ESANN'1994 proceedings - European Symposium on Artificial Neural Networks
Brussels (Belgium), 20-21-22 April 1994, D-Facto public., ISBN 2-9600049-1-4, pp. 147-152

A comparison of two weight pruning methods

Olivier Fambon and Christian Jutten

TIRF-INPG
46 Av. Felix Viallet
F-38000 Grenoble
E-mail: fambon@tirf.grenet.fr

Abstract: Two methods that attempt to remove useless weights after training are reviewed and compared. Both make use of the second derivative information but use different approaches. Both lead, under the same set of hypothesis, to the same selection criterion for pruning unrelevant weights. Practical comparison is also carried out on a small toy problem.

KeyWords: Layered Networks, pruning criterion, saliency.

1 Introduction: The design issue

Feed-forward Layered Neural Networks (LNN) are well known to be universal approximators provided they are composed of at least one hidden layer and a sufficient number of hidden units. Teaching them with a sufficient database of input/output pairs provides an *approximation* of the phenomenon underlying the database or a *model* of that phenomenon. It is theoretically proven that such architectures are universal approximators [1]. But the number of hidden units required for the approximation is unknown. So usually, a sigmoidal function is used such as tanh and 'enough' units are provided, making the learning phase all the more laborious and uncertain. Moreover, it leads to 'bad generalisation' properties for the network: this phenomenon is known as over-fitting. One way to cope with it is to identify useless parameters and delete (prune) them. Other techniques such as weight decay or weight elimination [2] also aim at reducing the number of parameters, but while learning. They make use of a second term in the objective function to be minimized. We will not discuss them here, but they have been found to be difficult to tune (proportion of the second term).

In fact, pruning parameters from a known acceptable solution, that can be used as a target, seems simpler than adding parameters properly so as to reach a solution. Strikingly, this is what happens with nervous cells.

In this paper, we compare two pruning methods: Optimal Brain Damage [3] and Statistical Stepwise Method [4]. Basically the methods seem quite different, but we show that they are theoretically equivalent under the same set of hypothesis. Experimental results are finally presented on a simple example.

2 Optimal Brain Damage (OBD)

2.1 The idea

Originally, OBD was proposed by Yann Le Cun and al. [3] as an alternative to the naive approach that leads to suppress the smallest weights. OBD was derived to move beyond this approximation and to give a theoretically justified pruning method. The scheme can be stated as : "Prune the weights that, if pruned, will cause the least increase of the overall error".

Suppose we have a LNN that has learnt by minimising an average squared error (or any additive measure) E. The vector of adjusted parameters $P = [p_i]$ is a point in the space of parameters that realises a (local) minimum of the error. It is possible to expand the error into a Taylor serie around that point and thus to quantify the influence of a (little [1]) variation of a parameter p_i over the error. The idea is that deleting a parameter p_i is understood as bringing it to zero, thus causing a variation $\delta p_i = (0 - p_i)$. Using a Taylor expansion of E around P:

$$\delta E = \sum_i \frac{\partial E}{\partial p_i}\delta p_i + \sum_i \frac{\partial^2 E}{\partial p_i^2}\delta p_i^2 + \sum_{i,j/i\neq j} \frac{\partial^2 E}{\partial p_i \partial p_j}\delta p_i \delta p_j + O(\| \delta P \|^3) \quad (1)$$

and making the following hypothesis: 1) E is almost quadratic (order 2 development is enough), 2) The Hessian is diagonal (cross terms vanish), and 3) The network is at a local minimum (first term vanishes), leads to the simplified expression of saliency:

$$s_i = \frac{\partial^2 E}{\partial p_i^2}(p_i)^2. \quad (2)$$

This is read as "deleting parameter p_i (setting it to 0) causes an increase of E of s_i". s_i is used to seek parameters, and the p_i's associated with the smallest s_i's are pruned.

The original pruning procedure integrates this idea in the following form: 1. Start off with a sufficiently large network, 2. Teach the net until a reasonable solution is reached, 3. Calculate saliencies s_i for each parameter p_i, sort the p_i using s_i and delete lowest saliency p_i's, 4. Loop in 2 as long as some stopping condition is not met.

Note that the retraining phase is theoretically essential in order to guarantee that the parameter vector P realises a minimum of E.

2.2 Diagonal Hessian?

The hypothesis of a diagonal Hessian seems strong and only justified by simplification matters. But if we wish to delete only one parameter p_i at a time, it is

[1] A Taylor expansion makes sense only for small variations of P

easy to see in (1) that the saliency s_i holds even without a diagonal Hessian. In this case, the third sum (cross terms) is null because $\delta p_j = 0$ thus $\delta p_i \delta p_j = 0$ for $j \neq i$. In fact, off diagonal terms are useful only if we wish to delete *several* parameters at a time (they express the dependency between deletion of different parameters). However, as the computation of the diagonal of the Hessian is already heavy, it is desirable to 'factorize' it for several parameters. But, this is theoretically correct only when assuming the stronger hypothesis of a diagonal Hessian.

