REGULARISATION OF MIXTURE DENSITY NETWORKS

Lars U. Hjorth and Ian. T. Nabney

Neural Computing Research Group, Aston University
Birmingham B4 7ET, UK
hjorthl@aston.ac.uk

ABSTRACT

Mixture Density Networks (*MDN*s) are a well-established method for modelling complex multi-valued functions where regression methods (such as *MLP*s) fail. In this paper we develop a Bayesian regularisation method for *MDN*s by an extension of the evidence procedure. The method is tested on two data sets and compared with early stopping.

1 Introduction

For prediction problems where the output is a multi-valued function of the inputs, a regression approach, (for example using multi layer perceptron (*MLP*) networks) fails because the mean of the conditional distribution is not a good description of the data. A synthetic example is shown in Figure 2; the prediction is very poor for x values in the range $[0.3, 0.7]$ where the mapping is multi-valued (*i.e.* the conditional distribution of t given x is multi-modal). In practical applications this feature often arises in inverse problems (see Section 3.2) and also occurs in hysteresis loops. To make useful predictions for such data sets more complex models are needed. One alternative is to use the Mixture Density Network (*MDN*) [1], which models the conditional distribution with a mixture of Gaussians where the parameters are input dependent.

For *MLP* networks there are well known Bayesian regularisation methods available, for example the evidence framework, that let us control the complexity of the model in a principled way. Previous workers [2, 10] extended this approach to networks that model both the conditional mean and variance (spherical in the first case and a full covariance in the latter). These networks cannot cope with multi-modal data. Husmeier [4] developed a theory of Bayesian regularisation for a special case of *MDN*, which is discussed in more detail in Section 2.1. This paper elaborates the evidence framework for a special case of the *MDN* and evaluates the method on two data sets.

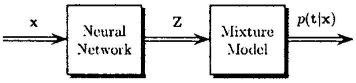

Figure 1: A block diagram showing the structure of an *MDN*. When a pattern $\mathbf{x}$ is presented to the network, a parameter vector $\mathbf{Z}$ is generated from the outputs of the network, which in turn is used as input to the mixture model to generate the conditional probability, $p(\mathbf{t}|\mathbf{x})$.

1.1 Mixture Density Networks

A popular method for modelling *unconditional* densities is a mixture model. In an *MDN* the *conditional* density of the targets is modelled by making the parameters of a mixture model input-dependent: the mapping from inputs to parameters is accomplished using a neural network (either *MLP* or *RBF*). This is illustrated in Figure 1.

We model the conditional density, $p(\mathbf{t}|\mathbf{x})$, by

$$p(\mathbf{t}|\mathbf{x}) = \sum_{i=1}^{M} \eta_i(\mathbf{x})\phi_i(\mathbf{t}|\mathbf{x}), \qquad (1)$$

for a mixture of M kernels, where η_i is the *mixing coefficient* for kernel i. In this paper the kernels, ϕ_i, are chosen to be spherical Gaussian density functions, which are defined by

$$\phi_i(\mathbf{t}|\mathbf{x}) = \frac{1}{(2\pi)^{c/2}\sigma_i(\mathbf{x})^c} \exp\left(-\frac{||\mathbf{t} - \mu_i(\mathbf{x})||^2}{2\sigma_i(\mathbf{x})^2}\right), \qquad (2)$$

where μ_i and σ_i^2 are the mean and variance respectively for each kernel and $\mathbf{t} \in \mathcal{R}^c$.

Considering a data set $\mathcal{D} = \{\mathbf{x}^n, \mathbf{t}^n\}$, where n is an index running over all patterns, the likelihood for the model is

$$\mathcal{L} = \prod_{n=1}^{N} p(\mathbf{t}^n|\mathbf{x}^n). \qquad (3)$$

There are three different types of mixture model parameters that need to be determined by the network: the means, the variances, and the mixing coefficients. We write $\mathbf{z}$ for the outputs of the neural network, which are assumed to be a linear combination of the hidden unit activations. The means are modelled as $\mu_{ik} = z^{\mu}_{ik}$ for $k = 1, 2, \ldots, c$. We have to ensure that the variance remains positive, which can be done with an exponential, $\sigma_i = \exp(z^{\sigma}_i)$. Finally, we need to ensure that the mixing coefficients lie between zero and one and sum to one, which can be achieved by applying the 'softmax' transformation to the corresponding network outputs.

$$\eta_i = \frac{\exp(z^{\sigma}_i)}{\sum_{i'=1}^{M} \exp(z^{\sigma}_i)}. \qquad (4)$$

This treatment of the different parameters ensures that the model output can always be interpreted as a probability density.

