BAYESIAN REGULARISATION METHODS IN A HYBRID MLP–HMM SYSTEM

Steve Renals* and David MacKay†

Cambridge University
* *Engineering Department, Trumpington Street, Cambridge CB2 1PZ, UK*
† *Cavendish Laboratory, Madingley Road, Cambridge CB3 0HE, UK*

ABSTRACT

We have applied Bayesian regularisation methods to multi-layer perceptron (MLP) training in the context of a hybrid MLP–HMM (hidden Markov model) continuous speech recognition system. The Bayesian framework adopted here allows an objective setting of the regularisation parameters, according to the training data. Experiments were carried out on the ARPA Resource Management database.

Keywords: *Speech Recognition, HMM, Neural Network, Regularisation, Bayesian*

1. INTRODUCTION

Over the past 2 years, high performance connectionist-statistical talker-independent continuous speech recognisers have been constructed [1, 2]. These systems have exploited a statistical understanding of feed-forward (and recurrent) networks, enabling them to be used within a hidden Markov model (HMM) speech recognition framework as state output probability estimators.

We typically use a multi-layer perceptron (MLP) to estimate the HMM output probabilities. The best performing systems of this type have used very large MLPs, with 1000 hidden units and 250,000 weights. Training these networks is a formidable problem, both in the computational and algorithmic effort required to estimate 250,000 parameters from 1.3 million samples and in the avoidance of overtraining, that is, the memorisation of fluctuations in the training data.

2. MLP PROBABILITY ESTIMATION

MLPs may be used to estimate probabilities. Several authors (e.g. Hopfield [3]) have studied feed-forward networks in terms of learning probability distributions and Bourlard and Wellekens [4] proved that a MLP trained to perform classification is a class-conditional posterior probability estimator. That is, if an MLP is trained to perform '1–from–N' classification (where N output units correspond to N pattern classes) then the ith output (corresponding to class c_i) of the trained network to a new input pattern $\mathbf{x}$ will be an estimate of $P(c_i|\mathbf{x})$, the posterior probability of the class given the data.[1] In this work the class c_i corresponds to a phone.

The posterior probability is what is required if the MLP is to be used directly for pattern classification. However, we use the MLP as a component of a larger speech recognition system, incorporating HMMs to model the temporal structure of speech. The MLP outputs are used as state output probability estimators [5] in the HMM. In this case we do not want the MLP posterior estimates, as these estimate the priors $p(c_i)$ using the relative frequencies of each class in the training data. HMMs implicitly specify the priors $p(c_i)$ through the model topology and state transition probabilities. Thus we divide the MLP outputs by the relative frequencies $R(c_i)$ to obtain scaled likelihoods, $(P(c_i|\mathbf{x})/R(c_i) = p(\mathbf{x}|c_i)/p(\mathbf{x}))$.

3. TRAINING PROCEDURES

We use a stochastic gradient descent procedure to train the MLP. This involves the per sample estimate of the gradient, $\partial E/\partial w_{ij}$, where E is the sample error (usually cross-entropy or sum squared error) and w_{ij} is a weight. For large redundant datasets, stochastic training is much preferable to "batch mode", where the exact gradient is computed by summing over all patterns.

Previous work has used a cross-validation procedure to avoid overtraining. In this approach, a portion of the training set is held out and not used directly in the parameter estimation process. Training is stopped 'early' when classification performance on the validation set fails to improve. This has proven to be a successful strategy, in terms of speech recognition accuracy. By training with early stopping, we have been able to train large networks which generalise well to an unseen data set. Indeed, cross-validation has allowed networks of increasing size to be trained, with a concomitant improvement in generalisation.

However, the MLPs that we use in our speech recognition systems do not seem as parsimonious or as efficient as might be desired. Reduction in network size becomes particularly important as training data sets increase in size and when real-time implementation of a recogniser is considered. Additionally, stopping early when training a large network results in many weights being noisily estimated or undertrained. Furthermore, cross-validation procedures do not allow us to use all the training data for parameter estimation. It should be possible to train smaller networks using all the training data and obtain performance equivalent to or better than that of our current networks.

