

# Selection of training samples for learning with hints

Jouko Lampinen, Paula Litkey and Harri Hakkarainen  
Laboratory of Computational Engineering, Helsinki University of Technology  
P.O.Box 9400, FIN-02015 ESPOO, FINLAND

## Abstract

*Training with hints is a powerful method for incorporating almost any type of prior knowledge into neural network models. In this contribution we demonstrate how the hints can be constructed from numerical approximation of the regularization cost function, and discuss the problem of selecting the hint samples. We give a simple algorithm for placing the hint samples in such regions in the input space where the hint error is large, and for selecting the minimum sufficient set of hint samples by removing the correlated samples.*

## 1 Introduction

Advanced regularization of neural networks or other non-linear models with background knowledge requires error measures that compare the properties of the network model  $y(\mathbf{x}; \mathbf{w})$  to the prior assumptions. Such error measures depend on the input  $\mathbf{x}$  where the error is evaluated, causing extra complexity over simple measures that depend only on the weights  $\mathbf{w}$  (as the weight decay).

Examples of this kind of regularization include e.g., smoothness priors, such as in Tikhonov regularization [2], and other criteria based on model derivatives. In the tangent prop [3] the tangent space of the model is rotated so that the model becomes invariant with respect to selected distortion operations. Training by fuzzy rules [4] is based on adjusting the model partial derivatives to obey set of expert given fuzzy-like rules.

A very powerful method for incorporating almost any type of prior knowledge into a neural network (or other non-linear statistical model) is *training by hints*, introduced by Abu-Mostafa [1]. The network performance is tested with artificial samples and undesired performance is converted to error signals for training. The technique allows use of the same algorithms and program code for the gradients of the data data error and the prior term. Although very flexible technique, training by hints have had rather limited application in neural network literature. An expert can easily figure out the hint samples but there are no straightforward procedures for constructing the hints systematically, which may have slowed down broader use of the technique.

In this contribution we show, by an example, how a regularization cost function can be converted to hints by a systematic procedure, and discuss the issue of selecting the hint samples. We provide a simple algorithm for placing the hint samples in such regions in the input space where the hint error is large, and for selecting the minimum sufficient set of hint samples by removing the correlated samples.

## 2 Systematic procedure for constructing the hints

Consider a regression problem of fitting a function  $y(\mathbf{x}; \mathbf{w})$  to samples  $\{\mathbf{x}^n, \mathbf{t}^n\}$ . In the following vectors are indicated by bold font, element of the vector by subscript, and index of a vector in a set of vectors by superscript. In regularized model fitting, or MAP estimation of the model parameters, the minimized error is of the form

$$E = \frac{1}{2} \sum_n \|\mathbf{y}^n - \mathbf{t}^n\|^2 + \alpha\Omega, \quad (1)$$

where the first term is the squared sum of residuals, corresponding to Gaussian noise model with constant variance, and the second term is the regularization cost function, or negative log-prior in MAP estimation, that penalizes models with low prior probability. The form of  $\Omega$  depends on the prior, and in, e.g., Tikhonov regularization it consists of integrals of squared derivatives [2]

$$\Omega = \frac{1}{2} \sum_{r=0}^R \int h_r(x) \left( \frac{d^r y}{dx^r} \right)^2 dx, \quad (2)$$

where  $h_r$  is a weighting function for the  $r^{th}$  order derivative.

In the hint approach the cost function  $\Omega$  must be stated in terms of hint samples  $\{\mathbf{x}_h, \mathbf{t}_h\}$  such that minimizing the residuals of the hint samples is equal to minimizing the cost function  $\Omega$ . In this section we show how to construct a hint samples for constraining the model derivative. Similar procedure can be used for many types of priors, such as higher derivatives or invariances.

Let the prior be that the slope of the  $k^{th}$  element of the target function with respect to the  $i^{th}$  input is about  $L$ . In the following the slope, or Jacobian of the model, is denoted by

$\frac{\partial \mathbf{y}_k}{\partial \mathbf{x}_i} |_{\mathbf{x}} = J_{ki}(\mathbf{x})$  and for an input sample  $\mathbf{x}^m$  shortly by  $J_{ki}^m$ . If we assume Gaussian distribution for the slope around  $L$ , the negative log-prior is

$$\Omega = \frac{1}{2} \sum_m (J_{ki}^m - L)^2,$$

where  $\mathbf{x}_m$  are sampled from the input probability distribution, producing a Monte Carlo approximation of the integral over  $p(\mathbf{x})$  in (2).