The weights of the network can now be found by non-linear minimisation of the negative logarithm of (3), which is

$$E(\mathbf{w}; \mathcal{D}) = -\sum_{n=1}^{N} \ln \sum_{i=1}^{M} \eta(\mathbf{x}^n)\phi_i(\mathbf{t}^n|\mathbf{x}^n), \qquad (5)$$

once the derivatives of $E(\mathbf{w}; \mathcal{D})$ with respect to the network weights have been calculated (see [1]).

2 Regularisation of Mixture Density Networks

Maximum likelihood estimation is always prone to over-fitting. In earlier work with *MDN*s 'early stopping' (training until validation error increases) was often used as a method for avoiding over-fitting. However, for *MLP*s Bayesian regularisation [5] has proved effective, so we have extended this approach to *MDN*s.

For our initial work, we have modified the model reviewed in Section 1.1. The mean and variance of the kernels are fixed and only the mixing coefficients are found by non-linear optimisation. This restricts the flexibility of the model but it can still, in theory, model any distribution if there is a sufficient number of kernels. For more efficient training, we have used an *RBF* for the neural network. These modifications simplify the analysis but they also give us a new problem; how should the fixed parameters be initialised?

In [4] a similar regularisation scheme was developed for a different form of *MDN*. In this model the mixing coefficients in the mixture model were fixed and the means and variances were adjusted. As a further simplification, the *MLP* used for modelling the mixture model parameters had fixed input layer weights (sampled from a Gaussian distribution whose variance was a hyperparameter). This model is reasonable for conditional distributions where the number of modes (and the distribution shape) do not change significantly for different inputs. Our approach is driven by the requirement for a model where these properties can change dramatically over the input space.

2.1 Bayesian Regularisation

Regularisation is achieved by applying the evidence procedure [5, 7, 6] to the *MDN* model. The function to minimise is no longer the likelihood function but the *misfit function*, $M(\mathbf{w})$, which is

$$M(\mathbf{w}; \mathcal{D}) = E(\mathbf{w}; \mathcal{D}) + \alpha E_W(\mathbf{w}), \qquad (6)$$

where $E(\mathbf{w}; \mathcal{D})$ is the negative log likelihood of our model and α is a regularisation parameter. Here, E_W denotes a Gaussian regulariser for the k weights in the neural network,

$$E_W(\mathbf{w}) = \sum_{j=1}^{k} w_j^2/2. \qquad (7)$$

By fixing the mean and variance the kernels in (5) become independent of the input vector $\mathbf{x}^n$ and the negative log likelihood reduces to

$$E(\mathbf{w}; \mathcal{D}) = -\sum_{n=1}^{N} \ln \sum_{i=1}^{M} \eta(\mathbf{x}^n)\phi_i(\mathbf{t}^n). \qquad (8)$$

The gradient of E, for the nth pattern, is

$$\frac{\partial E}{\partial w_{ir}} = \Psi^n_r \left[\eta_i(\mathbf{x}^n) - \pi^n_i\right], \qquad (9)$$

where $\Psi_r^n = \Psi_r(\mathbf{x}^n)$ is the activation of centre r of the *RBF* and r ranges over all centres. π_i^n denotes the posterior distribution for the ith kernel

$$\pi_i^n = \frac{\eta_i(\mathbf{x}^n)\phi_i(\mathbf{t}^n)}{\sum_{j=1}^{M} \eta_j(\mathbf{x}^n)\phi_j(\mathbf{t}^n)} \tag{10}$$

The Hessian for the nth pattern is given by

$$\mathbf{H} = \Psi_r^n \Psi_s^n \Big(\delta_{ij}(\eta_i(\mathbf{x}^n) - \pi_i^n) - \eta_i(\mathbf{x}^n)\eta_j(\mathbf{x}^n) + \pi_i^n \pi_j^n\Big). \tag{11}$$

The weights that minimise (6), $\mathbf{w}_{MP}$, are found by optimisation and are used to calculate the regularisation parameter α, using the following re-estimation formula

$$\alpha = \frac{\gamma}{2E_W(\mathbf{w}_{MP})}, \tag{12}$$

where γ, the *effective number of parameters*, measures how much structure from the data is incorporated into the network parameters or, to rephrase it, how many parameters are well determined by the data. γ is defined by

$$\gamma = \sum_{a=1}^{k} \frac{\lambda_a}{\lambda_a + \alpha}, \tag{13}$$

where λ_a denotes the ath eigenvalue of $\mathbf{H}$. Each term in the sum is a number between 0 and 1; thus γ can range between 0 to k.

