independent: It might be that the input distribution has a low density in a certain region, and that the fit of the student in that region is correspondingly bad, due to the poor sampling; however, when evaluating the generalisation performance of the student, those regions of input space with low density will hardly ever occur, and the student's poor performance in such regions will have little weight. Furthermore, the generalisation error bound in the PAC approach depends solely on a measure of the complexity of the function space, called the VC-dimension[1]. Once the VC dimension has been calculated, the array of results

[1]For the case of binary outputs, the VC dimension is the maximum number P' of training examples for

relating to PAC learning can be read off. However, determining the VC dimension is often difficult and to date the VC dimension has been determined for only a limited class of function spaces[Ant95]. Another drawback of PAC learning is that it tends to give a rather conservative estimate of the generalisation ability of the student - the typical generalisation performance of the student is often much better than the accuracy specified in the PAC approach[EvdB93]. Some efforts have been made recently to make PAC results more comparable to the typical performance of students by introducing specific classes of distributions for the model. For a general review of PAC learning see [Ant95].

1.1.1 Average Case Formalism

If we assume that we know the input distribution, teacher space, and student generating algorithm, we can attempt to calculate the generalisation error directly. However, if the variance of the test error is large, then the expected error (the generalisation error), in itself, does not shed much light on the test error distribution, and it is important to have an estimate of the variance of the test error. In attempting to perform the requisite averages inherent within an average case analysis, one invariably runs into technical difficulties and approximations need to be introduced. Much of the work carried out within the physics community in calculating the expected error has been through formal analogies drawn between data set averages and "quenched" averages in statistical physics (for a review, see[WRB93]). These calculations have typically been carried out with recourse to the *thermodynamic limit* in which the dimension of the inputs x is taken to be infinite. In the limit of an infinite input dimension, with the number of training examples $P \propto N$, the test error becomes *self-averaging* - the variance of the test error distribution is zero. As infinite networks are unrealistic, it is important to quantify the variance of the test error distributions explicitly in order to justify the relevance of the thermodynamic limit[Sol94a]. Other approaches have been made using statistical mechanics to calculate the maximum deviation of the test error from the generalisation error, which is an approach closely allied to the PAC worst case theory[EvdB93, EF93]. Another approach which employs statistical mechanics is that used by Haussler *et al.*[HKST94] to relate the entropy of the student space and the probability of minimising the test error on a random test set, although at the moment this theory has only been fully developed for cases in which the teacher space is a set of finite cardinality.

Within an average case Bayesian formalism, Amari[AF92] has examined the asymptotic decay of the generalisation error using the annealed approximation, which can lead to qualitatively correct results. Interestingly, Amari found that there exist essentially only four kinds of asymptotic decay with the number of examples P presented, classified according to whether or not the student is stochastic, the teacher output is corrupted by noise, the set of parameters specifying the teacher is unique, or has finite measure. However, some considerable care must be taken in employing the annealed approximation as this can lead to wildly incorrect results, as discussed in Seung *et al.*[SST92].

Throughout this thesis, we shall assume that the known input distribution is normal (gaussian), and similarly for the noise process. As more random examples are presented, the information content of each new training example decreases. In order to improve the efficiency of learning from examples, there has recently been much interest in *active learning* or *query*

---

which all possible $2^{P'}$ output configurations are achievable by appropriate settings of the student parameters.

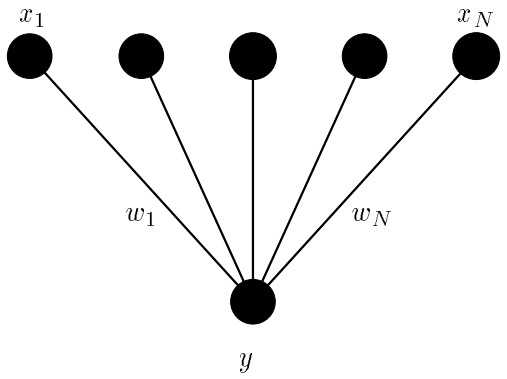

Figure 1.2. The simple perceptron. The input components are represented by the N dimensional vector $\mathbf{x}$. The activation is the weighted sum of these input components, $h = \sum_{i=1}^{N} N^{-\frac{1}{2}} w_i x_i = N^{-\frac{1}{2}} \mathbf{w} \cdot \mathbf{x}$. The final output of the perceptron is the transfer of the activation, $y = g(h)$.

learning in which new training examples are *selected* by the student in order to maximise some objective measure of their usefulness such as information (entropy) gain (for references, see [Plu94, Sol95]), although this framework is beyond the scope of this thesis.

1.2 The Perceptron

Artificial neural networks are composed of simple neuron like units which perform a mapping from $\mathcal{R}^N$ to $\mathcal{R}$, where the output y is some function of the weighted sum of the inputs into that neuron(figure(1.2)),

$$y = g\left(\frac{1}{\sqrt{N}} \mathbf{w} \cdot \mathbf{x}\right) \tag{1.1}$$

and each component of the vector $\mathbf{x}$ represents a real valued input to the network and $g(\cdot)$ is called the *activation function* (the factor $N^{-\frac{1}{2}}$ is for convenience). The vector of connection strengths $\mathbf{w}$ is called the *weight vector*. More complicated networks can be constructed from these simple devices by connecting the output of such a device to the input of another. In this thesis, we shall be concerned with a particular class of network architectures, namely *feedforward networks*, in which we assume that the outputs of each simple perceptron connect only to simple perceptron inputs in a subsequent layer. More complicated cases in which feedback connections are present are studied in the theory of *recurrent neural networks* (see *e.g.,* [Pin87]).

1.2.1 Training neural networks

Although the concept of neural networks has been around for many years[Heb49, Ros62] it is only comparatively recently that they have found widespread use. One of the reasons for this was the lack of a suitable training algorithm, especially for networks more complicated than the simple perceptron. A particularly fruitful approach to developing training algorithms comes from defining an *energy function*, or *training error*[Hop82]. For a set $\mathcal{P}$ consisting of the P

training example pairs $(\mathbf{x}^1, y^1)..(\mathbf{x}^P, y^P)$, the training error is defined to be the sum quadratic loss of the P examples,

$$E_{tr}(\mathbf{w}|\mathcal{P}) = \frac{1}{2}\sum_{\sigma=1}^{P}\left(y(\mathbf{w}, \mathbf{x}^\sigma) - y^\sigma\right)^2 .$$

where $y(\mathbf{w}, \mathbf{x}^\sigma)$ is the output of the student (with weight connection parameters $\mathbf{w}$), and y^σ is the output of the teacher for input $\mathbf{x}^\sigma$. The student parameters can be adapted to minimise the training error by (stochastically) descending the training error surface, updating the students parameters at time t by gradient descent (see *e.g.,* [WRB93]),

$$\frac{\partial w_i}{\partial t} = -\frac{\partial E_{tr}(\mathbf{w}|\mathcal{P})}{\partial w_i} + F_i(t),$$

where $F_i(t)$ is white noise such that $\langle F_i(t)F_j(t')\rangle = 2T\delta_{ij}\delta(t-t')$ and T is the *learning temperature*. This type of learning is known as *batch* learning as the student weights are updated according to their error on a batch of P training examples. The alternative approach, termed *on-line* learning can be thought of as a limiting case of batch learning in which the weights are modified from a stream of single examples. Learning is carried out at some finite temperature in order to avoid local minima in the error surface, and a heuristic annealing schedule for lowering the temperature is typically implemented. The equilibrium ($t \to \infty$) distribution of students that this algorithm produces is a *Gibbs* distribution,

$$P(\mathbf{w}|\mathcal{P}) = \frac{1}{Z}P^{pri}(\mathbf{w})\exp(-E_{tr}(\mathbf{w}|\mathcal{P})/T), \tag{1.2}$$

where $P^{pri}(\mathbf{w})$ represents prior constraints on the student and Z is a normalisation constant. The dynamics of batch learning will not concern us here, and the reader is referred to other works (for a discussion, see[KH92]). In the limit of zero learning temperature, the Gibbs distribution becomes uniform over the set of student weight vectors that exactly reproduce the training set, and zero elsewhere; the Gibbs algorithm then selects a student randomly from this distribution. This is also known as *exhaustive learning* and the space of zero training error students is termed the *version space*[WL92]. The performance of the students generated by the Gibbs learning algorithm is tested on a *test set* of M examples, $\mathcal{M} = \{(x^\mu, y^\mu), \mu = 1..M\}$, and measured by the *test error*, defined by

$$\epsilon_{test}(\mathbf{w}|\mathcal{M}, \mathbf{w}^0) = \frac{1}{2M}\sum_{\mu=1}^{M}\left(y(\mathbf{x}^\mu) - y^\mu\right)^2 \tag{1.3}$$

Ideally, one would like to know the *generalisation function*, *i.e.,* the expected error that a student drawn from $P(\mathbf{w}|\mathcal{P})$ will make on a random test example, and this is found by averaging the test error over the distribution of test sets. As the generalisation function is still dependent on the examples that were used to train the student and also on the Gibbs weight distribution, a further average over the Gibbs distribution and training set are taken in the definition of the *generalisation error*. In some cases it may be possible to carry out these averages exactly, and where we have been able to do so, we shall present such exact results, although these situations are rare[Han93]. Within this framework, there are several sources of randomness: The randomly distributed training data is employed by a stochastic learning algorithm which

is then tested on randomly distributed data. In order to measure these different sources of randomness, we calculate test error (co)variances, and use the notation,

$$\mathrm{var}\left(\epsilon_{test} : \mathcal{A}\right) \equiv \left\langle \left[\epsilon_{test} - \langle \epsilon_{test} \rangle_{\mathcal{A}}\right]^2 \right\rangle_{\mathcal{E}}, \tag{1.4}$$

$$\mathrm{cov}(\epsilon_{test}, \epsilon'_{test} : \mathcal{A}) \equiv \left\langle \left[\epsilon_{test} - \langle \epsilon_{test} \rangle_{\mathcal{A}}\right] \left[\epsilon'_{test} - \langle \epsilon'_{test} \rangle_{\mathcal{A}}\right] \right\rangle_{\mathcal{E}}, \tag{1.5}$$

where $\mathcal{E}$ denotes all sources of randomness and $\mathcal{A}$ is a set denoting one or more sources of randomness[2]. In words, $\mathrm{var}\left(\epsilon_{test} : \mathcal{A}\right)$ is the variance of ϵ_{test} over $\mathcal{A}$, averaged over all sources of randomness (and similarly for $\mathrm{cov}(\epsilon_{test}, \epsilon'_{test} : \mathcal{A})$). The different kinds of (co)variances that we can consider come from the different possible settings of $\mathcal{A}$, which are combinations of $\mathcal{P}, \mathcal{M}, \mathcal{L}, \mathcal{W}, \mathcal{W}^0$. $\mathcal{P}$ is the set of training examples, and $\mathcal{M}$ the set of test examples, with their union $\mathcal{L}$ denoting the *dataset*. $\mathcal{W}$ is the set of student weights consistent with the student post-training distribution, and $\mathcal{W}^0$ is the set of teacher weights consistent with the training set. All sources of randomness are therefore contained in the union, $\mathcal{E} = \mathcal{L} \cup \mathcal{W} \cup \mathcal{W}^0$.

Rather than enter into a detailed discussion of the possible measures of variance here, we shall introduce them when necessary in the text. Nevertheless, the quantity that we shall mainly be interested in measures the typical deviation of the test error from the average test error (generalisation function) and is given by,

$$\mathrm{var}\left(\epsilon_{test} : \mathcal{M}\right) = \left\langle \left(\epsilon_{test} - \langle \epsilon_{test} \rangle_{\mathcal{M}}\right)^2 \right\rangle_{\mathcal{E}}. \tag{1.6}$$

As an application of the techniques we use for calculating variances, we also evaluate the variance of cross-validation estimates of the generalisation error. Cross-validation is a widely used statistical technique used to estimate, for example, the generalisation error with a limited amount of data. Rather than splitting a dataset into a single training set on which a single student is trained, and then tested on the remaining data, cross-validation partitions the dataset into multiple training and test sets, with a separate student being trained on each test/training partition[Sto74, Sto77]. The cross-validation estimate of the error is then the average of the multiple cross-validation student errors. There has been a great deal of work carried out on the analysis of cross-validation, much of it, however, concerned with the asymptotic limit of a large amount of data. We show how analytic results can be obtained for cross-validation using ideas from statistical mechanics for all amounts of data.

1.3 Structure of thesis

A great deal of work has been carried out in the average case formalism for one of the simplest possible networks, the linear perceptron - a single layer perceptron with a linear activation function. In chapter(2) we examine the linear perceptron in the noiseless case (and hence without regularisation), introducing some of the methods that can be employed to calculate variances exactly and approximately. Details of the calculations will normally be relegated to appendices at the end of each chapter. An analysis of cross-validation is carried out for this simple model, with a comparison made between variants of cross-validation. For the more

[2]Strictly speaking, in general $\mathcal{A}$ is a set only in the limit of zero training temperature, otherwise it represents a distribution.

realistic case of a teacher corrupted with noise, we perform a detailed analysis of the variance of the linear perceptron with a regularisation parameter in chapter(3), continuing with a comparison of variants of cross-validation. With the introduction of a regularisation parameter, we consider issues of model selection and how we can use cross-validation to differentiate between two competing models. In chapter(4) we calculate test error variances for a representative non-linear student/teacher, the binary perceptron - a single layer perceptron with a sign activation function. These results enable us to make some connections between our work and the PAC formalism. We examine a two layer perceptron in chapter(5) using a different learning algorithm from batch learning, namely *online learning*, in which the student's weight vectors are updated after presentation of a single example in a stream of data examples. A detailed discussion is made of finite size effects for the soft-committee-machine architecture, including a scheme to both reduce the finite-size effects and also facilitate learning via the introduction of an extra student weight constraint. As this extra constraint on the student leads to a reduction in the generalisation error, we examine briefly in chapter(6) whether it is always the case that the extra knowledge in the form of additional student weight constraints necessarily leads to a reduction in the generalisation error. We conclude with a summary of the main results in the thesis in chapter(7) and an outlook on future research. Appendix(A) contains details of the replica formalism employed throughout much of the thesis in the calculation of (co)variances.

Chapter 2

The Linear Perceptron I: Spherical constraint

Abstract

We calculate the fluctuations in the test error induced by random, finite, training and test sets for the noise free, zero temperature linear perceptron of input dimension N with a spherically constrained weight vector. This variance enables us to address such issues as the partitioning of a data set into a test and training set, for which we find that the optimal assignment of the test set size scales with $N^{2/3}$. Furthermore, we examine the variance of cross-validation errors in the non-asymptotic data regime.

2.1 The Spherical Linear Perceptron

A *training set* is defined to be a set of P input/output pairs, $\mathcal{P} = \{(\mathbf{x}^\sigma, y^\sigma), \sigma = 1..P\}$. Each component of the input vectors $\mathbf{x}^\sigma$ is drawn from the zero mean, unit variance normal distribution, and the outputs y^σ are generated by a noiseless teacher perceptron, $y^\sigma = \frac{1}{\sqrt{N}}\mathbf{w}^0\cdot\mathbf{x}^\sigma$, characterised by the teacher weight vector $\mathbf{w}^0$. The student is of the same form as the teacher, namely a *linear perceptron* of dimension N with weight vector $\mathbf{w}$, where both student and teacher weight vectors are of length $\sqrt{N}$ (*spherical constraint*). The set of student perceptrons that agree with the teacher on the training set (*i.e.,* produce the same output as the teacher for the inputs from the training set) and that obey the spherical constraint is a subset of the set of all weight vectors, termed the *version space* (VS)[WRB93]. The *training error* is given by,

$$E_{tr}(\mathbf{w}|\mathcal{P}) = \frac{1}{2}\sum_{\sigma=1}^{P}\left(\frac{1}{\sqrt{N}}\mathbf{w}\cdot\mathbf{x}^\sigma - y^\sigma\right)^2. \tag{2.1}$$

The spherical constraint is imposed by adding to $E_{tr}(\mathbf{w}|\mathcal{P})$ an extra term, equivalent to a Lagrange multiplier. The resulting equilibrium distribution of students as $t\to\infty$ is a Gibbs distribution,

$$P(\mathbf{w}|\mathcal{P}) = \delta\left(\mathbf{w}\cdot\mathbf{w} - N\right)\frac{1}{Z}e^{-E_{tr}(\mathbf{w}|\mathcal{P})/T},$$

where Z is a normalising factor. The VS is then found as the set of weight vectors $\mathbf{w}$ of non-zero probability $P(\mathbf{w}|\mathcal{P})$ in the limit of zero temperature Gibbs learning. Students from the VS are tested against the teacher on a *test set* of M elements $\mathcal{M} = \{\left(\mathbf{x}^\mu, N^{-\frac{1}{2}}\mathbf{w}^0\cdot\mathbf{x}^\mu = y^\mu\right), \mu = 1..M\}$ [1], where the $\mathbf{x}^\mu$ are taken from the same normal distribution that was used to generate the training set inputs $\mathbf{x}^\sigma$. The training set and test set together form the *data set* of L elements, $\mathcal{L} = \mathcal{P} \cup \mathcal{M}$, with $L = P + M$. The average error that a student from the version space will make on a random example is given by the *generalisation function*,

$$\epsilon_f(\mathbf{w}|\mathbf{w}^0) = \frac{1}{2N}\left\langle \left(\left(\mathbf{w} - \mathbf{w}^0\right)\cdot\mathbf{x}\right)^2 \right\rangle_{\mathbf{x}},$$

where $\langle..\rangle_{\mathbf{x}}$ denotes an average over test example inputs. In a practical situation, this quantity is approximated by the *test error*,

$$\epsilon_{test}(\mathbf{w}|\mathcal{M}, \mathbf{w}^0) = \frac{1}{2M}\sum_{\mu=1}^{M}\left(\frac{1}{\sqrt{N}}\mathbf{w}\cdot\mathbf{x}^\mu - y^\mu\right)^2 = \frac{1}{2MN}\sum_{\mu=1}^{M}\left(\left(\mathbf{w} - \mathbf{w}^0\right)\cdot\mathbf{x}^\mu\right)^2, \tag{2.2}$$

which is an M sample estimator of the generalisation function. The generalisation function averaged over the version space of students and the possible training sets[2] that define the version space is the *generalisation error*,

$$\epsilon_g = \left\langle \epsilon_f(\mathbf{w}|\mathbf{w}^0)\right\rangle_{\mathcal{W},\mathcal{P}}. \tag{2.3}$$

Each of the M error contributions that sum to form the test error is independently and identically distributed and, applying the central limit theorem[3], one expects that the generalisation function will be $\epsilon_{test}(\mathbf{w}|\mathcal{M}, \mathbf{w}^0) + \mathcal{O}\left(1/\sqrt{M}\right)$.

The variance due to the random training and test sets, and also the different possible choices of students from the version space is given by,

$$\text{var}\left(\epsilon_{test} : \mathcal{M}\right) = \left\langle \left(\epsilon_{test}(\mathbf{w}|\mathcal{M}, \mathbf{w}^0) - \epsilon_f(\mathbf{w}|\mathbf{w}^0)\right)^2 \right\rangle_{\mathcal{E}}.$$

In section(2.2) we show how to calculate this variance, employing these results in section(2.3) to find the "optimal" test set size and in section(2.4) to gain insight into confidence in the testing/training procedure. In section(2.5) we examine the variance of the cross-validation error, and conclude with a summary in section(2.6).

2.2 Calculating the Averages - Geometrical approach

The P training examples constrain a student $\mathbf{w}$ to lie on the hyperplane,
$H = \{\mathbf{w}|\mathbf{w}\cdot\mathbf{x}^\sigma = \mathbf{w}^0\cdot\mathbf{x}^\sigma, \sigma = 1..P\}$. The version space is given by VS$= H \cap S$, where S is the spherical constraint, $\mathbf{w}\cdot\mathbf{w} = N$. The space of vectors that satisfy the intersection of a hyperplane and a hypersphere is a hypersphere of the dimension of the hyperplane (see

[1] Note that the indices σ and μ refer to training and test set inputs respectively.

[2] Isotropy of the problem in weight space ensures that $\epsilon_f(\mathbf{w}|\mathbf{w}^0)$ is the same for all teachers $\mathbf{w}^0$. We will, however, later include an average over $\mathbf{w}^0$ alongside the training set average for calculational simplifications.

[3] The central limit theorem holds for any input distribution.

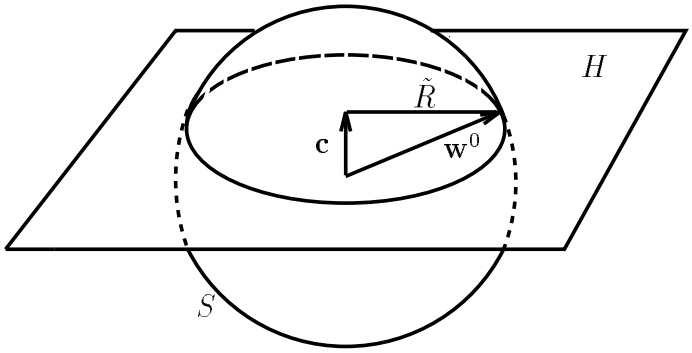

Figure 2.1. In three dimensions each training example is associated with a plane forming, for P examples, the subspace H (drawn here for only one example). The version space is the intersection of H with the spherical constraint, S. In the above example, this results in a circular version space. In general, the resulting version space is a hypersphere of dimension $N-P$, centred at $\mathbf{c} = \mathbf{w}^0 - \mathbf{P}\mathbf{w}^0$, where $\mathbf{P}$ is the projection onto the subspace H, and the radius of the version space is $\tilde{R} = |\mathbf{P}\mathbf{w}^0|$.

figure(2.1)). After training on P examples, therefore, the VS is a hypersphere of dimension $N-P$. For $\alpha = P/N > 1$, provided that at least N of the training examples are linearly independent, which is the generic case, the VS collapses to a single point, *i.e.,* the teacher weight vector, and the test errors become zero. We therefore limit the analysis to the case $\alpha < 1$.

2.2.1 Version Space Averages

We illustrate the techniques used in the calculation of the test error variance by demonstrating how to perform the averages over the test error, which itself is needed for the variance calculation. Equation (2.2) can be written as,

$$\epsilon_{test}(\mathbf{w}|\mathcal{M},\mathbf{w}^0) = \frac{1}{2MN}\sum_{\mu=1}^{M}(\mathbf{x}^\mu)^{\mathbf{T}}\left(\mathbf{w}-\mathbf{w}^0\right)\left(\mathbf{w}-\mathbf{w}^0\right)^{\mathbf{T}}\mathbf{x}^\mu,$$

where $(\mathbf{x}^\mu)^{\mathbf{T}}$ denotes the transpose of the vector $\mathbf{x}^\mu$. When written in component form, averages over the VS, $\langle..\rangle_{\mathcal{W}}$, are concerned only with the quantity

$$\left\langle\left(w_i - w_i^0\right)\left(w_j - w_j^0\right)\right\rangle_{\mathcal{W}}. \tag{2.4}$$

In order to understand the geometrical picture, it may be helpful to consider a specific example which we draw schematically in figure(2.1). For the perceptron of dimension three, the students (and teacher) are constrained to lie on a sphere of radius $\sqrt{3}$. One training example pair $(\mathbf{x}, y)$ forms a plane with normal in the direction of $\mathbf{x}$, a perpendicular distance $y/|\mathbf{x}|$ from the origin. This plane intersects the sphere to form a circular VS whose centre is along the direction of $\mathbf{x}$, a distance $y/|\mathbf{x}|$ from the origin. As the endpoint of the vector $\mathbf{w}^0$ lies on the VS, the centre of the VS can be found by subtracting from $\mathbf{w}^0$ its projection onto the plane. For the N dimensional case, the centre of the version space is given by $\mathbf{c} = \mathbf{w}^0 - \mathbf{P}\mathbf{w}^0$, where $\mathbf{P}$ projects onto the

subspace H.[4] Decomposing the vectors $\mathbf{w}$ and $\mathbf{w}^0$ into the centre of the VS and remaining contributions,

$$w_i = c_i + r_i, \quad w_i^0 = c_i + r_i^0,$$

and exploiting the symmetry of the hypersphere, equation(2.4) becomes simply,

$$\langle r_i r_j \rangle_{\mathcal{W}} + r_i^0 r_j^0. \tag{2.5}$$

Details for the calculation of the first term of the above equation are given in appendix(2.7.1) at the end of the chapter, which lead to the result,

$$\left\langle \epsilon_{test}(\mathbf{w}|\mathcal{M}, \mathbf{w}^0) \right\rangle_{\mathcal{W}} = \frac{1}{2MN} \sum_{\mu=1}^{M} \left(\frac{\mathbf{w}^0 \mathbf{P} \mathbf{w}^0}{N-P} \left(\mathbf{x}^\mu\right)^{\mathbf{T}} \mathbf{P} \mathbf{x}^\mu + \left(\left(\mathbf{x}^\mu\right)^{\mathbf{T}} \mathbf{P} \mathbf{w}^0 \right)^2 \right). \tag{2.6}$$

2.2.2 Teacher and Data Set Averages

Due to the isotropy of the teacher and student spaces, an average over teacher vectors is not strictly required; calculational simplifications are achieved, however, by including one. We thus average equation(2.6) over all teachers $\mathbf{w}^0$ satisfying the spherical constraint, $\left(\mathbf{w}^0\right)^{\mathbf{T}} \mathbf{w}^0 = N$. Evaluating the averages $\left\langle \left(\mathbf{w}^0\right)^{\mathbf{T}} \mathbf{P} \mathbf{w}^0 \right\rangle_{\mathcal{W}^0}$ and $\left\langle \left(\left(\mathbf{x}^\mu\right)^{\mathbf{T}} \mathbf{P} \mathbf{w}^0 \right)^2 \right\rangle_{\mathcal{W}^0}$, by using the result $\left\langle w_i^0 w_j^0 \right\rangle_{\mathcal{W}^0} = \delta_{ij}$, we obtain,

$$\left\langle \epsilon_{test}(\mathbf{w}|\mathcal{M}, \mathbf{w}^0) \right\rangle_{\mathcal{W},\mathcal{W}^0} = \frac{1}{2MN} \sum_{\mu=1}^{M} \left(\mathbf{x}^\mu\right)^{\mathbf{T}} \mathbf{P} \mathbf{x}^\mu \left(\frac{\mathrm{Tr}\mathbf{P}}{N-P} + 1 \right).$$

Tr$\mathbf{P}$ is the trace of the projection matrix $\mathbf{P}$, which is simply the dimension of the space onto which it projects, in this case that of the version space, Tr$\mathbf{P} = N - P$. We now perform the average over the possible test set input examples $\mathbf{x}^\mu$. Since the inputs are normally distributed, $\left\langle \left(\mathbf{x}^\mu\right)^{\mathbf{T}} \mathbf{P} \mathbf{x}^\mu \right\rangle_{\mathcal{M}} = \mathrm{Tr}\mathbf{P}$, and one obtains the well known result [SST92],

$$\epsilon_g \equiv \left\langle \epsilon_{test}(\mathbf{w}|\mathcal{M}, \mathbf{w}^0) \right\rangle_{\mathcal{W},\mathcal{W}^0,\mathcal{M}} = 1 - \alpha$$

where $\alpha = P/N$. Learning can be pictured in the following way: the root mean square distance of the centre of the hypersphere from the origin increases as $\sqrt{N\alpha}$; the radius decreases as $\sqrt{N(1-\alpha)}$, the VS 'shrinking' around the teacher weight vector.

2.2.3 Test Error Variance Results

The calculation of the test error variances follows on from the arguments presented in the previous two sections. Details are given in appendix(2.7.2) at the end of the chapter, and one obtains, for $P < N$:

$$\mathrm{var}\left(\epsilon_{test} : \mathcal{M}\right) = \frac{2\left(2 + N - P\right)\left(1 + N - P\right)}{MN(N+2)}. \tag{2.7}$$

[4] The projection matrix $\mathbf{P}$ can be found explicitly by orthonormalising the training set of input vectors $\{\mathbf{x}^\sigma\}$ to form an orthonormal set $\{\hat{\mathbf{x}}^\sigma\}$, from which $P_{ij} = \delta_{ij} - \sum_{\sigma=1}^{P} \hat{x}_i^\sigma \hat{x}_j^\sigma$.

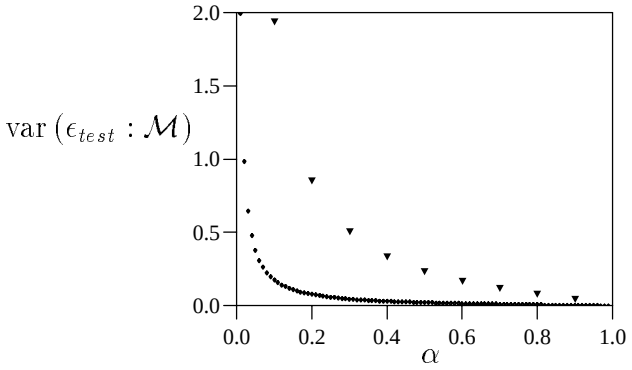

Figure 2.2. The variance, var $(\epsilon_{test} : \mathcal{M})$, in the test error induced by the random test sets, the version space, and the training sets. The triangles represent a perceptron of dimension $N = 10$, and the dots $N = 100$. The test set size is equal to the training set size.

$$\text{var}\left(\epsilon_{test} : \mathcal{E}\right) = \frac{(2+N-P)\,(1+N-P)\,(2+M)}{MN\,(N+2)} - \left(1-\frac{P}{N}\right)^2 . \tag{2.8}$$

Where var $(\epsilon_{test} : \mathcal{M})$ is the (average) test error variance over test sets (*cf.* section(1.2.1)), and var $(\epsilon_{test} : \mathcal{E})$ is the test error variance over all sources of randomness (*i.e.,* the weight space and dataset). For $M, P \propto N \gg 1$, one can readily verify that the variances are $\mathcal{O}\left(N^{-1}\right)$, and thus zero in the thermodynamic limit $(N \to \infty)$, which is the underlying assumption of self-averaging in statistical mechanics calculations. For $P = N$, there is zero variance in the test errors since the VS collapses to a single point. We shall primarily be interested in the deviation of the test error from the average test error over a fixed training set, and concentrate therefore on var $(\epsilon_{test} : \mathcal{M})$. We note, parenthetically, that var $(\epsilon_{test} : \mathcal{M}) \leq$ var $(\epsilon_{test} : \mathcal{E})$, which is generically true. A more detailed discussion of the relationship between these two types of variances is given (also for more general networks) in appendix(2.9) In figure(2.2), we plot var $(\epsilon_{test} : \mathcal{M})$ as a function of α for perceptrons of dimension N=10 and N=100, with the number of test examples set to the number of training examples $(M = P)$. For small values of α, there is a correspondingly large test error variation, decreasing monotonically with increasing α. The variance of the test error for α close to 1 is small, indicating that students in the version spaces generated by random training sets have almost equal test errors. For large N, var $(\epsilon_{test} : \mathcal{M})$ decays as $2(1-\alpha)^2/(\alpha N)$ which, for fixed α, scales with $1/N$.

2.3 Optimal test set size

We now turn our attention to the partitioning of a data set of examples into a training set and a test set. That is, given a data set of L elements, how many elements should be assigned to the training set, and the rest to the test set, given that we wish to produce a student with a low generalisation function.

A student that has a low test error will not necessarily have a low generalisation function, *unless* we can show that the test error will (at least on average) be close to the generalisation function. (Using nearly all the dataset examples for training may result in a student with a

low test error when tested on the remaining few examples, but our confidence that the test error is representative of the generalisation function is low because so few examples were used in the testing procedure). By applying the central limit theorem, the difference between the generalisation function and the test error will be distributed in a gaussian manner with mean zero[Fel70], where the standard deviation of this distribution is over the realisations of the test set. This means, for example, that with probability 0.84, the generalisation function will not lie more than one standard deviation above the test error. This bound, however, is dependent on the actual test error value, whereas we will here be interested in the typical upper bound when one takes into account the version space and different possible training sets. We therefore represent the test error by its average (the generalisation error) and the standard deviation over test sets alone by that over test sets, students, and training sets. Thus doing, we define the average probabilistic *upper bound* on the generalisation function as,

$$\epsilon_{ub}(M|L) = \epsilon_g + \tau\sqrt{\mathrm{var}\left(\epsilon_{test} : \mathcal{M}\right)}. \tag{2.9}$$

Setting $\tau = 1$, we will be 84% confident that the generalisation function will, on average, not be more than one standard deviation above the test error. Similarly, for $\tau = 2$, we will be 98% confident that $\epsilon_f(\mathbf{w}|\mathbf{w}^0)$ will, on average, be less than two standard deviations above the test error[5]. If we fix the size of the data set, L, we can consider the variance and generalisation error as a function of the test set size, M, the training set size being given by $P = L - M$. In figure(2.3) the generalisation error and standard deviation are plotted for a perceptron of dimension $N = 400$ and data set size $L = 200$. For small M, the standard deviation is large and the generalisation error is small, the perceptron having been trained on a relatively large number of examples. This situation reverses as M is increased, which gives rise to a minimum in the upper bound, $\epsilon_{ub}(M|L)$ for $M = M^*$. We note from figure(2.3) that this is at $M^* = 24$ for $\tau = 1$. The dependence of M^* on N and L is rather complicated; however, in the limit of large N and setting $L = \alpha_{tot}N$, an asymptotic expansion in N reveals the following scaling law for the optimal test set size,

$$M^* \sim \frac{1}{2}\left(2\tau\left(1-\alpha_{tot}\right)N\right)^{\frac{2}{3}}. \tag{2.10}$$

Or, writing this as the optimal fraction of the data set to be used for testing,

$$\frac{M^*}{L} \sim \frac{\left(\tau\left(1-\alpha_{tot}\right)\right)^{\frac{2}{3}}}{\alpha_{tot}}\frac{1}{(2N)^{\frac{1}{3}}}.$$

For fixed τ, α_{tot}, the optimal test fraction tends to zero as N increases to infinity. Even though the fraction of test examples tends to zero, there is still a very large number of test examples, enough that the test error will be close to the generalisation function. For fixed N, τ, the optimal test fraction tends to zero as α_{tot} approaches 1 as the perceptron then has increasingly more data at its disposal to learn the teacher, which results increasingly in a restriction on the possible student weights, and therefore a restriction on the variance of the test errors. For τ tending to zero, we recover the normal case in which we utilise all the data set as training examples, regardless of test error fluctuations.

[5]Here we have employed standard results about the percentage of the normal curve less than a certain number of standard deviations from the mean.

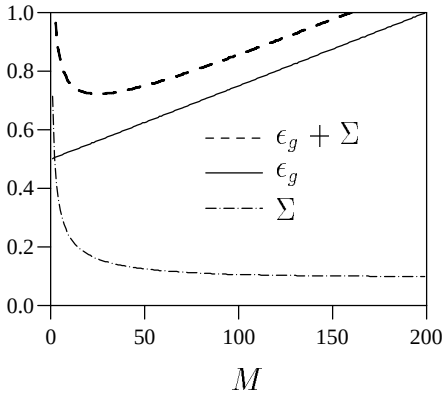

Figure 2.3. The standard deviation $\Sigma = \mathrm{var}\left(\epsilon_{test} : \mathcal{M}\right)^{1/2}$, generalisation error ϵ_g and upper bound ($\tau = 1$) plotted against the test set size M. The dimension of the perceptron is $N = 400$, with data set size $L = 200$, and test set size $P = L - M$. As M increases, the deviation of errors decreases, whereas the generalisation error increases (as the number of training examples decreases). The value, M^* for which the upper bound is minimised represents the optimal test set size; in this case, $M^* = 24$.

2.4 Confidence in the training/testing procedure

One way to quantify confidence in the training/testing procedure for a learning machine is to compare the result of training and testing the machine on different sets, and seeing whether or not the test errors are close. We have in mind the following scenario.