Next we approximate the Jacobian numerically by symmetric difference

$$J_{ki}(\mathbf{x}) \approx \frac{\mathbf{y}_k(\mathbf{x} + \Delta) - \mathbf{y}_k(\mathbf{x} - \Delta)}{2\Delta} = \frac{y^+ - y^-}{2\Delta},$$

where  $y^+$  and  $y^-$  denote the model outputs  $\mathbf{y}_k$  in the hint samples  $\mathbf{x} \pm \Delta$ , and  $\Delta$  is a zero vector with the difference in the element  $i$ . The index  $k$  is dropped for clarity, as the hint samples are defined only for the output  $\mathbf{y}_k$ . The gradient for one sample with respect to parameter  $\mathbf{w}_j$  is

$$\frac{\partial \Omega^m}{\partial \mathbf{w}_j} = (J_{ki} - L) \frac{\partial J_{ki}}{\partial \mathbf{w}_j},$$

where the gradient of the Jacobian is

$$\frac{\partial J_{ki}}{\partial \mathbf{w}_j} = \frac{1}{2\Delta} \left( \frac{\partial y^+}{\partial \mathbf{w}_j} - \frac{\partial y^-}{\partial \mathbf{w}_j} \right),$$

where  $\frac{\partial y^+}{\partial \mathbf{w}_j}$  denotes the gradient of  $\mathbf{y}$  with respect to  $\mathbf{w}_j$  evaluated at  $y^+$ . Denote the mean of the hint sample outputs by  $\bar{y} = (y^+ + y^-)/2$ . The gradient can be written as

$$\begin{aligned} \frac{\partial \Omega^m}{\partial \mathbf{w}_j} &= \left( \frac{y^+ - y^-}{2\Delta} - L \right) \frac{1}{2\Delta} \left( \frac{\partial y^+}{\partial \mathbf{w}_j} - \frac{\partial y^-}{\partial \mathbf{w}_j} \right) \\ &= \frac{1}{2\Delta^2} ((y^+ - \bar{y}) - L\Delta) \frac{\partial y^+}{\partial \mathbf{w}_j} \\ &\quad - \frac{1}{2\Delta^2} ((\bar{y} - y^-) - L\Delta) \frac{\partial y^-}{\partial \mathbf{w}_j} \\ &= \frac{1}{2\Delta^2} (y^+ - (\bar{y} + L\Delta)) \frac{\partial y^+}{\partial \mathbf{w}_j} + \\ &\quad \frac{1}{2\Delta^2} (y^- - (\bar{y} - L\Delta)) \frac{\partial y^-}{\partial \mathbf{w}_j} \end{aligned}$$

If we write the virtual targets for the hint samples as

$$\begin{aligned} t^+ &= \bar{y} + L\Delta \\ t^- &= \bar{y} - L\Delta \end{aligned} \quad (3)$$

then the cost function gradient can be computed as

$$\frac{\partial \Omega^m}{\partial \mathbf{w}_j} = \frac{1}{2\Delta^2} (y^+ - t^+) \frac{\partial y^+}{\partial \mathbf{w}_j} + \frac{1}{2\Delta^2} (y^- - t^-) \frac{\partial y^-}{\partial \mathbf{w}_j}, \quad (4)$$

which has the same form as the data error gradient

$$\frac{\partial E_{data}^n}{\partial \mathbf{w}_j} = (y^n - t^n) \frac{\partial y^n}{\partial \mathbf{w}_j}.$$

Thus we have to recompute the virtual targets for the hint samples by (3) after the forward pass of the hint samples, and then the same algorithms and program code can be used to compute the gradient of the data error and prior error with respect to the network weights, and train the network with any minimization technique. With MLP and backpropagation algorithm for the gradient  $\frac{\partial \mathbf{y}}{\partial \mathbf{w}}$  this is especially convenient, compared to the closed form backpropagation formulae for the gradient of the Jacobian with respect to network parameters, as derived in [4].

