PRUNING BOLTZMANN NETWORKS AND HIDDEN MARKOV MODELS

Morten With Pedersen[1] *David G. Stork*[2]

[1] Section for Digital Signal Processing, Department of Mathematical Modelling
Technical University of Denmark B321, DK-2800 Lyngby, DENMARK
`mwp@imm.dtu.dk`

[2]Machine Learning and Perception Group, Ricoh California Research Center
2882 Sand Hill Road Suite 115 , Menlo Park, CA 94025-7022 USA
`stork@crc.ricoh.com`

ABSTRACT

We present sensitivity-based pruning algorithms for general Boltzmann networks. Central to our methods is the efficient calculation of a second-order approximation to the true weight saliencies in a cross-entropy error. Building upon recent work which shows a formal correspondence between linear *Boltzmann chains* and Hidden Markov Models (HMMs), we argue that our method can be applied to HMMs as well. We illustrate pruning on *Boltzmann zippers*, which are equivalent to two HMMs with cross-connection links. We verify that our second-order approximation preserves the rank ordering of weight saliencies and thus the proper weight is pruned at each pruning step. In all our experiments in small problems, pruning reduces the generalization error; in most cases the pruned networks facilitate interpretation as well.

1. INTRODUCTION

There is an enormous body of simulation work demonstrating the value of architecture optimization for networks for pattern classification, and this has properly led to great interest in both theoretical foundations and in new algorithms. There are two basic viewpoints toward this issue: regularization (or penalty based) and sensitivity based. According to the viewpoint of regularization, one seeks to impose some desired property in the final solution, for instance smoothness. Thus in weight decay one penalizes large weights and therefore favors smoother decision boundaries. According to the viewpoint of sensitivity, one seeks to eliminate those parameters (e.g., weights) that have the smallest effect on the training error, thereby restricting the model without severely penalizing the training error. For instance, Optimal Brain Damage (OBD) [1] and Optimal Brain Surgeon (OBS) [2] eliminate weights that are predicted to have the least effect on the training error.

In fact both views stem from a deeper notion concerning the incorporation of model priors or structural risk minimization [3]. Despite their fundamental unity, in computational practice it is convenient to adopt one or the other of these views. For instance, regularization by weight decay is easy to incorporate *during* learning; pruning by sensitivity-based methods is traditionally performed *after* training with examples. (We note too that in practice one can use *both* methods when creating networks.) In this paper we mention briefly the application of weight decay but our primary contribution is a new sensitivity-based pruning algorithm for Boltzmann networks.

Most pruning methods have been developed for networks of nonlinear units — feed-forward or recurrent — typically trained by backpropagation or second-order methods. However pruning in another class of network, Boltzmann networks, has not received adequate attention. Though typically slower and a bit more difficult to train than feedforward neural networks, Boltzmann networks nevertheless have some desirable properties: natural handling of missing data (during training or recall), pattern completion, and superior avoidance of local energy minima during training. Based on the close relationship between certain Boltzmann networks and Hidden Markov Models [4,5], we now know that such Boltzmann networks can posess benefits of HMMs too, most particularly dynamic time adjustment.

Our paper is organized as follows: In Section 2. we provide a short review of Boltzmann networks. In Section 3., we show a second-order expansion of the cross-entropy cost function and present our sensitivity based pruning algorithm. In Section 4. we apply our method to a Boltzmann architecture that is of particular interest for integration of two channels having different time scales. We conclude in Section 5. with some thoughts on future directions.

2. BOLTZMANN NETWORKS

Boltzmann networks are stochastic networks with both visible and hidden units (cf., [6] for an introduction and the notation we use here). We let the subscript α denote the states of the visible units and β the states of the hidden units. The superscript $+$ denotes iterating the network with the visible units clamped to a desired pattern, and $-$ denotes the visible units running freely, or unclamped. The energy function for the Boltzmann network is usually defined as

$$E = -\frac{1}{2}\sum_{ij} w_{ij} s_i s_j , \qquad (1)$$

where w_{ij} is the (bi-directional) weight connecting units i and j, and s_i is the (binary) state of unit i. At *thermal equilibrium* the probability of finding the units in a given state configuration $\alpha\beta$ when the visible units are unclamped is given by the Boltzmann distribution

$$P^-_{\alpha\beta} = \frac{1}{Z} e^{-E_{\alpha\beta}}, \tag{2}$$

where Z is the normalizing partition function and $E_{\alpha\beta}$ is the energy (dependent on the weights) when the visible units are in states α and the hidden units are in states β; for clarity in the following, we have incorporated the temperature into the energy term. Thus, the probability P^-_α of finding the visible units in joint states α is found by summing over the possible hidden unit configurations β.