We divide a $2P$-dimensional data set into two disjoint sets of equal cardinality - a 'left' and a 'right' half. Perceptron $\mathbf{w}_1$ is trained on the right set and then tested on the left, and $\mathbf{w}_2$ is trained on the left set and tested on the right. This generates two test errors, $\epsilon^{(1)}_{test}$ and $\epsilon^{(2)}_{test}$ for perceptrons $\mathbf{w}_1$ and $\mathbf{w}_2$ respectively. If the difference between $\epsilon^{(1)}_{test}$ and $\epsilon^{(2)}_{test}$ is large, our confidence in the training/testing procedure would be small. A quantity that measures the mean square difference between the test errors of the perceptrons is

$$\Delta^2 = \left\langle \left(\epsilon^{(1)}_{test} - \epsilon^{(2)}_{test}\right)^2 \right\rangle_{\mathcal{W}_1,\mathcal{W}_2,\mathcal{L}} = 2\left(\mathrm{var}\left(\epsilon^{(1)}_{test} : \mathcal{E}\right) - \mathrm{cov}(\epsilon^{(1)}_{test}, \epsilon^{(2)}_{test})\right),$$

where we have defined the covariance,

$$\mathrm{cov}(\epsilon^{(1)}_{test}, \epsilon^{(2)}_{test}) = \left\langle \left(\epsilon^{(1)}_{test} - \epsilon_g\right)\left(\epsilon^{(2)}_{test} - \epsilon_g\right) \right\rangle_{\mathcal{W}_1,\mathcal{W}_2,\mathcal{L}}.$$

In figure(2.5), we present numerical simulations performed to calculate $\mathrm{cov}(\epsilon^{(1)}_{test}, \epsilon^{(2)}_{test})$ for $N = 64$, justifying the theoretical prediction detailed later in section(2.5.1). These covariances were found to be an order of magnitude smaller than the variances calculated from the results of section(2.2.3). The results in figure(2.4) demonstrate how the root mean square difference between $\epsilon^{(1)}_{test}$ and $\epsilon^{(2)}_{test}$ decreases as the data set size increases. For small α, there is minimal information supplied to both perceptrons about the teacher and the two students vary greatly in their errors. As α increases, the version spaces become more constrained around the teacher

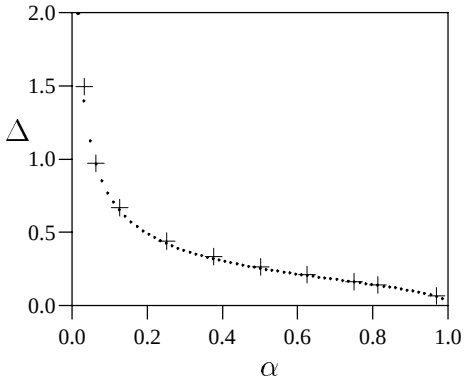

Figure 2.4. The crosses are the simulated values of Δ, the root mean square deviation between the two test error values generated by training perceptron (1) on the right half of the data set and testing it on the left, and vice versa for perceptron (2). The perceptron is of dimension $N = 64$. The dots are the approximation to Δ which neglects the covariance term $\mathrm{cov}(\epsilon_{test}^{(1)}, \epsilon_{test}^{(2)})$.

and the degree of belief in the training/testing procedure increases. As the dimension, N, of the perceptron is increased, Δ^2 scales with $1/N$.

Training and testing more than one student on the dataset is used extensively in practice in order to gather information about the performance of students trained with a certain algorithm, and this leads naturally onto the topic of cross-validation, discussed in the following section.

2.5 Cross-Validation

Cross-validation (CV) is a widely used statistical technique that can be employed to estimate the generalisation ability of a set of models, each model being trained and tested on the same finite data set [Sha93, Sto74, BFOS84]. From the set of possible models, the model which has the lowest CV error is then chosen as the "best" model, and a single student from this model trained on the whole dataset, and used as the single best estimator. In chapter(3), we discuss the problem of model selection; for the moment, however, we assume that a particular model has been selected, and concentrate on how to use the dataset in order to predict the error that a student from the selected model will make.

Leave out M cross-validation: CV(M)

Consider a set $\mathcal{L}$ containing L data points. This dataset is then partitioned into two disjoint subsets - a test set, $\mathcal{M}$ of dimension M, and a training set $\mathcal{P}$ of dimension $L - M$. In general, there are $\binom{L}{M}$ possible partitions, which we label by i and the size of each test set is given by $M = L/V$ for some chosen number of divisions of the data set, V. For example, for the case $V = 4$, we divide the dataset into four equal parts, which form 4 test sets of equal cardinality, $\mathcal{M}(1)..\mathcal{M}(4)$, which we depict schematically,

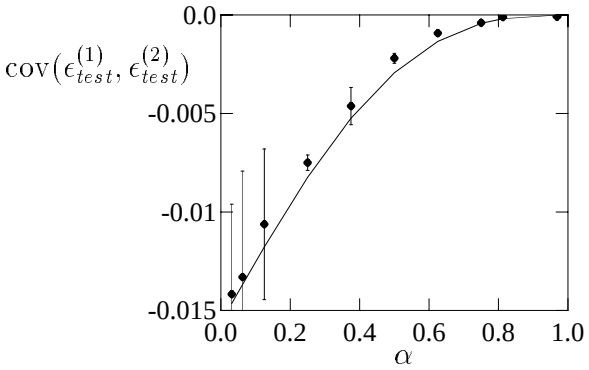

Figure 2.5. The covariance of two test errors under the scheme described in section(2.4) (Non-overlapping test sets of equal cardinality). The perceptron is of dimension $N = 64$. The dots are the simulation values plotted with one standard deviation error bars. The solid line is the theoretical value, $(\alpha^2 - 1)(1 - \alpha)^2/64$. Note that error bars are largest for small α as in this region, the version space is largest, resulting in relatively poor statistics.

$$\mathcal{L} = \boxed{\mathcal{M}(1)} \boxed{\mathcal{M}(2)} \boxed{\mathcal{M}(3)} \boxed{\mathcal{M}(4)}$$

The complement of each test set $\mathcal{M}(i)$ forms the training set $\mathcal{P}(i) = \mathcal{L} - \mathcal{M}(i)$. A student $s(i)$ of the model under consideration (*e.g.,* the spherical linear perceptron) is then trained on $\mathcal{P}(i)$ and tested on $\mathcal{M}(i)$, forming the test error $\epsilon(i)$. This is repeated for S students, from which the CV error is then found,

$$\epsilon^{CV} \equiv \frac{1}{S} \sum_{i=1}^{S} \epsilon(i). \tag{2.11}$$

The rationale behind this procedure is that the resulting CV error is an unbiased estimator of the generalisation function, with a variance less than that from only a single student.
The increase in computational expense incurred from (re)training S students is not necessarily a major factor if data is scarce. However, training students on all the possible $\binom{L}{M}$ partitions is typically prohibitive, and $S \ll \binom{L}{M}$ partitions are chosen either randomly, or selected to minimise their mutual overlap. We shall investigate different schemes for choosing the S partitions. Previous studies along these lines have been made by Burman[Bur89] who looks at the effect on the generalisation error and the variance of the test error for different numbers of divisions.

The variance of the CV error ϵ^{CV} that we shall ultimately calculate is the variance of this estimate over all sources of randomness (which is an upper bound on all the other possible variances).

By training S students on independently, identically distributed (iid) examples, and from

the definition (2.11), one obtains the decomposition[6],

$$\mathrm{var}\left(\epsilon^{CV} : \mathcal{E}\right) = \frac{1}{S}\mathrm{var}\left(\epsilon\left(1\right) : \mathcal{E}\right) + \left(1 - \frac{1}{S}\right)\mathrm{cov}(\epsilon\left(1\right), \epsilon\left(2\right) : \mathcal{E}). \tag{2.12}$$

(We typically shall drop the notational dependence on $\mathcal{E}$ for the remainder of our analysis of cross-validation errors). Note that, as the examples are iid, this variance is not dependent on the student number i, and we choose, without loss of generality, student $i = 1$ (and similarly, we choose students (1) and (2) for the covariance). Using the general result $\mathrm{cov}(\epsilon\left(1\right), \epsilon\left(2\right) : \mathcal{E}) \leq \mathrm{var}\left(\epsilon\left(1\right) : \mathcal{E}\right)$ in (2.12) we obtain immediately,

$$\mathrm{var}\left(\epsilon^{CV} : \mathcal{E}\right) \leq \mathrm{var}\left(\epsilon(1) : \mathcal{E}\right). \tag{2.13}$$

This result motivates the use of CV to improve the accuracy of prediction of errors over using single estimates. We shall primarily be aiming to quantify the improvement that one can expect from using the CV procedure over using a single student, and also to quantify the abilities of different CV schemes to minimize their variance. Other methods for estimating data dependencies such as the Akaike information criterion[Aka74], the jackknife, and bootstrap [Efr83, Efr86] are asymptotically equivalent to leave-one-out cross-validation, CV(1).

Using CV to estimate errors

Ideally, we would like to estimate the generalisation function $\epsilon_f(i)$ of a particular student - that is, the expected error that a perceptron trained on a set $\mathcal{P}(i)$ will make on a random test example. In order to estimate the proximity of the CV estimate, we define,

$$\Psi^2 = \left\langle \left(\epsilon^{CV} - \epsilon_f(1)\right)^2 \right\rangle_{\mathcal{L}}, \tag{2.14}$$

where the average is taken over all data sets, $\mathcal{L}$. By simply adding and subtracting ϵ_g in (2.14), we obtain,

$$\Psi^2 = \left\langle \left(\epsilon^{CV} - \epsilon_g\right)^2 \right\rangle_{\mathcal{L}} + \left\langle \epsilon_f(1)^2 \right\rangle_{\mathcal{L}} + \epsilon_g^2 - 2\left\langle \left\langle \epsilon^{CV} \right\rangle_{\mathcal{M}} \epsilon_f(1) \right\rangle_{\mathcal{P}}. \tag{2.15}$$

At this point, however, there is a problem: Even if we choose the size of the CV training sets to be equal to the training set size of the single perceptron, there is little that can be said about the quantity $\left\langle \left\langle \epsilon^{CV} \right\rangle_{\mathcal{M}} \epsilon_f(1) \right\rangle_{\mathcal{P}} = \sum_{i=1}^{S} \left\langle \epsilon_f(i)\epsilon_f(1) \right\rangle_{\mathcal{P}} / S$ without knowledge of the correlation between the generalisation functions of perceptrons trained on different subsets of $\mathcal{L}$. Theoretically, one may be able to calculate this for the specific model under consideration. Alternatively, the approach we take here is to assume that learning has reached the stage such that there is little difference between the generalisation functions of perceptrons trained on different subsets *i.e.,* that they are almost fully correlated. Under this assumption, we write,

$$\Psi^2 \approx \left\langle \left(\epsilon^{CV} - \epsilon_g\right)^2 \right\rangle_{\mathcal{L}}. \tag{2.16}$$

Hence, in order to minimise the average square difference between the CV error and the generalisation function/error, we seek to minimise the variance of the CV error alone (with respect to the different types of CV schemes).

[6]Correlations between the examples used to train/test the different students affect only the covariance term $\mathrm{cov}(\epsilon\left(1\right), \epsilon\left(2\right) : \mathcal{E})$. Such correlations do not affect the variance term as we average over all possible datasets. When we later address using different ways of generating (correlated) training/test sets for the different students from a dataset, these differences will manifest themselves in the covariance between two test errors.

Computational cost

As all the CV schemes we shall consider are unbiased, we shall be interested in comparing simply the variance of the different schemes under a given amount of computational resource. We define the *computational cost* C of a CV algorithm to be proportional to the total number of examples that are used in the *training* of the S students,

$$C \equiv S\frac{P}{L} = S\left(1 - \frac{1}{V}\right), \tag{2.17}$$

where P/L is the fraction of examples in the dataset used to train each student, and S is the number of students trained. This definition is motivated from the experience that testing students is computationally inexpensive compared to training them. When $C = 1$, a total of L examples have been used in training all the students.

2.5.1 Non-overlapping test sets $S \leq V$ (NOCV)

In each of the following four sections, we shall calculate the variance of different cross-validation schemes, relegating details to appendices at the end of the chapter.

For $S \leq V$, we are able to partition the dataset $\mathcal{L}$ into S disjoint sections, forming S test sets, $\mathcal{M}(1), ..., \mathcal{M}(S)$. S perceptrons are then trained in the following way: Perceptron (i) is trained on $\mathcal{L} - \mathcal{M}(i)$, and tested on $\mathcal{M}(i)$, forming the test error, $\epsilon(i)$. This procedure is repeated for all the S students, $i = 1..S$ and the resulting cross-validation error is defined[7]

$$\epsilon^{NOCV} = \frac{1}{S}\sum_{i=1}^{S}\epsilon(i). \tag{2.18}$$

As the examples in the dataset are iid distributed, $\langle \epsilon(i)\,\epsilon(j)\rangle = \langle \epsilon(1)\,\epsilon(2)\rangle$ for $i \neq j$ and using this, we express the variance of the CV error as,

$$\mathrm{var}(\epsilon^{\mathrm{NOCV}}) = \frac{1}{S}\mathrm{var}(\epsilon(1)) + \left(1 - \frac{1}{S}\right)\mathrm{cov}(\epsilon(1), \epsilon(2)\,|\mathrm{NOCV}) \tag{2.19}$$

Here, $\mathrm{var}(\epsilon(1))$ is the test error variance of a single perceptron having been trained on a set of size $L(1 - 1/V)$, and tested on a set of size L/V. Looking back at equation(2.8), we note that this is a quantity that we have already calculated. The covariance can be calculated by using the replica formalism, as outlined in appendix(2.8.1).

For the case of two divisions, $V = 2$, and two students, $S = 2$, the covariance is given by (see figure(2.5) and figure(2.6)),

$$\mathrm{cov}(\epsilon(1), \epsilon(2)\,|\mathrm{NOCV}) = \frac{1}{N}\left(\alpha^2 - 1\right)(\alpha - 1)^2, \tag{2.20}$$

which is negative throughout the range of possible α values from 0 to 1. As a partial explanation of this effect, without loss of generality, we consider one of the halves of the dataset, say $\mathcal{M}(1)$, to have examples which cover the input space more thoroughly than the other half of examples, $\mathcal{M}(2)$. This means that perceptron (1) will have a higher test error than perceptron (2). Generically, one of the test errors will be higher than average, and the other lower than average, thus giving rise to a negative covariance.

[7] This is sometimes termed "v-fold cross-validation" in the statistics literature[Bur89].

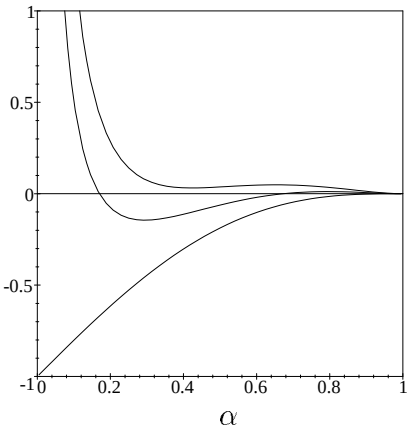

Figure 2.6. Scaled covariance $N\mathrm{cov}(\epsilon(1), \epsilon(2))$ of two test errors for the various CV schemes, for two divisions ($V = 2$). The lower curve is the case for a total of only two students (S=2) in the CV scheme, for which the optimal scheme OCV and non-overlapping scheme NOCV are the same. The middle curve is for four students, $S = 4$, plotted for the optimal scheme. The upper curve is for the Monte-Carlo scheme, which is independent of the number of students. The upper and lower curves represent limiting cases for the covariance - in the lower curve, there is no overlap between the test sets of the students, whereas for the upper curve, there is an overlap that would occur for randomly selected test sets. For the case in which the overlap between the test sets is maximal, then the covariance tends towards the variance as then the test and training sets of the two students are the same. Hence, as the test set overlap α_{12} increases, we expect an increase in the value of the covariance.

2.5.2 Random Partitioning, or Monte Carlo CV (MCCV)

For a set of L examples, and V divisions, there are $\binom{L}{L/V}$ different partitions of the dataset that could be constructed. As this is typically exponentially large, a computational compromise is given by randomly selecting a subset of the set of all the possible partitions[8]. (Note that the maximal number of partitions limits the reduction in the CV variance that can be obtained.) However, randomly selecting training/test sets for the student perceptrons may not be the optimal strategy as, with some probability, the same examples will be assigned to different students. We write the probability that an example lies in the test set of two student perceptrons as α_{12}, which for the random partitioning case gives $\alpha_{12} = V^{-2}$.

In figure(2.6) we plot the covariances of two students trained under different CV schemes and for different numbers of students. We see that the random scheme MCCV (upper curve) yields positive covariances, in contrast to the NOCV case (lower curve), where the covariance is negative. These results are difficult to interpret; however, some intuition may be gained from considering the extrapolation of the small test set overlap given by MCCV to that of maximal test set overlap, for which the covariance tends to the variance - a positive value. The middle curve of figure(2.6) is explained later in section(2.5.4).

[8]This scheme is termed "repeated v-fold cross-validation" by Burman[Bur89].

2.5.3 Block CV scheme (BCV)

The non-overlapping partition scheme NOCV is ideal for $S \leq V$ as the students will then be tested on disjoint sets, which will cover the input space better than overlapping sets.
The maximal number of students, however, for NOCV is $L-1$. As there is no restriction on the number of students for the random partition case, random partitioning will eventually yield a lower variance than the non-overlapping scheme for some $S > V$. This immediately brings us to consider another CV scheme, similar to the non-overlapping case, but which allows the same number of students as for the random case. We have in mind the following scenario: We randomly permute the members of the dataset, and then apply non-overlapping partitioning cross-validation (NOCV), training $S = V$ students. This procedure is repeated, each time randomly permute the dataset, and then applying non-overlapping CV. We call this scheme Block cross-validation (BCV) as the students are trained in blocks of V, such that the total number of students trained is $S = BV$.

We define the Block CV error to be the average of the NOCV errors over B blocks, namely,

$$\epsilon^{BCV} = \frac{1}{B}\sum_{i=1}^{B} \epsilon^{NOCV}(i) \tag{2.21}$$

where $\epsilon^{NOCV}(i)$, for each block i of V students, is defined in equation(2.18). The variance of the BCV error is then given (for a number of blocks, B) by,

$$\mathrm{var}\left(\epsilon^{BCV}\right) = \frac{1}{B}\mathrm{var}\left(\epsilon^{NOCV}\right) + \left(1-\frac{1}{B}\right)\mathrm{cov}(\epsilon(1),\epsilon(2)\,|\mathrm{MCCV}) \tag{2.22}$$

Writing this out more fully we get,

$$\begin{aligned}\mathrm{var}\left(\epsilon^{BCV}\right) &= \frac{1}{B}\left\{\frac{1}{V}\mathrm{var}(\epsilon(1)) + \left(1-\frac{1}{V}\right)\mathrm{cov}(\epsilon(1),\epsilon(2)\,|\mathrm{NOCV})\right\} \\ &+ \left(1-\frac{1}{B}\right)\mathrm{cov}(\epsilon(1),\epsilon(2)\,|\mathrm{MCCV})\end{aligned} \tag{2.23}$$

For the case of two divisions, $V = 2$ and two blocks, $B = 2$, we get

$$\mathrm{var}\left(\epsilon^{BCV}\right) = \frac{1}{4}\mathrm{var}(\epsilon(1)) + \frac{1}{4}\mathrm{cov}(\epsilon(1),\epsilon(2)\,|\mathrm{NOCV}) + \frac{1}{2}\mathrm{cov}(\epsilon(1),\epsilon(2)\,|\mathrm{MCCV}) \tag{2.24}$$

whereas for the random case it is,

$$\mathrm{var}\left(\epsilon^{MCCV}\right) = \frac{1}{4}\mathrm{var}(\epsilon(1)) + \frac{3}{4}\mathrm{cov}(\epsilon(1),\epsilon(2)\,|\mathrm{MCCV}). \tag{2.25}$$

Recalling the results from section(2.5.1) that the covariance under the NOCV($V = 2$) is negative, and that they were positive for the random case, we see that the variance of the BCV scheme will be lower than that for the random scheme. Indeed, using $\mathrm{cov}(\epsilon(1),\epsilon(2)\,|\mathrm{NOCV}) \leq \mathrm{cov}(\epsilon(1),\epsilon(2)\,|\mathrm{MCCV})$, it follows that $\mathrm{var}\left(\epsilon^{BCV}\right) \leq \mathrm{var}\left(\epsilon^{MCCV}\right)$. Hence for any Monte-Carlo CV scheme, we can find a block CV scheme that has a lower variance for the same computational cost[9].

[9]Provided the number of students in the random scheme is non-prime.

2.5.4 Optimal Partitioning (OCV)

The BCV scheme is an improvement over the random case because the overlap between the testsets is smaller, but can we do better - is there a way to assign the test sets to the various students that minimises their mutual overlap? The formulation of such a question leads to a straightforward linear optimisation problem. Solving this, one finds that the minimal overlap achievable is (P.Sollich, personal communication),

$$\frac{\alpha_{12}}{\alpha} = \frac{1}{V}\left(1 - \frac{S}{S-1}\right)(1 - \frac{1}{V}) + \frac{\Delta\,(1-\Delta)}{S(S-1)}, \tag{2.26}$$

where $\Delta = x - \lfloor x \rfloor$, and $x = S/V$. To construct such a partitioning we define $\epsilon(i)$ to be the fraction of examples on which exactly i students are tested and set,

$$\epsilon\,(\lfloor x \rfloor) = 1 - \Delta, \qquad \epsilon\,(\lceil x \rceil) = \Delta, \tag{2.27}$$

where $\lfloor x \rfloor$ is the nearest integer $\leq x$, and $\lceil x \rceil$ is the nearest integer $\geq x$. All other $\epsilon(i)$ are set to zero.

An example $V = 4$ and $S = 6$.

Suppose that we are going to partition the dataset into four equal parts ($V = 4$) and train 6 cross-validation students ($S = 6$). Hence $x = 1.5$ and $\Delta = 0.5$, giving $\epsilon(1) = 0.5$ and $\epsilon(2) = 0.5$, so that half of the examples are tested on only one student, and half on two students. In order to construct such a partitioning, lets say we had 12 data points, $x(1), .., x(12)$. This means that each test set will consist of 3 examples, and we assign the test sets for the 6 students as:
$\mathcal{M}(1) = \{x(1), x(2), x(3)\}$, $\mathcal{M}(2) = \{x(4), x(5), x(6)\}$,
$\mathcal{M}(3) = \mathcal{M}(4) = \{x(7), x(8), x(9)\}$, $\mathcal{M}(5) = \mathcal{M}(6) = \{x(10), x(11), x(12)\}$.
The training sets for each student are simply the complement of the test sets. In figure(2.6) we plot the covariances of individual students with $V = 2$. The graph can be interpreted as results for the optimal scheme OCV(2) with different numbers of students. The lower curve is for two students, the middle for four, and the upper curve is for an infinite number of students. The covariance essentially becomes more positive as the number of students is increased, reflecting the convergence of the OCV scheme to the random scheme as the number of students increases. In figure(2.7) we plot the (scaled by N) variance for a number of students $S = BV$ against B and V. These graphs serve as baseline values against which we shall compare other CV schemes.

Comparison of various CV schemes

In figure(2.8) we plot the errors that the Monte-Carlo CV and Block CV schemes make relative to optimal CV for two different values of α_{tot}. As the number of blocks is increased (remember that the number of students is given by $S = BV$), the performance of the optimal scheme becomes increasingly similar to both the random and block schemes. Although the relative performance for some schemes can degrade for larger datasets, the absolute values of the CV variances converge towards zero in the limit of a large amount of data. In the limit of an infinite amount of data, the three compared CV schemes will have zero variance and thus the same performance. For the same amount of computational cost, $C = B(V-1)$, BCV performs

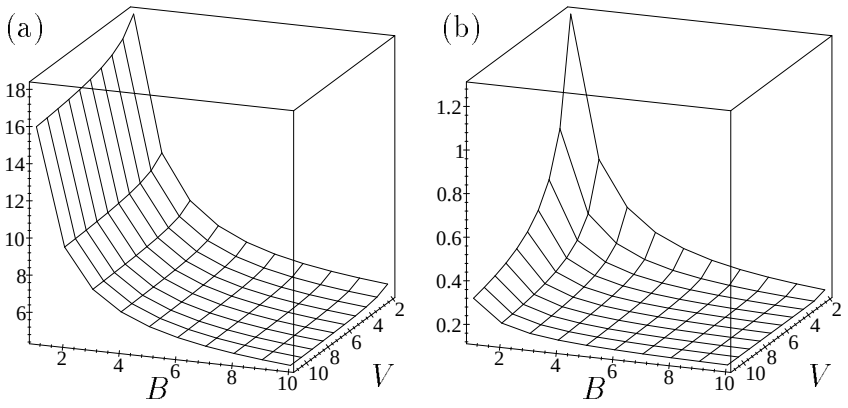

Figure 2.7. The (scaled) variance of the optimal partitioning scheme, $N\mathrm{var}\,(OCV)$, versus the number of divisions V and number of blocks B, such that the number of students is given by $S = BV$. (a) $\alpha_{tot} = 0.1$, (b) $\alpha_{tot} = 0.8$. Actual variances are given by dividing by N. Generally increasing the number of partitions and/or students lowers the variance, as does increasing the total amount of data available, α_{tot}.

better than MCCV, with a relative difference between the variance of the BCV compared to OCV of typically less than 5%. Although the MCCV scheme also performs well, it does worse than BCV, with a relative difference between the variance of the MCCV compared to OCV of typically less than 25%. From this we conclude that BCV is a very good approximation to the optimal scheme.

2.5.5 Optimising the upper bound

Having analyzed different CV schemes, we address the following question: Given that we wish to both minimise the generalisation error, and to maximise our confidence in the estimate of the error, how should we best use a CV scheme with a given amount of data and computing effort? This is essentially the same question that we asked in section(2.3) except that there we did not assume fully correlated generalisation functions, and thus restricted ourselves to training only a single perceptron[10]. Again, we form an upper bound function (*cf.* (2.9)),

$$\epsilon_{ub}^{CV} = \epsilon_g + \tau\sqrt{\mathrm{var}\,(\epsilon^{CV})}, \tag{2.28}$$

which we minimize with respect to the CV parameters S and V for a given cost, $C = S(1-1/V)$. For convenience, we set $\tau = 1$ throughout.
As there is generally only a small difference between the performance of the various CV schemes, we concentrate on the MCCV scheme as this is the most convenient to analyze. (In the large

[10]The resulting optimal test set size asymptotic scaling law, however, can be shown to be the same for estimation of the generalisation function *and* the generalisation error. The reason for this is that there is an implicit assumption that there is a large amount of data (N is large), as α_{tot} (the ratio of dataset size to the size of the perceptron) always takes a finite value. This means that the generalisation function will be close to the generalisation error, and that the scaling laws for the best approximation of the generalisation function and generalisation error will be the same.

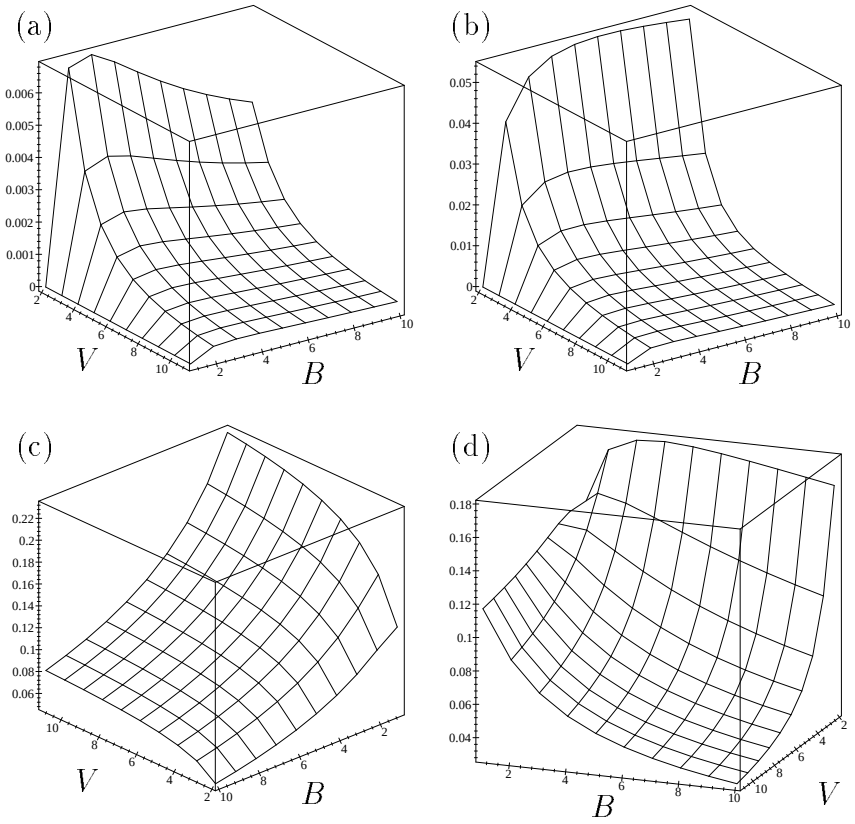

Figure 2.8. Relative performance of block CV and Monte-Carlo CV compared to optimal CV, versus the number of divisions V and blocks, B. (a),(b): $(\mathrm{var}\,(BCV) - \mathrm{var}\,(OCV))/\mathrm{var}\,(OCV)$ (a) $\alpha_{tot} = 0.1$, (b) $\alpha_{tot} = 0.8$. Figs. (c),(d): $(\mathrm{var}\,(MCCV) - \mathrm{var}\,(OCV))/\mathrm{var}\,(OCV)$ (c) $\alpha_{tot} = 0.1$, (d) $\alpha_{tot} = 0.8$. Note that in Fig.(d) the axes have been rotated. Although the dependence on the number of divisions is not straightforward, as the number of blocks is increased, the relative performance improves because optimal CV becomes more like Monte-Carlo CV.

N limit, there will essentially be no difference between the CV schemes.)
As in section(2.3), we empirically find that there is a $2/3$ power law scaling for the optimal test set size scaling. Upon making such a scaling Ansatz, we find the prefactor for the optimal number of divisions:

$$\mathrm{V}^* = (2C)^{\frac{1}{3}} \frac{\alpha_{tot}}{(1-\alpha_{tot})^{\frac{2}{3}}} N^{\frac{1}{3}}. \tag{2.29}$$

Alternatively, we can write this as an optimal test set size,

$$M^* = \left(\frac{(\alpha_{tot}-1)\,N}{4C^2} \right)^{\frac{2}{3}} \tag{2.30}$$

Comparing this with the optimal test set size we found for the single student (equation(2.10)), we see that,

$$M^*(CV) = \frac{1}{C^{\frac{1}{3}}} M^*(OTSS), \tag{2.31}$$

where $M^*(OTSS)$ is the optimal test set size calculated for a single student. We can also write,

$$\mathrm{V}^*(CV) = C^{\frac{1}{3}} \mathrm{V}^*(OTSS). \tag{2.32}$$

This could be viewed in a rather pessimistic light - namely that, for using say twice the amount of computing resource ($C = 2$) in the CV procedure, we have reduced the amount of examples required for testing by a factor of only $1/2^{1/3} = 0.79$. Given that we know that the relationship between the error and test set size is linear, it appears that a great deal more effort yields little in the way of improved performance.
There are interesting comparisons to be made between this work and that of Burman[Bur89], who (numerically) researches the best number of partitions to use for a linear student learning a quadratic teacher rule, although Burman does not explicitly optimise a measure of the bias-variance tradeoff by introducing a quantity such as the probabilistic upper bound. Although there is no general prescription given for the best number of partitions, Burman advises a number greater than $V = 2$. We find, from equation(2.29) that the number of divisions should scale with the $1/3$ power of the computational cost (normalised number of examples used in training).

2.6 Summary

We have explicitly calculated the variance in the test error of a linear N-dimensional spherical perceptron and found that it decays with the system size N as $1/N$ for a number of test examples and training examples proportional to N. Furthermore, the variance decreases monotonically to zero as the number of training examples approaches the system size. Using these variance results, we found the optimal test set size M^*, defined by minimising the average upper bound on the generalisation function given the test error. That is, for a data set of dimension L, an upper bound on the expected error that a student perceptron will make on a random test example by training on $L - M$ and testing on M examples, is minimised for $M = M^*$. For large N, M^* scales with $N^{2/3}$. A simple measure of the confidence in the training/testing

procedure was given, being the difference between the test error values for two identical perceptrons trained and tested on the same data set. This difference necessarily decays to zero as the number of training examples increases.
We have examined the performance of various cross-validation schemes for estimating the generalisation error. We found that there was little difference between the optimal scheme and the block cross-validation scheme (less than 5% difference), the Monte-Carlo scheme performing worst (less than 25% difference). Extensions to this work for the case of noise and weight decay are presented in chapter(3), and to non-linear systems in chapter(4).

2.7 Appendix: Geometrical approach

The first two appendices deal with the geometrical approach to calculating the variance of the test error.

2.7.1 Appendix: Version Space averages

For a single test example, the average of the test error $\epsilon_{test}(\mathbf{w}|\mathcal{M},\mathbf{w}^0)$ over the VS can be written,[11]

$$\frac{1}{2MN}x_i x_j\left(\langle r_i r_j\rangle_{\mathcal{W}} + r_i^0 r_j^0\right),$$

where $\mathbf{r}=\mathbf{w}-\mathbf{c}$, $\mathbf{c}$ is the centre of the VS, and $\mathbf{r}^0=\mathbf{P}\mathbf{w}^0$.
In order to perform the VS average, we transform the coordinate system, under a rotation matrix $\mathbf{R}$, to express the hyperspherical VS in canonical coordinates, $\widetilde{\mathrm{VS}}:\tilde{r}_1^2+\cdots+\tilde{r}_{N-P}^2=\tilde{R}^2$ where $\tilde{R}^2=\mathbf{w}^0\mathbf{P}\mathbf{w}^0$. Then

$$x_i x_j\langle r_i r_j\rangle_{VS} = \tilde{x}_c\tilde{x}_d R_{ic}R_{jd}R_{ia}R_{jb}\langle\tilde{r}_a\tilde{r}_b\rangle_{\widetilde{VS}}.$$

In the canonical system,

$$\langle\tilde{r}_a\tilde{r}_b\rangle_{\widetilde{VS}} = \frac{\tilde{R}^2}{N-P}\tilde{\delta}_{ab},$$

where,

$$\tilde{\delta}_{ab}=\begin{cases}\delta_{ab} & a,b\le N-P\\ 0 & otherwise.\end{cases}$$

For a rotation matrix, $R_{ic}R_{ia}=\delta_{ac}$, hence,

$$x_i x_j\langle r_i r_j\rangle_{VS} = \frac{\tilde{R}^2}{N-P}\tilde{x}_a\tilde{\delta}_{ab}\tilde{x}_b = \frac{\tilde{R}^2}{N-P}\mathbf{x^T}\mathbf{P}\mathbf{x}. \tag{2.33}$$

Using the definition $\mathbf{r}^0=\mathbf{P}\mathbf{w}^0$, we have

$$x_i x_j r_i^0 r_j^0 = \left(\mathbf{x}^T\mathbf{P}\mathbf{w}^0\right)^2.$$

Generalising the above argument to the case of M test inputs gives equation (2.6).

[11] The summation convention will be adhered throughout.

2.7.2 Appendix: Averaging the square generalisation function

The test error variance (2.1) can be written,

$$\text{var}\left(\epsilon_{test} : \mathcal{M}\right) = \left\langle \epsilon_{test}(\mathbf{w}|\mathcal{M}, \mathbf{w}^0)^2 \right\rangle_{\mathcal{M},\mathcal{W},\mathcal{P}} - \left\langle \epsilon_f(\mathbf{w}|\mathbf{w}^0)^2 \right\rangle_{\mathcal{W},\mathcal{P}},$$

and we demonstrate how to calculate the second term $\langle \epsilon_f(\mathbf{w}|\mathbf{w}^0)^2 \rangle_{\mathcal{W},\mathcal{P}}$. The first term can be calculated by employing similar techniques. A straightforward gaussian integration gives the generalisation function as $1 - \left(\mathbf{w}^T\mathbf{w}^0\right)/N$ and squaring this and averaging over the version space gives,

$$\left\langle \epsilon_f(\mathbf{w}|\mathbf{w}^0)^2 \right\rangle_{\mathcal{W}} = -1 + 2\frac{1}{N}\left(\mathbf{w}^0\right)^{\mathbf{T}}\mathbf{P}\mathbf{w}^0 + \frac{1}{N^2}\left(\left\langle \left(\mathbf{r}^T\mathbf{w}^0\right)^2 \right\rangle_{\mathcal{W}} + \left(\mathbf{c}^T\mathbf{w}^0\right)^2\right),$$

where, as before, $\mathbf{r} = \mathbf{w} - \mathbf{c}$, and $\mathbf{c} = \mathbf{w}^0 - \mathbf{P}\mathbf{w}^0$. The term in the above equation that still needs to be explicitly averaged over the version space can be calculated by employing equation (2.33), replacing $\mathbf{x}$ with $\mathbf{w}^0$. Further performing a teacher average leads to the equation[12],

$$\left\langle \epsilon_f(\mathbf{w}|\mathbf{w}^0)^2 \right\rangle_{\mathcal{W},\mathcal{W}^0} = \frac{N-P+1}{N^2\left(N-P\right)}\left\langle \left(\left(\mathbf{w}^0\right)^{\mathbf{T}}\mathbf{P}\mathbf{w}^0\right)^2 \right\rangle_{\mathcal{W}^0}$$

Writing the average in the above equation in component form, we need to find,

$$\left\langle w_j^0 w_k^0 w_p^0 w_q^0 \right\rangle_{\mathcal{W}^0} P_{jk} P_{pq}.$$

We show below that for a spherical constraint,

$$\left\langle w_j^0 w_k^0 w_p^0 w_q^0 \right\rangle_{\mathcal{W}^0} = \frac{N}{N+2}\left(\delta_{jk}\delta_{pq} + \delta_{jp}\delta_{kq} + \delta_{jq}\delta_{kp}\right), \tag{2.34}$$

which gives,

$$\left\langle \epsilon_f(\mathbf{w}|\mathbf{w}^0)^2 \right\rangle_{\mathcal{W},\mathcal{W}^0} = \frac{N-P+1}{N\left(N-P\right)\left(N+2\right)}\left[\left(\text{Tr}\mathbf{P}\right)^2 + 2\ \text{Tr}\left(\mathbf{P}^2\right)\right].$$

The final expression for $\langle \epsilon_f(\mathbf{w}|\mathbf{w}^0)^2 \rangle_{\mathcal{W},\mathcal{W}^0,\mathcal{P}}$ is obtained by using $\text{Tr}\mathbf{P} = N - P$ in the above expression.