2.3 Calculation

The original method requires the calculation of the diagonal of the Hessian. This is done in an exact way, inspired of back-propagation (BP). As a matter of fact, it is possible to propagate not only first derivatives but also second derivatives in the BP scheme (simply derive once more the BP equations). Calculating the second derivatives of the Mean Squared Error (MSE) with respect to the parameters $\{p_i\}$ leads to an "average" saliency over the whole test base. The complexity of the operation is roughly 3 times the complexity of one learning step. This is the reason why the hypothesis of the diagonal Hessian is interesting.

3 Statistical Stepwise Method (SSM)

3.1 The idea

SSM is a method for pruning weights under information criterion control proposed by Marie Cottrell et al. in [4]. Its principle is strongly linked to the Least Squares (LS) estimation theory, but simple in its philosophy. It can be stated as: "Prune the weights that are statistically non significant, but only if the resulting net is "better" than the previous one".

The notion of quality for a network (model) is defined as Akaike's B Information Criterion (BIC, see end of section for details).

The key is to consider that the vector of estimated parameters $P = [p_i]$ obtained after a LS optimization is a vector of random variables, $\hat{P}$, known as the LS estimator of the theoretical solution P^* that we seek. Thus we can try to describe the law of that LS estimator $\hat{P}$. As a matter of fact, due to the boundness properties of the sigmoidal functions and to the fact that we are seeking a LS estimator, $\hat{P}$ is asymptotically Gaussian, that is

$$\sqrt{T}(\hat{P} - P^*) \xrightarrow[T\to\infty]{\mathcal{L}} N(0, \sigma_r^2 \Sigma^{-1}) \qquad (3)$$

where T is the size of the learning base, σ_r^2 is the residual variance of the model (MSE) and Σ the Hessian of the error E with respect to the parameters at P^*. Now that we know the asymptotic law of vector $\hat{P}$, we can test (statistically) the nullity of each of it's components. The test is a standard Gaussian test, carried out on the mean of the estimator (P^* from (3)) in order to decide whether it is null or not. $p_i^* = 0$ is tested against $p_i^* \neq 0$, σ_r^2 is estimated by $\hat{\sigma}_r^2$, and the variance of $\hat{p}_i$ is then estimated by $\hat{\sigma}_r^2$ multiplied by the i^{th} diagonal term of the inverse of the Hessian taken in $\hat{P}$ (cf. (3)). The criterion is thus the quantity defined for the test rule

$$t_i = \left| \frac{\hat{p}_i}{\hat{\sigma}(\hat{p}_i)} \right| \tag{4}$$

For a fixed significance level of 5%, p_i is considered non significant as soon as $t_i < 1.96$. t_i is used to seek parameters, and the p_i associated to the smallest t_i such that $t_i < 1.96$ is selected as candidate for pruning. It is actually pruned only if the BIC of the pruned net, *after retraining*, is smaller than the BIC of the current net. If not, pruning is stopped. In the end, the effect of pruning is controlled using an information criterion

$$BIC = \log(\frac{\sigma_r^2}{T}) + n_p \frac{\log(T)}{T}$$

where n_p is the number of parameters of the net (model).

3.2 Calculation

Here again, the root of the method is the Hessian. In this case, the full matrix is computed, then inverted, and the diagonal terms are considered to compute the criterion. In fact, any heuristic can be used to get the Hessian, or better the diagonal of the inverse. At the moment, the full Hessian is computed using the approximation $\nabla^2 E \equiv (1/T) \cdot \sum \vec{\nabla} y(X_n) \cdot {}^t\vec{\nabla} y(X_n)$ where y is the output of the net and X_n the n^{th} training pattern. The complexity of the operation is roughly $T \times n_p^2 + n_p^3$, where n_p is the total number of parameters. The inversion is the heaviest task in the case of small databases.

4 Comparison

4.1 Theory

It is simple to notice that both use the second derivative information to select the parameters to be pruned. Indeed, if we suppose that the Hessian is diagonal (as in OBD) both criterion are rigorously equivalent. In fact, in equation (4) the term $\hat{\sigma}(\hat{p}_i)$ is proportional to the square root of the diagonal term of the inverse of the Hessian. Thus if the Hessian is diagonal, it's inverse is also diagonal and the terms $\hat{\sigma}(\hat{p}_i)$ are such that $\hat{\sigma}^2(\hat{p}_i) = \sigma_r^2 / \frac{\partial^2 E}{\partial p_i^2}$. Then it is simple to see that equation (2) is just the square of equation (4), except for the threshold. Thus selecting the parameters having a minimum criterion in either way is equivalent. The other hypotheses are equivalent and can be summed up as “convergence is reached and the error is quadratic”.