When training Boltzmann networks we want the probabilities of the freely running network P^-_α to match those of the environment/training examples P^+_α. As a measure of the difference between the two probability distributions we use the *Kullback-Leibler* measure, or relative entropy, as our cost function:

$$\begin{aligned} H(\mathbf{w}) &= \sum_\alpha P^+_\alpha \ln \frac{P^+_\alpha}{P^-_\alpha} \\ &= -\sum_\alpha P^+_\alpha \ln P^-_\alpha + const, \end{aligned} \tag{3}$$

where *const* is a constant determined solely by the environment, and is hence independent of the weights $\mathbf{w}$. When training using gradient descent we need the derivatives of $H(\mathbf{w})$ with respect to the bi-directional weights w_{ij} connecting units i and j:

$$\begin{aligned} \frac{\partial H(\mathbf{w})}{\partial w_{ij}} &= -\sum_\alpha P^+_\alpha \frac{\partial \ln P^-_\alpha}{\partial w_{ij}} \\ &= -\sum_\alpha P^+_\alpha \left\langle -\frac{\partial E_{\alpha\beta}}{\partial w_{ij}} \right\rangle^+_\alpha + \left\langle -\frac{\partial E_{\alpha\beta}}{\partial w_{ij}} \right\rangle^- \\ &= < s_i s_j >^- - < s_i s_j >^+, \end{aligned} \tag{4}$$

where $< \cdots >^+_\alpha$ is the mean value given that the visible units are clamped in states α, and $< \cdots >^-$ is the mean when all units are free running.

3. PRUNING

The first derivatives lead to the traditional first-order training methods [6]. However, for training using second-order methods and for our pruning algorithm we need the second derivatives of the entropic cost. These second derivatives are calculated as:

$$\begin{aligned} &\frac{\partial^2 H(\mathbf{w})}{\partial w_{ij} \partial w_{pq}} = \\ &-\left[\sum_\alpha \frac{P^+_\alpha}{P^-_\alpha} \cdot \frac{\partial^2 P^-_\alpha}{\partial w_{ij} \partial w_{pq}} - \sum_\alpha \frac{\partial P^-_\alpha}{\partial w_{ij}} \cdot \frac{P^+_\alpha}{(P^-_\alpha)^2} \cdot \frac{\partial P^-_\alpha}{\partial w_{pq}} \right] \\ &= \sum_\alpha \frac{\partial \ln P^-_\alpha}{\partial w_{ij}} \cdot P^+_\alpha \cdot \frac{\partial \ln P^-_\alpha}{\partial w_{pq}} - \sum_\alpha \frac{P^+_\alpha}{P^-_\alpha} \cdot \frac{\partial^2 P^-_\alpha}{\partial w_{ij} \partial w_{pq}} \\ &\approx \sum_\alpha \frac{\partial \ln P^-_\alpha}{\partial w_{ij}} \cdot P^+_\alpha \cdot \frac{\partial \ln P^-_\alpha}{\partial w_{pq}}. \end{aligned} \tag{5}$$