An elementary derivation of equation (2.34) is given by noting that the second factor follows from symmetry arguments, as only even power combinations of the teacher weight vector components contribute. The prefactor can be obtained by considering[13],

$$N^2 = \langle \mathbf{w}^4 \rangle_{\mathcal{W}^0} = N\langle w_1^4 \rangle_{\mathcal{W}^0} + N(N-1)\langle w_1^2 w_2^2 \rangle_{\mathcal{W}^0},$$

for which one can then explicitly calculate,

$$\langle w_1^4 \rangle_{\mathcal{W}^0} = N^2 \frac{\int_0^\pi d\theta \cos^4\theta \sin^{N-2}\theta}{\int_0^\pi d\theta \sin^{N-2}\theta} = \frac{3N}{N+2},$$

where, w_1 and w_2 are simply two independent directions. We note that for the case of a unit variance, zero mean gaussian measure, $\langle w_1^2 w_2^2 \rangle_{\mathcal{W}^0} = 1$, such that the difference between a spherical and a gaussian measure is $\mathcal{O}\left(N^{-1}\right)$, disappearing in the large N.

[12] The teacher space average is over the constraint that the $\mathbf{w}^0$ vectors are of length $\sqrt{N}$.

[13] We drop the teacher '0' index on teacher components raised to some power.

2.8 Appendix: Statistical Mechanics Formalism

The following appendices relate to the calculation of the covariance of two cross validation errors. The general formalism relating to these calculations is presented in appendix(A).

2.8.1 Non-overlapping test sets CV

In appendix(A), we construct a double replica formalism that allows general (co)variances to be calculated. In order to find the CV covariance, we consider the dataset as the union of the V disjoint test sets, $\mathcal{M}(1),...,\mathcal{M}(V)$. This means that we can write, explicitly, for perceptron (1), that $\mathcal{P}(1) = \mathcal{M}(1) \cup \mathcal{M}(2)... \cup \mathcal{M}(V-1)$. Similarly, for perceptron (2), we have, $\mathcal{P}(2) = \mathcal{M}(1) \cup \mathcal{M}(3) \cup ... \cup \mathcal{M}(V)$. Using these explicit test/training sets, we find that the required double replica free energy is,

$$F^{12} = {G_0}^{12} + \frac{\alpha_{tot}}{V}\left\{G^{12}(\gamma_1;\beta) + G^{12}(\beta;\gamma_2) + (V-2)G^{12}(\beta;\beta)\right\} \tag{2.35}$$

where ${G_0}^{12}$ and G_r^{12} are given by equations (A.32) and (A.35) respectively. Extremising this free energy gives the values of various order parameters which describe the state of the system. These take the form of overlap parameters, $q = \mathbf{w}^\rho \cdot \mathbf{w}^\sigma / N$, $R = \mathbf{w}^0 \cdot \mathbf{w}^\rho / N$, and $q_{12} = \mathbf{w}_1^\rho \cdot \mathbf{w}_2^\sigma / N$. The parameter q measures the overlap between student solutions in different replicated weight spaces, $\mathcal{W}^\rho$, and $\mathcal{W}^\sigma$. Similarly, R measures the overlap of the student weight vector with the teacher, and q_{12} measures the overlap between weights from the two different students. In principle, there is no difficulty in calculating results for the CV variance for any temperature, T. We concentrate our analysis, however, on the zero T limit as in this case analytic results are readily obtained. This also leads to the simplification $q = R$. The covariance is then found from F^{12} by differentiating with respect to the auxiliary fields γ_1 and γ_2 in the limit of vanishingly small fields and setting the order parameters to their saddle point values.

Zero temperature

At zero temperature, we have that $q = R = (1 - 1/V)\alpha_{tot}$. That is, the overlap of the students within the version space is equal to the overlap of a student from the version space with the teacher weight; the value of these overlaps increases linearly with the training set size. The value of the inter-replica overlap is given by,

$$q_{12} = q\frac{(q-1)V - 3q + 2}{(q-1)V + 1 - 2q} \tag{2.36}$$

For the case $V = 2$, we have $q_{12} = q^2 = R^2$. In the limit $V \to \infty$, we have $q_{12} = q$. A straightforward explanation of these results is found by considering the decomposition, $\mathbf{w}_i^\sigma = R\mathbf{w}^0 + \hat{\mathbf{w}}_i^\sigma$ for i=1,2 where $\hat{\mathbf{w}}_i^\sigma$ is a zero mean, random vector perpendicular to the teacher, with variance $\left\langle (\hat{\mathbf{w}}_i^\sigma)^2 \right\rangle = N(1-R^2)$, and covariance $\langle \hat{\mathbf{w}}_i^\sigma \cdot \hat{\mathbf{w}}_i^\rho \rangle = N(q-R^2)$. (This decomposition guarantees $\langle \mathbf{w}_i^\sigma \cdot \mathbf{w}^0 \rangle_{\mathcal{W}} = R$; the spherical constraint; and the intra-replica constraint.) Using this decomposition, we write the inter-replica overlap, $q_{12} = R^2 + \langle \hat{\mathbf{w}}_1^\sigma \cdot \hat{\mathbf{w}}_2^\rho \rangle$. For the case of $V = 2$, the training sets of the two perceptrons are independent, and therefore the average of $\hat{\mathbf{w}}_1^\sigma \cdot \hat{\mathbf{w}}_2^\rho$ is simply zero, leaving $q_{12} = R^2$. As V tends to infinity, the training sets become fully correlated, and $\hat{\mathbf{w}}_1^\sigma = \hat{\mathbf{w}}_2^\rho$, giving $q_{12} = R^2 + q - R^2$.

2.8.2 Appendix: Monte Carlo CV

Following similar logic to that of appendix(2.8.1), by explicitly constructing partitions that will satisfy that the probability of overlap of two test sets is given by $\alpha_{12} = V^{-2}$, we find that the double replica free energy is given by,

$$\begin{aligned} F^{12} &= G_0{}^{12} + \frac{\alpha_{tot}}{V^2}\Big\{(V-1)G^{12}(\gamma_1;\beta) + (V-1)G^{12}(\beta;\gamma_2) \\ &+ (V-1)^2 G^{12}(\beta;\beta) + G^{12}(\gamma_1;\gamma_2)\Big\} \end{aligned} \tag{2.37}$$

This leads to a rather complicated q_{12} value, which reduces for $MCCV(2)$ to $q_{12} = q/(2-q)$. Again, we have that q_{12} must approach q as V grows.

General test set overlap

Parenthetically, we note that specification of the test set overlap α_{12} is sufficient to determine all other relevant probabilities, such that we can write the double replica free energy as.

$$\begin{aligned} F^{12} &= G_0{}^{12} + \alpha_{tot}\Big\{\left(\frac{1}{V} - \alpha_{12}\right)G^{12}(\gamma_1;\beta) + \left(\frac{1}{V} - \alpha_{12}\right)G^{12}(\beta;\gamma_2) \\ &+ \left(1 - \frac{2}{V} + \alpha_{12}\right)G^{12}(\beta;\beta) + \alpha_{12}G^{12}(\gamma_1;\gamma_2)\Big\} \end{aligned} \tag{2.38}$$

2.9 Appendix: More general Networks

Up to now we have been looking at the relatively simple case of the linear perceptron. In this appendix we study (for a broader class of activation functions than the linear one studied so far) the relationship between the variance due to all sources of randomness, $\mathrm{var}(\epsilon_{test} : \mathcal{E})$ and the average variance due to the test set $\mathrm{var}(\epsilon_{test} : \mathcal{M})$.

Writing,

$$\mathrm{var}(\epsilon_{test} : \mathcal{M}) = \frac{1}{M}\left\langle \left\langle \epsilon_{test}^2 \right\rangle_{\mathcal{M}} - \langle \epsilon_{test} \rangle_{\mathcal{M}}^2 \right\rangle_{\mathcal{W},\mathcal{P}} \tag{2.39}$$

we remark that since the input distribution over which the test error is averaged is isotropic, the absolute directions of the student and teacher weight vectors are irrelevant - it is only their relative separation that is important. This means that $\epsilon_f = \langle \epsilon_{test} \rangle_{\mathcal{M}}$ can only be a function of the overlap between the student and teacher vector. Let us now consider calculating $\mathrm{var}(\epsilon_{test} : \mathcal{M})$ when there have been no training examples yet presented.

2.9.1 Overlap distribution at $\alpha = 0$

If we can find the overlap distribution at $\alpha = 0$, we shall be able to calculate the variance of the generalisation function at $\alpha = 0$. The motivation for doing this is that it will lead us to an approximation for $\mathrm{var}(\epsilon_{test} : \mathcal{M})$. As mentioned above, the term $\langle \epsilon_{test} \rangle_{\mathcal{M}} = \epsilon_f(R)$ is a function of the overlap,

$$R = \frac{1}{N}\mathbf{w}\cdot\mathbf{w}^0 \tag{2.40}$$

Hence, in order to perform the average over the student weight space, $\langle..\rangle_{\mathcal{W}}$, we need to find the distribution of R at $\alpha = 0$ which, fortunately, has a particularly simple form. Due to isotropy, without loss of generality, we may take the teacher weight vector to be,

$$(\mathbf{w}^0)^{\mathbf{T}} = (\sqrt{N}, 0, 0, 0, ...0). \tag{2.41}$$

Remember that both student and teacher weights satisfy the spherical constraint, $\mathbf{w} \cdot \mathbf{w} = N$, $\mathbf{w}^0 \cdot \mathbf{w}^0 = N$. This means that the overlap is simply, $R = w_1/\sqrt{N}$ and the overlap distribution is given by,

$$P(R) \propto \int d\mathbf{w} \delta \left(R - \frac{w_1}{\sqrt{N}} \right) \delta \left(\mathbf{w} \cdot \mathbf{w} - N \right). \tag{2.42}$$

By expressing the delta functions in integral representation form and performing a subsequent saddle point calculation, one finds that the overlap distribution becomes,

$$P(R) \propto \left(1 - R^2\right)^{\frac{N}{2}}, \tag{2.43}$$

which is highly peaked around $R = 0$. In the limit $N \to \infty$, this distribution approaches

$$P(R) = \sqrt{\frac{N}{2\pi}} \exp -NR^2/2, \tag{2.44}$$

a gaussian of variance $1/N$, mean zero. Hence, at $\alpha = 0$, the average over the student weight space of the square of the test error becomes simply,

$$\left\langle \epsilon_f^2(R) \right\rangle_{\mathcal{W}} = \int_{-\infty}^{\infty} Dx \;\; \epsilon_f^2 \left(\frac{x}{\sqrt{N}} \right) \tag{2.45}$$

where Dx is a unit variance, zero mean gaussian measure. By straightforward Taylor expansion, one readily finds that the variance over the overlap distribution of $\epsilon_{test}(R)$ is,

$$\left\langle \epsilon_f^2(R) \right\rangle_R - \langle \epsilon_f(R) \rangle_R^2 = \frac{1}{N} \left(\frac{d\epsilon_f}{dR}(R = 0) \right)^2 + \mathcal{O}\left(\frac{1}{N^2} \right) \tag{2.46}$$

As patterns begin to be presented, the overlap distribution becomes more peaked around its mean value. This means that the variance of $\epsilon_f(R)$ is maximal at $\alpha = 0$, decreasing monotonically with α. For bounded $\epsilon_f(R)$, equation(2.46) shows that one can bound the difference between the average of the square of ϵ_f and the square of the average of ϵ_f by order $1/N$. This being the case, we can write $\mathrm{var}\left(\epsilon_{test} : \mathcal{M}\right)$ as,

$$M \mathrm{var}\left(\epsilon_{test} : \mathcal{M}\right) = \mathrm{var}\left(\epsilon_{test} : \mathcal{E}\right)(M = 1) + \mathcal{O}\left(N^{-1}\right). \tag{2.47}$$

Hence, the only extra work involved in calculating $\mathrm{var}\left(\epsilon_{test} : \mathcal{M}\right)$ is in finding the average of the squared test error. The variance on the right side of equation(2.47) is simply the full variance calculated for one example. In appendix(A), we show how one can apply replica methods to find this variance, with the result in the form,

$$\mathrm{var}\left(\epsilon_{test} : \mathcal{E}\right) = \frac{1}{N} f(\mu), \tag{2.48}$$

where the test set size is given by $M = \mu N$. The full variance $\mathrm{var}\left(\epsilon_{test} : \mathcal{E}\right)(M = 1)$ for one example can then be obtained from equation(2.48) by taking the limit,

$$\mathrm{var}\left(\epsilon_{test} : \mathcal{E}\right)(M = 1) = \lim_{\mu \to 0} \mu f(\mu) \tag{2.49}$$

Hence, by applying replica methods, we can find the variance $\mathrm{var}\left(\epsilon_{test} : \mathcal{E}\right)$, and then by the above procedure, the variance $\mathrm{var}\left(\epsilon_{test} : \mathcal{M}\right)$ can be derived up to order N^{-1}.

Chapter 3

The Linear Perceptron II: Weight Decay

Abstract

By finding the variance of the test error due to randomness present in both the data set and algorithm for a noisy linear perceptron of dimension N, we are able to address such questions as the optimal test set size. We find that the optimal test set size possesses a phase transition between linear and 2/3 power law scaling in the system size N, dependent on the level of noise and the available amount of data. Cross-validation is assessed in terms of its variance, and results concerning model selection are presented.

3.1 Learning from noisy examples

In chapter(2) we built up a general framework for calculating variances, and used a variety of techniques from geometrical methods to statistical mechanics. Introducing noise into the formalism is a step towards a more realistic learning scenario, which we briefly review.

We consider the scenario in which the inputs are represented by N dimensional real vectors, $\mathbf{x} \in \Re^N$, and the output is a real variable, $y \in \Re$. A *data set* $\mathcal{L}$ is a set of L input-output pairs, $\mathcal{L} = \{(\mathbf{x}^\rho, y^\rho), \rho = 1..L\}$. The inputs $\mathbf{x}^\rho$ are assumed drawn independently and identically from a zero mean, unit covariance matrix Gaussian distribution. The (corrupted) outputs are $y^\rho = y^0(\mathbf{x}^\rho) + \eta^\rho$ for some *teacher* function $y^0(\cdot)$, where η^ρ is additive noise. For the purpose of learning from examples, $\mathcal{L}$ is split into two disjoint sets, the *training set*, $\mathcal{P} = \{(\mathbf{x}^\sigma, y^\sigma), \sigma = 1..P\}$ and the *test set*, $\mathcal{M} = \{(\mathbf{x}^\mu, y^\mu), \mu = 1..M\}$, where $L = P + M$[1]. The aim is to find, using the information in $\mathcal{P}$, a *student* function $y(\mathbf{x})$ that matches as closely as possible the output of a randomly chosen input-output pair. That is, we search for student functions that *generalise* well. Clearly, the optimal student is identical to the teacher, and we shall assume that this function is accessible to the student, *i.e.,* that the learning problem is *realisable*[WRB93, BS95b]. In this chapter, we shall again deal with one of the simplest input-output mappings considered in

[1] A σ index will refer to a training input, and μ to a test input.

the learning from examples literature, namely the *noisy linear perceptron*[HP91], for which the output y is related to the input $\mathbf{x}$ by

$$y(\mathbf{x}) = \frac{1}{\sqrt{N}}\mathbf{w}\cdot\mathbf{x},$$

where the *weight vector*, $\mathbf{w} \in \Re^N$. The data set outputs are generated by a 'noisy' teacher, $y^\sigma = \mathbf{w}^0\cdot\mathbf{x}^\sigma/\sqrt{N} + \eta^\sigma$, where $\mathbf{w}^0$ is the teacher vector, and the noise is drawn from a gaussian distribution of mean zero, variance σ^2, such that $\langle \eta^\rho \eta^\tau \rangle = \sigma^2\delta_{\rho,\tau}$. In addition, the *spherical constraint* is assumed on the teacher, namely that it lies on the hypersphere $\mathbf{w}^0\cdot\mathbf{w}^0 = N$.

Student perceptrons that match the outputs of the training set well are found by minimising the *training energy*[2],

$$E_{tr} = \sum_{\sigma=1}^{p}\left(y(\mathbf{x}^\sigma) - y^0(\mathbf{x}^\sigma)\right)^2 = \sum_{\sigma=1}^{p}\left(\tilde{\mathbf{w}}\cdot\mathbf{x}^\sigma - \eta^\sigma\right)^2,$$

where, for convenience, we have defined $\tilde{\mathbf{w}} = (\mathbf{w} - \mathbf{w}^0)/\sqrt{N}$. However, to prevent the student learning the noise in the training set we add a regularising term, $\lambda\mathbf{w}^2$, to the training energy to form an energy function, $E = E_{tr} + \lambda\mathbf{w}^2$ [KH92, DW93]. This extra *weight decay* term penalises large weights and prevents overfitting, improving generalisation performance. As $E_{tr}(\mathbf{w}|\mathcal{P}) \propto P$, as the amount of data increases, the relative importance of the weight decay decreases. The equilibrium ($t \to \infty$) distribution of students that this *Gibbs* algorithm produces is a *Gibbs* distribution,

$$P(\mathbf{w}|\mathcal{P}) = \frac{1}{Z}\exp(-E/T),$$

where Z is a normalisation constant. The *test error*, defined by

$$\epsilon_{test}(\mathbf{w}|\mathcal{M},\mathbf{w}^0) = \frac{1}{M}\sum_{\mu=1}^{M}\left(y^\mu - y(\mathbf{x}^\mu)\right)^2 = \frac{1}{M}\sum_{\mu=1}^{M}\left(\tilde{\mathbf{w}}\cdot\mathbf{x}^\mu - \eta^\mu\right)^2, \tag{3.1}$$

measures how well a student performs on (corrupted) examples from the test set. Ideally, one would like to know the *test* or *generalisation function*, *i.e.,* the expected error $\epsilon_f(\mathbf{w}|\mathbf{w}^0) = \langle\epsilon_{test}(\mathbf{w}|\mathcal{M},\mathbf{w}^0)\rangle_{\mathcal{M}}$ that a student drawn from $P(\mathbf{w}|\mathcal{P})$ will make on a random test example[3]. The generalisation function averaged over $P(\mathbf{w}|\mathcal{P})$ and all possible training sets $\mathcal{P}$ is termed the *generalisation error*, ϵ_g.

The test error forms an M sample estimate of the generalisation function and, according to the central limit theorem, the generalisation function will be distributed in a gaussian manner around the test function[Fel70]. It is the central aim in this chapter to calculate the variance of this distribution. The fluctuations due to random training sets for a particular student generated from the training set $\mathcal{P}$ are quantified by $\langle(\epsilon_{test}(\mathbf{w}|\mathcal{M},\mathbf{w}^0) - \epsilon_f(\mathbf{w}|\mathbf{w}^0))^2\rangle_{\mathcal{M}}$; the

[2]In comparison to the spherical linear perceptron, there is no 1/2 factor in the definition of the training error, or test error. This is to ensure that the generalisation error in the absence of any examples is 1.

[3]Although $\epsilon_f(\mathbf{w}|\mathbf{w}^0)$ is a function of the teacher, due to isotropy of the teacher space, the results of this chapter depend only on the length of the teacher vector, which is fixed. To simplify the calculation, however, we include later a teacher average which is implicit in the average over the data set. For convenience, we denote the generalisation function as an average over all possible test sets $\mathcal{M}$ of size M, although this average is naturally independent of the number of test examples.

average fluctuation of students generated by the training set can be found by averaging this over $P(\mathbf{w},\mathcal{P}) = P(\mathbf{w}|\mathcal{P})P(\mathcal{P})$. We then write the average fluctuation for a P dimensional training set as[4],

$$\mathrm{var}\,(\epsilon_{test}:\mathcal{M}) = \left\langle (\epsilon_{test}(\mathbf{w}|\mathcal{M},\mathbf{w}^0) - \epsilon_f(\mathbf{w}|\mathbf{w}^0))^2 \right\rangle_{\mathcal{M},\mathcal{W},\mathcal{P}} = \frac{1}{M}{\Sigma_1}^2, \tag{3.2}$$

where $\langle..\rangle_{\mathcal{M},\mathcal{W},\mathcal{P}}$ denotes an average over test sets, post-training student, and training set distributions respectively. ${\Sigma_1}^2$ is the variance of the test error calculated for a single test example. If the vast majority of the data examples are assigned to the training set and very few to the test set, the confidence in how well the test error matches the generalisation function will generally be small. Indeed, the test error in this case would typically fluctuate wildly over different test sets, *i.e.,* the variance, $\mathrm{var}\,(\epsilon_{test}:\mathcal{M})$ would be relatively large. We really want to use the data in a dual manner: to minimise the test error, yet remain confident that it will be representative of the generalisation function. That is, given a data set of size L, we aim to know how many examples, M, should constitute the test set, assigning the other $P = L - M$ examples to the training set. In order to address this, we form the generalisation function *upper bound* $\epsilon_{ub}(M|L) = \epsilon_g(M) + \tau\sqrt{\mathrm{var}\,(\epsilon_{test}:\mathcal{M})}$, where τ is a confidence parameter to be chosen. We view $\epsilon_{ub}(M|L)$ as an average probabilistic upper bound on the generalisation function of students trained on P examples and tested on M examples. In order to calculate the optimal scheme to satisfy the above dual requirement, we minimise $\epsilon_{ub}(M|L)$ with respect to M to find the optimal test set size, M^*. This requires the calculation of the variance, $\mathrm{var}\,(\epsilon_{test}:\mathcal{M})$.

In the following section(3.2), we calculate the variance exactly for a restricted region of the space of parameters λ, T, σ^2, and give results that hold for all parameter values, but are valid only for the large N regime in section(3.3). Using these results, we present the optimal test set calculations in section(3.4), and in section(3.6) we extend our analysis of cross-validation to look at model selection, concluding with a summary and discussion in section(3.7).

3.2 Exact variances

In the following two sections, we present briefly results of calculations that are exact in the sense that they hold for all N. These results represent the continuation of work presented in chapter(2), in which the variance of the noise-free spherical linear perceptron was calculated under exhaustive learning[5].

The exact calculations, however, were performed *without* a weight decay term in the training energy E. We defer presentation of results including weight decay until section(3.3) as these results rely on a large N approximation.

3.2.1 Gibbs learning without weight decay ($\lambda = 0$)

Recently, the generalisation error for the finite N Gibbs learning algorithm, without weight decay, was given[Han93] and as the calculation of the variance employs these results, we briefly present the line of argument.

[4] An average over the noise is implicit in the average over the test and training sets.

[5] In the exhaustive learning scenario considered in [BS95c], $P(\mathbf{w}|\mathcal{P})$ is given by the distribution that is uniform over those student weights that reproduce the training set exactly and that satisfy the constraint $\mathbf{w}\cdot\mathbf{w} = N$.

The average of the test error, given by equation(3.1), over the noise distribution, test sets, and student distribution becomes, after straightforward gaussian integrations,

$$\left\langle \epsilon_{test}(\mathbf{w}|\mathcal{M},\mathbf{w}^0)\right\rangle_{\mathcal{M},\mathcal{W}} = \left\langle \tilde{\mathbf{w}}^2\right\rangle_{\mathcal{W}} + \sigma^2. \tag{3.3}$$

By explicitly evaluating the first term of equation(3.3), we find,

$$\left\langle \epsilon_{test}(\mathbf{w}|\mathcal{M},\mathbf{w}^0)\right\rangle_{\mathcal{M},\mathcal{W}} = \left(\frac{T}{2}+\sigma^2\right)\mathrm{tr}'(\mathbf{A}^{-1}) + \sigma^2, \tag{3.4}$$

where the *covariance matrix* $\mathbf{A}$ is defined,

$$\mathbf{A} = \frac{1}{N}\sum_{\sigma=1}^{P}\mathbf{x}^\sigma(\mathbf{x}^\sigma)^{\mathbf{T}}, \tag{3.5}$$

and $\mathrm{tr}'(\cdot) = \mathrm{Tr}(\cdot)/\mathrm{N}$, where $\mathrm{Tr}(\cdot)$ is the trace. The generalisation error is found by taking an average of $\mathrm{tr}'(\mathbf{A}^{-1})$ over the gaussian inputs of the training set, which we denote by $\langle..\rangle_{\mathbf{x}}$, and using the fact that $\mathbf{A}^{-1}$ is distributed according to an inverse Wishart distribution, $W^{-1}(\mathbf{I},P)$ [Eat83, Han93] where $\mathbf{I}$ is the identity matrix. In order that the average of the inverse is finite[6], we require $P>N+1$, and have the result, $\langle \mathrm{tr}'(\mathbf{A}^{-1})\rangle_{\mathbf{x}} = N/(P-N-1)$, which gives,

$$\epsilon_g = \left(\frac{T}{2}+\sigma^2\right)\frac{N}{P-N-1} + \sigma^2. \tag{3.6}$$

For the variance, we rewrite equation(3.2) as,

$$\mathrm{var}\left(\epsilon_{test}:\mathcal{M}\right) = \left\langle \epsilon_{test}(\mathbf{w}|\mathcal{M},\mathbf{w}^0)^2\right\rangle_{\mathcal{M},\mathcal{W},\mathcal{P}} - \left\langle \epsilon_f(\mathbf{w}|\mathbf{w}^0)^2\right\rangle_{\mathcal{W},\mathcal{P}}, \tag{3.7}$$

where, as before, $\epsilon_f(\mathbf{w}|\mathbf{w}^0) = \langle \epsilon_{test}(\mathbf{w}|\mathcal{M},\mathbf{w}^0)\rangle_{\mathcal{M}}$. After carrying out the average over $\mathcal{M}$, equation (3.7) gives

$${\Sigma_1}^2 = 2\left\langle \tilde{\mathbf{w}}^4 + 2\sigma^2\tilde{\mathbf{w}}^2 + \sigma^4\right\rangle_{\mathcal{W},\mathcal{P}}. \tag{3.8}$$

A straightforward gaussian average over $P(\mathbf{w},\mathcal{P})$ gives

$${\Sigma_1}^2 = 2\left\langle \frac{1}{2N}\mathrm{tr}'(\mathbf{A}^{-2})\left(T+2\sigma^2\right)^2 + \left[\mathrm{tr}'(\mathbf{A}^{-1})\left(\frac{T}{2}+\sigma^2\right)+\sigma^2\right]^2\right\rangle_{\mathbf{x}}. \tag{3.9}$$

This can be explicitly evaluated for $P>N+3$ by employing [Eat83],

$$\left\langle (\mathrm{tr}'\mathbf{A}^{-1})^2\right\rangle_{\mathbf{x}} = \frac{(PN+2-N^2-2N)N}{(P-N)(P-N-1)(P-N-3)}, \tag{3.10}$$

and

$$\left\langle \mathrm{tr}'(\mathbf{A}^{-2})\right\rangle_{\mathbf{x}} = \frac{N^2(P-1)}{(P-N)(P-N-1)(P-N-3)}. \tag{3.11}$$

The full expression for ${\Sigma_1}^2$ is somewhat cumbersome, and we present here only the large N limit,

$${\Sigma_1}^2 = \frac{1}{2}\left(\frac{2\sigma^2\alpha+T}{\alpha-1}\right)^2 + \mathcal{O}\left(N^{-1}\right), \tag{3.12}$$

where $\alpha = P/N > 1$. Thus both the generalisation error and variance diverge for $\alpha\rightarrow 1$. As α increases beyond 1, ${\Sigma_1}^2$ decreases to its asymptotic value $2\sigma^4$.

[6]For $P<N$, there are unconstrained directions for the student, which lead to a divergent integral in the average.

3.2.2 Pseudo Inverse

The pseudo inverse algorithm is a limiting case of the general Gibbs algorithm in which the temperature and weight decay both tend to zero such that $T/\lambda \ll 1$ [KH92, DW93]. The benefit from the point of view of the analysis here is that we are able to calculate exactly the variance for both $P<N$ and $P>N$, rather than being restricted to $P>N$ as in section(3.2.1)

The generalisation error for the pseudo inverse algorithm for $P>N+1$ is given by employing the $T=0$ limit of equation(3.6)[Han93]. Similarly, the results for the variance for $P>N+3$ can be readily obtained from equation (3.9) by setting $T=0$. For $P<N$, the pseudo inverse algorithm is given by $\mathbf{w}=\mathbf{P}\mathbf{w}^0$, where $\mathbf{P}$ is the projection onto the subspace spanned by the training inputs[HP91]. Thus $P(\mathbf{w}|\mathcal{P})$ is zero except for the single point, $\mathbf{w}=\mathbf{X}^T(\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{Y}$, where $\mathbf{Y}^T=(y^1,..,y^P)$ and $\mathbf{X}^T=(\mathbf{x}^1,..,\mathbf{x}^P)$. This gives,

$$\epsilon_g = 1-\frac{P}{N}+\sigma^2\left(1+\left\langle \mathrm{tr}'(\mathbf{B}^{-1})\right\rangle_{\mathbf{x}}\right),$$

where $\mathbf{B}=\mathbf{X}\mathbf{X}^T/N$. Comparing $\mathbf{B}$ with the $N\times N$ correlation matrix for P patterns, $\mathbf{A}=\mathbf{X}^T\mathbf{X}/N$, (*cf.* equation (3.5)), we remark that $\mathbf{B}$ is also a correlation matrix, distributed identically to $\mathbf{A}$, but with the roles of P and N reversed. The results from section(3.2.1) concerning the averages of the correlation matrix can then be employed directly by interchanging P and N. For $P<N-1$, we obtain,

$$\epsilon_g = 1-\frac{P}{N}+\sigma^2\frac{N-1}{N-P-1},$$

in agreement with known results for $N\to\infty$, $\alpha=P/N$=const [KH92][7].

A straightforward calculation of the variance for $P<N-3$ leads to

$$\begin{aligned}\frac{N^2}{2}{\Sigma_1}^2 &= \sigma^4\left[2\left\langle \mathrm{tr}'(\mathbf{B}^{-2})\right\rangle_{\mathbf{x}}+\left\langle \mathrm{tr}'(\mathbf{B}^{-1})^2\right\rangle_{\mathbf{x}}+2\left\langle \mathrm{tr}'(\mathbf{B}^{-1})\right\rangle_{\mathbf{x}}+1\right]\\ &+ (N-P)\left[2\sigma^2\left(\left\langle \mathrm{tr}'(\mathbf{B}^{-1})\right\rangle_{\mathbf{x}}+1\right)+\frac{N}{N+2}(2+N-P)\right]. \qquad (3.13)\end{aligned}$$

The results in equations (3.10) and (3.11) can then be employed to find the variance explicitly. In figure(3.1), we plot the generalisation error and $\Sigma_1/\sqrt{2}$ against α for a system of size $N=20$. We remark that the two curves are very similar, a result which we show in the next section is not coincidental. Note that both curves possess the characteristic divergence as the training set size P approaches the system size N.

3.3 Weight Decay

In this section we present results for the general Gibbs learning algorithm for arbitrary temperature, weight decay, and noise.

After carrying out the gaussian integrations over the noise and test set inputs, the resulting generalisation error and variance are necessarily of the same form as equations (3.3) and (3.8) respectively, the only difference being in the distribution $P(\mathbf{w}|\mathcal{P})$ which now includes a weight

[7]Note that in [KH92] the generalisation error is calculated for uncorrupted test sets.

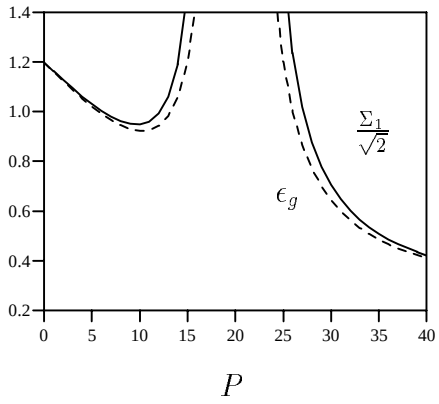

Figure 3.1. Pseudo inverse rule, $T = 0$, $\lambda = 0$. Dashed curve generalisation error. Solid curve, scaled standard deviation. The noise is $\sigma^2 = 0.2$, $N = 20$.

decay term. By continuing the gaussian integrations required for the average over $P(\mathbf{w}|\mathcal{P})$, we obtain,

$$\epsilon_g = \sigma^2 + \left(\frac{T}{2} + \sigma^2\right) \left\langle \mathrm{tr}'(\mathbf{M}^{-1}) \right\rangle_{\mathbf{x}} + \lambda \left(\lambda - \sigma^2\right) \left\langle \mathrm{tr}'(\mathbf{M}^{-2}) \right\rangle_{\mathbf{x}}, \tag{3.14}$$

where

$$\mathbf{M} = \mathbf{A} + \lambda \mathbf{I}.$$

Here $\mathbf{A}$ is the correlation matrix defined earlier in equation (3.5)[8]. The difficulty arises in the calculation of the averages of inverse powers of the matrix $\mathbf{M}$. $\mathrm{tr}'(\mathbf{M}^{-1})$ is termed the *response function*, $\mathcal{G}$, which can be shown to be self averaging in the thermodynamic limit, with $\langle (\mathcal{G} - G)^2 \rangle_{\mathbf{x}} = \mathcal{O}(N^{-2})$, where $G = \langle \mathcal{G} \rangle_{\mathbf{x}}$ [Sol94a]. Moreover, Sollich [Sol94a] obtained the first order corrections to the average of the finite N response function in the form,

$$G = G_0 + G_1/N + \mathcal{O}\left(1/N^2\right),$$

where G_0 is the averaged response function in the thermodynamic limit, and has the value,

$$G_0 = \frac{1}{2\lambda} \left(1 - \alpha - \lambda + \sqrt{(1 - \alpha - \lambda)^2 + 4\lambda}\right).$$

G_1 is related to G_0 by the equation, $G_1 = G_0^2 (1 - \lambda G_0) / (1 + \lambda G_0^2)^2$. Using these results, the first order approximation to the average $\langle \mathrm{tr}'(\mathbf{M}^{-1}) \rangle_{\mathbf{x}}$ can readily be found. Similarly, $\langle \mathrm{tr}'(\mathbf{M}^{-2}) \rangle_{\mathbf{x}}$ can be found by using $\langle \mathrm{tr}'(\mathbf{M}^{-2}) \rangle_{\mathbf{x}} = -(\partial/\partial\lambda) \langle \mathrm{tr}'(\mathbf{M}^{-1}) \rangle_{\mathbf{x}}$.

[8]Note that the pseudo inverse rule is best explained as the limiting case of using the matrix $\mathbf{M}$ for no weight decay.

At this point, however, we note that for the linear perceptron under consideration, we can rewrite the equation for the variance as

$$\frac{1}{2}{\Sigma_1}^2 = \epsilon_g^2 + \mathrm{var}(\tilde{\mathbf{w}}^2)_{\mathcal{W},\mathcal{P}}, \tag{3.15}$$

where $\tilde{\mathbf{w}} = (\mathbf{w} - \mathbf{w}^0)/\sqrt{N}$ and $\mathrm{var}(\tilde{\mathbf{w}}^2)_{\mathcal{W},\mathcal{P}}$ is the variance of $\tilde{\mathbf{w}}\cdot\tilde{\mathbf{w}}$ over the distribution $P(\mathbf{w},\mathcal{P})$. By straightforward gaussian integration, one finds that this variance is a function of the average of terms involving $\mathrm{tr}'\mathbf{M}^{-i}$, $i = 1..4$. Furthermore, the resulting expression is $\mathcal{O}(N^{-1})$, such that any finite size corrections to $\mathrm{tr}'\mathbf{M}^{-i}$ will be $\mathcal{O}(N^{-2})$ corrections to Σ_1^2. Whilst these corrections are straightforward to obtain, the resulting lengthy expressions do not merit inclusion here. Hence, up to order $\mathcal{O}\left(\frac{1}{N}\right)$, the standard deviation of the test error scales linearly with the generalisation error. Indeed, looking back at equations 3.12,3.6, we note that the large N expansion of the variance satisfies

$${\Sigma_1}^2 = 2\epsilon_g^2 + \mathcal{O}\left(N^{-1}\right). \tag{3.16}$$

Evaluating (3.14) and expanding for small λ gives

$$\epsilon_g = \frac{1}{2}\frac{2\sigma^2\alpha + T}{\alpha - 1} - \frac{\alpha\lambda}{2}\frac{T + 4\sigma^2}{(\alpha-1)^3} + \mathcal{O}\left(N^{-1}\right), \tag{3.17}$$

where $\alpha \gg 1 + N^{-1/4}$, $\lambda \ll 1$. (A similar expansion holds for $\alpha < 1$). A weight decay term is therefore advantageous in reducing the generalisation error and the variance.