This means that evaluating γ requires the calculation of the Hessian of $M(\mathbf{w}_{MP}; \mathcal{D})$. If this is not possible or too computationally expensive, a simpler numerical approximation can be used, for example the following approximate re-estimation formula

$$\alpha = \frac{k}{2E_W}, \tag{14}$$

where k is the number of parameters [5].

We interpret this as no longer distinguishing between well and poorly determined parameters. The advantage with this expression is that because it is very cheap to compute we can afford to update the regularisation frequently whereas in the case where we calculate the effective number of parameters we have to find the balance between updating frequency and the computational cost.

2.2 Initialisation

The fixed parameters in the *MDN* need to be initialised. The means of the Gaussian kernels are placed uniformly within the range of the targets.

The variance should be chosen as small as possible, to enable the model to model small uncertainties (*i.e.* sharp peaks in the posterior distribution). On the other hand it is crucial that by varying the mixing coefficients a peak can be positioned in an arbitrary position in the target range. For a two kernel mixture it is possible to analytically calculate the optimal variance, which is $\sigma^2 = d(3 + \sqrt(6))/12$, where d is the distance between two adjacent kernel means, and this choice is adequate for mixtures with more than two components. [3]

The centres of the *RBF* network were also fixed and set by using K-means followed by a few iterations of the EM algorithm to position the centres roughly in accordance with the distribution of input vectors.

Finally, by making the simplifying assumption that the outputs are independent logistics, the output layer weights can be initialised relatively close to a solution [3]. The near quadratic form of $M(\mathrm{w}; \mathcal{D})$ as a function of the network weights allows us to take full Newton steps during training without risking over-shooting the minima. This leads to a particularly efficient training algorithm for this form of *MDN*, which is important due to the high cost of computing the Hessian.

3 Evaluation

The regularisation method has been evaluated on two data sets. The first data set is the synthetic data introduced by Bishop [1]. The second is a real-life data set, where the task is to infer the wind direction near the sea surface from scatterometer measurements taken by a weather satellite.

3.1 Synthetic Data

3.1.1 Model Training

The data used was generated by the function

$$x = f(t) = t + 0.3\sin(2\pi t) + 0.2\epsilon \tag{15}$$

where ϵ is Gaussian noise with zero mean and unit variance; it is shown in Figure 2.

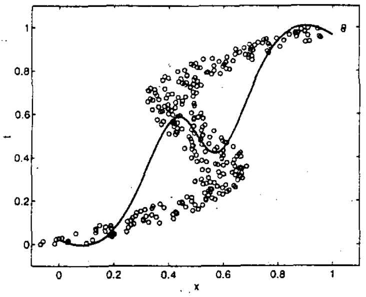

Figure 2: The data was generated by 300 samples from the function $x = t + 0.3\sin(2\pi t) + 0.2\epsilon$ where ϵ is Gaussian noise with zero mean and unit variance. The solid line represents the predictions of an *MLP* network.

300 samples from this function were used to train both the standard *MDN*, as described in Section 1.1, and the regularised *MDN* from Section 2. The validation set consisted of 300 additional samples, and the test set of 900 samples.

For the standard *MDN* with all the mixture model parameters adjustable, we used our knowledge of the mapping and trained the network with 3 kernels, since this is the maximum number of branches in the function, and with adjustable means, the kernels can be moved to lie on the function branches. We let the the number of hidden units vary (5, 10 and 15). The training was done with the quasi-Newton algorithm for up to 2000 iterations with 'early stopping'. This was repeated three times with different random seeds resulting in a total of 9 networks.

For the regularised model, it is interesting to vary, in addition to the hidden units and random seed, the number of kernels (10, 30, 50). In total 27 networks were trained with the quasi-Newton optimiser until the error function converged with a change smaller than 0.0001. The regularisation consisted of one parameter class for all second layer weights, with the biases excluded. The regularisation parameters was updated every 4 iterations with the modified evidence procedure using the update rule (12).