These virtual targets can be easily constructed heuristically: the slope of the function according to the hint is  $L$ . With respect to a point  $\{\mathbf{x}^h, \mathbf{y}_k(\mathbf{x}^h)\}$  the right hint point should be at  $\{\mathbf{x}^h + \Delta, \mathbf{y}_k(\mathbf{x}^h) + L\Delta\}$  and the left hint point analogically at  $\{\mathbf{x}^h - \Delta, \mathbf{y}_k(\mathbf{x}^h) - L\Delta\}$  so that the hint turns the slope towards to prior assumption but causes no change to the output value of the model. Note that a typical derivative hint is the monotonicity, e.g.,  $\frac{\partial \mathbf{y}_k}{\partial \mathbf{x}_i} \geq 0$  (see [5, 6]), which is implemented by setting the hint error to zero if the model is increasing in the hint sample and  $L = 0$ , or  $t^+ = t^- = \bar{y}$  otherwise.

### 3 Selection of the hint samples

Typical priors, or regularization terms, in neural networks depend only on the model parameters, such as the 'weight decay' in which  $\Omega = \sum_j w_j^2$ , corresponding to zero mean Gaussian prior for the parameters. Note that for linear model it also corresponds to simple Tikhonov regularization, or zero mean Gaussian prior for the first derivatives of the model.

An additional complication in hint training, and other similar regularization methods, is the selection of the locations where the cost function is evaluated. Typically the hint error is minimized over the probability distribution of input data, implemented by selecting the data samples as hint samples [1, 2]. However, the need for the hints, as for any prior knowledge, is large when the number of data samples is small. Then evaluating the hints in the few training samples may produce model that contradicts the hints in between the samples or just outside the domain of the samples. Monotonicity hint, for example, may lead to a fitting that is increasing in each hint sample, but has negative slope between the samples as demonstrated in the next section.

For high dimensional problems this cannot be alleviated by adding new samples in between the training samples as the number of hint samples would grow exponentially. A possible approach is active learning, i.e., selection of the training

samples according to some criteria that determine the most informative samples. In this section we give a simple method for selecting the hint samples according to their importance, so that each hint sample enforces the hint in as large volume of the input space as possible.

Let  $e(\mathbf{x}^m; \mathbf{w})$  denote the error for hint sample at  $\mathbf{x}^m$  and  $\mathbf{g}^m$  denote the gradient of the hint error with respect to the model parameters,  $\mathbf{g}^m = \nabla_{\mathbf{w}} e(\mathbf{x}^m)$ . In gradient based minimization of the hint error the direction of the gradient  $\mathbf{g}^m$  determines the direction of change in the parameters, and thus hint samples that produce gradients in same direction are redundant. To measure the importance of the hint sample we compute the total decrease of the hint error due to small change of parameters in the gradient direction of the sample. This is also equal to summed directed gradient of all hint samples in the gradient direction of the evaluated sample. The effect of hint sample  $m$  on the error of hint sample  $k$  is  $\Delta e = \sum_j \frac{\partial e^k}{\partial w_j} \frac{\partial e^m}{\partial w_j} = (\mathbf{g}^k)^T (\mathbf{g}^m)$  which is the inner product of the gradients of the hint errors in these two samples. For all hint samples we obtain a correlation matrix with elements  $C_{km} = (\mathbf{g}^k)^T (\mathbf{g}^m)$ .

The importance of hint sample  $m$  can be measured by the row sum of the correlation matrix  $C$ , which gives the total error reduction in all hint samples due to the sample  $m$ . After selecting the sample with the maximum row sum, the effect of that sample need to be removed from the correlation matrix. We use the orthogonal component of the remaining samples on the already selected samples as a measure of novelty of the remaining samples. The updated correlation matrix for selecting the next sample can be computed as

$$C_{kj}^p = C_{kj} - (\mathbf{u}_s^T \mathbf{g}^j)(\mathbf{u}_s^T \mathbf{g}^k),$$

where  $\mathbf{u}_s$  is a unit vector in the direction of the selected sample gradient. After selecting a given number of samples, or until the remaining row sums in  $C$  are below some threshold, we have a set of hint samples, such that weighting the errors in the hint samples by the corresponding row sums gives a rough approximation to the hint error gradient due to all the hint samples in  $C$ .

We use simple stochastic search for the most informative hint samples: we add number of candidate samples randomly in the vicinity of current set of hint samples and training samples and then select the optimal set of the samples. This way the hint samples can spread also to regions where there is no training data, increasing the generalization ability of the solution.