Up till now, we have considered an isotropic input distribution. More general input distributions can be considered in which the inputs are 'spatially' correlated, $P(\mathbf{x}) \propto \exp(-\mathbf{x}^{\mathbf{T}}\mathbf{\Gamma}^{-1}\mathbf{x}/2)$ (see *e.g.,* [Sol94a], [TL93]). For this correlated input distribution, equation (3.15) remains true on replacing $\tilde{\mathbf{w}}$ with $\mathbf{\Gamma}^{1/2}\tilde{\mathbf{w}}$. The variance of a single test example can then be well approximated as before by twice the square of the generalisation error under the spatially correlated input distribution.

3.4 Optimal test set size

Now that the variance has been calculated, we can proceed to establish the optimal test set size.

A data set $\mathcal{L}$, consisting of L elements, is split into the two disjoint subsets, $\mathcal{P}$ and $\mathcal{M}$. As before, $\mathcal{P}$ is the training set consisting of P examples, and $\mathcal{M}$ is the test set of M examples, such that $\mathcal{L} = \mathcal{P} \cup \mathcal{M}$, and $L = P + M$. Given a data set of L elements, we can then set $P = L - M$ in the equations for the variance and generalisation error, and let M vary between 1 and $L - 1$.

For small M, the standard deviation is relatively large and the generalisation error is small, as the perceptron has been trained on a relatively large number of examples and tested on only a few. This situation reverses as M is increased. The resulting competition between the generalisation error and standard deviation leads to the following definition:

The probabilistic *upper bound* on the generalisation function is defined by $\epsilon_{ub}(M|L) = \epsilon_g + \tau\sqrt{\mathrm{var}\left(\epsilon_{test} : \mathcal{M}\right)}$, where τ is a confidence parameter.

From the central limit theorem, the generalisation function will be distributed in a gaussian manner around the test error [Fel70] and, on average, the generalisation function will

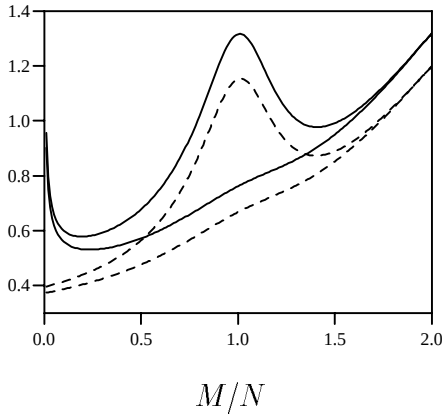

Figure 3.2. Solid curves, upperbounds for $\lambda = 0.01$ (upper curve), and 0.05. Dashed curve generalisation error. The noise is $\sigma^2 = 0.2$, $N = 100$, $L = 200$, $T = 0$. The global minimum in each upperbound represents the optimal test set size.

be distributed similarly around the generalisation error. Setting $\tau = 1$, we will be 84% confident that the generalisation function will lie below $\epsilon_{ub}(M|L)$. Similarly, for $\tau = 2$, we will be 98% confident[9]. For convenience, we set $\tau = 1$ throughout. In figure(3.2), we plot the generalisation error and upper bound function for two values of the weight decay for $N = 100$, $L = 200$, $\sigma^2 = 0.2$, and $T = 0$. We note that the two graphs are qualitatively similar, differing maximally for small M. This can be explained by using the approximation to the variance, and writing the upper bound as[10],

$$\epsilon_{ub}(M|L) = \left(1 + \sqrt{\frac{2}{M}}\right) \epsilon_g + \mathcal{O}\left(\frac{1}{\sqrt{MN}}\right). \tag{3.18}$$

We see from figure(3.2) that the optimal test set size, M^*, for both weight decays is $M^* \approx 24$. Empirically increasing the system size, N, we find that M^* scales like $N^{2/3}$. Further experiment leads to the conclusion that, in general, there exist two scaling laws for M^*. One is the aforementioned 2/3 scaling, and the other is linear. These different scaling phases occur due to the existence of two competing local minima in the upper bound function. 2/3 scaling implies a relatively small test set compared with linear scaling. We would expect that, for small noise levels, or large weight decay, the optimal test set size, M^*, would be minimal, and that as we increase the noise, M^* grows. This conjecture is borne out in figure(3.3), where we plot the prefactors of the linear and 2/3 scaling laws for $L = 0.6N$, $\sigma^2 = 0.8$, $T = 0$. For $\lambda < 0.15$, the scaling is linear (M^* large), and the prefactor reduces quickly as λ tends to 0.15. There is then a transition to 2/3 scaling (M^* small) as λ increases beyond this transition point. Initially, the prefactor for the 2/3 scaling is large, reducing as λ increases.

[9]Here we have quoted the percentage of the normal curve less than a certain number of standard deviations from the mean[Fel70].

[10]Equation (3.18) also holds for (spatially) correlated inputs on replacing ϵ_g with the generalisation error calculated for correlated inputs, from which the modified optimal test set size can be calculated accordingly.

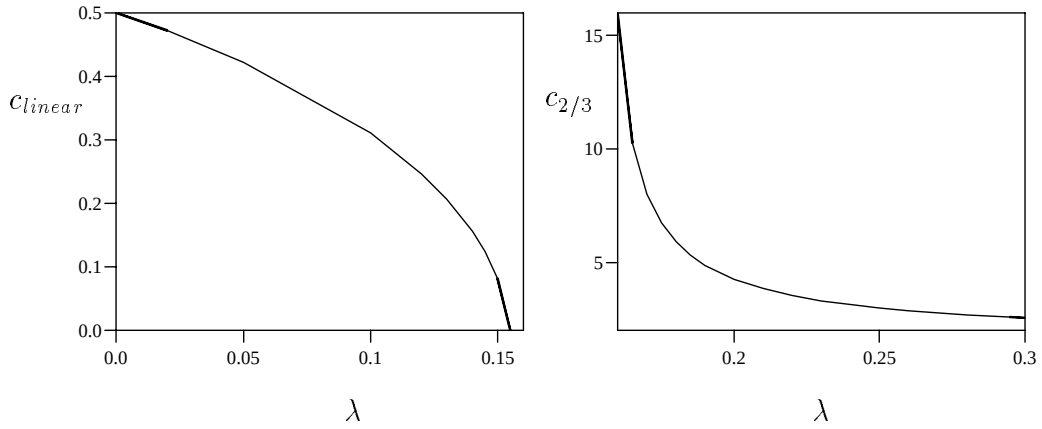

Figure 3.3. Scaling law prefactors for the optimal test set size for a data set of size $L = 0.6N$, $\sigma^2 = 0.8$, $T = 0$.

In general, isolating the phase boundaries involves the solution of a rather complicated expression and, as such, the boundary needs to be found numerically. For the pseudo inverse algorithm, however, analytical expressions for the the large N limit are readily found. In figure(3.4) we plot the phase diagram for the pseudo inverse rule ($N \gg 1$). The values of the prefactors in the regions (a), (b), and (c) are respectively:

$$\frac{1}{2^{1/3}}\left[\frac{\lambda(\alpha_{tot}-1)(\sigma^2+(\alpha_{tot}-1)^2)}{\sigma^2-(\alpha_{tot}-1)^2}\right]^{2/3}, \quad \sigma+\alpha_{tot}-1, \quad \frac{1}{2^{1/3}}\left[\alpha_{tot}(\alpha_{tot}-1)\right]^{2/3},$$

where $\alpha_{tot} = L/N$.

For large N, the variance is essentially zero, and the transition regions are simply given by consideration of the generalisation error. If this is a monotonically decreasing function of α, such phase transitions will not exist as the 'optimal' scheme in this sense is to simply take the smallest test set. For a large enough value of λ, the generalisation error will necessarily be monotonic, and we will have $2/3$ scaling. Thus, small test sets are reasonable for a large weight decay or small noise levels, in that the test error will be a good estimate of the generalisation function. The existence of a phase transition in the scaling law of the optimal test set size is an effect of the non-monotonicity of the generalisation error in the presence of noise. The overfitting phenomenon around $\alpha = 1$ is therefore the origin of the linear phase transition wedge drawn in figure(3.4) - due to overfitting, it is better to use less examples in the training procedure, and more for testing.

General scaling argument

Using the approximation in equation(3.16) that we found for the variance of the linear perceptron, we can differentiate the upper bound equation(3.18) explicitly as a function of α, and find

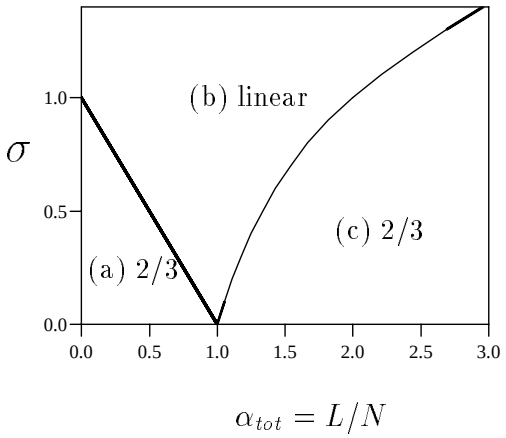

Figure 3.4. Phase diagram for the pseudo inverse rule. In each region, the optimal test set size scales either linearly with N, or like $N^{2/3}$

.

the optimal test set size prefactor for 2/3 scaling is given by,

$$N^{-\frac{2}{3}} M^* = \left(\frac{-\epsilon_g}{\sqrt{2}\epsilon_g'} \right)^{\frac{2}{3}}, \tag{3.19}$$

and the asymptotic ($\alpha \gg 1$) generalisation error is given by[11],

$$\epsilon_g = \frac{1}{2} \frac{2\sigma^2\alpha + T}{\alpha - 1}. \tag{3.20}$$

For the noiseless case, the asymptotic generalisation error is given by,

$$\epsilon_g = \frac{T}{2\alpha}, \tag{3.21}$$

and using equation(3.19), we note that the linear dependence on T cancels, and does not affect the optimal test set size. This gives the optimal test set size prefactor as,

$$N^{-\frac{2}{3}} M^* = 2^{-\frac{1}{3}} \alpha^{\frac{2}{3}}, \tag{3.22}$$

and for the case of noise we find,

$$N^{-\frac{2}{3}} M^* = \left(\frac{\sqrt{2}\sigma^2}{2\sigma^2 + T} \right)^{\frac{2}{3}} \alpha^{\frac{4}{3}}. \tag{3.23}$$

[11]The pseudo inverse rule prefactor is found by setting $T = 0$ in (3.20). Note that this means that the noise enters only as a prefactor in the error. From (3.19) we see therefore that the optimal test set size will be independent of the noise.

With noise therefore, more test examples are needed than in the noiseless case. We note that the prefactor is bounded for large noise, which is a reflection of the fact that the noise affects both the training and test examples. Examining equation(3.23) we see that as we increase the temperature, the prefactor reduces - increasing the temperature means that there is increased uncertainty in the training procedure, and more examples should be devoted to training.

3.5 Cross-validation

An examination of cross-validation for the noisy linear perceptron is complicated by the parameters of noise and weight decay, extra to those of the spherical linear perceptron in chapter(2). Perhaps the more interesting situation that we shall now be able to address is that of model selection in terms of cross-validation in a region where asymptotic theories of statistics are not valid. We leave a brief discussion of these results until section(3.6).

3.5.1 Student Error Covariance

As mentioned in chapter(2), a comparison of the effectiveness of different CV schemes boils down to an analysis of the covariance of two cross-validation student errors. The calculation of these covariances follows the method outlined in appendix(2.8) with, however, a slight modification as explained in appendix(3.8.1). For simplicity, we examine here the case of leave-out-half CV, comparing the results for the covariance with those for the spherical linear perceptron in section(2.5). The most striking qualitative difference between the spherical and weight decay constraints is for the case in which there is no overlap between the test sets, $\alpha_{12} = 0$. Comparing figure(2.6) and figure(3.5)(a), we see that the covariance for the weight decay constraint begins at zero, whereas it begins at negative 1 for the spherical case.

As a partial explanation of the results for the covariances for small α, let us consider the simple case of a dataset consisting of only two examples, and $V = 2$, such that each student is trained and tested on only one example. At zero temperature, the resulting one dimensional constraint from the requirement of zero training error gives (after minimising the Gibbs error with respect to w),

$$\left(\tilde{w}^{(1)} x^{(1)} - \eta^{(1)}\right) x^{(1)} = \lambda w^{(1)}, \tag{3.24}$$

and similarly for the second perceptron, $w^{(2)}$, where $\tilde{w} = (w - w^0)/\sqrt{N}$. Setting $w^0 = \sqrt{N}$, $\lambda = 1$, and the noise variance equal to 1, we can explicitly calculate the covariance of the errors formed by the weight vectors which are solutions to (3.24), assuming the two examples $x^{(1)}$ and $x^{(2)}$ are independent.[12] Expanding these results for large N, the covariance of the two errors is given by,

$$\mathrm{cov}(\epsilon^{(1)}, \epsilon^{(2)}) = \frac{16}{N} + \mathcal{O}\left(N^{-2}\right). \tag{3.25}$$

[12]Parenthetically, we note that the student solutions to (3.24) are $w = \mathcal{O}\left(N^{-\frac{1}{2}}\right)$. For the spherical linear perceptron, we have $w = \mathcal{O}\left(N^{\frac{1}{2}}\right)$.

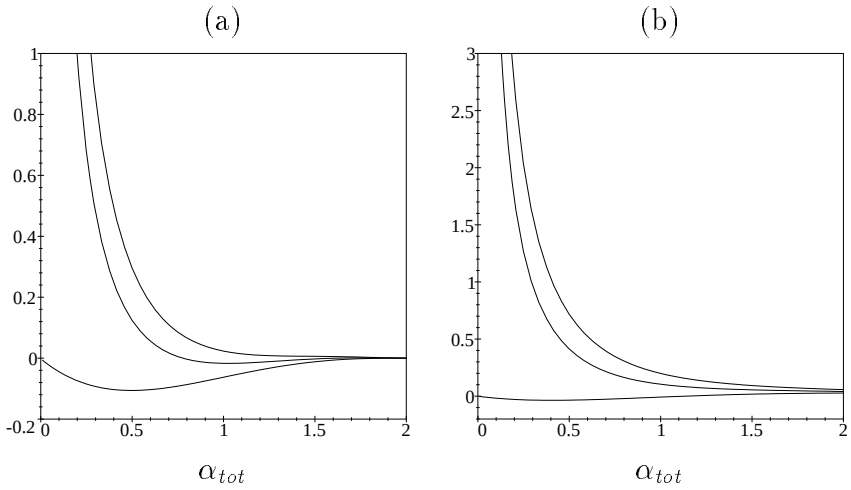

Figure 3.5. Covariance of the error of two cross-validation students trained on half of the data set (V=2) for different values of the test set overlap α_{12} and noise. The lower curve is $\alpha_{12} = 0$, middle $\alpha_{12} = 1/6$ (corresponding to OCV for S=4), upper $\alpha_{12} = 1/4$ (random, MCCV)(a) $\sigma^2 = 0, \lambda = 10^{-3}$ (b) $\sigma^2 = 0.2, \lambda = 0.2$.

Similarly, the variance is given by,

$$\text{var}\left(\epsilon^{(1)}\right) = 8 + \frac{16}{N} + \mathcal{O}\left(N^{-2}\right). \tag{3.26}$$

These results suggest that in the limit of no training data, the covariance is an order smaller than the variance. For the case of some correlation between the examples in the training and testing sets of the two perceptrons, the variance of the two errors can be thought of as the limiting case in which the two halves of the dataset are fully correlated. Since we have seen that for the case of full correlation (variance), the covariance is an order larger than with no correlation, we expect that for a non-zero level of correlation, the covariance will be an order larger. For both the spherical constraint and the weight decay, for correlated test sets ($\alpha_{12} > 0$), there is a divergence in the covariance as the amount of data decreases to zero. Again, this is explicable when we consider that for the fully correlated case, corresponding to the variance, there is always a $1/M$ prefactor for the variance.

For the case of noise, with the weight decay set optimally, we see in figure(3.5)(b), that the covariance for the correlated test sets is larger than for the clean case. For the case of no test set correlation, in the limit of an infinite amount of noise, the student CV errors become random, yielding zero covariance.

In figure(3.6) we again plot covariances: in (a) we show the result of under-regularised students($\lambda < \sigma^2$), finding that optimal partitioning (OCV) can yield a big improvement over random partitioning (MCCV). In figure(3.6)(b) we see that using an over-regularised student can be less risky than an under-regularised student in that it's covariance is closer to that of the optimal weight decay ($\lambda = \sigma^2$). Indeed, we see that for larger values of the weight decay, we can actually reduce the covariance of the CV errors below that for the optimal weight decay setting, although this is not particularly of interest, as an over-regularised student will perform badly in terms of it's generalisation error.

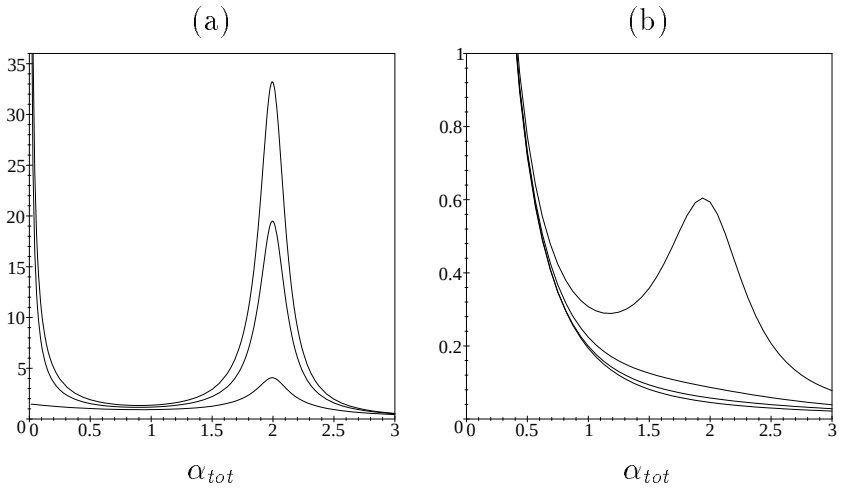

Figure 3.6. Covariance of the error of two cross-validation students trained on half of the data set (V=2). (a)$\sigma^2 = 0.5, \lambda = 10^{-3}$: upper curve MCCV, middle curve OCV(S=4), lower OCV(S=2). (b) $\sigma^2 = 0.2$, $\alpha_{12} = 1/4$ (MCCV): from below $\lambda = 0.4,\ 0.2,\ 0.1,\ 0.01$.

3.6 Model selection using Cross-validation

3.6.1 Introduction

In the previous discussion concerning cross-validation, we have assumed that a particular model has been chosen (or a hyperparameter set to a particular value), and we subsequently wish to evaluate the performance of that model. We now turn our attention to the problem of choosing a model from a collection of possible student models. In particular, we shall be interested in the case where one wishes to set the value of some (hyper)parameter - the weight decay parameter, for example. Typically, there will be some global optimal setting of the hyperparameter such that one student will perform better than another. We then select that model for which the error, estimated by cross-validation, is lower than that for all others. In this section, we examine what would be the best CV scheme to use, given that we wish to discriminate between two different models.

This work is related to work by Shao[Sha93], who discusses linear model selection by cross-validation in the asymptotic data regime. The type of linear models that Shao considers are, in our language, essentially low dimensional linear perceptrons in which certain teacher components are set to zero. The possible students can then be classified as too powerful (student contains more nonzero components than the teacher), optimal (nonzero student components correspond to nonzero teacher components only), or too weak (not enough nonzero student components). There is some similarity in Shao's model selection scenario to the weight decay case in which the weight decays are set either too weakly, optimally, or too strongly. Shao examines the behaviour of leave-one-out CV for selecting a linear model, in terms of the *consistency*. A model selection procedure is defined to be consistent if, in the limit of an infinite amount of data, it is unbiased and the variance of the parameter selection distribution tends to zero, so that the probability of selecting the model with the best predictive ability tends to one. Leave-one-out CV is a popular choice amongst practitioners of CV, arguably because it

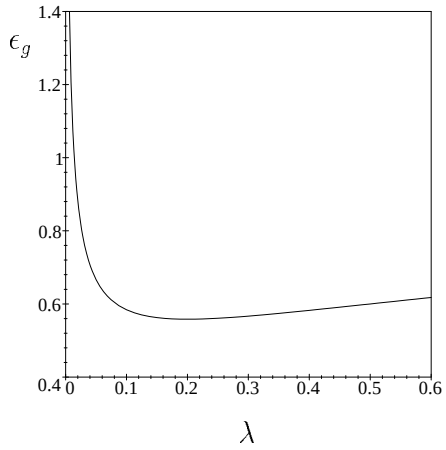

Figure 3.7. The generalisation error vs λ, for two partitions, $V = 2$, and a noise level $\sigma^2 = 0.2$, with a dataset of size $\alpha_{tot} = 2$.

is easily implementable, and has an intuitive feel. Shao's main result is that leave-one-out CV is inconsistent and that in order to have a consistent procedure, one needs to use leave-out-M CV with $M/L \to 1$ as $L \to \infty$. That is, we need to make the cross-validation set size as large as the the whole data set as the amount of data increases. On reflection, however, this is not as surprising as it might at first sound. Let us consider the weight decay case. For consistency, we need the variance of the cross-validation estimate of the optimal weight decay parameter to tend to zero as the amount of data increases without bound. However, as we increase the amount of data, the prior (*i.e.,* weight decay) becomes increasingly less important, and the error surface increasingly insensitive to the choice of weight decay - no matter what weight decay we use, the errors will tend to the same thing. Indeed Marion[MS96] has shown that the variance of the optimal weight decay diverges as the amount of data increases. This highlights, therefore, the difficulty of judging different CV schemes on the basis of the consistency. We prefer to concentrate on how they *perform* - *i.e.,* what error they have. For this reason, we shall judge the various CV schemes on the basis of the variance of their errors, and *not* on the variance of their parameter estimates.

3.6.2 Discriminating between two models

The issue we are here concerned with is the following: given two models (two linear perceptrons with different weight decay parameters), how can we use cross-validation to best decide which is the better model? We denote the cross-validation errors of the models ϵ_1, ϵ_2, where the models have been trained with weight decays λ_1 and λ_2 respectively. One way to discriminate between the two models is to look at the difference between their CV errors, $\epsilon_1 - \epsilon_2$. If this has large modulus, it should be clear which model is the better. However, the cross-validation errors are random variables (due to the random dataset), and we therefore need to consider the joint distribution of CV errors. If the expectation of $[\epsilon_1 - \epsilon_2]^2$ is large, *and* the variance of $[\epsilon_1 - \epsilon_2]^2$ is small we can be sure that one model will be consistently better than the other.

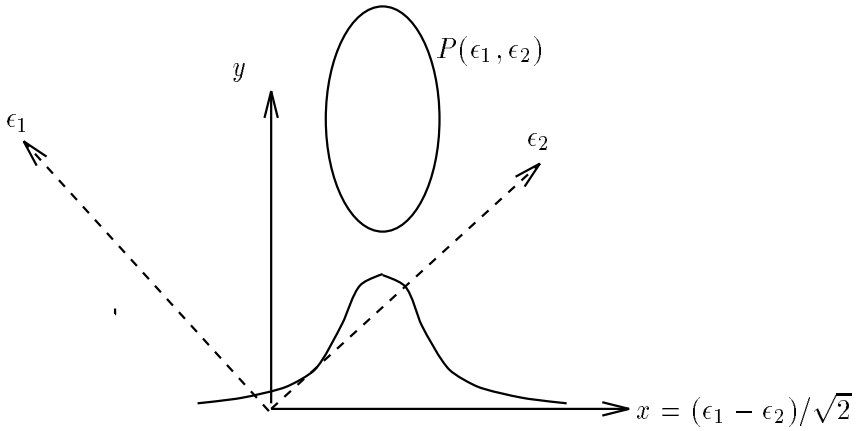

Figure 3.8. Two models are evaluated by their cross-validation errors, ϵ_1 and ϵ_2. The joint probability distribution $P(\epsilon_1, \epsilon_2)$ is represented by the ellipse, and the projection onto the x-axis $(\epsilon_1 - \epsilon_2)/2^{1/2}$ is drawn. In order to discriminate between the two models, we desire a large modulus of $\epsilon_1 - \epsilon_2$, and essentially a small variance. These criteria are embodied in the minimisation of the separation ratio, Ψ.

This is equivalent to prefering a minimal value of the *similarity ratio*,

$$\Psi(\lambda_1, \lambda_2) = N \frac{\mathrm{var}\,(\epsilon_1) + \mathrm{var}\,(\epsilon_2) - 2\mathrm{cov}(\epsilon_1, \epsilon_2)}{\left(\langle \epsilon_1 \rangle - \langle \epsilon_2 \rangle\right)^2} \tag{3.27}$$

As the variance and covariance typically are order $\mathcal{O}\left(N^{-1}\right)$, the similarity ratio will typically be order $\mathcal{O}\left(1\right)$. We shall consider here only the case of MCCV (random partitioning) which means that the covariance in equation(3.27) is equal to simply the covariance of two cross-validation students - *i.e.,* independent of the number of students S.[13] As we know how to work out the (co)variance of cross-validation errors we can, in principle, determine the best CV scheme to use - *i.e.,* how many divisions/students to use for a given computational cost.

In figure(3.9), we plot the similarity ratio $\Psi(\lambda_1, \lambda_2)$ against the number of divisions for a computational cost of $C = 10$. (Remember that the number of students is related to the cost by $S = VC/(V-1)$). Three cases are considered: (a) both models are under-regularised (b) both models are over-regularised, (c),(d) one model is under-regularised, the other over-regularised. In (a),(b), and (d) there is little difference between the sizes of the optimal divisions, all of which are of the order of $V = 2$. For (c), however, there is a greatly increased optimal division size, with a much larger value of the similarity ratio. Looking at figure(3.7) we see that for the two values $\lambda_1 = 0.1$ and $\lambda_2 = 0.4$, the average CV errors are almost the same. In this case, it is extremely difficult to differentiate between the two models, giving rise to a large value of the similarity ratio. The best that can be done in this circumstance is to train the students on a large fraction of the dataset in order to distinguish between their average errors. The situation

[13]This covariance is easy to work out: we have already found the covariance of two CV students for the case in which they are both trained using the same value for the weight decay. It is straightforward to show that for the case in which the students have different weight decays, the resulting expression for the covariance is equivalent to the case in which they are both equal, however with their single replica values calculated with different values of the weight decay.

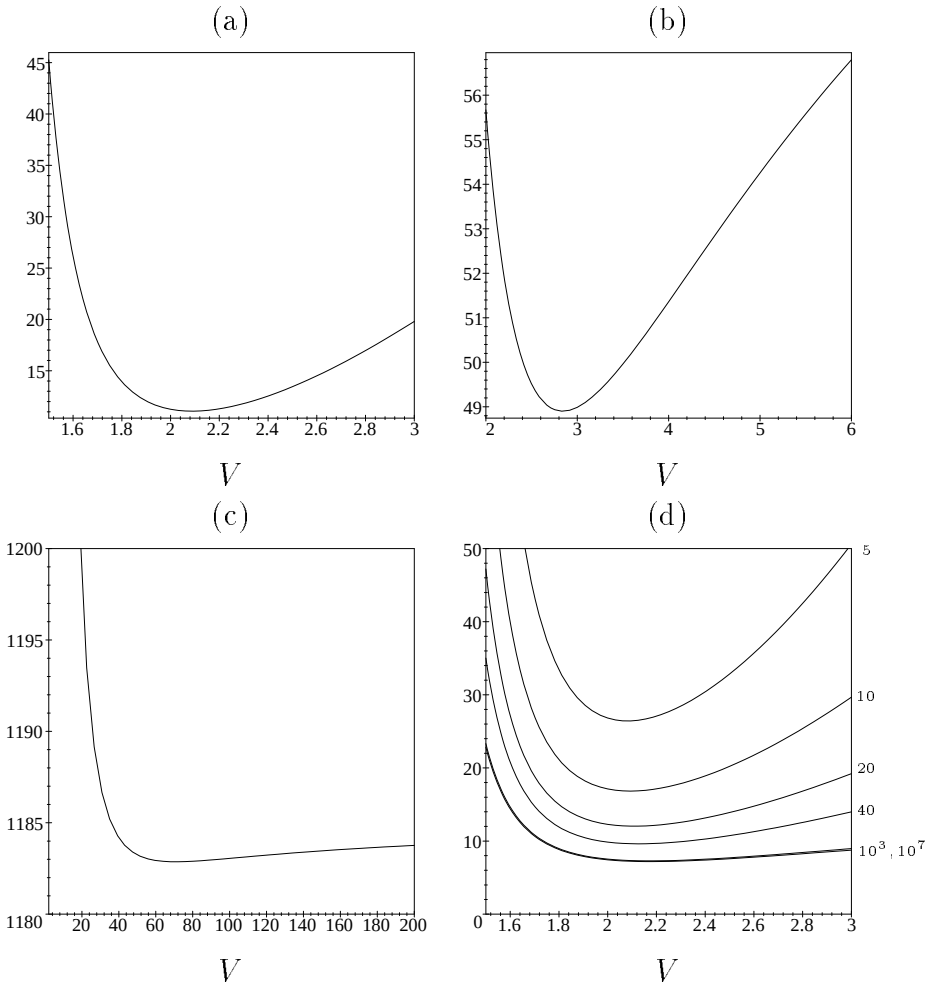

Figure 3.9. The similarity ratio $\Psi(\lambda_1, \lambda_2)$ of the cross-validation errors for the random partitioning scheme plotted against the number of divisions V, with a fixed computational cost $C = 10$ and $\alpha_{tot} = 2$. The optimal setting of the weight decay is $\lambda = \sigma^2 = 0.2$. The minimum in each graph represents the best partitioning to maximise the discrimination between the two models. (a) $\lambda_1 = 0.05$, $\lambda_2 = 0.1$. (b) $\lambda_1 = 0.6$, $\lambda_2 = 0.4$ (c) $\lambda_1 = 0.4$, $\lambda_2 = 0.1$. (d) $\lambda_1 = 0.4$, $\lambda_2 = 0.05$. The different curves correspond to different values of the computational cost, C.

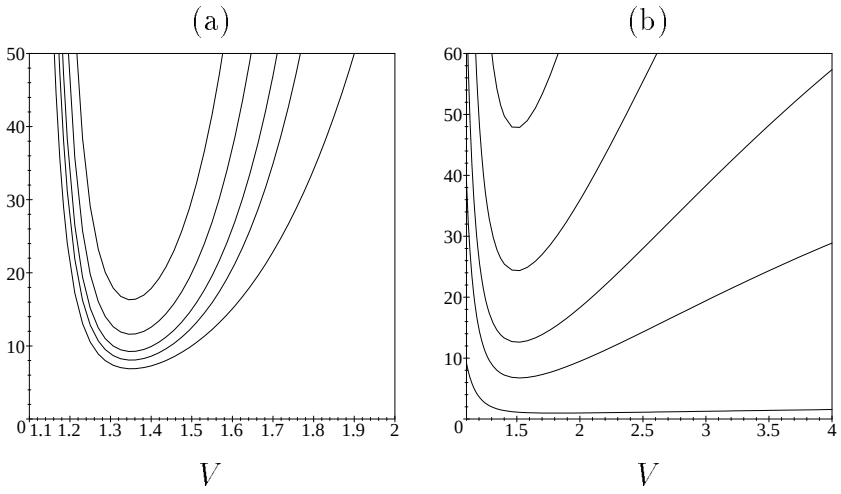

Figure 3.10. The similarity ratio $\Psi(\lambda_1, \lambda_2)$ of the cross-validation errors for the random partitioning scheme plotted against the number of divisions V, for $\alpha_{tot} = 4$. The optimal setting of the weight decay is $\lambda = \sigma^2 = 0.2$. The curves in each figure represent different computational costs: from above, C=5,10,20,40,1000. The minimum in each graph represents the best partitioning to maximise the discrimination between the two models. (a) $\lambda_1 = 0.05$, $\lambda_2 = 0.4$. (b) $\lambda_1 = 0.4$, $\lambda_2 = 0.6$.

in (c) is not generic but demonstrates the effect of trying to differentiate between two models which perform very similarly.
As can be seen from figure(3.9)(d), the dependence of the optimal division on the amount of computing resource is weak. Increasing the computational cost translates into using more students, but not changing the number of divisions. This leads to reduced values of the similarity ratio and increased distinction between the two models.

In a similar fashion, in figure(3.10) we look at the similarity ratio, now for a larger dataset ($\alpha_{tot} = 4$), (a) for an under and an over-regularised model, and (b) for two over-regularised models. The case of two under-regularised students is very close to that in graph (a). Again, there is a weak dependence on the computational cost, with however, a reduced value of the optimal division number from that of using a smaller dataset.

In order to determine the dependence of the optimal number of divisions on the size of the dataset, we plot in figure(3.11) the decay of the optimal number of divisions for two over-regularised models and one over, one under-regularised model. (For the case of both models under-regularised, we found there was little change from the over-under regularised case). Asymptotically, there is power law decay, $V \sim 1 + \mathcal{O}\left(\alpha_{tot}^{-1}\right)$ towards $V = 1$ which corresponds to using all the dataset to test the students, with only a limitingly small fraction used to train the students. As the size of the dataset increases, the importance of the weight decay diminishes and we enter a "data dominated regime." All models will perform similarly, and it becomes increasingly difficult to differentiate between two models, based upon their test error performance/cross-validation error. This is a similar situation to that found by Shao, which we discussed in section(3.6.1). For the linear model that Shao considers, a power law decay of the number of divisions $V \sim 1+\alpha_{tot}^{-1/4}$ is used in order to ensure consistency of the model parameter selection scheme. Therefore, although we used a different criterion to that of consistency (which

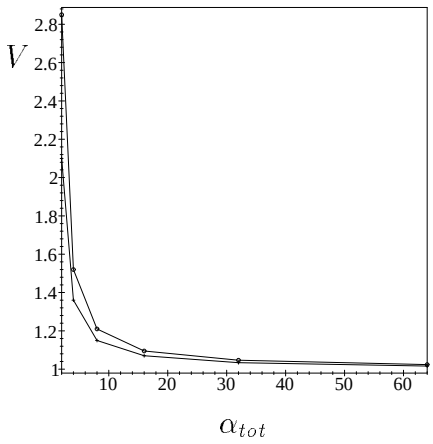

Figure 3.11. The optimal division size V versus the size of the data set α_{tot}, plotted for a noise level $\sigma^2 = 0.2$ The upper curve (circle) is for both models over-regularised, $\lambda_1 = 0.6$, $\lambda_2 = 0.4$ The lower curve (cross) is for one model over-regularised, and the other under-regularised $\lambda_1 = 0.05$, $\lambda_2 = 0.4$. Asymptotically, the optimal partition decays as $1 + \mathcal{O}\left(\alpha_{tot}^{-1}\right)$.

is based on asymptotically selecting the correct parameters of the model), namely the similarity ratio (based on differentiating between the errors of two models), a similar conclusion is reached - the test set size should be increased as the amount of data is increased in order to optimise the cross-validation procedure. Note that this apparant paradox, that as more data is available, it becomes increasingly difficult to choose the 'right' model, is not necessarily apparant in other methods of model selection. The optimal setting of the weight decay, as we have considered above, is to set the weight decay equal to the noise level. Clearly, there *are* measures of noise that will become more sharply defined as the amount of data increases, and hence with those measures, one *can* say which would be the better prior/noise model - however, what we have shown above is that the *performance* of models with different weight decays (priors) becomes increasingly similar as the amount of data increases.

3.7 Summary and Outlook

We have calculated the variance in the test error of the linear perceptron due to randomness present in both the data set and algorithm. Where an exact calculation was not tractable, we showed that the variance can be very well approximated by a simple scaling of the square of the generalisation error. We applied these results to address the question of the best assignment of a data set into a test and training set. We found that there exist two different regions for the scaling of the optimal test set size with the system dimension, one linear, which operates for example for relatively large noise, and one 2/3 scaling. We demonstrated how one can apply the techniques of statistical mechanics in order to analyse cross-validation in a model selection problem, and how to optimise the test set size of cross-validation students in order to help discriminate between two models.