3.1.2 Results

It is clear that an error measure such as root mean squared error is inappropriate for evaluating the model, since the optimal prediction for this measure is the conditional mean of the test data. Instead, the performance of the network was evaluated in terms of its inverse modelling performance. For a given test input $\mathbf{x}^*$, a target value $\mathbf{t}^*$ is made using $p(\mathbf{t}|\mathbf{x})$. This prediction is made in one of two ways: the first is to optimise of $p(\mathbf{t}|\mathbf{x})$ as a function of $\mathbf{t}$ to find the most probable target; the second is to use the mean of the most probable kernel, which is the kernel with the largest mixing coefficient, η_i. As a measure of the fit of our model to the underlying data generator, we then calculate the RMS error between $\mathbf{x}^*$ and $f(\mathbf{t}^*)$ on the test dataset. This is possible since we know the true function and the inverse function is unambiguous (the same technique was used in [1]). For both the unregularised and the regularised models the model with the lowest *NRMS* error on the training set was chosen to represent its class. Table 1 contains the results.

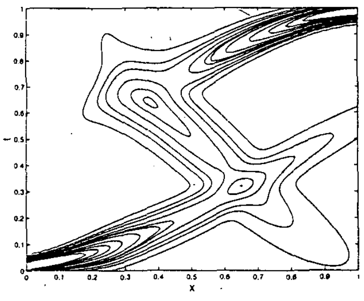

Figure 3: The conditional probability density for the unregularised network.

It was found that the regularisation parameter quickly converges close to its final value. Figure 3 and 4 shows the conditional probability density for the two different models. Most of the probability is well correlated with the density of the training set but some of it is smeared out in other areas of the target space.

The *NRMS* errors of the regularised models are smaller than for the unregularised models, indicating that regularisation can be used to improve the results on this data set. We also found that the regularised models are more consistent; the difference between the same models trained using different random seeds are smaller for the regularised models than the unregularised models.

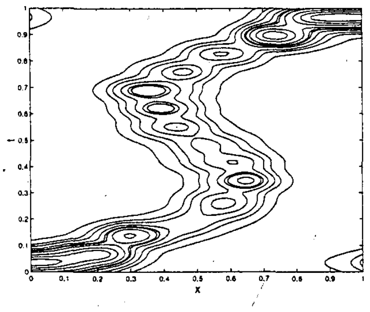

Figure 4: The conditional probability density for the regularised network.

MDN	(unreg)	(reg)
Hidden units/centr.	15	10
Kernels	3	30
Avg. *NRMS* (opt)	0.00458	0.00073
Avg. *NRMS* ($\max \eta$)	0.00548	0.00096
Flops	6×10^8	1×10^9
Norm. $-\ln \mathcal{L}$	-0.98	-1.02

Table 1: *NRMS* test set error averaged over the different runs with the same configuration for the best networks from each category. The first column of *NRMS* errors is when the predictions are based on the highest mode, while the second is based on the largest mixing coefficient (the kernel with most probability mass).

3.2 Radar Scatterometer Data

This is a geophysical application where the final goal is to improve weather predictions. Forecasts are currently made by a numeric weather prediction (*NWP*) model that, given the current 'state' (*i.e.* measurements of relevant variables), estimates weather conditions for the future. One of these state variables is the wind field near the ocean surface. The *NWP* is very computationally intensive to run and generates overly smooth wind fields — hence the interest in using satellite data for more direct measurement. This is done by measuring radar backscatter from the sea surface with a scatterometer on the ERS-1 satellite. Our task is to build the inverse mapping from the scatterometer data back to wind speed and direction that can be used by the *NWP*.

The input data consists of a triplet, σ_n^0, of scatterometer measurements and the incidence angle of the middle beam. The target data comes from a *NWP* model.

3.2.1 Aliases in the Wind Direction

The main problem is that the measurements from three different antennae on the satellite are ambiguous, *i.e.* certain measurements do not have a one-to-one correspondence with a unique wind vector. The problem exists for the wind direction and we often get one or more *aliases*, typically 180 degrees from the true wind direction.