## 4 Numerical experiments

In this section we present some numerical experiments that show the effect of selecting the training samples in hint train-

ing. Fig. 2 shows an example of learning a simple surface shown in Fig. 1 with monotonicity hint for one variable. In the upper row the hint error was evaluated in a dense grid of  $21 \times 21$  samples, but still the model can adjust the illegal performance to occur between the hint samples. Note that additional regularization, such as weight decay, would indirectly constrain the second derivative, making fast changes in the first derivatives costly. However, the hints or similar regularization methods are particularly useful when the more simple methods are not applicable (see [4]), and the interference of other regularization with the hints is not straightforward to control. When the hint samples are allowed to track the regions where the hint error is emerging, very few hint samples are sufficient to suppress the undesired performance, as shown in the bottom row.

Fig. 3 shows an example of reducing the hint gradient correlation matrix during the selection of the samples. Table 1 shows the effect of the hint selection methods to the generalization ability of the model.

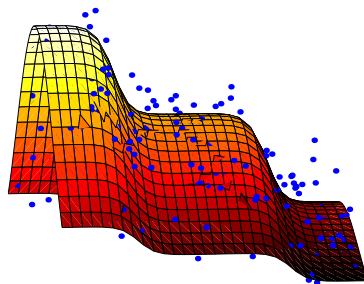


Figure 1: The target function and noisy training data used in Fig. 2

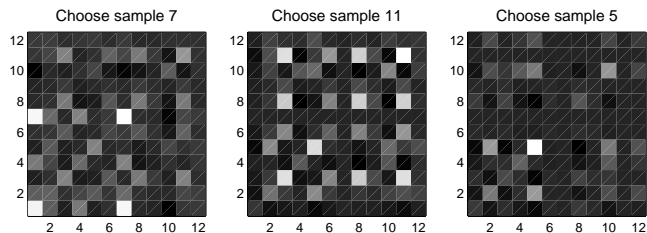


Figure 3: The evolution of the correlation matrix of the hint error gradients during the selection of hint samples in the proposed procedure. Note how the gradients of the samples 1 and 7 have high correlation, which is removed after selection of sample 7.

## 5 Conclusion

We have demonstrated that in hint training it is beneficial to select the samples where the error function is evaluated in such a way that all regions where the error occurs are accumulated to the minimized cost function. This decreases the