3.8 Appendix

3.8.1 Replica methods

The replica calculation for the weight decay linear perceptron differs only slightly from that for the spherical linear perceptron. In terms of the free energy contributions G_0 and G_r, only the entropic term, G_0, which expresses the form of prior student constraints is affected. Rather than introducing a gaussian average over a delta function representation of the spherical constraint, we now have simply a gaussian weight decay measure. In carrying out the replica method, we will then need to make a slightly more general replica symmetry ansatz, as the length of students trained on the same data is no longer fixed (as was the case for the spherical constraint). Specifically, the replica symmetry ansatz now takes the form (*cf.* (A.13)),

$$q^{\tau,\tau'} = q^0 \delta_{\tau,\tau'} + (1 - \delta_{\tau,\tau'})\, q$$

$$\hat{q}^{\tau,\tau'} = \hat{q}^0 \delta_{\tau,\tau'} + (1 - \delta_{\tau,\tau'})\, \hat{q}$$

The resulting entropic term is given by,

$$G_0 = -R\hat{R} + \frac{1}{2} q\hat{q} - q^0 \hat{q}^0 - \frac{1}{2} \ln\left(\beta\lambda + \hat{q} - 2\hat{q}^0\right) + \frac{1}{2} \frac{\hat{q} - \hat{R}^2}{\beta\lambda + \hat{q} - 2\hat{q}^0} \tag{3.28}$$

and

$$G_r = \frac{1}{2} \ln\left(1 + \beta\left(q^0 - q\right)\right) + \frac{\beta}{2} \frac{q - 2R + 1 + \sigma^2}{1 + \beta\left(q^0 - q\right)}. \tag{3.29}$$

The saddle point equations resulting from extremising the free energy $G_0 - \alpha G_r$ are,

$$q^0 = q + \frac{1}{\beta\lambda + \hat{R}}$$

$$q = \left(\hat{R} + \hat{q}\right)\left(q^0 - q\right)^2$$

$$\hat{q}^0 = \frac{1}{2}\left(\hat{q} - \hat{R}\right)$$

$$\hat{q} = \alpha\beta \frac{1 + \sigma^2 + q - 2R^2}{1 + \beta\left(q^0 - q\right)}$$

$$R = \hat{R}\left(q^0 - q\right)$$

$$\hat{R} = \frac{\alpha\beta}{1 + \beta\left(q^0 - q\right)}$$

The solution of these equations is given by [DW93],

$$q^0 = q + \mathcal{Q}$$

$$q = \alpha \frac{1 + \sigma^2 + \alpha}{\phi^2 - \alpha}$$

$$R = \frac{\alpha}{\phi}$$

where

$$\mathcal{Q} = \frac{1}{2\beta\lambda}\left(1-\alpha-\lambda\right) + \sqrt{\left(1-\alpha-\lambda\right)^2 + 4\lambda} \tag{3.30}$$

$$\phi = \lambda + \beta\lambda\mathcal{Q} + \alpha \tag{3.31}$$

3.8.2 Double replica

Because there is also noise present in the weight decay calculation, the double replicated Hamiltonian changes form slightly to accommodate this (*cf.* (A.35)):

$$G_r^{12}(\beta_1,\beta_2) = -\frac{1}{2}\frac{\beta_1}{1+\beta_1\left(q_1^0-q_1\right)}\frac{\beta_2}{1+\beta_2\left(q_2^0-q_2\right)}\times$$

$$\left\{\left(q_{12}-R_1R_2+R_1-1\right)\left(q_{12}-R_1R_2+R_2-1\right)+2\sigma^2\left(q_{12}-R_1-R_2+1\right)+\sigma^4\right\} \tag{3.32}$$

where subscripts 1,2 denote the replica system. The entropic term is given by

$$G_0^{12} = -\frac{1}{2}\frac{q_{12}-R_1R_2}{\left(q_1^0-q_1\right)\left(q_2^0-q_2\right)}, \tag{3.33}$$

where the single replica solutions are given in appendix(3.8.1).

Chapter 4

The Binary Perceptron

Abstract

The binary perceptron is fundamentally different from the linear perceptron in that the output is discontinuous. Cross-validation schemes performing random, or optimised partitioning are found to perform to a greater degree of similarly than for the linear perceptron. By calculating the variance of the generalisation function over the version space, we make a tentative connection with the PAC worst case analysis by approximating the distribution of errors. Even for small system sizes, the probability of an error close to the worst case bound is extremely small.

4.1 Introduction

Having studied in some detail the linear perceptron in the previous two chapters, we turn our attention to a simple non-linear system, the binary perceptron. The motivation in so doing is to check some of the conclusions inferred for the linear perceptron against a non-linear system and we shall again be working within the framework of batch learning with the teacher of the same form as the student. As for the linear perceptron, a considerable body of work already exists for the binary perceptron, for which many calculations have been carried out with recourse to the thermodynamic limit [GT90, WRB93]. We aim, therefore, both to calculate variances for the binary perceptron, and also to check the performance of cross-validation for which much of the work carried out on cross-validation has been on linear and/or continuous models[Sha93, Bur89].

4.2 The Binary Perceptron

The binary perceptron has the same structure as the simple perceptron introduced in chapter(1), now with a binary valued activation function, such that the output for real valued inputs $\mathbf{x}$ is given by:

$$\mathbf{y} = \mathrm{sgn}\left(\frac{1}{\sqrt{N}}\mathbf{w}\cdot\mathbf{x}\right), \tag{4.1}$$

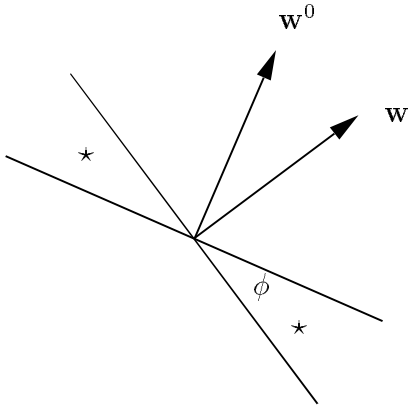

Figure 4.1. Geometrical interpretation of the generalisation error between a binary perceptron student and teacher with weight vectors $\mathbf{w}$ and $\mathbf{w}^0$, respectively. Shown is the projection of the input space onto the plane spanned by $\mathbf{w}$ and $\mathbf{w}^0$; the input regions for which the outputs of student and teacher disagree are marked by asterisks. The generalisation error ϵ_g is equal to the probability with which a random input vector will 'land' in one of these regions. For isotropically distributed inputs, this probability is simply $2\phi/\pi$ where ϕ, the angle between $\mathbf{w}^0$ and $\mathbf{w}$, is given by $\phi = \arccos(\mathbf{w}^0 \cdot \mathbf{w}/N)$ due to the normalisation $(\mathbf{w}^0)^2 = \mathbf{w}^2 = N$.

where $\mathrm{sgn}(h) = +1$ for $h \geq 0$, and -1 otherwise. The spherical constraint, $(\mathbf{w}^0)^2 = \mathbf{w}^2 = N$ is again imposed as a convenient normalisation. Geometrically, the output of the binary perceptron depends on which side of the hyperplane, specified by the weight vector $\mathbf{w}$, the input example $\mathbf{x}$ lies; the output is positive for a positive projection of the example onto the hyperplane, and negative for a negative projection. As we do not consider a threshold (which simply adds a constant to the activation h), the hyperplane passes through the origin.

As before, we shall be interested in the generalisation performance of a binary student perceptron, specified by weight vector, $\mathbf{w}$, learning a binary teacher perceptron specified by $\mathbf{w}^0$. Students are generated by minimising the *training error* on a set $\mathcal{P}$ of P examples, given by

$$E_{tr} = 2 \sum_{k=1}^{p} \theta \left(-t_k s_k \right), \tag{4.2}$$

where $\theta(x) = 0$ for $x \leq 0$ and $+1$ otherwise, and t_k , s_k are the outputs of the teacher and student on input example $\mathbf{x}^k$ respectively. The inputs are selected randomly with each component drawn from a zero mean, unit variance normal distribution. Note that the training error equation(4.2) counts the number of errors that the student makes on the training set. We again take an extensive number of training examples $P = \alpha N$ such that the training error itself will be extensive. The Gibbs algorithm selects (spherical) student weight vectors from the distribution $\propto \exp -\beta E_{tr}(\mathbf{w})$[1].

[1]One way to achieve this for example is to simply randomly sample candidate spherical student weight vectors (with uniform probability over student weight space), which are then selected with the Gibbs probability.

The generalisation error

In order to test the performance of trained binary perceptron students, we again form the *test error*,[2]

$$\epsilon_{test} = \frac{2}{M} \sum_{m=1}^{M} \theta \left(-t_m s_m \right), \tag{4.3}$$

where the test set is composed of the M input-output pairs, $\mathcal{M} = \left\{ (\mathbf{x}^1, t(\mathbf{x}^1)), .., \left(\mathbf{x}^M, t(\mathbf{x}^M) \right) \right\}$ and the generalisation error is defined as the test error averaged over the test set distribution. In figure(4.1), we show geometrically how the generalisation function is related to the average overlap between the student and teacher vectors,

$$\epsilon_f = \frac{2}{\pi} \arccos(R), \tag{4.4}$$

where we define the overlap parameter, $R = \frac{1}{N} \mathbf{w}^0 \cdot \mathbf{w}$. In order to calculate the generalisation error, we need to average equation(4.4) over the weight space and training/test sets. The non-linearity of the activation function increases the complexity of the calculation of the generalisation error, compared to that for the linear perceptron, although these difficulties can be overcome using the replica formalism of statistical mechanics, following the procedure outlined in appendix(A). The generalisation error calculation was initially carried out by Gyorgyi and Tishby[GT90], and we briefly restate some of their results[3]. Details of the replica method as applied to the binary perceptron are found in appendix(4.7.1).

Zero-mean additive gaussian noise on the weight vectors (of variance $\sigma^2_{weights}$) and input components (of variance σ^2_{inputs}) noise have similar effects, and manifest themselves in the noise parameter,

$$\gamma = \left(\left(1 + \sigma^2_{inputs} \right) \left(1 + \sigma^2_{weights} \right) \right)^{-1/2}, \tag{4.5}$$

so that a noise free system is modelled by the selection $\gamma = 1$, and the generalisation function in the presence of noise is given by,

$$\epsilon_f = \frac{2}{\pi} \arccos(\gamma R). \tag{4.6}$$

The resulting generalisation error is plotted in figure(4.2) for different values of the noise parameter, γ. For zero noise and large α, the generalisation error decays algebraically,

$$\epsilon_g = \frac{1.25}{\alpha} + \mathcal{O}\left(\alpha^{-2} \right). \tag{4.7}$$

In the presence of noise, the residual generalisation error as $\alpha \rightarrow \infty$ is given by,

$$\epsilon_r = \frac{2}{\pi} \arccos\left(\gamma \right), \tag{4.8}$$

[2]The test error is scaled so that $e.g.,(\alpha = 0) = 1$, as for the linear perceptron.

[3]Whereas in [GT90] the generalisation error is defined such that the zero α value is a half, we define the zero α value to be 1, as we did for the linear perceptron.

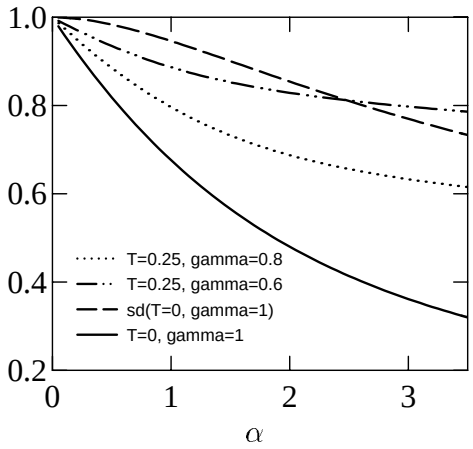

Figure 4.2. The generalisation error plotted for different settings of the noise parameter γ against the training set size, $\alpha = P/N$. The standard deviation of the test error for a student trained at zero temperature on noiseless examples is also plotted.

and the asymptotic decay of the generalisation error is given by,

$$\epsilon_g - \epsilon_r \propto \alpha^{-\frac{1}{2}}. \tag{4.9}$$

In contrast to the noise free, zero temperature linear perceptron, there is no critical value of α for which the generalisation error becomes zero. For the binary perceptron, the student and teacher outputs identify only whether a given input lies between the planes defined by the student and teacher. As more examples are presented, the student rotates toward the teacher, but there will always be a chance that an example arises that lies between the student and teacher planes, thus giving an error for finite α.

4.3 The Variance

Once the generalisation error has been calculated, the desired variance is straightforward to obtain. Referring back to the discussion in section(2.9) we remark that, up to order $\mathcal{O}(N^{-1})$, the variance $\mathrm{var}(\epsilon_{test} : \mathcal{M})$ obeys,

$$M\,\mathrm{var}(\epsilon_{test} : \mathcal{M}) = \mathrm{var}(\epsilon_{test} : \mathcal{E})(M = 1) + \mathcal{O}\left(N^{-1}\right) \tag{4.10}$$

As the output of the binary perceptron is the sign function, we arrive immediately at the result,

$$\mathrm{var}(\epsilon_{test} : \mathcal{E})(M = 1) = 2\epsilon_g - \epsilon_g^2 + \mathcal{O}\left(\frac{1}{N}\right), \tag{4.11}$$

and hence that

$$\mathrm{var}(\epsilon_{test} : \mathcal{M}) = \frac{1}{M}\left\{2\epsilon_g - \epsilon_g^2\right\} + \mathcal{O}\left(\frac{1}{MN}\right) \tag{4.12}$$

This shows that the asymptotic decay of the variance for the binary perceptron is much slower than that for the linear perceptron: for the binary perceptron, the variance decays only with the generalisation error, whereas it decays as the square of the generalisation error for the linear perceptron.

4.3.1 Optimal test set size

Analogous to the optimal test set size analysis that we carried out for the linear perceptron (see sections(2.3,3.4)), we again look for the best partitioning of a data set into a test and training set in order to find a low test error, yet remain confident that the test error is close to the generalisation function. Rephrasing this, we wish to (confidence) bound the generalisation function from above: this can be achieved by adding on to the average error say one standard deviation of the generalisation function distribution, forming the upper bound $\epsilon_{ub}(M|L)$; with probability 0.84, the generalisation function will be lower than $\epsilon_{ub}(M|L)$, and we seek to minimise this upperbound with respect to the freedom we have in choosing the fraction of examples assigned to the test/training procedure. Using the approximation to $\mathrm{var}(\epsilon_{test} : \mathcal{M})$, given in (4.12), we have for a one standard deviation confidence,

$$\epsilon_{ub}(M|L) = \epsilon_g + \left(\frac{2\epsilon_g - \epsilon_g^2}{M}\right)^{\frac{1}{2}} + \mathcal{O}\left(\frac{1}{N}\right) \tag{4.13}$$

Optimising this upper bound with respect to α, we make again the scaling ansatz, $M = cN^{\frac{2}{3}}$. This ansatz is motivated by numerical optimisation of the upper bound or, alternatively, by consistency arguments. For large N, we find,

$$c = \left(\frac{\sqrt{2\epsilon_g - \epsilon_g^2}}{2\epsilon_g'}\right)^{\frac{2}{3}}, \tag{4.14}$$

where ϵ_g is the value of the generalisation error at the optimal value of α. Unlike the linear perceptron, the binary perceptron generalisation error is a monotonically decreasing function of α, regardless of the (fixed) noise level. This means that there will not be a phase transition to a different (linear) scaling law as the noise level rises, as was the case for the linear perceptron. For large N, $\alpha^* = \alpha_{tot} + \mathcal{O}\left(N^{-\frac{1}{3}}\right)$, and we therefore approximate $\epsilon_g(\alpha^*)$ by $\epsilon_g(\alpha_{tot})$. Similarly, ϵ_g' is the value of the derivative of the generalisation error at the optimal value of α, which we approximate by $\epsilon_g'(\alpha_{tot})$. For the case of no noise, using equation(4.7) gives the asymptotic value of the prefactor for large α,

$$c = 2.5^{-\frac{1}{3}}\alpha = 0.733\alpha, \tag{4.15}$$

which corresponds well with the gradient of the curves in figure(4.3) for large α. For the case of noise present we use equation(4.8), and again find, asymptotically, a linear scaling of the prefactor c with α, which may at first seem surprising. On deeper reflection, however, we realise that noise enters the process only through the variable γ which is bounded between 0 and 1 (an artifact of the binary nature of the problem) so that the residual generalisation error is bounded, even for infinite levels of weight and input example noise. We see from figure(4.3) that the asymptotic values of the prefactors are indeed linear, and have bounded gradient. This is to be contrasted with the noisy linear perceptron, described in section(2.3), where noise gives rise to a different prefactor scaling in α, namely a 4/3 power law.

As the temperature is increased, the optimal test set size tends to decrease, which is a similar effect to that found for the linear perceptron. The explanation is that as T is increased, there is less confidence in the training procedure, and more examples are required to reduce the training error.

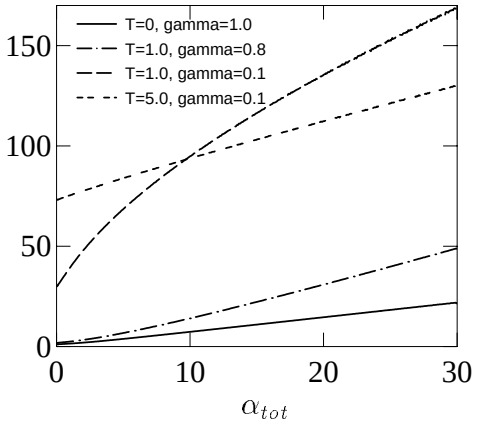

Figure 4.3. Prefactor c for the optimal test set size $M^* = cN^{\frac{2}{3}}$ for various values of the noise and temperature. All curves tend to a linear prefactor in the limit of large α_{tot}. Note that there is little change in the curves for different temperature. The effect of noise is to increase the required size of the test set in order to compensate for the higher implicit variance of noisy examples.

4.4 Cross-validation

Similar to the analysis carried out on cross-validation in the previous two chapters, we briefly examine here the relative performance of different cross-validation schemes in the context of a highly non-linear rule. We refer the reader to chapter(2) for the general aims of our analysis here.

We again perform a replica analysis as we did previously for the linear perceptron, and an overview of the requisite results for the method is given in appendix(4.7.1). Although the procedure is essentially the same as for the linear perceptron, the resulting saddle point problem necessitates the application of more sophisticated numerical techniques. Also, for the case of noise, the replica method breaks down if the learning temperature is too low, and we try to avoid such regions. As pointed out in chapter(2), an analysis of the relative performance of different cross-validation schemes boils down to a comparison of the covariance of two individual cross-validation students.

In figure(4.4) we plot covariances for the case of 2 divisions, $V = 2$, under the different CV schemes described in chapter(2). We see from figure(4.4) that the covariance is most negative for the scheme where there is the smallest test set overlap. This figure is to be compared to figure(2.6) in section(2.5) where we plotted for the spherical linear perceptron the behaviour of cross-validation under the same conditions. Noteworthy is the similarity of the curves for the binary scheme relative to the curves for the linear model, under the different types of CV schemes. We therefore expect very little difference between the performance (*i.e.,* CV error variance) of CV schemes for the binary perceptron, relative to the differences in the linear perceptron. Learning with a small temperature has little effect on the covariance. (The graphs of zero noise and $T = 0.25$ are indistinguishable from figure(4.4)(a)). The effect of noise is to increase slightly the the tail of the covariance curve as it tends towards the zero asymptotic value. Furthermore, the optimal CV scheme is seen to be very close to the random (Monte

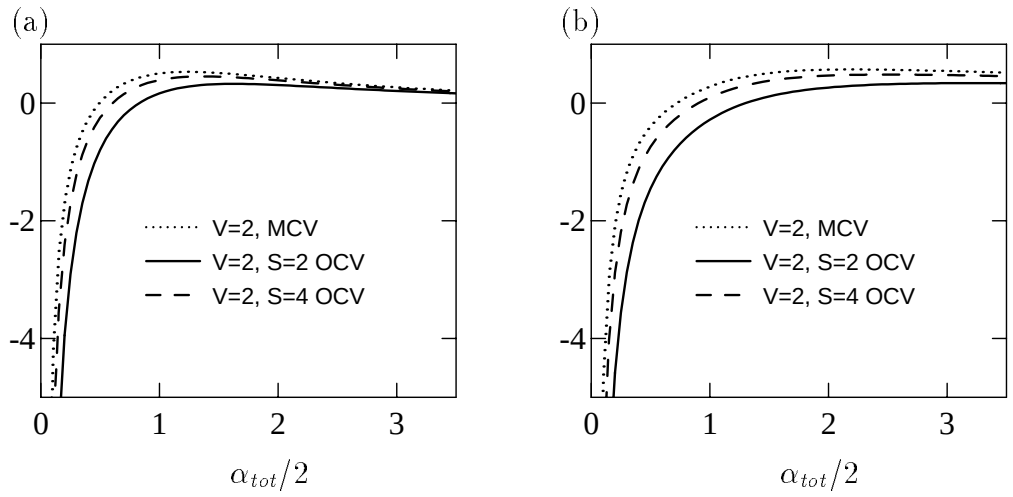

Figure 4.4. The (scaled) covariance of two test errors for different CV schemes, all for two partitions, $V = 2$. The lower curve is for the non-overlapping scheme for S=2, equivalent to the OCV scheme for $S \leq V$. The middle curve is the OCV scheme for $S = 4$. The upper curve is the MCCV scheme, which is independent of the number of students. (a) The noise is set to zero, and so is the temperature. (b) $\gamma = 0.6$, $T = 0.25$.

Carlo) CV scheme *cf.* figure(4.4)(b), and we conclude that the effect of noise does not bring about significant advantages of using one CV scheme in favour of another. In conclusion therefore, the CV procedure for the binary perceptron is more robust to noise and temperature changes than is the linear perceptron.

4.5 Connection with PAC learning

The standard PAC analysis deals with binary output systems[Ant95], and we are now in a position to make some connections between the average case and the PAC analysis. The work in this section is linked to that by Engel and Fink [EvdB93] who use techniques of statistical mechanics to calculate a worst case analysis for the performance of the binary perceptron.

Let us review briefly the picture that we have of zero temperature learning: A set of training examples $\mathcal{P}$ is used by the (zero temperature) Gibbs learning algorithm to generate a set of candidate students - the version space - and a student is picked at random from the version space. Although all students in the version space have zero training error, they will in general have different generalisation functions, and a measure of the scale of this difference is given by the variance of the generalisation function over the version space. A worst case analysis is concerned with bounding the performance of the worst student from the version space. Engel and Fink are concerned with checking how tight the bounds from the distribution free PAC theory are in the case of a specific input distribution. However, it may still be the case that these bounds are not tight in the sense that, for all but pathological input distributions, the overwhelming proportion of the error mass lies far from the worst case boundary. Equally, while Engel and Fink can calculate the worst case error for a specific input distribution, we can

calculate how likely it is that we come close to saturating that bound for the same input distribution. Given our understanding of the gaussian nature (for large N) of the error distribution around its mean value, with a variance of the order of $\mathcal{O}(N^{-1})$, it is intuitively clear that for iid distributed inputs, the probability that an error occurs close to the worst case error will be exponentially small. However, what is not immediately clear is whether the prefactor of this variance is so large that for "moderate" system sizes, there is still an appreciable chance that an error occurs close to the worst case.

We restate briefly the theory of PAC learning as explained by Engel and Fink. Initially, Engel and Fink are interested in bounding the difference between the training error (which will be set to zero later), and the generalisation function. For a *fixed* student, the probability that the training error E_{tr} and the generalisation function ϵ_f differ more than a quantity ϵ is given by the Hoeffding inequality,

$$\text{Prob}\left\{|E_{tr}-\epsilon_f|>\epsilon\right\} \leq 2\exp\left(-2\epsilon^2 P\right), \tag{4.16}$$

which, for a constant bound gives $\epsilon \sim 1/\sqrt{P}$, corresponding to the central limit theorem picture that we have been using throughout. In a worst case analysis, the maximal difference between the training error and generalisation function over a class of possible students is given by the Vapnik and Chervonenkis bound,

$$\text{Prob}\left\{max_{\mathbf{w}}|E_{tr}-\epsilon_f|>\epsilon\right\} \leq c\Delta(2P)\exp\left(-2\epsilon^2 P\right), \tag{4.17}$$

where for $P > N$, the growth function $\Delta(m)$ is given by $\Delta(m) = 2\sum_{i=0}^{N-1}\binom{m-1}{i}$. Using Stirling's formula, and approximating the summation by a saddle point integration gives, for $\alpha \geq 1$, $N \gg 1$,

$$\Delta(2\alpha N) = 2\sum_{i=0}^{N-1}\binom{m-1}{i} \sim \exp\left(N\left[2\alpha\log(2\alpha)-(2\alpha-1)\log 2\alpha - 1\right]\right). \tag{4.18}$$

Using this in the VC bound (4.17) we obtain,

$$\text{Prob}\left\{max_{\mathbf{w}}|E_{tr}-\epsilon_f|>\epsilon\right\} \leq c\exp\left(N\left[2\alpha\log(2\alpha)-(2\alpha-1)\log 2\alpha - 1 - \alpha\epsilon^2\right]\right) \tag{4.19}$$

Hence, in the limit of an infinitely large perceptron, $N\to\infty$, the maximal difference ϵ_f^{vc} between the training error and generalisation function is bounded from above by,

$$\epsilon_f^{vc}(\alpha) = \sqrt{2\log 2\alpha - (2-1/\alpha)\log(2\alpha-1)} \tag{4.20}$$

with probability 1. For large α, this means that the maximal difference between the training error and generalisation function scales like, $\epsilon_f^{vc} \sim \sqrt{\log\alpha/\alpha}$.

4.5.1 Restriction to the version space

For the case in which the set of possible student vectors is restricted to the version space (zero training error), the bounds tighten, such that the maximal generalisation function of students from the version space is given for large α by,

$$\epsilon_f^{vc} \sim \frac{2\log\alpha}{\alpha\log 2}. \tag{4.21}$$

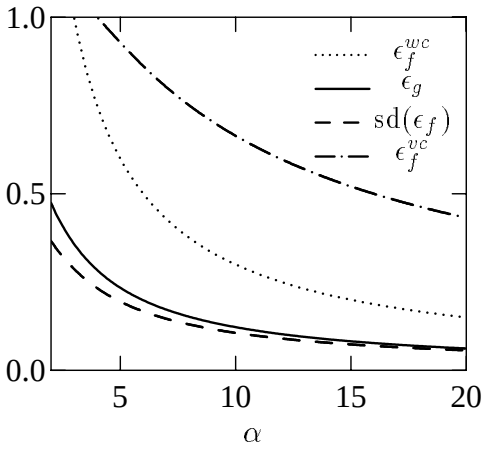

Figure 4.5. Bounds on the generalisation performance. The generalisation error is plotted as the solid curve. The dotted curve is the asymptotic curve for the performance of the worst student from the version space, $\epsilon_f^{wc} \sim 3/\alpha$. The dot-dash curve is the asymptote to the normal VC theory limiting value of the error, $\epsilon_f^{vc} \sim 2\log\alpha/(\alpha\log 2)$. The dashed curve is the (scaled) variance over the version space of the generalisation function. Note that we plot only from $\alpha = 2$ as below this, the asymptotes are wildly incorrect.

Engel and Fink provide a replica symmetric calculation for the performance of the worst student from the version space (given by the student in the version space with the smallest overlap with the teacher), which gives the large α result (see figure(4.5)),

$$\epsilon_f^{wc} \sim \frac{3}{\alpha}. \tag{4.22}$$

Although replica symmetry is found not to hold for $\alpha > 2$, a replica symmetry breaking calculation gives the same asymptotic scaling with α. Hence the VC bound over-estimates the generalisation function of the worst student by a factor $2\log\alpha/(3\log 2)$. According to Engel and Fink, however, the VC bounds can be tightened by using information theory to give (for any input distribution), a bound $2/\alpha$ for large α, which is very close to the (distribution specific) worst case bound given by Engel and Fink.

Given that it is possible to calculate the generalisation performance of the worst student from the version space, one might ask, how likely is it that a student sampled from the version space will have an error close to this worst case? Since we have been calculating variances (including those of errors over the version space), and given our usual assumption of a normal distribution, we see that we are in a position to say how likely, for a given value of α, a generalisation function so far from the average case is likely to occur. For the iid distributed input examples, the variance of the distribution of errors is order $\mathcal{O}(N^{-1})$ and hence, for any finite difference between the worst case generalisation performance and the typical performance, the probability of a randomly selected student with error close to the worst case will become exponentially unlikely as the size of the perceptron increases.

The variance that we require in order to calculate the probability of picking a student with a generalisation function worse than the average worst case student is the variance of the

generalisation function over the version space, $\mathrm{var}(\epsilon_f : \mathcal{W})$. Explicitly, this is not a quantity that we have previously calculated although, in principle, it is straightforward to calculate this variance[4]. However, as the number of test examples grows, the test error approaches the generalisation function, and we have $\epsilon_f = \epsilon_{test} + \mathcal{O}\left(M^{-\frac{1}{2}}\right)$. We therefore simply approximate the variance of the generalisation function by that of a large sample test error. There are, however, some subtleties hidden here. Engel and Fink calculate the worst case generalisation function by assuming a binary valued input distribution, namely, $P(\mathbf{x}) = \prod_{i=1..N}\left[\frac{1}{2}\delta(x_i - 1) + \frac{1}{2}\delta(x_i + 1)\right]$, for which the first two moments are the same as for a zero mean unit variance distribution. The calculation of the free energy, from which all statistical properties, including the thermal variance, are derived depends only on the first two moments of the input distribution. This means that we can take the results for the (auxiliary field) free energy of our replica calculation for gaussian inputs and the results for the thermal variance will hold for the binary input distribution.

We take several representative points from the learning curves for the worst case analysis (as computed by Engel and Fink), and calculate the probability that such a student will be chosen by the Gibbs learning algorithm. Technically, we should also take into consideration the variance induced by the fluctuations of the finite N corrections to the average worst performing student. This would give rise to two order $\mathcal{O}(N^{-1})$ variance bumps around the thermodynamic mean generalisation function and the worst generalisation function. Leaving aside such concerns, let us calculate the probability that a student would have an error larger than the average worst case analysis for a finite dimensional system. Using the assumed normal distribution for the generalisation function, the probability of an error worse than $\lambda\epsilon_f^{wc}$ (for some chosen $\lambda \leq 1$) is,

$$\mathrm{Prob}\left\{\epsilon_f > \lambda\epsilon_f^{wc}\right\} = \frac{1}{2}\mathrm{erfc}\left(\frac{-N^{\frac{1}{2}}}{\sqrt{2\mathrm{var}(\epsilon_f : \mathcal{W})}}\left(\lambda\epsilon_f^{wc} - \epsilon_g\right)\right), \tag{4.23}$$

where $\mathrm{var}(\epsilon_f : \mathcal{W})$ is the variance (not divided by N) of the generalisation function over the version space. We calculate the variance of the generalisation function by taking the thermal (weight space) variance of a large-sample test error ($M/N = 100$), checking that this is a good approximation for the case of zero α in the fashion outlined in section(2.9). The thermal variance of the generalisation function in terms of the overlap R is

$$\mathrm{var}(\epsilon_f : \mathcal{W}) = \frac{1}{\pi^2}\left\langle[\arccos(R)]^2\right\rangle_{\mathcal{E}} - \frac{1}{\pi^2}\left\langle[\arccos(R)]\right\rangle_{\mathcal{E}}^2. \tag{4.24}$$

As we know the distribution of the overlap at zero α, a straightforward calculation yields,

$$\mathrm{var}(\epsilon_f : \mathcal{W})(\alpha = 0) = \frac{1}{\pi^2}\left(\frac{\pi^2}{4} + \frac{1}{N}\right) - \frac{1}{4} + \mathcal{O}\left(N^{-2}\right) = \frac{0.405}{N} + \mathcal{O}\left(N^{-2}\right). \tag{4.25}$$

Using the large-sample test error ($M/N = 100$) predicts this value to the third decimal place. As a rather crude analysis, we take two values of α and read off the worst case analysis results from the paper by Engel and Fink[EvdB93], ($\alpha = 10, \epsilon_f^{wc} = 0.29, \epsilon_g = 0.1225, \mathrm{var}(\epsilon_f : \mathcal{W}) = 0.011$), ($\alpha = 20, \epsilon_f^{wc} = 0.14, \epsilon_g = 0.06, \mathrm{var}(\epsilon_f : \mathcal{W}) = 0.003$). In figure(4.6) we plot, using (4.23), the probability that a student from the version space will be chosen that has an error greater than

[4]One introduces the generalisation function as an auxiliary field in the Gibbs training energy. The second derivative of the resulting free energy gives the thermal variance of the generalisation function.

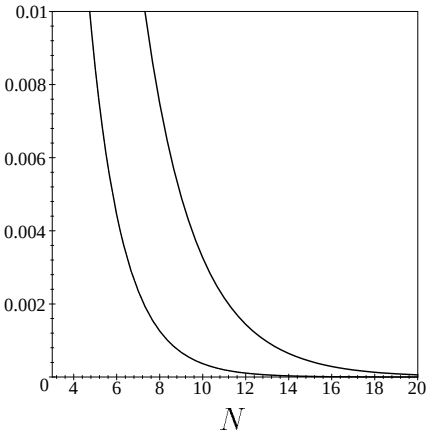

Figure 4.6. The probability of picking a student with a generalisation function worse than 0.8 of the error of the worst case student, versus the system size N. The upper curve is computed for $\alpha = 10$, and the lower for $\alpha = 20$.

the average worst case student, as calculated by Engel and Fink. It should be stressed that for such small system sizes as plotted, corrections to our finite size analysis will be relatively large. Nevertheless, we see that even for such small system sizes, the chance of picking a student that has an error close to that of the worst student is very small.

We are therefore lead to the conclusion that, although the PAC theory can yield "tight" *bounds* on the generalisation performance for the distribution independent case, with the assumption of a specific input distribution (here a gaussian), the overwhelming fraction of students in the version space perform similarly to the average case analysis, and not to the worst case, even for 'small' system sizes.

4.6 Summary

We have extended our analysis of finite size effects to a highly non-linear system, primarily by employing techniques from statistical mechanics. However, we have found a very simple relationship between the test error variance and the generalisation error. This simple relationship shows that the variance of the test error decays only with the generalisation error and not with the square of the generalisation error as for the linear perceptron. A (double) replica analysis showed that the performance of cross-validation is much less sensitive to the scheme used (random CV or block CV or optimal CV) than is the linear perceptron. For the optimal test set size, we again find a 2/3, power law scaling with the system size, where the prefactor is linear for both noisy and clean examples. As the worst case PAC analysis is generally concerned with binary output systems, we were able to calculate how likely a student is chosen with error close to the worst generalisation performance as predicted by the PAC theory. We found that this probability decreases exponentially with the system size, and that even for relatively small systems, errors close to the worst case generalisation performance are exceedingly unlikely.

4.7 Appendix

4.7.1 Replica Method

As explained in appendix(A.1.3), the thermal variance can be obtained by a straightforward extension of the single replica method. What is required is the average of the partition function, which was calculated in [GT90]. We refer the reader therefore to [GT90] and state here the final expression for the free energy which is composed of two contributions,

$$F = G_0 + \alpha G_r(\beta) \tag{4.26}$$

where the entropic term is given by,

$$-G_0 = \frac{1}{2}\left(\frac{q-R^2}{1-q} + \ln(1-q)\right) \tag{4.27}$$

and the replicated Hamiltonian is given by,

$$G_r(\beta) = 2\int_0^\infty Dy \int_{-\infty}^\infty \ln\left\{e^{-\beta} + \left(1-e^{-\beta}\right) H\left(\frac{t\sqrt{q-\gamma^2R^2} - y\gamma R}{\sqrt{1-q}}\right)\right\}, \tag{4.28}$$

where denote a gaussian measure, $Dy = (2\pi)^{-1/2}\exp(-y^2/2)\,dy$, and

$$H(u) = \int_u^\infty Dx = \frac{1}{2}\mathrm{erfc}\left(\frac{u}{\sqrt{2}}\right). \tag{4.29}$$

One must however bear in mind that the above formulae hold only in the region where replica symmetry is valid. This region is found by examining the stability of the replica solutions from the Hessian evaluated at the saddle point. As given in [GT90], the condition for stability is,

$$\Gamma = 1 - 2\alpha\int_0^\infty Dy\int_{-\infty}^\infty Dt\left(\frac{\partial^2}{\partial z^2}\ln\left[u_\beta + H(z)\right]\right)^2 > 0, \tag{4.30}$$

where $u_\beta = (\exp(\beta)-1)^{-1}$. However, for $T=0$, replica symmetry is stable and believed to be exact.

4.7.2 Double replica free energy

As the weight vector constraints for the binary perceptron are the same as those for the linear perceptron, the double replica entropic term is unchanged, as given by equation(A.32). The calculation of the Hamiltonian term follows the usual line of argument as presented in section(A.3), with the expression being of exactly the form given in equation(A.23). However, there does not exist any straightforward simplification of the general expression and as such, a numerical integration needs to be performed.

Chapter 5

On-line learning of multi-layer neural networks

Abstract

We complement the recent progress in thermodynamic limit analyses of mean on-line gradient descent learning dynamics in multi-layer networks by calculating the fluctuations that real, finite dimensional systems necessarily possess. Fluctuations from the mean dynamics are largest at the onset of specialisation as student hidden unit weight vectors begin to imitate specific teacher vectors, and increase with the degree of symmetry of the initial conditions. In light of this, we include a symmetry breaking term to stimulate asymmetry in the learning process, which typically also leads to a significant decrease in training time.