4. REGULARISATION

A simple prior, $\mathcal{H}_G$ for the parameters of the network states that all the parameters are expected to have similar magnitude. For

[1] This probability is also conditioned on the network and weights.

example, we could model all the parameters $\{w_i\}$ as coming from a single Gaussian distribution,

$$P(\{w_i\}|\alpha, \mathcal{H}_G) = \frac{1}{Z_W}\exp(-\alpha E_W), \tag{1}$$

where $E_W = \sum w_i^2/2$, and α is an unknown 'hyperparameter' which specifies the variance $\sigma_W^2 = 1/\alpha$ of the gaussian from which the parameters are drawn. The true prior for the parameters can be obtained by integrating over the hyperparameter α. But for the moment let us assume α is fixed. When we combine the log of the prior (1) with the log probability of the data and differentiate in order to maximise the posterior probability of the parameters, we find that the gradient $\partial \log P(\mathbf{w}|\alpha, \mathcal{H}_G)/\partial w_{ij}$ gives a 'weight decay' term $-\alpha w_{ij}$, which prevents w_{ij} from growing large. This simple 'regularisation' scheme is also known as ridge regression, and has often been found to enhance the performance of both interpolation and classification models, *as long as α is set appropriately*. We will shortly explain the Bayesian framework for setting α.

The prior that states that all parameters come from a single Gaussian is not always well–matched to the real world. Firstly, for reasons of dimensional consistency, it may be appropriate to divide the parameters into several classes c, with independent scales α_c (for example, the input weights, the biases, and the output weights may all have different dimensions) [6]. Assuming a Gaussian prior for each class, we can define $E_{W(c)} = \sum_{i\in c} w_i^2/2$, so the prior is:

$$P(\{w_i\}|\alpha_c, \mathcal{H}_{G'}) = \frac{1}{\prod Z_{W(c)}}\exp(-\sum \alpha_c E_{W(c)}), \tag{2}$$

we then obtain weight decay with a different decay rate for each class of weights. This prior allows the output weights to be much larger than the input weights, for example, if that is what the data want. The data can determine the multiple constants $\{\alpha_c\}$ using the Bayesian framework reviewed below.

A special case of this prior is used by the 'Automatic Relevance Determination' (ARD) model, $\mathcal{H}_{\text{ARD}}$ [7]. This prior is appropriate if we believe that some of the input variables for our classification may actually be irrelevant to the problem. A simple and 'soft' prior which embodies the concept of relevance is obtained if we take the prior (2) with multiple regularisation constants, and define the weight classes c so that there is a separate 'α' for each input. The Bayesian framework described below automatically embodies Occam's razor, and is able to infer large values for the regularisation constants of irrelevant inputs. This prevents those inputs from causing significant overfitting — whereas under the simple prior $\mathcal{H}_G$, random correlations between the irrelevant inputs and the output would hurt the model's performance. The Automatic Relevance Determination model produced the winning entry in the 1993 Energy prediction competition.

Continuing to richer priors, we might believe that there are regularities in the problem such that some parameters are expected to assume similar non–zero values to each other. The 'soft weight–sharing' model uses a mixture of Gaussians to capture this prior [8]. Here the label c runs over the Gaussian components of the mixture distribution.

$$P(\{w_i\}|\{p_c, \mu_c, \sigma_c\}, \mathcal{H}_{\text{SWS}}) = \prod_i \left(\sum_c p_c \mathcal{N}(w_i; \mu_c, \sigma_c)\right) \tag{3}$$

This prior gives weight decay towards the means μ_c, each weight being attracted most towards the mean to which it is most likely to belong. The parameters $\{p_c, \mu_c, \sigma_c\}$ of the mixture model can be re–estimated so as to adapt them to the data.

We may regard stopping early as a form of regularisation, in which training according to the data is moderated by a force toward a random (initial) weight matrix. This regulariser is a somewhat unsophisticated prior, relying on an excess of weights in the network.