The approximation in Eq. 5 is justified if we assume that the problem at hand is realizable, i.e., there exists a set of *optimal* weights $\mathbf{w}^*$, for which $P^-_\alpha = P^+_\alpha$, $\forall\alpha$ [7]. For these weights, the term in Eq. 5 involving second derivatives of the unclamped probabilities P^-_α reads

$$\begin{aligned} \sum_\alpha \frac{P^+_\alpha}{P^-_\alpha} \cdot \frac{\partial^2 P^-_\alpha}{\partial w_{ij} \partial w_{pq}} &= \sum_\alpha \frac{\partial^2 P^-_\alpha}{\partial w_{ij} \partial w_{pq}} \\ &= \frac{\partial^2}{\partial w_{ij} \partial w_{pq}} \left(\sum_\alpha P^-_\alpha \right) \\ &= 0, \end{aligned} \tag{6}$$

where in the first step we used the fact that $P^+_\alpha = P^-_\alpha$ at $\mathbf{w} = \mathbf{w}^*$, and in the last step that $\sum_\alpha P^-_\alpha = 1$. Thus, close to the optimal weights $\mathbf{w}^*$ the term in Eq. 5 involving second derivatives vanish. The remaining term involves terms of a form calculated from Eq. 4:

$$\begin{aligned} \frac{\partial \ln P^-_\alpha}{\partial w_{ij}} &= \left\langle -\frac{\partial E_{\alpha\beta}}{\partial w_{ij}} \right\rangle^+_\alpha - \left\langle -\frac{\partial E_{\alpha\beta}}{\partial w_{ij}} \right\rangle^- \\ &= < s_i s_j >^+_\alpha - < s_i s_j >^- . \end{aligned} \tag{7}$$

Thus we obtain the second derivatives:

$$\begin{aligned} \frac{\partial^2 H(\mathbf{w})}{\partial w_{ij} \partial w_{pq}} &\approx \sum_\alpha \frac{\partial \ln P^-_\alpha}{\partial w_{ij}} \cdot P^+_\alpha \cdot \frac{\partial \ln P^-_\alpha}{\partial w_{pq}} \\ &= \sum_\alpha P^+_\alpha < s_i s_j >^+_\alpha < s_p s_q >^+_\alpha \\ &\quad - < s_p s_q >^- \sum_\alpha P^+_\alpha < s_i s_j >^+_\alpha \\ &\quad - < s_i s_j >^- \sum_\alpha P^+_\alpha < s_p s_q >^+_\alpha \\ &\quad + < s_i s_j >^- < s_p s_q >^- \\ &= < s_i s_j s_p s_q >^+ - < s_p s_q >^- < s_i s_j >^+ \\ &\quad - < s_i s_j >^- < s_p s_q >^+ \\ &\quad + < s_i s_j >^- < s_p s_q >^- . \end{aligned} \tag{8}$$

We note that the second derivatives involves only terms already computed when calculating the gradient, thus implementation is straightforward and yields little computational burden beyond that needed for gradient descent learning.

3.1. Pruning using OBD and OBS

Two well-known methods for pruning traditional feed-forward and recurrent networks are Optimal Brain Damage (OBD) [1] and Optimal Brain Surgeon (OBS)

[2]. Both methods use second-order expansions of the error to estimate the importance of the parameters. OBD uses a diagonal approximation to the Hessian to calculate the saliency of a weight by estimating the change in error when the weight is set to zero. OBS uses the full Hessian, and the change in training error is estimated including the effect of reestimating the remaining parameters in the model to a new minimum within the quadratic approximation. The rationale behind both methods is that if we remove the least salient weights according to training error, we gracefully relieve the danger of overfitting, and thereby improve generalization.

The method we present here is firmly rooted in the logic of OBD/OBS; the key novelty is the equation for second derivatives, Eq. 8. We note that for traditional Boltzmann networks, removing a weight from the model is equal to setting it to zero, since the weight no longer provides contribution to the energy (Eq. 1). Thus, the logic of OBD and OBS can be applied directly for these models as well, though using the second derivatives derived above. Even though it is usually intractable to compute the numerical value of the entropic cost (Eq. 3), this does not affect the pruning methods since they only measure the *change* in cost, working from approximations using information already provided by the learning algorithms.

3.2. Pruning Hidden Markov Models

It has been shown that for certain tree-like connectivity of Boltzmann networks it is possible and computationally tractable to compute the expressions in Section 2. and 3. *exactly* [8]; this, therefore, provides greater accuracy for the saliency estimates provided by our method. Such computations have been used to a specific topology of Boltzmann networks called Boltzmann *chains* [4]. It was shown that any first-order HMM can be represented by an equivalent Boltzmann chain [4]. It was furthermore shown that under the condition that all state sequences have a mandatory end state, Boltzmann chains can be represented by first-order HMMs as well [5].