5.1 Introduction

The framework of the previous chapters has been that of *batch* learning in which the student is found from minimisation of the training error of many training examples. In the large input dimension limit, the training error approaches the average training error, and the error surface becomes 'smooth'. That is, the thermodynamic limit of the batch learning process is deterministic (in the limit of zero temperature). *On-line* learning can be thought of as a limiting case of batch learning in which the training error consists of only a single example. This leads to a simplified dynamics of learning, often more amenable to analysis, and recent advances in the theory of on-line learning have yielded insights into the training dynamics of multi-layer neural networks. In this chapter, we shall adopt the conventional notation adhered to in on-line learning, so that the *weights* parametrising the *student* network are successively updated according to the error incurred on a single example from a stream of input/output examples, $\{\boldsymbol{\xi}^\mu, \tau(\boldsymbol{\xi}^\mu)\}$, generated by a *teacher* network $\tau(\cdot)$[HP91, Ama67, HK94, BS92, BS95d, BSS95b]. The analysis of the resulting weight dynamics has previously been treated by assuming that the input dimension is infinite (the *thermodynamic limit*) such that a mean dynamics analysis is exact[SA95]. Here we present a more realistic treatment by calculating the corrections to the mean dynamics, induced by finite dimensional inputs[Sol94a, Hes94, BSS95a, BSS96].

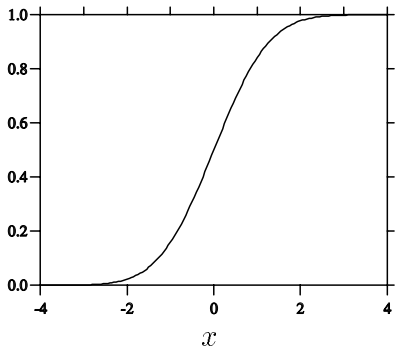

Figure 5.1. The error function, erf(βx) for $\beta = 1$ - a sigmoidal function and bounded between 0 and 1. For $\beta \to \infty$, erf(βx) approaches the step function.

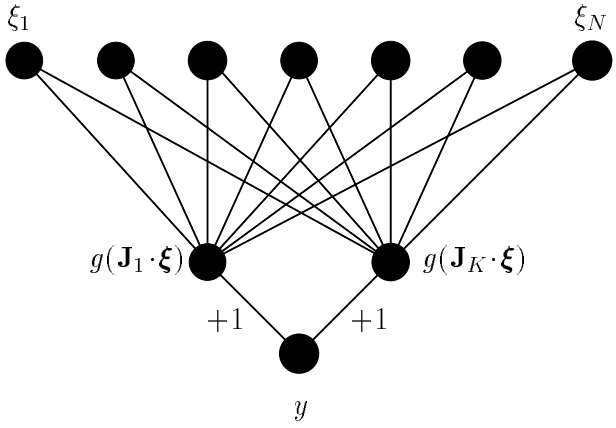

Figure 5.2. The soft committee machine. Each hidden unit computes the transfer of its activation. The final output value is given by the sum of the hidden unit outputs, $y = \sum_{m=1}^{K} g(\mathbf{J}_m \cdot \boldsymbol{\xi})$.

The Soft Committee Machine (SCM)

We assume that the *teacher* network the student attempts to learn is a *soft committee machine*[BS95d] of N inputs, and M hidden units, this being a one hidden layer network with weights connecting each hidden to output unit set to +1, and with each hidden unit n connected to all input units by $\mathbf{B}_n (n = 1..M)$. Explicitly, for the N dimensional training input vector $\boldsymbol{\xi}^\mu$, the output of the teacher is given by,

$$\zeta^\mu = \sum_{n=1}^{M} g(\mathbf{B}_n \cdot \boldsymbol{\xi}^\mu), \tag{5.1}$$

where $g(x)$ is the activation function of the hidden units, and we take $g(x) = \text{erf}(x/\sqrt{2})$, which is an analytically convenient sigmoidal function (figure(5.1)) . The teacher generates a stream of training examples $(\boldsymbol{\xi}^\mu, \zeta^\mu)$, with input components drawn from a normal distribution of zero mean, unit variance. The *student* network that attempts to learn the teacher, by fitting the training examples, is also a soft committee machine, but with K hidden units. For input $\boldsymbol{\xi}^\mu$, the student output is (see figure(5.1)),

$$\sigma(\mathbf{J},\boldsymbol{\xi}^{\mu}) = \sum_{i=1}^{K} g(\mathbf{J}_i \cdot \boldsymbol{\xi}^{\mu}), \tag{5.2}$$

where the student weights $\mathbf{J} = \{\mathbf{J}_i\}(i = 1..K)$ are sequentially modified to reduce the *error* that the student makes on an input $\boldsymbol{\xi}^{\mu}$,

$$\epsilon(\mathbf{J},\boldsymbol{\xi}^{\mu}) = \frac{1}{2}\left(\sigma(\mathbf{J},\boldsymbol{\xi}^{\mu}) - \zeta^{\mu}\right)^2 = \frac{1}{2}\left(\sum_{i=1}^{K} g(x_i^{\mu}) - \sum_{n=1}^{M} g(y_n^{\mu})\right)^2, \tag{5.3}$$

where the *activations* are defined $x_i^{\mu} = \mathbf{J}_i \cdot \boldsymbol{\xi}^{\mu}$, and $y_n^{\mu} = \mathbf{B}_n \cdot \boldsymbol{\xi}^{\mu}$. Gradient descent on the error equation(5.3) results in an update of the student weight vectors,

$$\mathbf{J}_i^{\mu+1} = \mathbf{J}_i^{\mu} - \frac{\eta}{N}\delta_i^{\mu}\boldsymbol{\xi}^{\mu}, \tag{5.4}$$

where,

$$\delta_i^{\mu} = g'(x_i^{\mu})\left[\sum_{n=1}^{M} g(y_n^{\mu}) - \sum_{j=1}^{K} g(x_j^{\mu})\right], \tag{5.5}$$

and g' is the derivative of the activation function g. The *learning rate* η is typically chosen to be small, so that convergence to a minimum is guaranteed (we shall later consider only very small η). Equation (5.4) is demonstrative of the general class of on-line learning rules in which a 'weight' parameter is updated according to a Markov process. As learning is a Markov process, the probability of the system being in a particular state depends on the state at the previous time step, and on the transition matrix between states. This leads to a master equation (an integro-differential equation) for the time evolution of the probability of being in a particular state, given initial conditions. A general solution of the master equation does not exist, and generally, one needs to make approximations such as a small learning rate. Fortunately, for the case we study here, the form of the update equation is simple enough for the average dynamics to be calculated exactly, without recourse to a small learning rate. Nevertheless, in the analysis of finite size effects, a small learning rate shall be assumed. Ultimately, the quantity of interest is the typical performance of the student on a randomly selected input example, given by the *generalisation error*,

$$\epsilon_g = \langle \epsilon(\mathbf{J},\boldsymbol{\xi}) \rangle, \tag{5.6}$$

where $\langle .. \rangle$ represents an average over the gaussian input distribution. As the dependence of the error on the input enters only through the student and teacher activations ξ and y_n, one can rewrite the probability of an input $\boldsymbol{\xi}$ in terms of the joint probability $\mathcal{P}(\mathbf{x},\mathbf{y}) = \int \mathrm{d}\boldsymbol{\xi} \Pi_i \delta\left(x_i - \boldsymbol{\xi}\cdot\mathbf{J}_i\right) \Pi_n \delta\left(y_n - \boldsymbol{\xi}\cdot\mathbf{B}_n\right) \exp\left(-\boldsymbol{\xi}\cdot\boldsymbol{\xi}/2\right)$, which gives,

$$\mathcal{P}(\mathbf{x},\mathbf{y}) = \frac{1}{\sqrt{(2\pi)^{M+K}|\mathcal{C}|}} \exp -\frac{1}{2}(\mathbf{x},\mathbf{y})^{\mathbf{T}}\mathcal{C}^{-1}(\mathbf{x},\mathbf{y}), \tag{5.7}$$

where $\mathcal{C}$ is the $(M+K)\times(M+K)$, correlation matrix,

$$\mathcal{C} = \begin{bmatrix} Q & R \\ R^T & T \end{bmatrix}, \tag{5.8}$$

and the elements of the submatrices are the *overlap parameters*, $R_{in} = \mathbf{J}_i \cdot \mathbf{B}_n$, $Q_{ij} = \mathbf{J}_i \cdot \mathbf{J}_j$, and $T_{nm} = \mathbf{B}_n \cdot \mathbf{B}_m (i, j = 1..K; n, m = 1..M)$. Using this distribution one finds that the generalisation error in terms of the order parameters is given by[SA95],

$$\epsilon_g(\mathbf{J}) = \frac{1}{\pi}\left\{ \sum_{ik} \arcsin \frac{Q_{ik}}{\sqrt{1+Q_{ii}}\sqrt{1+Q_{kk}}} + \sum_{nm} \arcsin \frac{T_{nm}}{\sqrt{1+T_{nn}}\sqrt{1+T_{mm}}} - 2\sum_{in} \arcsin \frac{R_{in}}{\sqrt{1+Q_{ii}}\sqrt{1+T_{nn}}} \right\}, \tag{5.9}$$

where $1 \leq i, k \leq K$ sum over the student hidden units, and $1 \leq n, m \leq M$. Using (5.4), we derive (stochastic) *update equations*,

$$R_{in}^{\mu+1} - R_{in}^{\mu} = \frac{\eta}{N} \delta_i^{\mu} y_n^{\mu}, \tag{5.10}$$

$$Q_{ik}^{\mu+1} - Q_{ik}^{\mu} = \frac{\eta}{N}\left(\delta_i^{\mu} x_j^{\mu} + \delta_k^{\mu} x_i^{\mu}\right) + \frac{\eta^2}{N^2} \delta_i^{\mu} \delta_k^{\mu} \boldsymbol{\xi}^{\mu} \cdot \boldsymbol{\xi}^{\mu}. \tag{5.11}$$

We have therefore reduced the N dimensional weight space update equation(5.4) to a set of $K(K+1)/2 + KM$ coupled difference equations. One could iterate these stochastic difference equations to find the order parameters at each discrete time step. However, the approach used in [SA95, BS92, BS95d, CC95] is to take the limit of an infinite input dimension, and form differential equations for the average overlaps from these stochastic difference equations.[1] We therefore average over the input distribution to obtain deterministic equations for the mean values of the overlap parameters, which are self-averaging in the thermodynamic limit. In this limit we treat $\mu/N = \alpha$ as a continuous variable and form differential equations for the *thermodynamic overlaps*, R_{in}^0, Q_{ik}^0,

$$\frac{dR_{in}^0}{d\alpha} = \eta \left\langle \delta_i y_n \right\rangle, \tag{5.12}$$

$$\frac{dQ_{ik}^0}{d\alpha} = \eta \left\langle \delta_i x_k + \delta_k x_i \right\rangle + \eta^2 \left\langle \delta_i \delta_k \right\rangle, \tag{5.13}$$

where, as before, $\langle .. \rangle$ represents an average over the input distribution. The expressions resulting from these gaussian averages are given in appendix(5.5.1). For given initial overlap conditions, the differential overlap equations can be integrated to find the mean dynamical behaviour of a student learning a teacher with an arbitrary numbers of hidden units [SA95] figure(5.2). Typically, ϵ_g decays rapidly to a *symmetric phase* in which there is near perfect symmetry between the hidden units. Such phases exist in learnable scenarios until sufficient examples have been presented to determine which student hidden unit will mimic which teacher hidden unit. For perfectly symmetric initial conditions, such *specialisation* is impossible in a mean dynamics analysis. The more symmetric the initial conditions are, the longer the trapping in the symmetric phase (see figure(5.3)).

[1] Whilst the transformation of these difference equations to differential equations is intuitively clear for infinite N, such that the differential equations and differential equations are equivalent, we mention that by choosing the discrete time updates to be Poisson distributed, the resulting differential equation is an exact model of the discrete dynamics[Hes94].

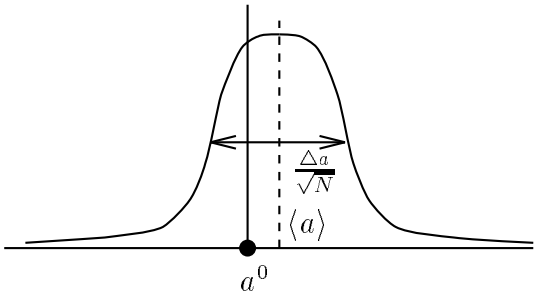

Figure 5.3. Schematic depiction of the small fluctuations ansatz for a learning rate set to $\eta = 1$. The thermodynamic distribution of the order parameter a is given by the delta peak centred at a^0. A finite dimensional system is characterised by a gaussian peak of variance $(\triangle a)^2/N$, centred at $\langle a \rangle = a^0 + a^1/N$.

5.2 Finite size effects

Large deviations from the mean dynamics can exist in the symmetric phase as a small perturbation from symmetry can determine which student hidden unit will specialise on which teacher hidden unit [BS95d].
We can rewrite (5.10,5.11) in the form

$$a^{\mu+1} - a^{\mu} = \frac{\eta}{N}\left(F_a + \eta G_a\right), \tag{5.14}$$

where $F_a + \eta G_a$ is the *update rule* for a general overlap parameter a. In order to investigate finite size effects, we make the following 'small fluctuations' ansaetze[Hes94] for the deviations of the update rules F_a (the same form is made for G_a) and overlap parameters a from their thermodynamic values,[2]

$$F_a = F_a^0 + \triangle F_a + \frac{1}{N}F_a^1, \qquad a = a^0 + \sqrt{\frac{\eta}{N}}\triangle a + \frac{\eta}{N}a^1, \tag{5.15}$$

where $\langle \triangle F_a \rangle = \langle \triangle a \rangle = 0$. The update rule ansatz is motivated by observing that the activations have variance $\mathcal{O}(1)$ which, iterated through (5.14), yield overlap variances of $\mathcal{O}(N^{-1})$. Terms of the form, $\triangle a$ represent *dynamic* corrections that arise due to the random examples. Terms like a^1 represent *static* corrections such that the mean of the overlap parameter a is given by $a^0 + \eta a^1/N$ - the thermodynamic average plus a correction. In order to simplify the analysis, we assume a small learning rate, η, so that the thermodynamic overlaps are governed by,

$$\frac{da^0}{d\tilde{\alpha}} = F_a^0, \tag{5.16}$$

where F_a^0 is the update rule F_a averaged over the input distribution, and the rescaled learning rate is given by

$$\tilde{\alpha} = \eta\alpha. \tag{5.17}$$

[2]If the order parameter represented by c is Q_{11}, then $c^0 = Q_{11}^0$, and $\triangle c = \triangle Q_{11}$.

Substituting (5.15) in (5.14) and averaging over the input distribution, we derive a set of coupled differential equations for the (scaled) covariances $\langle \triangle a \triangle b \rangle$, and static corrections a^1,

$$\frac{d\langle \triangle a \triangle b \rangle}{d\tilde{\alpha}} = \sum_c \langle \triangle a \triangle c \rangle \frac{\partial F_b^0}{\partial c^0} + \sum_c \langle \triangle b \triangle c \rangle \frac{\partial F_a^0}{\partial c^0} + \langle \triangle F_a \triangle F_b \rangle \tag{5.18}$$

$$\frac{1}{2}\frac{d^2 a^0}{d\tilde{\alpha}^2} + \frac{da^1}{d\tilde{\alpha}} = \sum_b b^1 \frac{\partial F_a^0}{\partial b^0} + \frac{1}{2}\sum_{bc} \langle \triangle b \triangle c \rangle \frac{\partial^2 F_a^0}{\partial b^0 \partial c^0} + G_a^1. \tag{5.19}$$

Summations are over all overlap parameters, $\{Q_{ij}, R_{in} | i,j = 1..K, n = 1..M\}$. The static corrections G_a^1 to the update rules are calculated analytically in the appendix(5.5.2). The elements $\langle \triangle F_a \triangle F_b \rangle$ are found explicitly by calculating the covariance of the update rules F_a, and F_b (see section(5.5.3)). [3] From this differential equation for the density, we obtain (5.18).

Initially, the fluctuations $\langle \triangle F_a \triangle F_b \rangle$ are set to zero, and equations (5.16,5.18) are then integrated to find the evolution of the covariances, $\mathrm{cov}(a,b) = (\eta/N)\langle \triangle a \triangle b \rangle$, and the corrections to the thermodynamic average values, $(\eta/N)a^1$. The average finite size correction to the generalisation error is given by

$$\epsilon_g = \epsilon_g^0 + \frac{\eta}{N}\epsilon_g^1, \tag{5.20}$$

where,

$$\epsilon_g^1 = \sum_a a^1 \frac{\partial e_g^0}{\partial a} + \frac{1}{2}\sum_{ab} \langle \triangle a \triangle b \rangle \frac{\partial^2 \epsilon_g^0}{\partial a^0 \partial b^0}. \tag{5.21}$$

These results enable the calculation of finite size effects for an arbitrary teacher/student learning scenario. For demonstration, we calculate the finite size effects for a student with two hidden units learning a teacher with one hidden unit. In this over-realisable case, one of the student hidden units eventually specialises on the single teacher hidden unit, while the other student hidden unit decays to zero. In figure(5.2), we plot the thermodynamic limit generalisation error alongside the $\mathcal{O}(N^{-1})$ correction. In figure(5.2a) there is no significant symmetric phase, and the finite size corrections (figure(5.2b)) are small. For a finite size correction of less than 10%, we would require an input dimension of around $N > 25\eta$. For the more symmetric initial conditions (figure(5.3a)) there is a very definite symmetric phase, for which a finite size correction of less than 10% (figure(5.3b)) would require an input dimension of around $N > 50,000\eta$. As the initial conditions approach perfect symmetry, the finite size effects diverge, and the mean dynamical theory becomes inexact. Using the covariances, we can analyse the way in which the student breaks out of the symmetric phase by specialising its hidden units. For the isotropic teacher scenario $T_{nm} = \delta_{nm}$, and $M = K = 2$, learning proceeds such that one can approximate, $Q_{22} = Q_{11}, R_{22} = R_{11}$. By analysing the eigenvalues of the covariance matrix $\langle \triangle a \triangle b \rangle$, we found that there is a sharply defined principal direction, the components of which we show in figure(5.4). Initially, all components of the principal direction are similarly correlated, which corresponds to the symmetric region. Then, around $\tilde{\alpha} = 20$, as the symmetry breaks, R_{11} and

[3] The derivation of the above equations for the fluctuations is consistent with the Van Kampen expansion approach, whereby the Kramers-Moyal representation of the master equation is expanded under the 'small fluctuation' ansatz, yielding a partial differential equation for the fluctuation probability density [Hes94].

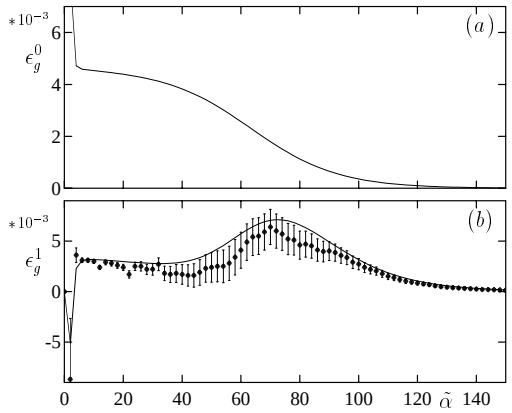

Figure 5.2.

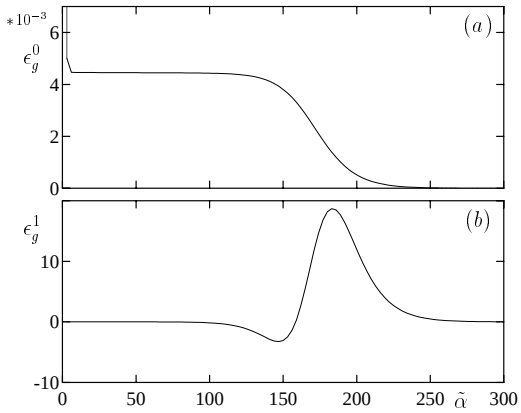

Figure 5.3.

Figure 5.2. Two student hidden units, one teacher hidden unit. Non zero initial parameters: $Q_{11} = 0.2, Q_{22} = R_{11} = 0.1$. (a) Thermodynamic generalisation error, ϵ_g^0. (b) $\mathcal{O}(N^{-1})$ correction to the generalisation error , ϵ_g^1. Simulation results for $N = 10, \eta = 0.1$ and (half standard deviation) error bars are drawn.

Figure 5.3. Two student hidden units, one teacher hidden unit. Initially, $Q_{11} = 0.1$, with all other parameters set to zero. (a) Thermodynamic generalisation error ϵ_g^0. (b) $\mathcal{O}(N^{-1})$ correction to the generalisation error, ϵ_g^1.

R_{21} become maximally anti-correlated, whilst there is minimal correlation between the Q_{11} and Q_{12} components. This corresponds well with predictions from perturbation analysis [SA95]. Essentially, the symmetry breaking is characterised by a specialisation process in which each student vector increases its overlap with one particular teacher weight, whilst decreasing its overlap with other teacher weights. After the specialisation has occurred, there is a growth in the anti-correlation between the student length and its overlap with other students. The asymptotic values of these correlations are in agreement with the convergence fixed point, $R^2 = Q = 1$.

5.3 Breaking the symmetry

In light of possible prolonged symmetric phases, we explicitly break the symmetry of the student hidden units by imposing an ordering on the student lengths, $Q_{11} \geq Q_{22} \geq ... \geq Q_{KK}$. This constraint is enforced in a 'soft' manner by including an extra term to (5.3),

$$\epsilon^{\dagger} = \frac{1}{2} \sum_{j=1}^{K-1} h\left(Q_{j+1j+1} - Q_{jj}\right), \tag{5.22}$$

where $h(x)$ approximates the step function,

$$h(x) = \frac{1}{2}\left(1 + \mathrm{erf}\left(\frac{\beta}{\sqrt{2}}x\right)\right). \tag{5.23}$$

This leads to an easily implementable modification involving the addition of a gaussian term in the student weight lengths to the weight update rule (*cf.* (5.4)). In figure(5.5), we show

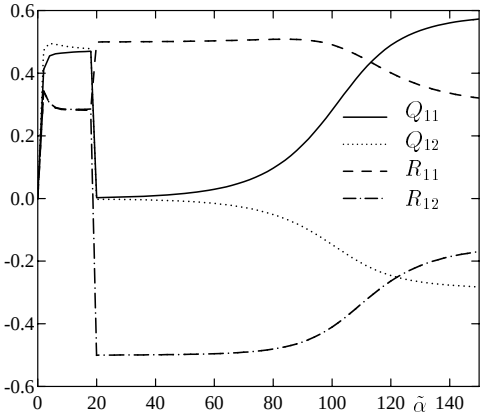

Figure 5.4.

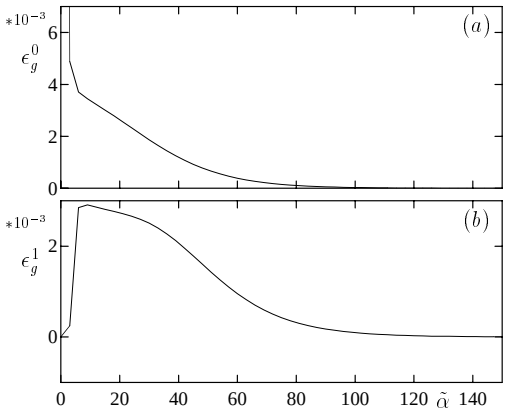

Figure 5.5.

Figure 5.4. (a) The normalised components of the principal eigenvector for the isotropic teacher. $M = K = 2$, $(Q_{22} = Q_{11}, R_{22} = R_{11})$. Non zero initial parameters $Q_{11} = 0.2$, $Q_{22} = 0.1$, $R_{11} = 0.001$, $R_{22} = 0.001$.

Figure 5.5. Two student hidden units, one teacher hidden unit. The initial conditions are as in figure(5.3). (a) Thermodynamic generalisation error, ϵ_g^0 . (b) $\mathcal{O}(N^{-1})$ correction to the generalisation error, ϵ_g^1.

the overlap parameters and their fluctuations for β=10, $K = 2, M = 1$. This graph is to be compared to figure(5.3) for which the initial conditions are the same. There is now no collapse to an initial symmetric phase from which the student will eventually specialise. Also, the initial convergence to the optimal values is much faster. As there is essentially no symmetric phase, the finite size corrections are much reduced. They are now largest around the initial value of $\tilde{\alpha}$ where the overlap parameters are very symmetric, becoming rapidly smaller due to the large driving force away from this near-symmetric region. For the case in which the teacher weights are equal, the constraint (5.22) will prevent the student from converging optimally. In light of this, we need to adapt the constraint as learning proceeds. A naive scheme is to adapt the steepness, β, such that it is inversely proportional to the average of the gradients Q_{ii}, which decreases as the dynamics converge asymptotically.

5.4 Summary

In this work we have complemented the recent significant advances in the theory of multi-layer networks by finding the conditions under which thermodynamic limit calculations in on-line learning can be expected to be representative of real learning scenarios. We provided, also, more detailed insights into the specialisation process out of the symmetric phase. In addition, we found that by breaking the internal symmetries of the network, we were able to reduce both finite size effects and training time. We conjecture that such symmetry breaking is potentially of great benefit in the practical field of neural network training.

5.5 Appendix: On-line learning

In the following appendices, we outline the derivation of the update equations in the thermodynamic limit, and demonstrate how finite size corrections to these equations, both static and dynamic, can be obtained.

5.5.1 Appendix: Thermodynamic equations

From (5.13) and (5.12), we require the averages of two types of multivariate gaussian integrals over the distribution $\mathcal{P}(\mathbf{x}, \mathbf{y})$ given in (5.7). The terms which are proportional to η involve the three dimensional integral,

$$I_3 \equiv \langle g'(u) v g(w) \rangle \,, \tag{5.24}$$

where u is one of the components of $\mathbf{x}$ while u and w can be components of either $\mathbf{x}$ or $\mathbf{y}$. Similarly, for the terms proportional to η^2, we need to evaluate integrals of the form,

$$I_4 \equiv \langle g'(u) g'(v) g(w) g(z) \rangle \,, \tag{5.25}$$

where u and v are components of $\mathbf{x}$ while w and z can be components of either $\mathbf{x}$ or $\mathbf{y}$. Also required are the integrals of the above form given for I_3 and I_4 when two of the arguments are equal. However, this simply means that those two arguments are fully correlated, and the resulting integral is found by modifying the correlation matrix accordingly. The full equations for the dynamics of the thermodynamic overlaps is given by,

$$\frac{dR_{in}}{d\tilde{\alpha}} = \eta \left\{ \sum_m I_3(i,n,m) - \sum_j I_3(i,n,j) \right\}, \tag{5.26}$$

$$\begin{aligned} \frac{dQ_{ik}}{d\tilde{\alpha}} &= \eta \left\{ \sum_m I_3(i,k,m) - \sum_j I_3(i,k,j) \right\} + \eta \left\{ \sum_m I_3(k,i,m) - \sum_j I_3(k,i,j) \right\} \\ &+ \eta^2 \left\{ \sum_{n,m} I_4(i,k,n,m) - 2 \sum_{j,n} I_4(i,k,j,n) + \sum_{j,l} I_4(i,k,j,l) \right\}, \end{aligned} \tag{5.27}$$

The function $I_3(i,n,j)$ is defined as the three dimensional gaussian average, $I_3(i,n,j) \equiv \langle g'(x_i) y_n g(x_j) \rangle$, where the covariance matrix is given by the projection of the full covariance matrix (5.8) onto the subspace spanned by x_i, y_n, and x_j:

$$\mathcal{C}_3 = \begin{pmatrix} Q_{ii} & R_{in} & Q_{ij} \\ R_{in} & T_{nn} & R_{jn} \\ Q_{ij} & R_{jn} & Q_{jj} \end{pmatrix} \tag{5.28}$$

The value of I_3 is then given by evaluating,

$$I_3 = \frac{2}{\pi} \frac{1}{\sqrt{\Lambda_3}} \frac{C_{23}(1 + C_{11}) - C_{12} C_{13}}{1 + C_{11}}, \tag{5.29}$$

where, in terms of the projection onto the four dimensional covariance matrix we have defined,

$$\Lambda_3 = (1 + C_{11})(1 + C_{33}) - C_{13}^2 \tag{5.30}$$

The integral I_4 is found similarly by projecting the full covariance matrix onto the subspace spanned by its four arguments, for which,

$$I_4 = \frac{4}{\pi^2}\frac{1}{\sqrt{\Lambda_4}}\arcsin\left(\frac{\Lambda_0}{\sqrt{\Lambda_1\Lambda_2}}\right), \tag{5.31}$$

where

$$\begin{aligned}
\Lambda_4 &= (1+C_{11})(1+C_{22}) - C_{12}^2 \\
\Lambda_0 &= \Lambda_4 C_{34} - C_{23}C_{24}(1+C_{11}) - C_{13}C_{14}(1+C_{22}) + C_{12}C_{13}C_{24} + C_{12}C_{14}C_{23} \\
\Lambda_1 &= \Lambda_4(1+C_{33}) - C_{23}^2(1+C_{11}) - C_{13}^2(1+C_{22}) + 2C_{12}C_{13}C_{23} \\
\Lambda_2 &= \Lambda_4(1+C_{44}) - C_{24}^2(1+C_{11}) - C_{14}^2(1+C_{22}) + 2C_{12}C_{14}C_{24}
\end{aligned} \tag{5.32}$$

The resulting dynamical equations enable the average case dynamics of soft-committee machine learning to be found for arbitrary student and teacher architectures.

5.5.2 Appendix: The static correction to the update rule

We turn our attention to the term in (5.11),

$$\frac{1}{N}\left\langle \delta_i\delta_k \boldsymbol{\xi}\cdot\boldsymbol{\xi}\right\rangle = \left\langle \delta_i\delta_k\right\rangle + \mathcal{O}\left(N^{-1}\right), \tag{5.33}$$

where we are interested in calculating the order N^{-1} contribution. The thermodynamic term $\langle\delta_i\delta_k\rangle$ has already been found, and is given as the factor of the η^2 term in (5.27). Each student weight vector gives the constraint $x_i = \mathbf{J}_i\cdot\boldsymbol{\xi}$, and similarly, each teacher weight vector gives the constraint, $y_n = \mathbf{B}_n\cdot\boldsymbol{\xi}$. As the contribution of the student and teacher weight vectors only occur through these $K+M$ scalar products with the input vector, we can consider the weight vectors to be $K+M$ dimensional, setting the remaining $N-K-M$ components to zero. This means that we need also only consider the first $K+M$ components of the input vector, $\hat{\boldsymbol{\xi}} = (\xi_1, .., \xi_{K+M})$. We write the constraints in the matrix form,

$$\begin{pmatrix} x_1 \\ . \\ x_K \\ y_1 \\ . \\ y_M \end{pmatrix} = \begin{pmatrix} \mathbf{J}_1{}^{\mathbf{T}} \\ . \\ \mathbf{J}_K{}^{\mathbf{T}} \\ \mathbf{B}_1{}^{\mathbf{T}} \\ . \\ \mathbf{B}_M{}^{\mathbf{T}} \end{pmatrix} \cdot \hat{\boldsymbol{\xi}} \tag{5.34}$$

Inverting the above equation, we can write

$$\hat{\boldsymbol{\xi}}^2 = \mathbf{x}^{\mathbf{T}}\left(\mathrm{W}\mathrm{W}^{\mathbf{T}}\right)^{-1}\mathbf{x}, \tag{5.35}$$

where W is the matrix of student/teacher weight vector components given in (5.34). The prefactor of the order N^{-1} correction to the thermodynamic value of (5.33) is then given by,

$$\left\langle \delta_i\delta_k \mathbf{x}^{\mathbf{T}}\left(\mathrm{W}\mathrm{W}^{\mathbf{T}}\right)^{-1}\mathbf{x}\right\rangle - (K+M)\left\langle\delta_i\delta_k\right\rangle. \tag{5.36}$$

As $\mathrm{WW}^{\mathbf{T}}$ is just the covariance matrix (5.8), after some algebra, we can write the static correction as,

$$-2\frac{\partial}{\partial\lambda}\left\langle\delta_i\delta_k(\lambda^{-1}\mathcal{C})\right\rangle\Big|_{\lambda=1}, \tag{5.37}$$

where $\langle\delta_i\delta_k(\lambda^{-1}\mathcal{C})\rangle$ is the value of the expression $\langle\delta_i\delta_k\rangle$ calculated using the modified covariance matrix $\lambda^{-1}\mathcal{C}$.

5.5.3 Appendix: Covariance of the update rules

In section(5.2) we mentioned that one needs to find the covariance of the update rules explicitly by calculation. Ignoring terms higher than η^2 we note that in order to calculate the covariance of two update rules, we need only find the average of the general quantity $\langle\delta_i\delta_k x_\alpha x_\beta\rangle$ where x_α and x_β stand for arbitrary activations. The method of calculating these is straightforward: we know how to find $\langle\delta_i\delta_k\rangle$ (this is just I_4), and the gaussian measure $\mathcal{P}(\mathbf{x},\mathbf{y})$ is the exponential of a quadratic form containing all the possible $x_\alpha x_\beta$. Therefore, by adding to the each element of the inverse of the full covariance matrix (5.8) the parameter λ, and by differentiating the average $\langle\delta_i\delta_k\rangle$ with this modified covariance matrix, we 'pull down' the factor $x_\alpha x_\beta$:

$$\langle\delta_i\delta_k x_\alpha x_\beta\rangle = -\frac{\partial}{\partial\lambda}\int\frac{d\mathbf{x}d\mathbf{y}}{(2\pi)^{M+K}|\mathcal{C}|^{\frac{1}{2}}}\exp\left\{-\frac{1}{2}(\mathbf{x},\mathbf{y})^{\mathbf{T}}\left(\mathcal{C}^{-1}+\Lambda\right)(\mathbf{x},\mathbf{y})\right\}\delta_i\delta_k\Big|_{\lambda=0} \tag{5.38}$$

where the matrix Λ has zero elements, except where those elements contribute to the term $x_\alpha x_\beta$, which then contain λ (the diagonal elements contain 2λ). After a few lines of algebra, we find,

$$\langle\delta_i\delta_k x_\alpha x_\beta\rangle = -\frac{\partial}{\partial\lambda}\left\langle\delta_i\delta_k\left(\hat{\mathrm{C}}^{(4)}\right)\right\rangle\Big|_{\lambda=0} + [\mathcal{C}]_{\alpha,\beta}\langle\delta_i\delta_k\rangle, \tag{5.39}$$

where $\hat{\mathrm{C}}^{(4)}$ is the projection onto the four dimensional subspace of the modified full covariance matrix $(\mathcal{C}^{-1}+\Lambda)^{-1}$.

Chapter 6

Does extra knowledge necessarily improve generalisation?

Abstract

The generalisation error is a widely used performance measure employed in the analysis of adaptive learning systems. This measure is generally critically dependent on the knowledge that the system is given about the problem it is trying to learn. We examine to what extent it is necessarily the case that an increase in the knowledge that the system has about the problem will reduce the generalisation error. Using the standard definition of the generalisation error, we present simple cases for which the intuitive idea of 'reducivity' - that more knowledge will improve generalisation - does not hold. Under a simple approximation, however, we find conditions to satisfy 'reducivity'. Finally, we calculate the effect on the generalisation error for weights constrained to a particular sign. This particular restriction results in a significant improvement in generalisation performance.

6.1 Introduction

The employment of a priori knowledge in designing a learning machine is crucial to the success of the machines ability to generalise well. Given that knowledge affects the generalisation ability, our aim here is to address the following question: does more knowledge necessarily improve generalisation? Intuitively, the answer to this question would seem to be 'yes', depending, of course, on the definitions of knowledge and generalisation. Nevertheless, this question phrases a possible desiderata, which itself can affect the design of learning machines. Again, we formulate the problem in the language of learning from examples[Bar96, BS95a]

A training set of input/output pairs is generated by some teacher function, and the task is to find a student function whose outputs match closely the outputs of the teacher function on the training set. Constraints on the set of possible teacher functions that generate the training set are critical in narrowing down the search for a good student. Indeed, without any constraints it is an impossible task to find a student that generalises to unseen examples (see the discussion in chapter(1)). A priori assumptions are therefore made as to the form of the teacher, that is,

restrictions are imposed on the space of teacher functions. Throughout this chapter we assume that the spaces of the teacher and student functions are the same. The learning problem is then realisable in the sense that amongst the student space, there is a student that will match perfectly the output of the teacher on all possible inputs. We denote the teacher/student space of functions by $F(\Psi)$, and a particular mapping as $y = f(x, \theta)$ for $f \in F(\Psi)$ and $\theta \in \Psi$, where the output is denoted by y, and the input by x. A particular mapping that a function performs is represented by the point θ in the parameter space Ψ. We assume that a single teacher θ^0 generates the noise free set of training data $\mathcal{L} = \{x^\sigma, f(x^\sigma, \theta^0)\}$, where σ indexes each element of the training set $\mathcal{L}$. In the learning problem, one attempts to find a student $f(x, \theta)$ that matches the teacher $f(x, \theta^0)$ on the training set.[1] To measure the extent to which the student has learnt the teacher, an error measure $\epsilon_g(\theta, \theta^0, x)$ is defined. The set of admissible students, represented by the parameter space $\Theta \in \Psi$, is determined by the requirement of minimising the error measure on all examples in the training set, and satisfying a priori constraints on the student. Hence Θ expresses all the information that the student has about the teacher.[2] In section(6.2) we review briefly the definition of the generalisation error, before formulating the original question more rigourously. We subsequently consider specific cases, beginning with the simplest possible - a one dimensional version space. In section(6.3), we analyse higher dimensions, using the linear perceptron as the function space $F(\Psi)$ and present results for sign constrained weights. In section(6.4) we conclude with a summary of the main results.