5. THE BAYESIAN FRAMEWORK

In the Bayesian framework, the overall aim is to find models that are well-matched to the data, and make optimal predictions with those models. Neural network learning is interpreted as an **inference** of the most probable parameters for the model, given the training data. The search in model space (*i.e.*, the space of architectures, noise models, preprocessings, regularisers and weight decay constants) can then also be treated as an inference problem, where we infer the relative probability of alternative models, given the data. Bayesian model comparison embodies Occam's razor, *i.e.* a preference for simple models, so that the model complexity is automatically tuned by Bayesian inference and the model is matched to the data. In principle there is no need for test data to be left out: all the data can be devoted to both parameter optimisation and to model comparison.

The Bayesian framework for neural nets can be implemented numerically by Monte Carlo methods [9] or by deterministic methods involving Gaussian approximations, the 'evidence' framework [10, 6]. Here we review the way in which the evidence framework can optimise multiple regularisation constants on-line during learning. The framework has numerous other spin-offs, including the evaluation of error bars on parameters and predictions; marginalisation over these error bars to improve classifier predictions; objective functions for pruning; and data selection criteria for active learning.

The evidence framework optimises the decay constants by finding their most probable value, *i.e.*, the maximum over $\{\alpha_c\}$ of the 'evidence', $P(D|\{\alpha_c\}, \mathcal{H})$. This maximum has useful properties which allow it to be located efficiently by on-line re-estimation techniques. We summarise here the method for the case of a single regularisation constant α. As shown in [10], the maximum evidence α satisfies the following self-consistent equation:

$$\alpha \sum_i w_i^{\text{MP}\,2} = \gamma \tag{4}$$

where $\mathbf{w}^{\text{MP}}$ is the parameter vector which minimises the objective function $M = E_D + \alpha E_W$, where E_D, is the data error (*e.g.* relative entropy), and γ is the 'number of well-determined parameters', given by $\gamma = k - \alpha \text{Trace}(\mathbf{A}^{-1})$, where k is the total number of parameters, and $\mathbf{A} = -\nabla\nabla \log P(\mathbf{w}|D, \mathcal{H})$. The matrix $\mathbf{A}^{-1}$ measures the size of the error bars on the parameters $\mathbf{w}$. Thus $\gamma \to k$ when the parameters are all well-determined;

otherwise, $0 < \gamma < k$. Recalling that α corresponds to the variance $\sigma_w^2 = 1/\alpha$ of the assumed distribution for $\{w_i\}$, equation (4) specifies an intuitive condition for matching the prior to the data, $\sigma_w^2 = \langle w^2 \rangle$, where the average is over the γ effective parameters; the other $k - \gamma$ effective parameters having been set to zero by the prior.

A re-estimation formula for optimising α can be based on equation (4), namely $\alpha := \gamma / \sum_i w_i^{\text{MP}2}$. However, in large problems, the evaluation of the trace matrix inverse is computationally intensive, so in this work we have used the approximation $\gamma \simeq k$, to give the cheap re-estimation formula $\alpha := k / \sum_i w_i^{\text{MP}2}$.

6. EXPERIMENTAL

6.1. BASELINE SYSTEM

The baseline recognition system that we use is similar to that described in Robinson *et al.* [11]. We use a 12th order perceptual linear prediction (PLP) front end, including temporal derivatives of the features, resulting in 26 features per frame. The input to the MLP incorporates a context of ± 3 frames, giving a total of 182 (7×26) inputs. 68 output units were used corresponding to phone classes, and the MLPs incorporated varying numbers of hidden units, from 64 to 1000.

Each phone was modelled by a single distribution HMM, with the state output probabilities for all states in the phone model coming from the associated MLP output. Phone HMMs were left-to-right models of $d/2$ states with no skip transitions, where d was the average phone duration, computed from the training data. Transition probabilities were set to 0.5, thus giving a duration model approximating a (truncated) Poisson distribution with mean and variance d, and minimum duration $d/2$.