In traditional research on HMMs, the topology of HMMs for a given task has been chosen by hand or found by 'exhaustive' search. However we suggest that the topology can be optimized by *converting the HMM into a corresponding Boltzmann chain and performing pruning on this model.* The resulting chain is then converted back into an HMM for reestimation of the remaining parameters or, alternatively, the reestimation is done for the Boltzmann chain, only converting back the *optimal* model.

Pruning a weight in a Boltzmann chain representing an HMM should be equivalent to setting the corresponding transition probability in the HMM to zero, thus preventing hidden state sequences including the transition in question from contributing to the probability of a given observation sequence. This means that the weight should be *set equal to $-\infty$ if pruned in the Boltzmann chain*, yielding zero contribution to the partition function for state sequences including the transition in question. This is consistent with the expressions in [4] for converting the parameters in an HMM into the weights of a Boltzmann chain, which is accomplished by applying the natual logarithm to the transition probabilites.

When pruning Boltzmann chains representing HMMs we must modify our algorithms somewhat however, since these methods estimate the effect on the cost when a parameter in the chain is set to zero. Instead we are forced to set the weights to $-\infty$ one by one and compute the resulting change in error. This is possible since we are able to perform exact calculations of the entropic cost function for this special topology of Boltzmann networks. Also, we should be careful if/when pruning weights representing observation probabilities, since this means that the observation is no longer possible when in a particular hidden state.

4. EXPERIMENT

It has been shown how to model correlated discrete time series on disparate timescales using cross-connected parallel Boltzmann chains [4], which we call Boltzmann *zippers* (Fig. 1). These models can be interpreted as interconnected HMMs. Such models are of particular interest where one must integrate information from two time series having different inherent time scales, as for instance speechreading, where the fast acoustic information must be integrated with the slow visual information [9]. Here we use pruning to investigate the utilization of the cross-connection links.

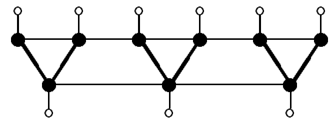

Figure 1. Topology of a Boltzmann zipper. White circles represent groups of visible units/states, dark circles represent groups of hidden units/states.

We generated synthetic patterns by two left-right HMMs. In order to generate observations on two different time scales, a *fast* model was iterated twice as often as a *slow* model. The fast model had three hidden states and three observation symbols, the slow had two hidden states and two observation symbols. The last states in the models were connected, increasing the probability of making transitions to the last hidden state in one model if the other model was already in its last state. Thus, the fast model was able to make direct transitions to its last state *if* the slow model was already in its last state.

Three hundred pairwise sequences of lengths 14 and 7 (respectively) were generated, and a fully connected

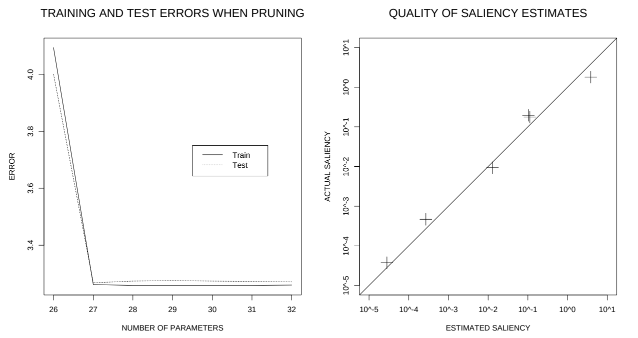

Figure 2. Left panel: Cross-entropy error on training and independent test sets (150 patterns each) versus the number of bidirectional weights between units. Pruning proceeds from right to left, and at the extreme left (26 weights), all cross connections have been removed. Right panel: The true saliences in a full network versus the saliencies estimated to second order. Note the excellent agreement over six orders of magnitude.