6.2 General Theory

6.2.1 The Generalisation Error

To measure how well the student performs on the training set, the training energy is formed, $E_{tr}(\mathbf{w}|\mathcal{P}) \propto \sum_{\sigma=1}^{p} \epsilon(\theta, \theta^0, x^\sigma)$. The student is found by minimising the training error with respect to the parameter θ, whilst also adhering to additional a priori constraints. This is typically achieved by stochastic gradient descent, resulting in a post training distribution of students, $P(\theta|\mathcal{L}) \propto P^{pri}(\mathbf{w}) \exp(-E_{tr}(\mathbf{w}|\mathcal{P})/T)$ where the temperature, T, controls the randomness of the stochastic algorithm (see *e.g.,* [WRB93]). $P^{pri}(\mathbf{w})$ is the a priori constraint on the student. In the limit of zero T, the distribution of students becomes uniform over those that have zero training error and satisfy the a priori constraints; this space of student functions is known as the *version space*, which we denote by Θ.[3] In section(6.3.3), we present results for non-zero T, but for the rest of this chapter, zero T is implied. To find the expected error that a student makes on a random example input, termed the generalisation function, we average the error over the input distribution, $P(x)$, giving $\epsilon_f(\theta, \theta^0) = \int dx P(x) \epsilon_g(\theta, \theta^0, x)$.[4] Hence, given the teacher, $\epsilon_f(\theta, \theta^0)$ measures the expected error that a student θ makes, given that the teacher is θ^0 and that the student is θ. As the student does not know the teacher, we assume that Θ

[1] Extra regularisation conditions on the student, such as weight decay, will not be considered here.

[2] We briefly note that the assumption that the set of admissible functions is all that is known about the teacher function is found also in the PAC approach (see *e.g.,* [Hau94]); we addresses, however, somewhat different issues.

[3] The student distribution we consider is known also as exhaustive learning (see *e.g.,* [SSS90]).

[4] An extension to this framework is to consider the off-training-set error (see *e.g.,* [Wol92]) in which the expected error of the student is calculated for test examples not included in the training set.

expresses all the information that the student has about the teacher. The generalisation error is then defined as the expected performance of a random student from Θ, given a random teacher from Θ,

$$\epsilon_g(\Theta) = \left\langle \epsilon_f(\theta, \theta^0) \right\rangle_{\theta \in \Theta, \theta^0 \in \Theta} , \tag{6.1}$$

where $\langle .. \rangle_{\theta^0 \in \Theta}$ and $\langle .. \rangle_{\theta \in \Theta}$ represent averages over the version space Θ.[5] We write $\epsilon_g(\Theta)$ to emphasize that the generalisation error is a function of the version space.

Intuitively, one expects that any further restrictions or a priori assumptions, resulting in a smaller version space, must necessarily reduce the generalisation error. Furthermore, an understanding of the relation between the geometry of the version space and the generalisation error is desirable in light of possible algorithms that are based upon the geometry of the version space (*e.g.,* some forms of query learning which perform version space bisection). To formulate this more precisely, we make the following definition.

Definition
$F(\Theta')$ is an 'error reduced' function space of $F(\Theta)$ if $\epsilon_g(\Theta') < \epsilon_g(\Theta)$ for $\Theta' \subset \Theta$, and we say that 'reducivity' holds.

In this chapter we examine which subsets Θ' of Θ are error reducing, according to the preceding definition. We mention briefly that one can also consider the generalisation error for a fixed teacher, $\epsilon_g(\theta^0, \Theta) = \langle \epsilon_f(\theta, \theta^0) \rangle_{\theta \in \Theta}$, and check reducivity with the teacher assumed known. We show in a later section, however, that the main results also hold for $\epsilon_g(\theta^0, \Theta)$, and concentrate accordingly on $\epsilon_g(\Theta)$.

6.2.2 One Dimensional Version Space

We begin with the simplest possible case of a one dimensional version space, assuming that it can be paramaterized by a connected interval on the real line, which we write, without loss of generality, as [0,a]. Furthermore, we assume that the generalisation function can be written as, $\epsilon_f(\theta, \theta^0) = dist(\mid \theta - \theta^0 \mid)$, for some function $dist(\cdot)$.[6] $\epsilon_g(\Theta)$ is then simply $\epsilon_g(a) = \int_0^a d\theta P(\theta) \int_0^a d\theta^0 P(\theta^0) dist(\mid \theta - \theta^0 \mid)$, where $P(\cdot)$ is the parameter space distribution. For a uniform distribution, $P(\theta) = P(\theta^0) = 1/a$, and we can write,

$$\epsilon_g(a) = \frac{2}{a^2} \int_0^a dy \int_0^y dx dist(x),$$

for which the requirement of reducivity *i.e.,* $\frac{d\epsilon_g(a)}{da} > 0$ becomes

$$\int_0^a dx dist(x) > \frac{2}{a} \int_0^a dy \int_0^y dx dist(x),$$

[5] In this joint average of $\epsilon_f(\theta, \theta^0)$ over the version space, we assume independence of the student and the teacher: As the training set is fixed, we write $P(\theta^0, \theta | \mathcal{L}) = P(\theta | \theta^0, \mathcal{L}) P(\theta^0 | \mathcal{L})$. With the assumption $P(\theta | \theta^0, \mathcal{L}) = P(\theta | \mathcal{L})$, we have that θ and θ^0 are independently distributed over Θ.

[6] In this assumption as to the form of the generalisation function we have in mind a larger class of error measures than the square error measure, $\epsilon(\theta, \theta^0, x) = 1/2 \left[f(x, \theta) - f(x, \theta^0) \right]^2$, for which the assumption $\epsilon_f(\theta, \theta^0) = dist(\mid \theta - \theta^0 \mid)$ would hold only for the linear function $f(x, \theta) = x\theta$ and $g(s) = s^2$.

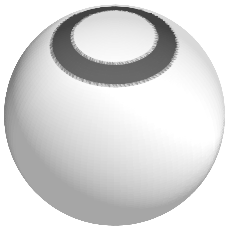

Figure 6.1. A sphere of radius $\sqrt{3}$. The shaded region represents the version space, $\Theta = \{\theta \in [0.4, 0.6], \phi \in [0, 2\pi]\}$. Making Θ smaller by pushing the inner boundary towards the outer boundary does not result in a reduction in generalisation error.

This is equivalent to

$$a\langle dist\rangle_a - \frac{2}{a}\int_0^a dx x\langle dist\rangle_x > 0,$$

where $\langle dist\rangle_x$ is the average value of $dist(\cdot)$ over the interval $[0, x]$. For a monotonically increasing function, $\langle dist\rangle_a > \langle dist\rangle_x$ $(a > x)$, and thus reducivity holds for all monotonic increasing functions defined on the real line.

Unfortunately, for higher dimensional cases, it is not generally possible to separate the dependence of the generalisation function into a summation over the individual components of the parameter vector, *i.e.,* $\epsilon_f(\theta, \theta^0) \neq \sum_i^n dist(\mid \theta_i - \theta_i^0 \mid)$, where n is the dimension of the parameterisation, and more complicated effects can appear. In the following sections we concentrate on the linear perceptron, beginning with an explicit example of a two dimensional version space which violates the error reduction property.

6.3 The Linear Perceptron

For the noise free linear perceptron, the inputs are represented by N dimensional real vectors, $\mathbf{x} \in \Re^N$, and the output is a single valued real variable, $y \in \Re$ (see *e.g.,* [HP91]). The inputs $\mathbf{x}$ are assumed drawn independently and identically from a zero mean, unit covariance matrix Gaussian distribution. The teacher outputs are given by $f(\mathbf{x}, \mathbf{w}^0) = \mathbf{w}^0 \cdot \mathbf{x}/\sqrt{N}$. Similarly, the student outputs are $f(\mathbf{x}, \mathbf{w}) = \mathbf{w} \cdot \mathbf{x}/\sqrt{N}$. We also impose the additional a priori spherical constraint on both the student and teacher, $\mathbf{w} \cdot \mathbf{w} = \mathbf{w}^0 \cdot \mathbf{w}^0 = N$. The error measure is taken to be proportional to the squared difference between the teacher and student outputs, $\epsilon(\mathbf{w}, \mathbf{w}^0, \mathbf{x}) = (\mathbf{w} \cdot \mathbf{x} - \mathbf{w}^0 \cdot \mathbf{x})^2/2N$. We proceed to analyse this model for a specific version space.

6.3.1 A Two Dimensional Version Space

We look now at the three dimensional linear perceptron. A point on the surface of a three dimensional sphere of radius $r = \sqrt{3}$ is given by the ordered pair (ϕ, θ), which represents the usual spherical polar coordinate parameterisation.[7]

[7] $w_1 = r\cos(\phi)\sin(\theta), w_2 = r\sin(\phi)\sin(\theta), w_3 = r\cos(\theta)$ where, $r = \sqrt{3}$ for the spherical normalisation condition.

Assuming a zero mean, unit covariance matrix gaussian input distribution, the generalisation function is $\epsilon_f(\mathbf{w}, \mathbf{w}^0) = 1 - \mathbf{w}\cdot\mathbf{w}^0/N$. We write the scalar product in this expression in spherical coordinates and average over the version space given by $\Theta = \{(\phi, \theta), \phi \in [a, b], \theta \in [c, d]\}$. A straightforward calculation gives

$$\epsilon_g(\Theta) = 1 - \frac{1}{(d-c)^2}\left(\lambda\left(\cos(d) - \cos(c)\right)^2 + \left(\sin(d) - \sin(c)\right)^2\right),$$

where $\lambda = 2\left(1 - \cos(b-a)\right)/(b-a)^2$. To violate reducivity we look for regions such that when we reduce the width of, for example, the interval $[c, d]$, the generalisation error increases. Without loss of generality, then, we search for regions for which $\partial\epsilon_g(\Theta)/\partial c > 0$, and we plot one such region in figure(6.1). To find such a region explicitly, we look for the boundary at which $\partial\epsilon_g(\Theta)/\partial c = 0$, and define $\Lambda(c, d) = \lambda(\partial\epsilon_g(\Theta)/\partial c = 0)$, which is given by the equation,

$$\Lambda = \frac{\sin c - \sin d}{\cos d - \cos c}\left\{\frac{\sin c - \sin d + (d-c)\cos c}{\cos d - \cos c + (d-c)\sin c}\right\}.$$

In figure(6.2)(a), we show how this relates to the reducivity. In region (1), Λ varies between 0 and 1, and $\partial\epsilon_g(\Theta)/\partial c$ can be of either sign, depending on the value of λ; thus in region (1), reducivity depends critically on $\delta = b - a$. For $\lambda > \Lambda$, $\partial\epsilon_g(\Theta)/\partial c < 0$, and for $\lambda < \Lambda$, $\partial\epsilon_g(\Theta)/\partial c > 0$. In both regions (2) and (3) $\Lambda \notin [0, 1]$ and, as $\lambda \in [0, 1]$ (figure(6.2)(b)), the sign of $\partial\epsilon_g(\Theta)/\partial c$ is fixed, independent of $[a, b]$. In fact, in regions (2) and (3), reducivity is guaranteed. In region (2), as δ decreases (*i.e.,* [a,b] shrinks), $\partial\epsilon_g(\Theta)/\partial c$ becomes increasingly negative, whereas in region (3), for decreasing δ, $\partial\epsilon_g(\Theta)/\partial c$ becomes less negative. The boundary between regions (2) and (3) is given by the solution of $\cos d - \cos c + (d-c)\sin c = 0$. Despite the simplicity of the example, the behaviour of reducivity on the sphere is non trivial.

At this point, the reader may well conjecture that reducivity would be guaranteed for convex regions Θ and $\Theta' \subset \Theta$. (In general, a region is convex if the geodesic connecting any two points lies wholly within the region itself). Perhaps somewhat surprisingly, we demonstrate in the next section that convexity is not a sufficient condition for reducivity.

6.3.2 Euclidean Approximation To The Version Space

For simplicity, we concentrate on version spaces small enough such that the region can be considered Euclidean. For the linear perceptron described above, this corresponds to a region small enough such that the curved surface of the hypersphere appears 'flat'. By writing $\mathbf{w} = \mathbf{c} + \mathbf{w}$, and $\mathbf{w}^0 = \mathbf{c} + \mathbf{w}^0$, where $\mathbf{c}$ lies in the space Θ, we write the generalisation error as

$$\epsilon_g(\tilde{\Theta}) = \frac{1}{2N}\left\langle\left(\mathbf{w} - \mathbf{w}^0\right)^2\right\rangle_{\mathbf{w},\mathbf{w}^0\in\tilde{\Theta}},$$

where $\tilde{\Theta}$ is the approximately flat region on the sphere. As $\mathbf{w}$ and $\mathbf{w}^0$ are uncorrelated, this can be written in the form,

$$\epsilon_g(\tilde{\Theta}) = \frac{1}{N}\left(\left\langle\mathbf{w}^2\right\rangle_{\tilde{\mathbf{w}}\in\tilde{\Theta}} - \langle\mathbf{w}\rangle^2_{\tilde{\mathbf{w}}\in\tilde{\Theta}}\right).$$

We now consider an infinitesimal decrease in the space $\tilde{\Theta}' = \tilde{\Theta} - \Delta$. For a uniform distribution over the space, and ignoring terms in Δ^2, we can write, with a slight abuse of notation,

$$\epsilon_g(\tilde{\Theta}') - \epsilon_g(\tilde{\Theta}) \approx \frac{\Delta}{N\tilde{\Theta}}\left(\left\langle\mathbf{w}^2\right\rangle_{\tilde{\mathbf{w}}\in\tilde{\Theta}} - \langle\mathbf{w}^2\rangle_{\mathbf{w}\in\Delta}\right), \tag{6.2}$$

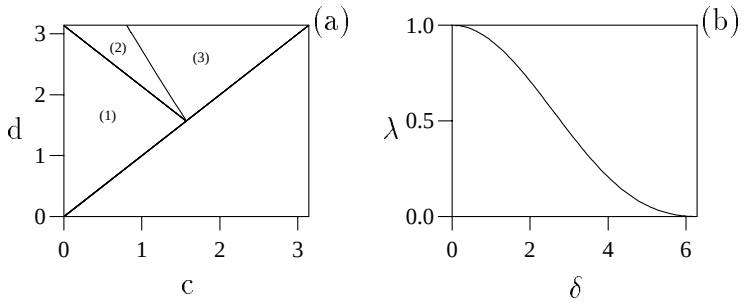

Figure 6.2. The version space is the region on the sphere given by $\Theta = \{(\phi, \theta), \phi \in [a, b], \theta \in [c, d]\}$.(a) in (1) reducivity depends on the region [a,b]. In (2) and (3) reducivity is guaranteed ($\partial \epsilon_g(\Theta)/\partial c < 0$). In (2), as $[a, b]$ shrinks, $\partial \epsilon_g(\Theta)/\partial c$ becomes more negative, and vice versa in region (3). The region $c > d$ is unphysical. (b) The function λ versus $\delta = b - a$.

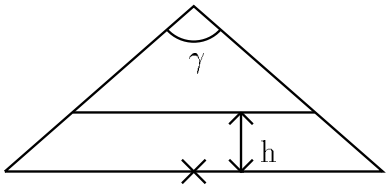

Figure 6.3. Counter example used to show that convexity is not a sufficient condition for reducivity. We take the hypotenuse to have length 2. The cross marks the position of the teacher for the example of reducivity violation for a given teacher.

where Δ and $\tilde{\Theta}$ are the surface contents of Δ and $\tilde{\Theta}$ respectively. In equation(6.2), we have assumed, without loss of generality, that $\langle \mathbf{w} \rangle_{\tilde{\mathbf{w}} \in \tilde{\Theta}} = 0$, *i.e.,* that the origin, $\mathbf{c}$, is taken to be the centroid of $\tilde{\Theta}$. Reducivity holds then for the condition

$$\langle \mathbf{w}^2 \rangle_{\mathbf{w} \in \Delta} > \left\langle \mathbf{w}^2 \right\rangle_{\tilde{\mathbf{w}} \in \tilde{\Theta}} . \tag{6.3}$$

Note that this is a general condition, holding for any dimension. Using this, we can show that convexity (for the linear perceptron at least) is not a sufficient condition for reducivity to hold. In order to do this, we observe that equation(6.3) will not be satisfied for regions, Δ, sufficiently close to the centroid, since the left hand side of equation(6.3) will be small. This observation leads to the following two dimensional counter example. Let the convex region $\tilde{\Theta}$ be the larger triangle as shown on figure(6.3). By explicit calculation, one finds $\epsilon_g(\text{tri})=4/9$ for the marked angle $\gamma = \pi/2$. We now take $\tilde{\Theta}'$, a convex subset of $\tilde{\Theta}$, to be the trapezium as shown, for which, in the limit $h \to 0$, $\epsilon_g(\text{trap})=2/3$. Hence $\epsilon_g(\tilde{\Theta}') > \epsilon_g(\tilde{\Theta})$, demonstrating the insufficiency of convexity as a condition for reducivity.

At this point we refer back to section(6.2.1) and note that we can readily find an example of a *fixed* teacher for which an increase in the students knowledge results in an increase in

$\epsilon_g(\theta^0, \Theta)$. In the above trapezium/triangle example, consider a very flat triangle, for which γ tends to π. We take the teacher to be positioned at the cross marked in figure(6.3), for which, $\epsilon_g(\times, tri) = 1/6$. Taking again, $\tilde{\Theta}'$ to be the infinitely thin trapezium, we have $\epsilon_g(\times, trap) = 1/3$, which is larger than $\epsilon_g(\times, tri)$.

The geometry of the above situation may appear somewhat pathological. Such non-reducive situations can, however, be constructed for essentially any version space Θ. In passing, we mention another example to help clarify the situation.

For a two dimensional ellipse with minor and major axes a and b respectively, one readily finds $\langle \mathbf{w}^2 \rangle_{ellipse} = (a^2+b^2)/4$. We see then that for a circle ($b = a$), all infinitesimal enlargements of the circle are 'expansions' in the sense that they satisfy equation(6.3). Without loss of generality, let $b > a$ such that for an ellipse, we can violate equation(6.3) by choosing the point on the perimeter about which we wish to expand to be close to the centroid ($\langle \mathbf{w}^2 \rangle_\Delta = a^2$) and $b > \sqrt{3}a$. We note that this violation of reducivity occurs for an eccentricity (b/a) that is not much larger than unity. In general, such non-expansive enlargements can occur for the following reason: the centroid represents the best-guess student (within the euclidean approximation); adding space as close as possible to this student increases the weight on the distribution of weight space close to this best-guess, decreasing ϵ_g.

By examining equation(6.2), we note that the greatest decrease in generalisation error is to be found for a region Δ furthest away from the centroid of the set. This is in line with the intuitive notion that we can improve generalisation most by increasing our knowledge about the teacher in those regions that contribute most to the generalisation error. One way to obtain this knowledge is to choose an input x such that the reply from the teacher yields information about the teacher in the desired region; this is the concept of query learning (see *e.g.,* [Sol94b]). However, we have shown here that there is no general realtionship between the volume of the version space (entropy) and the generalistion error, so that query algorithms that reduce the volume of the version space cannot be guaranteed to reduce the generalisation error.

The previous arguments have been aimed at infinitesimal, local alterations to $\tilde{\Theta}$, and we consider briefly an example of global enlargement. We envisage situations in which the boundary of $\tilde{\Theta}$ can be expressed in a spherical coordinate system, $r = r(\phi, \theta, ..)$, which is the case for convex regions. The enlarged version space $\tilde{\Theta}'$ can then be defined by a new boundary, $r' = \lambda(\phi, \theta, ..)r(\phi, \theta, ..)$, for some $\lambda(\phi, \theta, ..) > 1$. Assuming we can bound λ by some extremum values, $\lambda_{min} < \lambda(\phi, \theta, ..) < \lambda_{max}$, it is then a simple matter to form an inequality such that the generalisation error of the larger version space is greater than the generalisation error of the smaller. For an enlargement $\lambda(\phi, \theta, ..)$ which preserves the origin as the centroid of both Θ and $\tilde{\Theta}'$ a two dimensional case it is $\lambda_{min}^2 > \lambda_{max}$, a sufficient, but by no means necessary condition for reducivity.

6.3.3 Sign Constrained Weights

In this section, our motivation is twofold. Firstly, up till now we have considered, for specific examples, low dimensional version spaces; here we calculate the generalisation error for an infinitely large perceptron under a new weight constraint. Secondly, the performance of a perceptron with sign constrained weights is of some interest in itself. The constraint we examine corresponds to predetermining the sign of each weight: $\mathrm{sgn}(w_i) = \rho_i$, where each ρ_i (i=1..N) is pre-set to ± 1. This constraint has been studied previously in the context of pattern storage for the Hopfield network, for which it was found that the sign-constrained capacity was half that

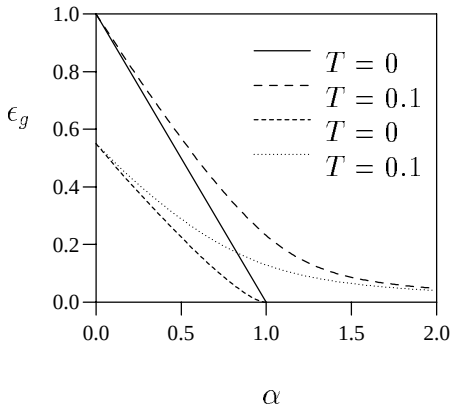

Figure 6.4. Comparison of the generalisation error for the spherical constraint and the spherical sign-constraint. The curves beginning at 1 for $\alpha = 0$ are the spherical constraint; the sign-constraint law curves begin at 1-$\sqrt{2}/\pi$ for α=0.

of the unconstrained case[AWC89].

By writing the output of the perceptron as $y = \sum x_i \mathrm{sgn}(w_i)|w_i|$, where $|..|$ is the modulus, and transforming the inputs according to, $x_i' = \rho_i x_i$, the output can be written $y = \sum x_i'|w_i|$. As the input distribution is Gaussian and hence symmetric, the analysis of the sign-constraint is equivalent to that of constraining the weights to be positive. In addition, we retain the spherical constraint. The method of calculation is that of statistical mechanics, following closely the exposition given in appendix(A). This will enable us to obtain results for any temperature, and without recourse to the euclidean approximation employed is section(3.2). As is required in statistical mechanics calculations, we define the limit of the dimension of the perceptron such that the number of training patterns is proportional to the dimension of the perceptron, *i.e.*, $P = \alpha N$.

A sketch of the calculation is given in appendix(6.5); as the calculation follows so closely that given by [SST92], we refer the reader to that work, and point out only the major differences between our and their analysis. We note however, that constraining the signs of the weights breaks the isotropy of weight space under the usual spherical constraint alone. This will leave the generalisation error as a function of the specific teacher chosen, which we necessarily average over the version space according to (6.1). In an isotropic weight space, such an average is not actually required as the generalisation error is independent of the specific teacher chosen.

For the spherical constraint alone, the dimension of the version space ($T = 0$) reduces linearly with α, resulting in a linear reduction of the generalisation error, $\epsilon_g = 1 - \alpha$, $\alpha \leq 1$. For the sign-constraint, however, boundary effects result in a small deviation from linearity (figure(6.4)). For $T = 0$ and $\alpha \geq 1$, the subspace collapses to a single point for both the sign-constrained and spherical perceptron, and $\epsilon_g = 0$. Non zero T results in an increase in generalisation errors, affecting both the sign-constrained and spherical perceptron similarly, such that for a given $(\alpha, T), \epsilon_g^{sign} < \epsilon_g^{sph}$. For $\alpha = 0$, the perceptron has no information about the teacher other than that imposed by the a priori constraint, and we have $\epsilon_g^{sph} = 1$, and $\epsilon_g^{sign} = 1 - \sqrt{2}/\pi$. The Gibbs learning algorithm itself can thus be regarded to be error

reducing; the extra knowledge contained in the training examples as α increases leads, via the Gibbs learning algorithm, to a reduction in generalisation error.

6.4 Summary

We have examined the effect of constraints on the generalisation error of simple learning systems, concentrating in particular on the linear perceptron. Assuming that both the student and teacher lie in the version space of constraints, we studied what effect increasing the constraint, by decreasing the version space, has on the generalisation error. For a connected one dimensional case, in which we assumed that the error function is simply a monotonically increasing function of the separation between the student and teacher, we showed that decreasing the version space necessarily decreases the generalisation error. This however, is not the case for higher dimensional version spaces, and we presented an explicit example. Furthermore, convexity of the version spaces is not a sufficient condition for the smaller version space to have lower generalisation error. In general it is a non-trivial problem to predict whether reducing the version space will reduce the generalisation error, and each case must be treated explicitly. For sign-constrained weights, we carried out a statistical mechanics calculation for the generalisation error, finding that an increase in constraints over the normal spherical constraint does lead to a reduction in generalisation error. The above analysis concentrated on the situation in which we are able to choose the version space at will. In the situation of learning from examples, after incorporation of a priori knowledge, the version space is subsequently modified by a learning algorithm, to which the concept of reducivity can be applied, opening an area of further research.

6.5 Appendix: Replica method for sign-constrained weights

The calculation for the linear perceptron with sign-constrained weights follows closely that presented in appendix(A) and rather than entering into great detail, we sketch here the main differences between the two calculations.

The free energy is separated into two terms, $F = G_0 - \alpha G_r$, where only the term G_0 is affected by the constraints upon the weights. Hence we need calculate only the term G_0 for the new weight distribution. As the new weight distribution arising from the sign constraint retains connectedness, we envisage no problems with the replica symmetry ansatz, and expect the results to be exact. Note that throughout, we use the same notation for the order parameters as those in [SST92], namely that q is the normalised overlap between two replicas, R is the overlap between the student and the teacher. $\hat{q}$ and $\hat{R}$ are conjugate order parameters arising from the definition of the order parameters q and R. We write G_0 as,

$$G_0 = -\frac{1}{2}(1-q)\hat{q} - R\hat{R} + \frac{1}{N} I(\hat{q}, \hat{R}, \mathbf{w}^0),$$

where

$$I = \int_{-\infty}^{\infty} D\mathbf{z} \ln \int_{-\infty}^{\infty} d\mu(\mathbf{w}) \exp[\mathbf{w} \cdot (\mathbf{z}\sqrt{\hat{q}} + \mathbf{w}^0 \hat{R})].$$

$D\mathbf{z}$ is the N dimensional gaussian measure, $(2\pi)^{-N/2}\exp(-\mathbf{z}\cdot\mathbf{z}/2)d\mathbf{z}$. The weight vector distribution for the sign-constraint is given by

$$P(\mathbf{w}) = \frac{2^N}{V}\delta(\mathbf{w}\cdot\mathbf{w} - N)\theta(\mathbf{w})d(\mathbf{w}),$$

and the corresponding measure is $d\mu(\mathbf{w}) = P(\mathbf{w})d(\mathbf{w})$, where V is the surface content of an N-sphere, and $\theta(\cdot)$ is the theta function. Introducing the integral representation for the delta function (which gives rise to the parameter λ) and performing the saddle point approximation, we obtain,

$$I = const. + N\left(\lambda + \frac{1}{4\lambda}\left(\hat{q} + \hat{R}^2\right) + \frac{1}{2}\ln\left(\frac{\pi}{4\lambda}\right)\right) + \sum_{i=1}^{N}\int_{-\infty}^{\infty} Dz \ln \mathrm{erfc}(-u_i),$$

where

$$u_i = \frac{z_i\sqrt{\hat{q}} + w_i^0\hat{R}}{\sqrt{4\lambda}}.$$

By comparison with the results for the spherical perceptron, we observe that we can write

$$G_0^{sign} = G_0^{sph} + \sum_{i=1}^{N} J_i\left(\lambda, \hat{q}, \hat{R}, \mathbf{w}^0\right),$$

where

$$J_i = \frac{1}{2\pi}\sqrt{\frac{4\lambda}{\hat{q}}}\int_{-\infty}^{\infty} du \exp\left(-\frac{1}{2\hat{q}}\left(4\lambda u^2 - 4\sqrt{\lambda}\hat{R}w_i^0 u + (w_i^0)^2\right)\right)\ln\mathrm{erfc}(-u),$$

and G_0^{sph} is the contribution to the free energy given by the normal spherical constraint, given in [SST92].

There remains an explicit dependence on the teacher weight $\mathbf{w}^0$ and we thus average G_0 over possible teachers, having the same measure as the students. This results finally in the expression for the contribution to the free energy,

$$G_0^{sign} = G_0^{sph} + \sqrt{\frac{\phi^2}{2\pi}}\int_{-\infty}^{\infty} du \exp\left(-\phi^2 u^2\right)\mathrm{erfc}\left(\frac{u\phi\hat{R}}{\sqrt{\hat{q}}}\right)\ln\mathrm{erfc}(u),$$

where

$$\phi = \sqrt{\frac{\lambda + \hat{q}}{\hat{q} + \hat{R}^2}}.$$

For completeness, we state the further results necessary to find the free energy, namely

$$G_0^{sph} = \lambda - \frac{1}{2}(1-q)\hat{q} - R\hat{R}\frac{1}{2}q\hat{q} - R\hat{R} - \frac{1}{2}\ln(4\lambda) + \frac{\hat{R}^2 + \hat{q}}{4\lambda},$$

and

$$G_r = \frac{1}{2}\ln[1 + \beta(1-q)] + \frac{\beta(q - 2R + 1)}{2(1 + \beta(1-q))}.$$

Now that the free energy has been found, the order parameters are set to those values that actually extremise the free energy. The generalisation error is then found from the relation, $\epsilon_g = 1 - R$. Unfortunately, in all but the simplest of limiting cases, the solution to this extremisation problem needs to be solved numerically.

Chapter 7

Conclusion

In this thesis we have conducted an investigation of finite size corrections to infinite system size calculations generally employed by physicists in the analysis of neural networks.

In chapters (2) and (3) we studied one of the simplest possible neural network models, the linear perceptron. In chapter(2) we concentrated on exact analytical results for the variance of the test error for a spherically constrained student learning a spherically constrained teacher when there is no noise on the examples. Intuitively, we expect that the variance of the errors scales like the inverse of the system size, which we found to be true. As an application of these results, we showed how one can address the issue of an optimal test set size, in which one is concerned with minimising the test error, yet wishes that the resulting error remains close to the average test error. For a large system size N, we found that the optimal test set size scales with $N^{2/3}$. We also conducted an in depth analysis of the performance of different cross-validation schemes when they are used to estimate the generalisation error. This is one of few analyses that are able to tackle cross-validation without recourse to the limit of a great deal of data. In particular, three different schemes, corresponding to different overlaps of the test sets for two cross-validation students were analysed. These correspond to choosing the cross-validation sets at random, to minimise their mutual overlap, and to minimise in a blockwise fashion their mutual overlap. The optimal scheme, in the sense that the variance of the cross-validation estimate of the generalisation error is the lowest, is given by the scheme which minimises the mutual test set overlaps. Comparing the performance of the other two schemes against the optimal, we found that there is less than a 25% relative difference between the optimal scheme and the random scheme. This figure drops to less than 5% when the block scheme is employed. Hence, in cases where the optimal scheme might seem too time consuming to apply, the easily implementable block scheme is a good compromise.

In chapter(3) we extended our analysis of the linear perceptron to the inclusion of noise on the examples. In order to suppress the learning of the noise, we included a weight decay in the learning procedure and found that there are different phases for the scaling of the optimal test set size with the system dimension. For small noise levels, or large weight decay, we found that the scaling is again of a 2/3 power type. However, for large levels of noise, the test set size needs to be much larger, and we found a linear scaling behaviour. With the introduction of the weight decay parameter, we were able to address the issue of how to best use cross-validation in order to discriminate between two different models. We found that the optimal scheme in this sense is to choose a number of divisions that scales like $1 + \mathcal{O}(\alpha^{-1})$, that is, that the test set size should approach the size of the total amount of data in the data set as the data set grows.

This is a similar conclusion to that reached by Shao who considered linear model selection using cross-validation based on the statistical requirement of consistency, although Shao advocates a slightly different scheme, namely that the test set size should scale like $1 + \mathcal{O}\left(\alpha^{-1/4}\right)$.

As a representative of a non-linear rule, we analysed the binary perceptron in chapter(4) in a similar manner to that for the linear perceptron. We found that the optimal test set size scales like $N^{2/3}$, with a linear prefactor for both noisy and clean examples. In terms of the cross-validation error performance, we found that there is a much greater similarity between the performance of the different CV schemes for the binary perceptron than for the linear perceptron. We were able to make some connections between our work and the PAC worst-case approach by calculating the probability that such worst-cases occur. Even for rather small system sizes, we found that this probability is remarkably small.

In chapter(5) we analysed a different learning strategy, namely that of on-line learning, in which the student weights are updated after presentation of a single example from a stream of input examples. This work complements the recent significant advances in the study of on-line learning in cases involving multiple hidden units. This analysis confirms that, provided the initial conditions are not too symmetric, the thermodynamic learning curves calculated by the average case theory are representative. Finite size effects are largest around the symmetry breaking point, when the students hidden units begin to specialise on those of the teacher. By stimulating asymmetry between the student's hidden units, we showed that a considerable reduction in both finite size effects and generalisation error can be achieved. We conjectured that such symmetry breaking constraints can be employed to potentially great benefit in the practical field of training neural networks.

With the motivation that extra constraints may reduce the generalisation error, we investigated in chapter(6) to what extent it is necessarily true that increasing the knowledge we have about the teacher reduces the generalisation error. Perhaps, counter-intuitively, we showed that increasing this knowledge can increase the generalisation error, even for such cases as a metric generalisation measure and convex constraints. For the linear perceptron, we showed how reducing the generalisation error is related to increasing the knowledge far from the centroid of the posterior student distribution. For the sign-constrained weight case, we found that there is a considerable reduction in the generalisation error.

7.1 Outlook on future research

There are many questions left unanswered by this thesis. The most glaring is that we have restricted our batch learning analysis to realisable cases only. In principle, having found a general statistical mechanics framework in which finite size variances can be calculated, it should be straightforward to extend these cases to unrealisable rules. This is particularly interesting for the cross-validation analysis as there has been relatively little work carried out on non-asymptotic data regimes. There is much work to be done on extending the preliminary results on model selection, and there is potentially a great deal that can be said about how to select cross-validation schemes from this type of analysis. Again, in principle, there is no difficulty in extending this analysis to multi-layer structures, and in particular, the tree committee machine, for which a generalisation error calculation already exists[SH92]. An exciting prospect is also to extend the connection between the finite size results for the binary perceptron to the PAC analysis. Indeed, if one were able to calculate the variance of the errors for a class of input

distributions, one would be able to make a confidence bound connection to the worst case results as formulated by the VC theory. It may be that this is a realistic way to bridge the gap between the average and worst case analyses.

In terms of on-line learning, there is more work being carried out on unrealisable rules and more general architectures. This work is important as the on-line learning approach represents one way to avoid some of the difficulties inherent in the statistical mechanics analysis of batch learning.

We hope therefore that we have shown how powerful tools from statistical mechanics can be employed to calculate effects also for finite system sizes. These results, in addition to being of interest in themselves, may lead to a much better understanding of the connection between average and worst case analyses.

Appendix A

Statistical Mechanics Formalism

A.1 Introduction

In this appendix we explain in some detail calculational tools that can be employed in calculating both average test errors and their variances. For the case of the linear perceptron, we have seen that geometrical arguments can lead in a straightforward manner to the evaluation of the averages for the average test error and the variance of the test error (chapter(2)). For more general network architectures, however, such calculational simplifications may not exist, and we therefore describe a more general method which has its roots in the theory of statistical physics. Statistical mechanics provides a mechanism for bridging the gap between a microscopic description of a process and a macroscopic picture. For example, from the microscopic description of a molecule as a hard sphere, and a description of the dynamical interaction of such hard spheres, statistical mechanics is able to say something about macroscopic properties of the system, such as the pressure. In order to do so, however, the number of particles (system dimension) is taken to be arbitrarily large (the *thermodynamic limit*). Such a restriction need not necessarily result in unrepresentative results compared to those for finite size systems [Sol94a]. In order to say something about the macroscopic picture, an average over the possible configurations is taken, each of which occurs with some probability which is physically related to the energy of the system. For many physical systems, the resulting average behaviour is equivalent to the typical behaviour of the system - this is called *self-averaging*[BY86]. Much of the work in making the connection between statistical mechanics and neural networks was pioneered by Amit *et al.*[AGS85a, AGS85b] and Gardner [Gar87, Gar88, GGY89].

A.1.1 The Partition Function, Z, and the Free Energy

The starting point of the statistical mechanics treatment is the description of the probability of a particular microstate. For the case of neural networks, stochastic gradient descent results in a Langevin process and the equilibrium distribution of students is given by the *Gibbs* distribution,

$$P(\mathbf{w}|\mathcal{P}) = \frac{1}{Z} P^{pri}(\mathbf{w}) \exp(-\beta E),$$

where E is the *training error*, and $\beta = 1/T$ is the training temperature. The *partition function* Z is a constant such that the probability distribution is correctly normalised. The student

prior $P^{pri}(\mathbf{w})$ expresses additional *a priori* constraints on the student, such as the spherical constraint. The partition function is given by the integral,

$$Z(\beta) = \int d\mathbf{w} P^{pri}(\mathbf{w}) \exp\left(-\beta \sum_1^P \epsilon\left(\mathbf{w}, \mathbf{x}\right)\right),$$

where $\epsilon\left(\mathbf{w}, \mathbf{x}\right)$ is the error that a student makes on an example $\mathbf{x}$.