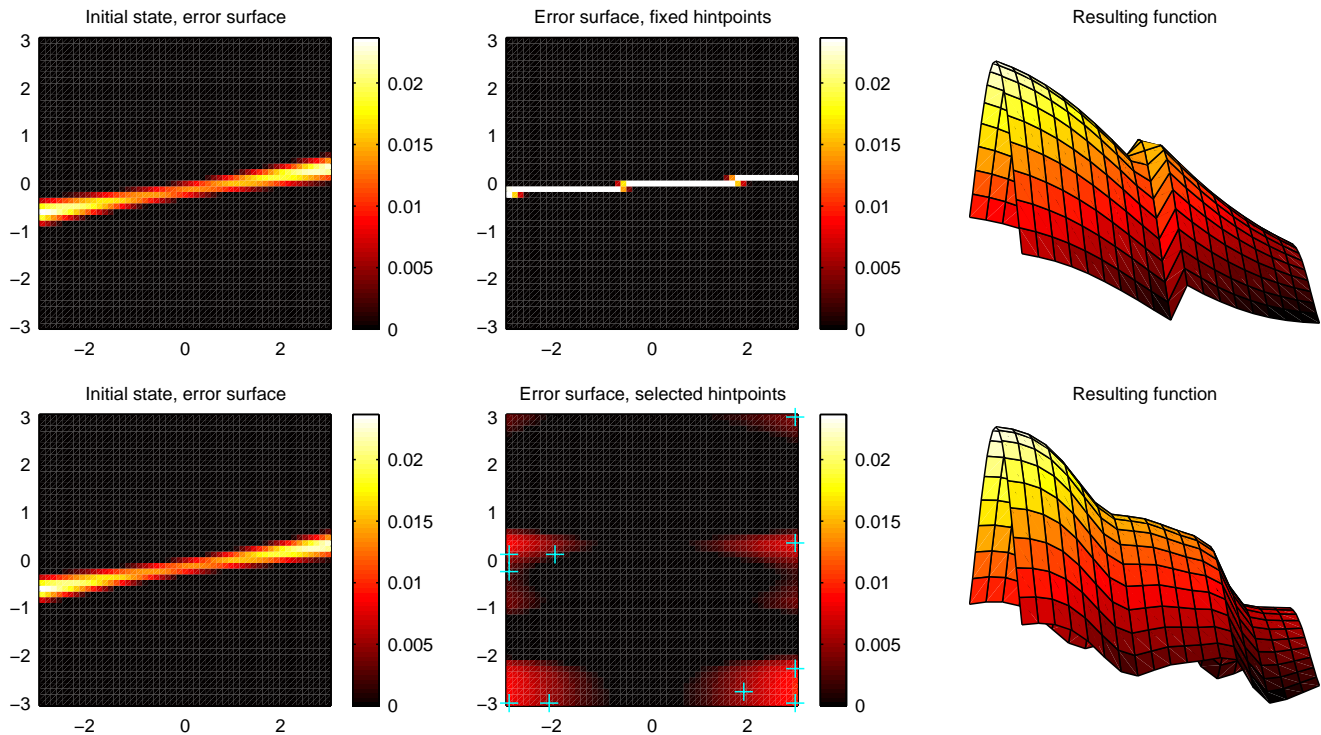


Figure 2: The difference of fixed or adaptive hint samples. In the upper row large number of monotonicity hint samples were used (a  $21 \times 21$  grid), but still the model produced a steep step, aligned to be between the hint samples, in order to fit the noisy data samples better. The left and middle images show the hint error in the 2D coordinate grid, and the right figure shows the resulting function estimate. The lower row show the result when 10 hint samples were reselected after each 200 iterations with RPROP minimization algorithm. The samples efficiently track and suppress the emerging hint error, so that the model correctly identifies the steps in the target function.

Hint samples	Error
No hints	10.2
Hints in data samples	6.2
Hint samples selected by the proposed method	3.6

Table 1: Sum squared errors from the true target function in a very dense grid with various hint sample selection methods.

computational load of hint training as the number of hint samples can be reduced, and especially it allows the hint samples to concentrate to erroneously modelled regions even if there were no training samples.

A practical problem in this type of regularization is that the error minimization becomes much harder than in standard regularization: the hints limit the search space so that bad initial states often lead to bad local minima. Global optimization methods, with tunneling through the barriers generated by the hints may prove useful, or the weight of the hint error ( $\alpha$  in (1)) need to be adjusted during the training, which seems to be difficult to automatize.

## References

- [1] Abu-Mostafa, Y. S. (1993). A method for learning from hints. In S. J. Hanson, J. D. Cowan & C. L. Giles, eds., *Advances in Neural Information Processing Systems*, vol. 5, pp. 73–80. Morgan Kaufmann, San Mateo, CA.
- [2] Bishop, C. M. (1993). Curvature-driven smoothing: A learning algorithm for feed-forward networks. *IEEE Transactions on Neural Networks*, 4(5):pp. 882–884.
- [3] Krogh, A. & Hertz, J. A. (1992). Tangent prop - a formalism for specifying selected invariances in an adaptive network. In P. Y. S. B. Victorri, Y. LeCun & J. Denker, eds., *Advances in Neural Information Processing Systems*, vol. 4. Morgan Kaufmann Publishers, San Mateo, CA.
- [4] Lampinen, J. & Selonen, A. (1997). Using background knowledge in multilayer perceptron learning. In M. Frydrych, J. Parkkinen & A. Visa, eds., *Proc. of The 10th Scandinavian Conference on Image Analysis*, vol. 2, pp. 545–549.
- [5] Selonen, A. & Lampinen, J. (1997). Experiments on regularizing MLP models with background knowledge. In W. G. et.al., ed., *Artificial Neural Networks - ICANN'97*, vol. 1327 of *Lecture Notes in Computer Science*, pp. 367–372. Springer.
- [6] Sill, J. & Abu-Mostafa, Y. S. (1997). Monotonicity hints. In M. C. Mozer, M. I. Jordan & T. Petsche, eds., *Advances in Neural Information Processing Systems*, vol. 9, pp. 634–640. MIT Press, Cambridge, MA.