Boltzmann zipper having the same number of hidden and visible states as the underlying HMM-models was trained, using 150 examples for training and 150 examples as a separate test set. The model, initially having 32 parameters, was trained using gradient descent followed by a second-order damped Gauss-Newton method. In order to ensure numerical stability and facilitate training, the entropic cost function was augmented by a small quadratic weight decay term.

In order to investigate the utilization of the cross-connections in the zipper, we used our method (in the diagonal Hessian approximation) on these six weights. Since it is not clear how these weights relate to the transition probabilities in the HMMs we chose to limit the degrees of freedom by setting the weights to zero when pruned. The remaining weights were retrained by the second-order method after each weight elimination. In the left panel of Fig. 2 we show the results of pruning. We note that the errors are left almost unchanged until we prune the final cross-connection, after which the errors increase significantly. This is consistent with the model from which the data was generated, and indicates that the zipper has captured the underlying structure well. In fact, the lowest test error was obtained using only one cross connection, and the error rises dramatically if this last connection is also pruned.

In the right panel of Fig. 2 we illustrate the quality of the saliency estimates. For the fully connected zipper, we plot the estimated saliency for the cross connections versus the actual saliency computed by setting the weight to zero and calculating the resulting change in training error. We note that the estimates are approximately equal to the actual saliencies. Note that the rank ordering of the estimated saliency is the same as the actual saliencies; and thus the correct weight is deleted.

In 12 pruning experiments such as just described (in which the generating model had a single cross-connection link), we found that all 12 resulting networks displayed better generalization than the unpruned network. Of these, we found that 9 displayed best generalization using a single link, as expected.

5. CONCLUSION AND FUTURE WORK

We have derived the second derivatives of the entropic cost function and shown how to use these for sensitivity-based pruning of general Boltzmann networks. We have described how to extend this approach to Hidden Markov Models by transforming these into equivalent Boltzmann chains. Finally, we illustrated the viability of pruning on the cross-connections of Boltzmann zippers.

Clearly our method must be further demonstrated on large and realistic problems, such as speechreading. It will be interesting to see if our method, when applied to Boltzmann chains, is computationally more efficient than the exhaustive search method used throughout speech research for highly trained models.

Acknowledgements

This work was completed during a visit to the Ricoh California Research Center, supported by the Danish Natural Science and Technical Research Councils through the Computational Neural Network Center (CONNECT). The authors would like to thank Lars Kai Hansen and Greg Wolff for support, and Lawrence Saul for valuable discussions and for making available his Boltzmann zipper code.

References

[1] Y. Le Cun, J. S. Denker & S. A. Solla (1990) "Optimal Brain Damage," in D. S. Touretzky (ed.), *Advances in Neural Information Processing Systems – 2*, pp. 598-605. San Mateo, CA: Morgan-Kaufmann.

[2] B. Hassibi & D. G. Stork (1993) "Second order derivatives for network pruning: Optimal Brain Surgeon," in S. J. Hanson, J. D. Cowan & C. L. Giles (eds.) *Advances in Neural Information Processing Systems – 5*, pp. 164-171. San Mateo, CA: Morgan-Kaufmann.

[3] V. Vapnik (1982) **Estimation of Dependences Based on Empirical Data** New York: Springer-Verlag.

[4] L. K. Saul & M. I. Jordan (1995) "Boltzmann Chains and Hidden Markov Models," in G. Tesauro, D. Touretzky & T. Leen (eds.), *Advances in Neural Information Processing Systems – 7*, pp. 435-442. Cambridge, MA: MIT Press.

[5] D. J. MacKay (1996) "Equivalence of Linear Boltzmann Chains and Hidden Markov Models," *Neural Computation* **8**(1) 178-181.

[6] S. Haykin (1994) **Neural Networks: A Comprehensive Foundation** New York: Macmillan.

[7] L. K. Hansen (1996) unpublished manuscript.

[8] L. K. Saul & M. I. Jordan (1994) "Learning in Boltzmann trees," *Neural Computation* **6**(6) 1174-1184.

[9] D. G. Stork & M. E. Hennecke (eds.) (1996) **Speechreading by Humans and Machines** New York: Springer-Verlag.