Auxiliary Field Methods

Although it is often straightforward to express the generalisation error in terms of the average overlap of student and teacher vectors, we demonstrate an alternative method which involves augmenting the partition function with an auxiliary field (this approach which will turn out useful later on), and we define,

$$Z(\beta, \gamma) = \int d\mathbf{w} \exp\left(-\beta \sum_1^P \epsilon\left(\mathbf{w}, \mathbf{x}\right) - \gamma \sum_1^M \epsilon\left(\mathbf{w}, \mathbf{x}\right)\right), \tag{A.1}$$

so that we can write the thermal average of the test error,

$$\langle \epsilon_{test} \rangle_{\mathcal{W}} = -\frac{1}{M} \lim_{\gamma \to 0} \frac{\partial}{\partial \gamma} \ln Z\left(\beta, \gamma\right) \tag{A.2}$$

In order to calculate the generalisation error, we need to further average (A.2) over the dataset inputs. By interchanging the derivative and the average, we can write the generalisation error as a function of the (dataset) averaged augmented log partition function,

$$\epsilon_g = -\frac{1}{M} \lim_{\gamma \to 0} \left\langle \frac{\partial}{\partial \gamma} \ln Z\left(\beta, \gamma\right) \right\rangle_{\mathcal{L}} = -\frac{1}{M} \lim_{\gamma \to 0} \frac{\partial}{\partial \gamma} \left\langle \ln Z\left(\beta, \gamma\right) \right\rangle_{\mathcal{L}}. \tag{A.3}$$

Thus, by interchanging the order of averaging and differentiation, the statistical properties of the system can be found from the *free energy*, $F(\beta, \gamma) = \left\langle \ln Z\left(\beta, \gamma\right) \right\rangle_{\mathcal{L}}$[1]. The free energy is similar to a generating function in statistics, and we can generate higher (thermal) moments from taking higher derivatives. Calculating the free energy by performing the dataset average (known as the "quenched" average in statistical mechanics) lies at the heart of the statistical mechanics formalism, and there have been many attempts to facilitate the technical difficulties inherent in the average of the logarithm of a function[BY86]. Depending on the complexity of the student/teacher architectures, it may well be possible to carry out the averages over the Gibbs distribution, and over the data sets (the *quenched variables*) by fairly direct methods. The method that we briefly review in the following section, however, is a rather general method that can, in principle, be applied to a large class of difficult calculations.

[1]The standard definition of the free energy is given by $-F/\beta$

A.1.2 Replica Methods - a brief introduction

The essential feature of the replica method comes from rewriting the logarithm in a more convenient way. For small x we can expand the logarithm as,

$$\ln(1+x) = x + \mathcal{O}\left(x^2\right).$$

For $n \ll 1$, we set $Z^n = 1 + x$, giving

$$\ln Z^n = Z^n - 1 + \mathcal{O}\left((1-Z^n)^2\right)$$

Using L'Hopital's rule, one then attains,

$$\langle \ln Z \rangle = \lim_{n \longrightarrow 0} \frac{\partial \langle Z^n \rangle}{\partial n}.$$

This means that the difficulty of averaging the logarithm of the partition function has been transferred to averaging powers of the partition function. We write Z^n as the product of n *replicas* of Z, and carry out the quenched average over this product, before performing an analytic continuation of a vanishingly small number of replicas ($n \rightarrow 0$). The techniques for carrying out such calculations are by now standard and we refer the reader to the works[WRB93, SST92] for more detailed discussions. The introduction of the different replicas artificially introduces extra degrees of freedom to the calculation which typically manifest themselves as *order parameters*, being overlaps between weight vectors from different replica solution spaces. In order to proceed with such calculations, one therefore needs to determine these extra degrees of freedom. The simplest possible assignment is that of replica symmetry, which assumes that the solutions represented by the different replica systems are indistinguishable. Intuitively, this corresponds to the case in which the weight space of solutions remains connected such that the replicated weight space overlaps are identical. Such an assumption, however, is by no means guaranteed to yield correct results. The validity of the results is commonly checked by calculating the entropy of the resulting calculation, which is simply related to the free energy. For the linear perceptron, however, the replica symmetry ansatz is exact. For discrete systems, the entropy must be positive, and any violation of this will necessitate a breaking of replica symmetry. In the case of the binary perceptron, we therefore restrict our analysis to the region of validity for the replica symmetry ansatz. The second essential ingredient to the statistical mechanics type calculations is the assumption of an infinite input dimension (*thermodynamic limit*), which enables averages to be carried with recourse to the saddle point method[Arf85]. The assumption of self-averaging assures that the saddle point approximation to the integrals required becomes exact in the thermodynamic limit. The reason that the saddle point approximation can be taken is that the training energy which appears in the Gibbs distribution is *extensive*(scales with N).

A.1.3 The Thermal Variance

We have seen that statistical mechanics can be used to find the average of the test error (the generalisation error), which is an order $\mathcal{O}(1)$ quantity, but how can it be employed to find the variance? The training error is extensive (scales with N), which means that the variance of the Gibbs distribution is order $\mathcal{O}(N^{-1})$, giving a generalisation error variance also of order

$\mathcal{O}(N^{-1})$. Thus, in the thermodynamic limit, the test error variance is zero! There is, however, a way around this difficulty.

In the previous section we noted that in order for the saddle point method to work, we need to introduce an extensive quantity in the exponent of the Gibbs distribution. Hence, in applying the auxiliary field method, we use an extensive test error, rescaling the variance at the end of the calculation. (In fact, we did exactly this in demonstrating how to calculate the generalisation error using the partition function, (A.1)). This will be only an approximation to the variance as the assumption that the test error variance scales exactly with $1/N$ means that higher order terms in $1/N$ will not be captured by this method. Using the single replica method only, one readily verifies that, from the definition of the partition function (A.1), the second moment of the thermally averaged test error can be found from,

$$\frac{1}{M^2}\lim_{\gamma\to 0}\frac{\partial^2}{\partial\gamma^2}\left\langle \ln Z\left(\beta,\gamma\right)\right\rangle = \left\langle \left(\epsilon_{test}\right)^2 - \left\langle \epsilon_{test}\right\rangle_{\mathcal{W}}^2\right\rangle_{\mathcal{E}} = \mathrm{var}\left(\epsilon_{test}:\mathcal{W}\right), \tag{A.4}$$

which we term the *thermal variance*. It is the expected variance over the posterior distribution of students trained and tested on a random data set. For zero temperature, we could also term this the *version space* variance. This variance, however, does not capture all the fluctuations in the test error caused by the stochastic algorithm and the random data sets. In order to calculate the full variance, $\mathrm{var}\left(\epsilon_{test}:\mathcal{E}\right)$, we need to extend the replica formalism.

A.2 Double Replica Method

In this section, we extend the replica formalism to what we shall term the *double replica* formalism in order to evaluate averages that depend upon two types of quenched averages. The motivation for the double replica method comes from the wish to calculate averages such as,

$$\mathrm{var}\left(\epsilon_{test}:\mathcal{E}\right) = \left\langle \left(\epsilon_{test}\right)^2\right\rangle_{\mathcal{W},\mathcal{L}} - \epsilon_g^2 \tag{A.5}$$

The difference between this and the thermal variance (A.4) is

$$\left\langle \left\langle \epsilon_{test}\right\rangle_{\mathcal{W}}^2\right\rangle_{\mathcal{L}} - \epsilon_g^2 \tag{A.6}$$

By introducing two different auxiliary fields, we can write,

$$\left\langle \epsilon_{test}\right\rangle_{\mathcal{W}} = -\lim_{\gamma_1\to 0}\frac{\partial}{\partial\gamma_1}\ln Z\left(\beta,\gamma_1\right)$$

$$\left\langle \epsilon_{test}\right\rangle_{\mathcal{W}} = -\lim_{\gamma_2\to 0}\frac{\partial}{\partial\gamma_2}\ln Z\left(\beta,\gamma_2\right)$$

Furthermore, using the identity,

$$\lim_{m,n\to 0}\frac{\partial^2 \ln\left\langle a^m b^n\right\rangle}{\partial m\partial n} = \left\langle \ln a\ln b\right\rangle - \left\langle \ln a\right\rangle\left\langle \ln b\right\rangle \tag{A.7}$$

we can write,

$$\left\langle \left\langle \epsilon_{test}\right\rangle_{\mathcal{W}}^2\right\rangle_{\mathcal{L}} - \epsilon_g^2 = \frac{1}{M^2}\lim_{\gamma_1,\gamma_2,m,n\to 0}\frac{\partial^2}{\partial\gamma_1\partial\gamma_2}\frac{\partial^2}{\partial m\partial n}\ln\left\langle Z^m\left(\beta,\gamma_1\right)Z^n\left(\gamma_2,\beta\right)\right\rangle \tag{A.8}$$

Hence the full variance $\mathrm{var}\left(\epsilon_{test}:\mathcal{E}\right)$ can be obtained simply adding equations (A.4) and (A.8). Thus, in theory, one can apply a combination of the single replica and double replica method to obtain most of the kinds of (co)variances that one wishes.

A.3 Double Replica Method for general Perceptron architecture

Having demonstrated, in principle, how one can calculate variances and covariances by a combination of the double and single replica methods, we present in more detail how such a double replica calculation proceeds. We outline the derivation of the double replica method for a general perceptron architecture, which has the single replica method embedded within it.
As we shall primarily be interested in using the double replica method to calculate cross-validation type covariances, in which the training data of one perceptron is used as the test data of the other, we derive the double replica results from the wish to calculate,

$$\left\langle Z^m\left(\beta_1,\gamma_1\right) Z^n\left(\beta_2,\gamma_2\right)\right\rangle_{\mathcal{L}} = \int \prod_{k=1}^{P+M} \left(d\mu(\mathbf{x}^k)\right) \int \prod_{\rho=1}^{m} \left(d\mu(\mathbf{w}_1^\rho)\right) \int \prod_{\sigma=1}^{n} \left(d\mu(\mathbf{w}_2^\sigma)\right)$$

$$\exp\left\{-\beta_1 \sum_{\rho=1}^{m}\sum_{\mathbf{x}\in\mathcal{P}_1} \epsilon\left(\mathbf{w}_1^\rho,\mathbf{x}\right) - \gamma_1 \sum_{\rho=1}^{m}\sum_{\mathbf{x}\in\mathcal{M}_1} \epsilon\left(\mathbf{w}_1^\rho,\mathbf{x}\right) - \beta_2 \sum_{\sigma=1}^{n}\sum_{\mathbf{x}\in\mathcal{P}_2} \epsilon\left(\mathbf{w}_1^\rho,\mathbf{x}\right) - \gamma_2 \sum_{\sigma=1}^{n}\sum_{\mathbf{x}\in\mathcal{M}_2} \epsilon\left(\mathbf{w}_2^\sigma,\mathbf{x}\right)\right\}$$

For simplicity, let us assume that the size of the training set for the two systems are equal, and similarly, the size of the two test sets are equal. Then,

$$\left\langle Z^m\left(\beta_1,\gamma_1\right) Z^n\left(\beta_2,\gamma_2\right)\right\rangle_{\mathcal{L}} = \int \prod_{\rho=1}^{m} \left(d\mu(\mathbf{w}_1^\rho)\right) \int \prod_{\sigma=1}^{n} \left(d\mu(\mathbf{w}_2^\sigma)\right)$$

$$\left\{\int d\mu(\mathbf{x}) \exp\left\{-\beta_1 \sum_{\rho=1}^{m} \epsilon\left(\mathbf{w}_1^\rho,\mathbf{x}\right) - \beta_2 \sum_{\sigma=1}^{n} \epsilon\left(\mathbf{w}_1^\rho,\mathbf{x}\right)\right\}\right\}^P$$

$$\left\{\int d\mu(\mathbf{x}) \exp\left\{-\gamma_1 \sum_{\rho=1}^{m} \epsilon\left(\mathbf{w}_1^\rho,\mathbf{x}\right) - \gamma_2 \sum_{\sigma=1}^{n} \epsilon\left(\mathbf{w}_2^\sigma,\mathbf{x}\right)\right\}\right\}^M$$

which we write as

$$\left\langle Z^m\left(\beta_1,\gamma_1\right) Z^n\left(\beta_2,\gamma_2\right)\right\rangle_{\mathcal{L}} = \int \prod_{\rho=1}^{m} \left(d\mu(\mathbf{w}_1^\rho)\right) \int \prod_{\sigma=1}^{n} \left(d\mu(\mathbf{w}_2^\sigma)\right) \exp\left\{\left(P\mathcal{G}\left(\beta_1;\beta_2\right) + M\mathcal{G}\left(\gamma_1;\gamma_2\right)\right)\right\} \quad \text{(A.9)}$$

where,

$$\mathcal{G}\left(\beta_1;\beta_2\right) = -\ln \int d\mu(\mathbf{x}) \exp\left\{-\beta_1 \sum_{\rho=1}^{m} \epsilon\left(\mathbf{w}_1^\rho,\mathbf{x}\right) - \beta_2 \sum_{\sigma=1}^{n} \epsilon\left(\mathbf{w}_2^\sigma,\mathbf{x}\right)\right\}$$

The evaluation of this replicated Hamiltonian follows directly the standard method presented in [SST92], and we refer the reader therefore to that work for further details. The input examples $\mathbf{x}$ appear always through the activation for some student, $\mathbf{x}\cdot\mathbf{w}$. This means that by substituting for the activations, we can write the integral over the inputs as an integral over activations. Using the gaussian measure shorthand,

$$D\mathbf{x} = \frac{1}{(2\pi)^{N/2}} \exp\left(-\frac{1}{2}\mathbf{x}\cdot\mathbf{x}\right), \quad \text{(A.10)}$$

we have

$$\exp\left\{\mathcal{G}\left(\beta_1;\beta_2\right)\right\} = \int d\mathbf{x_1} d\mathbf{x_2} dy \exp\left\{-\frac{\beta_1}{2}\sum_{\rho=1}^{m}\left[g(x_1^\rho)-g(y)\right]^2 - \frac{\beta_2}{2}\sum_{\sigma=1}^{n}\left[g(x_2^\sigma)-g(y)\right]^2\right\}$$

$$\int D\mathbf{x}\prod_{\rho=1}^{m}\left\{\delta\left(x_1^\rho - \frac{1}{\sqrt{N}}\mathbf{w_1^\rho}\cdot\mathbf{x}\right)\right\}\prod_{\sigma=1}^{n}\left\{\delta\left(x_2^\sigma - \frac{1}{\sqrt{N}}\mathbf{w_2^\sigma}\cdot\mathbf{x}\right)\right\}\delta\left(y-\frac{1}{\sqrt{N}}\mathbf{w}^0\cdot\mathbf{x}\right) \tag{A.11}$$

The integral over the input examples can be carried out by introducing the integral representation of the delta function,

$$\delta(x) = \frac{1}{2\pi}\int_{-\infty}^{\infty} d\hat{x}\exp ix\hat{x} \tag{A.12}$$

and integrating over the gaussian measure. We find that the weight vectors appear only in the form of their mutual overlaps, namely

$$Q_i^{\tau,\tau'} = \frac{1}{N}\mathbf{w}_i^\tau\cdot\mathbf{w}_i^{\tau'} i = 1,2$$

$$Q_{12}^{\tau,\tau'} = \frac{1}{N}\mathbf{w}_1^\tau\cdot\mathbf{w}_2^{\tau'}$$

$$R_i^\tau = \frac{1}{N}\mathbf{w}_i^\tau\cdot\mathbf{w}^0 \qquad i = 1,2$$

$Q_i^{\tau,\tau'}$ is the overlap between student weights from two replicas of the same perceptron. $Q_{12}^{\tau,\tau'}$is the overlap between student weights from two replicas of different perceptrons. $R_i^\tau$is the overlap between the the replicated student weight and teacher weight. The integral over the weight space to find the free energy can then be transformed to an integral over the overlap parameters. The great simplification of the replica method then comes about through imposing specific forms for these overlap (order) parameters. The simplest possible assumption for the form of overlap parameters is called the Replica Symmetric (RS) ansatz. This is the assumption that the overlap between weight vectors from *different* replica systems is independent of the replica system. Bearing in mind the spherical constraint, this means that the RS ansatz takes the form,

$$Q_i^{\tau,\tau'} = \delta_{\tau,\tau'} + (1-\delta_{\tau,\tau'})q_i$$

$$R_i^\tau = R_i$$

$$Q_{12}^{\tau,\tau'} = q_{12} \tag{A.13}$$

The calculation then proceeds by substituting in the RS ansatz to make the replicated Hamiltonian a function of the order parameters. Note that by using the identity,

$$\int_{-\infty}^{\infty} Dt\exp\left(\sqrt{2}xt\right) = \exp\left(x^2\right) \tag{A.14}$$

where Dt is a zero mean, unit variance gaussian measure, the quantities in exponentials that are squared can be linearised by the introduction of an auxiliary variable, t. After some straightforward algebra, one obtains,

$$\exp\left\{\mathcal{G}\left(\beta_1;\beta_2\right)\right\} = \int DyDt_1Dt_2h^m(\beta_1)k^n(\beta_2) \tag{A.15}$$

where

$$h(\beta) = \int Dx \exp\frac{-\beta}{2}\left\{g\left(x\sqrt{1-q_1}+R_1y-t_1\sqrt{q_1-R_1^2}\right)-g(y)\right\}^2 \tag{A.16}$$

$$\begin{aligned} k(\beta) &= \int Dx \exp\frac{-\beta}{2}\left\{g\left(x\sqrt{1-q_2}+R_2y-t_1\frac{(q_{12}-R_1R_2)}{\sqrt{q_1-R_1^2}}\right.\right. \\ &- \left.\left. t_2\sqrt{q_2-R_2^2-\frac{(q_{12}-R_1R_2)^2}{q_1-R_1^2}}\right)-g(y)\right\}^2 \end{aligned} \tag{A.17}$$

We mentioned earlier that the single replica method is embedded within the double replica method. To retrieve the single replica results, we set $n = 0$ throughout the derivation of the double replica method, essentially 'turning off' one of the replica systems. We see therefore, that the corresponding single replica result for the replicated Hamiltonian would simply contain an integral over the function $h(\beta_1)$. A more intuitive way of expressing (A.15) comes from realising that the functions h and k are related to the single replica function G_r

$$\exp\left\{G_r\right\} = \int DyDt\ h^n(\beta,t,q,R) \tag{A.18}$$

This means that one can write,

$$\exp\left\{\mathcal{G}\left(\beta_1;\beta_2\right)\right\} = \int DyD(t_1,t_2)h_1^mh_2^n \tag{A.19}$$

where

$$h_1 = h\left(\beta_1,t_1,q_1,R_1\right), \qquad h_2 = h\left(\beta_2,t_2,q_2,R_2\right) \tag{A.20}$$

and the measure $D(t_1,t_2)$ expresses the coupling between the two replica systems,

$$D(t_1,t_2) = dt_1dt_2\frac{1}{2\pi}\frac{1}{\det\left(\mathbf{A}\right)}\exp\left(-\frac{1}{2}\mathbf{t^T}\mathbf{A}^{-1}\mathbf{t}\right) \tag{A.21}$$

where $\mathbf{t^T} = (t_1,t_2)$, and the covariance matrix $\mathbf{A}$ is given by,

$$\mathbf{A} = \begin{bmatrix} 1 & \frac{q_{12}-R1R_2}{\sqrt{q_1-R_1^2}\sqrt{q_2-R_2^2}} \\ \frac{q_{12}-R1R_2}{\sqrt{q_1-R_1^2}\sqrt{q_2-R_2^2}} & 1 \end{bmatrix} \tag{A.22}$$

Taking the derivatives with respect to the replica numbers m,n, one obtains, in the limit $m,n\to 0$,

$$G(\beta_1,\beta_2) = \langle \ln h_1 \ln h_2\rangle - \langle \ln h_1\rangle\langle \ln h_2\rangle \tag{A.23}$$

where $\langle..\rangle$ represents an average over the measure $DyD(t_1,t_2)$. From (A.8), the variance is given by taking also the limit of the two auxiliary field terms going to zero, *i.e.,* $\gamma_1,\gamma_2\to 0$. In this limit, $q_1,q_2,q_{12}\to q$, $R_1,R_2\to R$, where q and R are the single replica order parameters in the absence of an auxiliary field (*i.e.,* they take their standard single replica values). Therefore, when we take the derivatives with respect to γ_1 and γ_2, we use the fact that

$$\frac{\partial^2}{\partial\gamma_1\partial\gamma_2}\langle\ln h_1\rangle\langle\ln h_2\rangle=\left\{\frac{\partial}{\partial\gamma_1}\langle\ln h_1\rangle\right\}^2 \tag{A.24}$$

This term is straightforward to calculate - it is simply the square of the derivative of the single replica G_r with respect to the auxiliary field, evaluated in the limit of a zero auxiliary field.

A.3.1 Double Replica Entropic term

We have seen that the student and teacher vectors occur in the replica calculation only through their respective overlap parameters. This being the case, we can transform the integral over the weight vectors to an integral over the overlap parameters by introducing the definition of the overlap parameters through delta functions.

The entropic term G_0 represents the weight space constraints for the system, and is independent of the functional form of the transfer function. The method of calculation parallels that in [SST92], and we derive briefly below the final form of G_0. We write, symbolically, the double replicated free energy,

$$\langle Z_1^m Z_2^n\rangle=\int\left\{\prod_{i,j,\rho,\sigma}dQ_{ij}^{\rho,\sigma}\delta\left(Q_{ij}^{\rho,\sigma}-\frac{1}{N}\mathbf{w}_i^\rho\cdot\mathbf{w}_j^\sigma\right)\right\}\exp\left\{-N\left(\alpha G(\beta_1,\beta_2)+\mu G(\gamma_1,\gamma_2)\right)\right\} \tag{A.25}$$

where $Q_{ij}^{\rho,\sigma}$ stands for a general overlap parameter, and $\delta\left(Q_{ij}^{\rho,\sigma}-\frac{1}{N}\mathbf{w}_i^\rho\cdot\mathbf{w}_j^\sigma\right)$ expresses the definition of the overlap parameter in terms of the overlap of two weight vectors. The integral is over all overlap parameters. As usual, one represents the delta functions in integral representations, which introduce conjugate variables $\hat{Q}_{ij}^{\rho,\sigma}$, each of which are integrated over the imaginary axis. Again, symbolically, we write,

$$G_0=-\sum_{i,j,\rho,\sigma}\hat{Q}_{ij}^{\rho,\sigma}Q_{ij}^{\rho,\sigma}+\frac{1}{N}\ln\int\prod_\rho\left\{d_\mu(\mathbf{w}_1^\rho)\right\}\prod_\sigma\left\{d_\mu(\mathbf{w}_2^\sigma)\right\}\exp\left\{\sum_{i,j,\rho,\sigma}\hat{Q}_{ij}^{\rho,\sigma}\mathbf{w}_i^\rho\cdot\mathbf{w}_j^\sigma\right\} \tag{A.26}$$

Applying replica symmetry, and removing the squares of the weight vectors in the exponential by introducing auxiliary variables, z, we find,

$$G_0=-(m^2-m)\hat{q}_1q_1-m\hat{R}_1R_1-(n^2-n)\hat{q}_1q_2-n\hat{R}_2R_2-mn\hat{q}_{12}q_{12}+\frac{1}{N}\ln\langle f_1^m f_2^n\rangle \tag{A.27}$$

where

$$\langle..\rangle=\int D(\mathbf{z_1},\mathbf{z_2}).. \tag{A.28}$$

$D(\mathbf{z_1},\mathbf{z_2})$ is the same as for (A.21), but with the matrix $\mathbf{A}$ replaced by $N\mathbf{A}$. As before, f is simply the corresponding term from the single replica formalism,

$$f(\mathbf{z})=\int d_\mu(\mathbf{w})\exp\left\{(1-\hat{q})\mathbf{w}\cdot\mathbf{w}+\mathbf{w}\cdot\left(\hat{R}\mathbf{w}^0+\sqrt{q-R^2}\mathbf{z}\right)\right\} \tag{A.29}$$

A.3.2 Determining the order parameters

At this stage a little care is needed. The double replica entropic term is given by the limit of differentiating G_0 with respect to m and n in the limit $m, n \to 0$. If we do this, we see that the only terms that will remain for the double replica entropic term will be,

$$G_0 = -\hat{q}_{12} q_{12} + \frac{1}{N} \left\{ \langle \ln f_1 \ln f_2 \rangle - \langle \ln f_1 \rangle \langle \ln f_2 \rangle \right\} \tag{A.30}$$

Thus, from the double replicated free energy alone, we can only determine the saddle point equations for the order parameters which express the interaction between the two replica systems. The issue is then how to determine the saddle point equations for the other order parameters. Intuitively, these must reduce to the saddle point equations for the single replica system. One can demonstrate that this is indeed the case by examining (A.27) more carefully.
Before we take the double replica limit, the terms in G_0 of order m^2, n^2, and, more specifically mn, are an order smaller than the terms in m and n alone. At this point, therefore, the double replica free energy is dominated by the single replica system contributions, meaning that the single replica order parameter saddle point equations can be obtained, *before* taking the double replica index derivative.
The integrals in (A.30) above are straightforward to carry out, and one obtains,

$$G_0 = (1 - q_1)(1 - q_2) \left(4 \hat{R}_1 \hat{R}_2 \hat{q}_{12} + 2 \hat{q}_{12} \right) - q_{12} \hat{q}_{12} \tag{A.31}$$

As $\hat{q}_{12}$ appears only in G_0, $\hat{q}_{12}$ can be eliminated by solving the saddle point equation resulting from differentiating the double replica free energy with respect to $\hat{q}_{12}$. From this one finds,

$$G_0 = -\frac{1}{2} \frac{q_{12} - R_1 R_2}{(1 - q_1)(1 - q_2)} \tag{A.32}$$

Again, we mention that this holds for any (single layer) perceptron, regardless of the activation function.

A.4 Linear Perceptron

Single replica

From [SST92], we have the single replica results,

$$G_0 = \frac{q - R^2}{1 - q} + \ln (1 - q) \tag{A.33}$$

$$G_r = \frac{1}{2} \ln \left[1 + \beta (1 - q) \right] + \frac{1}{2} \frac{\beta (q - 2R + 1)}{1 + \beta (1 - q)} \tag{A.34}$$

The zero temperature minimum of the free energy $G_0 - \alpha G_r$ is given simply by $q = R = 1 - \alpha$, resulting in the generalisation error value $\epsilon_g = 1 - R = 1 - \alpha$.

Double replica

Following the procedure mentioned in section(A.3), the double replicated Hamiltonian can be found from the variance of the single replica (see (A.23)). For the linear perceptron, this is particularly straightforward to find, giving,

$$G_r^{12}(\beta_1,\beta_2) = -\frac{1}{2}\frac{\beta_1}{1+\beta_1(1-q_1)}\frac{\beta_2}{1+\beta_2(1-q_2)}\times$$

$$\left\{(R_1-1)^2(R_2-1)^2 + (q_{12}-R_1R_2)(R_1-1)(R_2-1) + (q_{12}-R_1R_2)^2\right\} \tag{A.35}$$

Again, the entropic term is given by (A.32).

Bibliography

[AF92] S. Amari and N. Fujita. Four types of learning curves. *Neural Computation*, 4:605–618, 1992.

[AGS85a] D. Amit, H. Gutfreund, and H. Sompolinsky. Spin-glass models of neural networks. *Physical Review A*, 32:1007–1018, 1985.

[AGS85b] D. Amit, H. Gutfreund, and H. Sompolinsky. Storing infinite numbers of patterns in a spin-glass model of neural networks. *Physical Review Letters*, 55:1530–1533, 1985.

[Aka74] H. Akaike. A new look at statistical model indentification. *IEEE transactions on automatic control*, 19:716–723, 1974.

[Ama67] S. Amari. Theory of adaptive pattern classifiers. IEEE Trans., EC-16:299–307, 1967.

[Ant95] M. Anthony. Probabilistic analysis of learning in artificial neural networks: The pac model and its variants. Technical report, London School of Economics and Political Science, London, 1995. NeuroCOLT tech report: NC-TR-94-3.

[AR88] J.A. Anderson and E. Rosenfeld, editors. *Neurocomputing: Foundations of Research.* MIT Press, Cambridge, 1988.

[Arf85] Arfken. *Mathematical Methods for Physicists.* Academic Press, Orlando, Florida, 1985.

[AWC89] D. J. Amit, K. Y. M. Wong, and C. Campbell. Perceptron learning with sign-constrained weights. *Journal of Physics A*, 22:2039, 1989.

[Bar96] D. Barber, D. Saad. Does extra knowledge necessarily improve generalisation? *Neural Computation*, 8:202–214, 1996.

[BFOS84] L. Breiman, J. Friedman, R. Olshein, and C. Stone. *Classification and Regression Trees.* Wadsworth, Belmont, California, 1984.

[BS92] M. Biehl and H. Schwarze. Online learning of a time-dependent rule. *Europhysics Letters*, 20:733–738, 1992.

[BS95a] D. Barber and D. Saad. Knowledge and generalisation in simple learning systems. In *ESANN'95*, pages 161–166, Brussels, 1995. D Facto.

[BS95b] D. Barber, D. Saad and P. Sollich. Finite-size effects and optimal test set size in linear perceptrons. *Journal of Physics A*, 28:1325–1334, March 1995.

[BS95c] D. Barber, D. Saad and P. Sollich. Test error fluctuations in finite linear perceptrons. *Neural Computation*, 1995.

[BS95d] M. Biehl and H. Schwarze. Learning by online gradient descent. *Journal of Physics A*, 28:643–656, 1995.

[BSS95a] D. Barber, P. Sollich, and D. Saad. Finite size effects in on-line learning in multilayer neural networks. *Annals of Mathematics and Artificial Intelligence*, 1995. In Press.

[BSS95b] N. Barkai, H.S. Seung, and H. Sompolinsky. On-line learning of dichtomies. In G. Tesauro, D.S. Touretzky, and T.K. Leen, editors, Advances in Neural Information Processing Systems *NIPS 7*. MIT Press, 1995.

[BSS96] D. Barber, D. Saad, and P. Sollich. Finite size effects in on-line learning of multilayer neural networks. *Europhysics Letters*, 34:151–156, 1996.

[Bur89] P. Burman. A comparative study of ordinary cross-validation, v-hold cross-validation, and the repeated learning testing methods. *Biometrika*, 76:503–514, 1989.

[BY86] K. Binder and A. P.. Young. Spin glasses: Experimental facts, theoretical concepts, and open questions. *Reviews of Modern Physics*, 58(4):802–963, October 1986.

[CC95] M. Copelli and N. Caticha. On-line learning in the committee machine. *Journal of Physics A*, 28:1615, 1995.

[DW93] A. P. Dunmur and D. J. Wallace. Learning and generalisation in a linear perceptron stochastically trained with noisy data. *Journal of Physics A*, 26:5767–5779, 1993.

[Eat83] M. Eaton. *Multivariate Statistics - A Vector Space Approach*. John Wiley, New York, 1983.

[EF93] A. Engel and W. Fink. Statistical mechanics calculation of Vapnik Chervonenkis bounds for perceptrons. *Journal of Physics A*, 26:6893–6914, 1993.

[Efr83] B. Efron. Estimating the error rate of a prediction rule: improvement on cross-validation. *Journal of the American Statistical Association*, 78:316–331, 1983.

[Efr86] B. Efron. How biased is the apparent error rate of a prediction rule? *Journal of the American Statistical Association*, 81:461–470, 1986.

[EvdB93] A. Engel and C. van den Broek. Systems that can learn form examples: replica calculation of uniform convergence bounds for the perceptron. *Physical Review Letters*, 71:1772–1775, 1993.

[Fel70] W. Feller. *Introduction to Probability Theory and Its Applications*, volume 1. John Wiley, New York, 1970. (1st edition 1950).

[Gar87] E. Gardner. Maximum storage capacity in neural networks. *Europhysics Letters*, 4:481–485, 1987.

[Gar88] E. Gardner. The space of interactions in neural network models. *Journal of Physics A*, 21:257–270, 1988.

[GGY89] E. Gardner, H. Gutfreund, and I. Yekutieli. The phase space of interactions in neural networks with definite symmetry. *Journal of Physics A*, 22:1995–2008, 1989.

[GT90] G. Gyorgyi and T. Tishby. Statistical theory of learning a rule. In *Neural Networks and Spin Glasses.*, pages 3–36. World Scientific, Singapore, 1990. Editors Theumann W, Koeberle R.

[Han93] L. K. Hansen. Stochastic linear learning: exact test and training error averages. *Neural Networks*, 6:393–396, 1993.

[Hau94] D. Haussler. *The probably approximately correct (pac) and other learning models.* Kluwver, 1994. In A Meyrowitz and S Chipman, editors, *Foundations of knowledge acquisition: Machine Learning.*

[Heb49] D.O. Hebb. *The Organization of Behavior.* Wiley, New York, 1949. Partially reprinted in [AR88].

[Hes94] T. Heskes. On Fokker-Planck approximations of on-line learning processes. *Journal of Physics A*, 27:5145, 1994.

[HK94] T. Heskes and B. Kappen. Learning processes in neural networks. *Physical Review A*, 44:2718–2762, 1994.

[HKST94] D Haussler, M Kearns, H S Seung, and N Tishby. Rigorous learning curve bounds from statistical mechanics. In *Proceedings of the Seventh Annual ACM Workshop on Computational Learning Theory (COLT '94)*, pages 76–87, 1994.

[Hop82] J.J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences, USA*, 79, 1982. Reprinted in [AR88].

[HP91] A. Hertz, J. Krogh and G. Palmer. *Introduction to the theory of Neural Computation.* Addsion-Wesley, Redwood City California, 1991.

[KH92] A. Krogh and J. A. Hertz. Generalization in a linear perceptron in the presence of noise. *Journal of Physics A*, 25:1135–1147, 1992.

[MS96] G. Marion and D. Saad. Finite size effects in Baysian model selection and generalisation. 1996. Accepted for *J.Phys. A.*

[Pin87] F.J. Pineda. Generalization of back-propagation to recurrent neural networks. *Physical Review Letters*, 59:2229–2232, 1987.

[Plu94] M Plutowski. *Selecting Training Exemplars for Neural Network Learning.* PhD thesis, University of California, San Diego, 1994.

[Ros62] F. Rosenblatt. *Principles of Neurodynamics*. Spartan, New York, 1962.

[SA95] D. Saad and Solla S. A. Online learning in soft committee machines. *Physical Review E*, 52:4225, 1995.

[SH92] H Schwarze and J Hertz. Generalization in a large committee machine. *Europhysics Letters*, 20:375–380, 1992.

[Sha93] J. Shao. Linear model selection by cross-validation. *Journal of the American Statistical Association*, 88:486–494, 1993.

[Sol94a] P. Sollich. Finite size effects in learning and generalization in linear perceptrons. *Journal of Physics A*, 27:7771–7784, 1994.

[Sol94b] P. Sollich. Query construction, entropy and generalization in neural network models. *Physical Review E*, 49:4637–4651, 1994.

[Sol95] P. Sollich. *Asking Intellligent Questions - The Statistical Mechanics of Query Learning*. PhD thesis, University of Edinburgh, Scotland, 1995.

[SSS90] D. B. Schwarze, S. A. Solla, and V. K. Samamlan. Exhaustive learning. *Neural Computation*, 2:374, 1990.

[SST92] H. S. Seung, H. Sompolinsky, and N. Tishby. Statistical mechanics of learning from examples. *Physical Review A*, 45:6056–91, 1992.

[Sto74] M Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society, Series B*, 36:111–147, 1974.

[Sto77] C J Stone. Cross-validation: A review. *Math. Operationsforschung, Series Statistics*, 9:1–51, 1977.

[TL93] W. Tarkowski and M. Lewenstein. Learning from correlated examples in a perceptron. *Journal of Physics A*, 26:3669–3679, 1993.

[Val84] L G Valiant. A theory of the learnable. *Communications of the ACM*, 27:1134–1142, 1984.

[VC71] V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory Prob. Appl.*, 16:264–80, 1971.

[WL92] D H Wolpert and A Lapedes. An investigation of exhaustive learning. Technical Report SFI TR 92-04-20, Santa Fe Institute, Santa Fe, 1992.

[Wol92] D. H. Wolpert. On the connection between in-sample testing and generalisation error. *Complex Systems*, 6:47–94, 1992.

[Wol95] D H Wolpert. Off-training set error and a priori distinctions between learning algorithms. Technical Report SFI TR 95-01-003, Santa Fe Institute, Santa Fe, 1995.

[WRB93] T. L. H. Watkin, A. Rau, and M. Biehl. The statistical mechanics of learning a rule. *Reviews of Modern Physics*, 65:499–556, 1993.