6.2. TASK

We have performed experiments on the DARPA Resource Management (RM) task to evaluate this approach. This has a vocabulary of 998 words and a word pair grammar of perplexity 60. Initial experiments were performed on a speaker dependent (SD) task, using a single talker, `dtd0_5`. In this case there was a training set of 600 sentences, with a further 100 sentences for development and an evaluation set of 100 sentences. The RM speaker independent (SI) task is larger and this was attempted after algorithms had been tested on the SD task. In this case a training set of 3990 sentences uttered by 109 speakers was used, of which 300 were held out for cross-validation, with a test set of 300 sentences (February 1989).

For both SI and SD cases, training alignments were produced using the Wernicke MLP–HMM speaker independent RM recogniser described in [11].

6.3. TRAINING

In these experiments we used a training schedule based on that described in section 3.. This was optimised empirically for the standard MLP–HMM system. Thus, the weights were initialised to small random values and the gradient descent step-size was set using a cross-validation procedure. In this schedule, training starts with a fixed step-size and continues until there fails to be an improvement on a cross-validation set. The step-size is then halved each epoch, with training be halted when there is no further improvement on the cross-validation set. We used the Bayesian framework described above to set the regularisation constant, assuming that the number of weights was equal to the number of well-determined parameters.

Hidden Units	Word error (%) No Reg	Decay	ARD
64	8.6	7.7	10.0
256	5.3	5.4	5.7

Table 1. Recognition error on the 100 sentence (818 word) speaker dependent evaluation set.

6.4. SPEAKER DEPENDENT EXPERIMENTS

MLPs containing 64 and 256 hidden units were used in these experiments. As a contrast an non-regularised MLP with 1000 hidden units has achieved an error of 3.8% on this task [5]. The results are shown in table 1. Here the regularisation had little effect, particularly with the larger network. With this small task the difference between a 5.4% and 5.3% word error is only 1 word.

6.5. SPEAKER INDEPENDENT EXPERIMENTS

Initial experiments on the speaker-independent database, using a 1000 hidden unit network trained using weight decay, have produced an error rate of 7.5% on the February 1989 testset. This compares with an error of 5.1% using a similar non-regularised MLP[2]. It should be noted that the score of 7.5% error is significantly better than our preliminary results with the non-regularised MLP.

7. DISCUSSION

7.1. EFFECT OF REGULARISATION

Figure 1 shows the distribution of input-to-hidden weight values after training with and without regularisation (weight decay). These weights are from a 1000 hidden unit network, trained using the speaker independent dataset. The non-regularised network has much longer tails on the weight distribution (ranging between ± 2), compared with the regularised network (± 0.25). Also note that the network that was not regularised has a high density of zero (or almost zero) weights (*i.e.* more weights "switched off"). A comparison of these two distributions seems to indicate that the regularised network was probably over-regularised.

7.2. RESEARCH ISSUES

These results are preliminary, and there are several open research issues:

- *Optimisation.* We have used an optimised schedule that was developed for cross-validation training. It may well be an inappropriate schedule for directly regularized training.
- *Initialisation.* In these experiments, the weights were initialised to small random values. In the case of early-stopping, this corresponds to setting a prior for the weight matrix. When we regularise with a parameterised prior, it may not be necessary to initialise the weights in this fashion.

[2] Although the initialisation and subsequent early-stopping did provide a heuristic form of regularisation

