**Magnitude-based Pruning**

This method eliminates weights with the minimum absolute values. Although it is simple and fast, it does not produce accurate weight pruning, i.e. it does not eliminate weights in an optimized manner relative to the effect on the error function. Moreover, this algorithm requires full retraining after each step, which may cause a computational overhead if the size of the training samples is very large.

Another intuitive idea of pruning is just to set the weight to zero and evaluate the change of error by running forward pass. If the change is too low the weight is removed, otherwise it is restored. This is obviously very inefficient since for each weight, a forward pass takes $O(w)$ operations, where $w$ is the total number of network weights, and this in total would require $O (mw^{2})$ operations where *m* is the number of training samples.

**Sensitivity-based Pruning**

Several measures have been used to evaluate the sensitivity of removing a weight from the network. One method (Mozer & Smolensky, 1989) measures the relevance $ρ\_{i}$ of the unit $u\_{i}$; the importance of the unit is then approximated by using the training feedback of the ANN provided by the backpropagation algorithm. If $ρ\_{i}$ falls below a certain threshold, the unit can be deleted.

Another sensitivity measure introduced in (Karnin, 1990) calculates the sensitivity $S\_{ij}$ of the network weight $w\_{ij} $by monitoring the sum of all changes $Δw\_{ij}$ experienced by the weight $w\_{ij} $during training. After the training, each weight has an estimated sensitivity and the lowest sensitivity weight can be removed.

The above-mentioned methods consider the sensitivity measure evaluations based somehow on the first-order information of the error function (gradients). Other sensitivity measures estimate the error “saliency” by making use of the second derivative information via Hessian matrix of the ANN error function. Using the Hessian provides additional curvature information of the error function and thus makes these methods more accurate, i.e. the removal of an ANN weight is done in an optimized manner. These methods, such as Optimal Brain Surgeon (OBS) and its variants are the focus of this study. They have several advantages that will be elaborated through computational experimental results.

**Hessian-based Pruning**

The Hessian-based approach to ANN pruning provides possibility to generate simpler structure with the least sacrifice in accuracy. In this section, we first explain the original OBS formulation. We also explain a modification we suggest, which makes use of the faster and fewer Hessian inverse approximations.

**Hessian-based Pruning Formulation**

The main algorithms in this category are Optimal Brain Surgeon (OBS) and Optimal Brain Damage (OBD).

Optimal Brain Damage (OBD) (LeCun et al., 1989) uses the second-order derivative of the objective function with respect to the networks’ weights. OBD avoids the complexity of computing the Hessian by assuming that the Hessian matrix is diagonal, which is not always the case and may lead to OBD removing wrong weights as argued in (Hassibi et al., 1993) . Furthermore such an assumption does not always lead to high accuracy.

**Optimal Brain Surgeon (OBS)**

This method is a Hessian-based pruning that utilizes the second-order derivative of the objective function (Hassibi et al., 1993) . In contrast to OBD, it does not assume diagonal Hessian and does not require retraining unless the error increases by a large amount. OBS can update the weights to reflect the network new changes after each weight removal. Additionally, OBS provides Hessian inverse approximation. This method claims superiority over simple pruning algorithms such as the one based on the magnitude of weights and OBD, as it is more likely to prune the correct weights, which yields the least increase in the networks error *E*. The OBS formulation of the change in *E* obtained from the second term of Taylor series of approximation is given by

$∆E= \frac{1}{2} ∆w^{T}H∆w $ (1)

The goal is to eliminate the *i*th weight $w\_{i}$ to minimize the increase in error given by (1). Therefore, eliminating $w\_{i}$ can be achieved by adding a constraint$ w\_{i}+ ∆w\_{i}=0$, which is the same as ( $w+ ∆w)^{T}e\_{i}=0$, where $e\_{i}$ is the unit vector corresponding to weight $w\_{i}$*.*

$min\frac{1}{2} ∆w^{T}H∆w$ (2)

$subject to ( w+ ∆w)^{T}e\_{i}=0$ (3)

Solving this problem by Lagrange’s method, the change in error is given by

$∆E\_{i}= \frac{1}{2}\frac{w\_{i}^{2}}{[H^{-1}]\_{ii}}$ (4)

and the optimal weight change is given by

$∆w=-\frac{w\_{i}}{[H^{-1}]\_{ii}} H^{-1} e\_{i}$ (5)

Eq. 5 is used to update the weights without the need for ANN retraining unless the error grows by a large amount. This is one of the most attractive features of OBS. Overall, OBS finds the weight that causes the overall least increase in the error function, removes it and updates the remaining weights according to Eq. 5.

**Multiple weights OBS (MWOBS)**

As it was shown in the standard OBS that the time complexity to remove only one weight requires $O (pw^{2}) $operations in the binary output unit case where the number *k* of output units is *k* = 1. In this variant we explore and implement the possibility of removing multiple weights instead of one with same complexity of removing one weight. The error increase $∆E\_{i}$ is evaluated individually for all network weights using original OBS Eq. 4$. $ We then choose $n$ weights that resulted in the least error increase. This simplifying assumption would not be as accurate as evaluating every combination of weights as described in the general formulation of pruning *n* weights using OBS in (Stahlberger & Riedmiller, 1997). However, we take advantage of the original OBS error equation to evaluate individual weights, sort them and then choose the *n* weights that result in the minimum error increase $∆E\_{i}$. Removing *n* weights requires computing the Hessian approximation once for *n* weights instead of one for each weight as in the standard OBS.

REFERENCES

Hassibi, B., Stork, D. G., & Wolff, G. J. (1993). *Optimal brain surgeon and general network pruning.* Paper presented at the Neural Networks, 1993., IEEE International Conference on.

Karnin, E. D. (1990). A simple procedure for pruning back-propagation trained neural networks. *Neural Networks, IEEE Transactions on, 1*(2), 239-242.

LeCun, Y., Denker, J. S., Solla, S. A., Howard, R. E., & Jackel, L. D. (1989). *Optimal brain damage.* Paper presented at the NIPs.

Mozer, M. C., & Smolensky, P. (1989). *Skeletonization: A technique for trimming the fat from a network via relevance assessment.* Paper presented at the Advances in neural information processing systems.

Stahlberger, A., & Riedmiller, M. (1997). Fast network pruning and feature extraction by using the unit-OBS algorithm. *Advances in neural information processing systems*, 655-661.