SSM provides a theoretically justified threshold for the selection phase. OBD, doesn't, but it is still possible to design a simple threshold allowing only a small

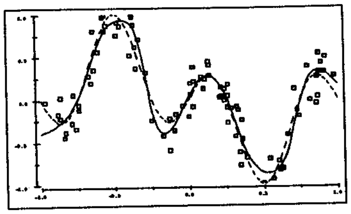

Fig 1. Function f (dashed), Samples, Net output (solid).

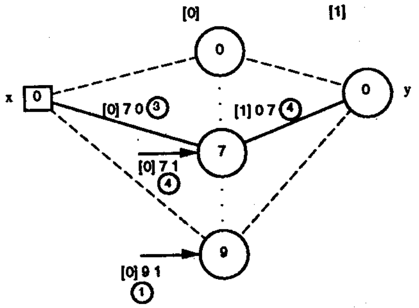

Fig 2. Pruned connections (solid).

Stage	OBD	SSM	Eval
1	[1] 0 9 [0] 9 1	[0] 9 1 [0] 9 0	-0.30 0.17%
2	[0] 9 0 [0] 8 1	[0] 8 1 [0] 9 0	-0.77 11.6%
3	[0] 7 0 [0] 7 1	[0] 7 0 [0] 7 1	-1.63 5.1%
4	[1] 0 7 [0] 7 1	[0] 7 1 [1] 0 7	-1.75 4.8%
5	[0] 4 1 [0] 2 1	[0] 4 1 [0] 4 0	-1.98 4.1%
6	[0] 2 1 [0] 2 0	[0] 2 1 [0] 2 0	-2.14 3.7%
7	[0] 2 0 [1] 0 2	[0] 2 0 [1] 0 2	-2.22 3.6%
8	[0] 9 0 [1] 0 9	[0] 9 0 [1] 0 9	-2.31 3.58%
9	[0] 4 0 [0] 8 0	[0] 8 0 [1] 0 8	-2.41 3.57%

Table 1. Pruning results.

percentage of error increase as the saliencies are defined in terms of error variation. Finally, the idea of post-pruning verification using the BIC criterion could be applied to any pruning procedure. It could be compared to a sophisticated try and error procedure, the last decision relying on the BIC, and the selection criterion pointing out the parameters to be pruned.

4.2 A small toy problem

In order to evaluate the criteria, we have built a little problem: learn function $f(x) = \sin(\pi x) \cdot \sin(2\pi(x + 0.5))$ defined on $[-1, 1]$ from a set of 100 samples randomly (uniform) distributed over the interval. Samples are noisy pairs $(x, f(x)+n)$ where n is a random number in $[-0.3, 0.3]$ (Fig 1). This was used to compare the criteria: do they prune the same weights under identical conditions? The results on a network composed of 10 hidden units are reported.

On Fig 2, [1]07 means weight to unit 0 layer [1], from unit 7 layer [0]. Small circled numbers indicate the order in which connections were pruned ('Stage' in Table 1). Arrows are the biases.

Pruning is achieved on the same network for both methods. This net is at a (local) minimum i.e training is performed between each stage. It is often carried out on the net resulting from SSM pruning.

In Table 1, each double line shows which weight is pruned (first line) and which is next when ordered by s_i (OBD) or t_i (SSM). The 'Eval' column shows BIC and MSE (in %) before pruning or, equivalently, after retraining. It is calculated on the net resulting from SSM.

Pruning stops for SSM because the selected weight has $t_i > 1.96$. However, no further pruning is achievable, as from experience, 6 units are required for this task, so the BIC of any over pruned net would be greater that the one obtained here (final net: BIC=-2.48, MSE=3.6%). Note that 3.6% error is very close to the variance induced by the noise (3%) so that no further learning is desirable.

It appears that both methods select the same weights in 5 stages, and select identical pairs in 2 stages. Note that the third weight for either method has great s_i or t_i compared with the two first so that the conclusion is relatively stable: both methods are comparable in practice on this task.

In conclusion, the small differences observed may come from the different procedures used to calculate the Hessian, or from the fact that it is hard in practice to guarantee that the net is at a minimum, which is the essential hypothesis of both methods. Current work aims at applying these methods to RBF networks.

Acknowledgements

We would like to thank Marie Cottrell and Morgan Mangeas for their collaboration, and especially Morgan for his precious advice and for the simulations. This work is partly funded by ELENA Nerves 2 (Esprit Project BR 6891).

References

[1] K. Hornick and al. “Multilayer feed-forward networks are universal approximators”, Neural Networks, Vol. 2, 1989, pp 359-366

[2] S. J. Hanson and L. Y. Pratt. “Comparing Biases for Minimal Network Construction With Back-Propagation”, Advances in Neural Information Processing I, Morgan Kaufmann, 1989.

[3] Y. Le Cun, and al. “Optimal Brain Damage”, Proceedings of The Neural Information Processing Systems, Denver 1990.

[4] M. Cottrell and al. “Times Series and Neural Network: a Statistical Method for Weight Elimination”, ESANN’93, pp 157-164