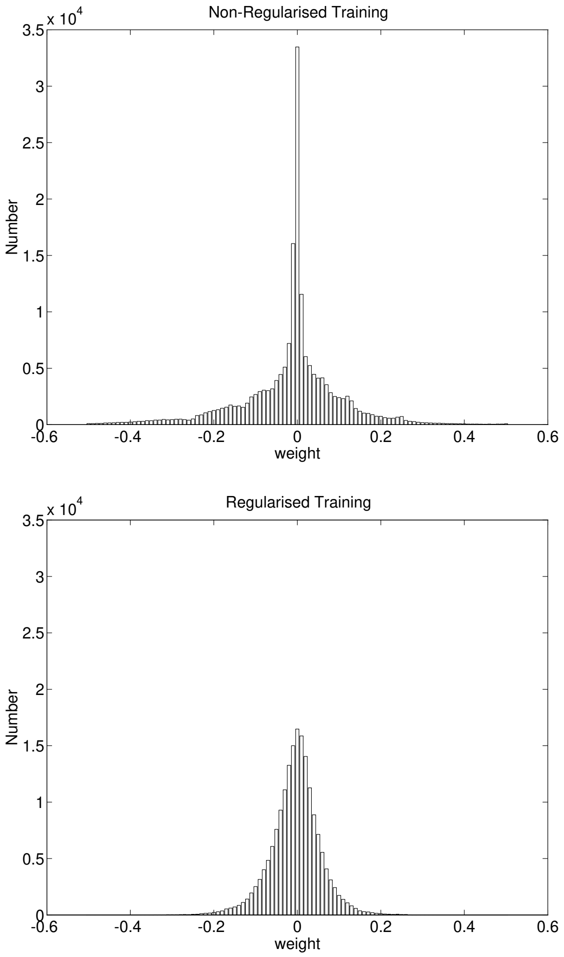

Figure 1. Distribution of input-to-hidden weights after non-regularised (above) and regularised (below) training. The regularisation took the form of a Gaussian prior on the input-to-hidden weights, *i.e.* a weight decay term.

- *Approximation.* The number of well-determined parameters was approximated by the actual number of parameters. If a substantial fraction of the parameters are in fact ill-determined, then this approximation will lead to over-regularisation. One way to avoid this would be the specification of a "fudge factor" of $1 - \delta$ where δ is the fraction of ill-determined parameters.

ACKNOWLEDGEMENTS

Steve Renals was supported by a SERC postdoctoral fellowship and David MacKay by a Royal Society fellowship. This work was supported by ESPRIT BRA 6487, WERNICKE. Thanks to Mike Hochberg for helpful conversations and comments.

REFERENCES

[1] S. Renals, N. Morgan, M. Cohen, and H. Franco. Connectionist probability estimation in the DECIPHER speech recognition system. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 601–604, San Francisco, 1992.

[2] T. Robinson. Recurrent nets for phone probability estimation. In *Proceedings DARPA Workshop on Continuous Speech Recognition*, September 1992. In press.

[3] J. J. Hopfield. Learning algorithms and probability distributions in feed-forward networks. *Proceedings of the National Academy of Sciences USA*, 84:8429–8433, 1987.

[4] H. Bourlard and C. J. Wellekens. Links between Markov models and multilayer perceptrons. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-12:1167–1178, 1990.

[5] H. Bourlard and N. Morgan. Continuous speech recognition by connectionist statistical methods. *IEEE Transactions on Neural Networks*, 1993. In press.

[6] D. J. C. MacKay. A practical Bayesian framework for backprop networks. *Neural Computation*, 4:448–472, 1992.

[7] D. J. C. Mackay and R. M. Neal. Automatic relevance determination for neural networks. 1993. In preparation.

[8] S. J. Nowlan and G. E. Hinton. Simplifying neural networks by soft-weight sharing. *Neural Computation*, 4:473–494, 1992.

[9] R. M. Neal. Bayesian learning via stochastic dynamics. In S. J. Hanson, J. D. Cowan, and C. L. Giles, editors, *Advances in Neural Information Processing Systems*, volume 5, pages 475–482. Morgan Kaufmann, San Mateo CA, 1993.

[10] D. J. C. MacKay. Bayesian interpolation. *Neural Computation*, 1992. 415–447.

[11] A. J. Robinson *et al*. A neural network based, speaker independent, large vocabulary, continuous speech recognition system: the WERNICKE project. In *Proceedings European Conference on Speech Communication and Technology*, 1993.