In order to make predictions in direction space we have to take the periodicity of the target variable into account. We choose to do this by using *Circular Normal* kernels [9, 8]. These kernel functions have the form

$$\phi(\theta|\mathbf{x}) = \frac{1}{2\pi I_0(s)} \exp\{s\cos(\theta - \mu)\} \quad (16)$$

where μ is the centre of each kernel which corresponds to the mean and s corresponds to the inverse variance of a traditional Gaussian distribution. $I_0(s)$ is a zeroth order modified Bessel function of the first kind. The parameters s and μ can be initialised in an analogous way to Gaussian kernels. The method presented in Section 2 can now be used to regularise the estimation of the mixing coefficients.

3.2.2 Results

Four different training methods were used for the *MDN* models: 1) evidence with Hessian α-update (12); 2) evidence with Hessian α-update where the estimate of the inverse Hessian from the quasi-Newton algorithm was used[1]; 3) evidence with approximate α-update (14); 4) early stopping. Re-estimation was carried out every 20 iterations for methods 1 and 2, every 5 iterations for method 3, and the validation set performance was measured every 25 iterations for method 4. Table 2 contains some results for the models on a test set of 19,000 patterns. We can also view the convergence of the hyperparameter α for the analytic Hessian update in Figure 5.

4 Discussion

The experiments have shown that our regularisation method is an alternative to 'early

[1] Recall that the quasi-Newton algorithm approximates the inverse Hessian to use in its quadratic update rule. This compromise is much more computationally efficient than the analytic Hessian, but gives better results for α than the approximate update.

Regularisation	Training $-\log\mathcal{L}$	Test $-\log\mathcal{L}$
1	0.5914	0.9204
2	0.7483	0.9939
3	0.4791	0.9622
4	0.5107	0.9334
1	0.5829	0.8287
3	0.5232	0.8239

Table 2: Average negative log likelihood on the wind data set for networks with 18 kernels and 36 centres in the *RBF* network. There were 5000 patterns in the training set for the upper set of results, and 10000 for the lower set of results. See text for description of regularisation methods

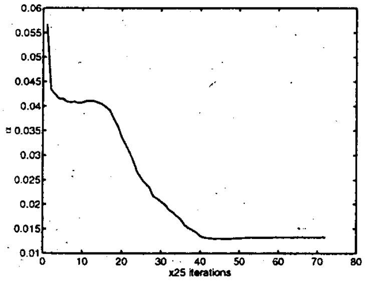

Figure 5: The convergence of α.

stopping' with similar results, but we can make more effective use of the data since we do not have to allocate a validation set. However, regularisation significantly increases the computational cost, especially when using the analytical Hessian, due to the fact that we need a large number of kernels in order to represent the posterior distribution to the same precision as in the case with adaptable kernels. We found that α had a tendency to decay to extremely small values with the less accurate update rules on smaller datasets (this was most marked for the approximate update (14), although it did occur to a lesser extent when using the approximate inverse Hessian calculated by the quasi-Newton optimisation algorithm).

One of the strengths of using fixed kernels is that there is no need to estimate the number of branches of the underlying function. The price we have to pay is that the number of parameters in the model grows very quickly with the number of kernels. The obvious solution is to allow the kernel mean and variance to be adaptive and we are currently working on extending our procedure to this case.

5 References

[1] Christopher M. Bishop. Mixture density networks. Technical Report NCRG/94/004, Department of Computer Science and Applied Mathematics, Aston University, UK, 1994.

[2] Christopher M. Bishop and Cazhaow S. Qazaz. Regression with input-dependent noise: A Bayesian treatment. *Advances in Neural Information Processing Systems*, 1997.

[3] Lars U. Hjorth. Regularisation of mixture density networks. Technical Report NCRG/99/004, Department of Computer Science and Applied Mathematics, Aston University, UK, 1999.

[4] Dirk Husmeier. *Modelling Conditional Probability Densities with Neural Networks.* PhD thesis, King's College London, University of London, 1997.

[5] D. J. C. MacKay. Bayesian interpolation. *Neural Computation*, 4(3):415–447, 1992.

[6] D. J. C. MacKay. The evidence framework applied to classification networks. *Neural Computation*, 4(5):698–714, 1992.

[7] D. J. C. MacKay. A practical Bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472, 1992.

[8] K. V. Mardia. *Statistics of Directional Data.* Academic Press, London, 1972.

[9] I T Nabney and C M Bishop. Modelling conditional probability distributions for periodic variables. In *4th International Conference on Artificial Neural Networks*, pages 177–182. IEE, 1995.

[10] Peter M Williams. Using neural networks to model conditional multivariate densities. *Neural Computation*, 8:843–854, 1996.