To appear in: Tricks of the Trade, Lecture Notes in Computer Science State-of-the-Art Surveys, edited by G. Orr and K.-R. Müller and R. Caruana, Springer Verlag, pp. 299–314, 1998.

Neural Network Classification and Prior Class Probabilities

Steve Lawrence[1], Ian Burns[2], Andrew Back[3], Ah Chung Tsoi[4], C. Lee Giles[1*]
{lawrence,giles}@research.nj.nec.com, ian.burns@oa.com.au, back@zoo.riken.go.jp, Ah_Chung_Tsoi@uow.edu.au

[1] NEC Research Institute[**], 4 Independence Way, Princeton, NJ 08540
[2] Open Access Pty Ltd, Level 2, 7–9 Albany St, St. Leonards, NSW 2065, Australia
[3] Brain Information Processing Group, The Institute of Physical and Chemical Research (RIKEN), Japan
[4] Faculty of Informatics, University of Wollongong, Northfields Ave, Wollongong, NSW 2522, Australia

Abstract. A commonly encountered problem in MLP (multi-layer perceptron) classification problems is related to the prior probabilities of the individual classes – if the number of training examples that correspond to each class varies significantly between the classes, then it may be harder for the network to learn the rarer classes in some cases. Such practical experience does not match theoretical results which show that MLPs approximate Bayesian *a posteriori* probabilities (independent of the prior class probabilities). Our investigation of the problem shows that the difference between the theoretical and practical results lies with the assumptions made in the theory (accurate estimation of Bayesian *a posteriori* probabilities requires the network to be large enough, training to converge to a global minimum, infinite training data, and the *a priori* class probabilities of the test set to be correctly represented in the training set). Specifically, the problem can often be traced to the fact that efficient MLP training mechanisms lead to sub-optimal solutions for most practical problems. In this chapter, we demonstrate the problem, discuss possible methods for alleviating it, and introduce new heuristics that are shown to perform well on a sample ECG classification problem. The heuristics may also be used as a simple means of adjusting for unequal misclassification costs.

1 Introduction

It has been shown theoretically that MLPs approximate Bayesian *a posteriori* probabilities when the desired network outputs are *1 of M* and squared-error or cross-entropy cost functions are used [6, 11, 12, 15, 23, 25, 26, 28, 29, 32]. This result relies on a number of assumptions for accurate estimation: the network

* Lee Giles is also with the Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742.
** `http://www.neci.nj.nec.com/`

must be large enough and training must find a global minimum, infinite training data is required, and the *a priori* class probabilities of the test set must be correctly represented in the training set.

In practice, MLPs have also been shown to accurately estimate Bayesian *a posteriori* probabilities for certain experiments [10]. However, a commonly encountered problem in MLP classification is related to the case when the frequency of the classes in the training set varies significantly[3]. If the number of training examples for each class varies significantly between classes then there may be a bias towards predicting the more common classes [3, 4], leading to worse classification performance for the rarer classes. In [5] it was observed that classes with low *a priori* probability in a speech application were "ignored" (no samples were classified as these classes after training). Such problems indicate that either the estimation of Bayesian *a posteriori* probabilities is inaccurate, or that such estimation may not be desired (e.g. due to varying misclassification costs (this is explained further in section 4)). Bourlard and Morgan [7] have demonstrated inaccurate estimation of Bayesian *a posteriori* probabilities in speech recognition. This chapter discusses how the problem may occur along with methods of dealing with the problem.

2 The Trick

This section describes the tricks for alleviating the aforementioned problem. Motivation for their use and experimental results are provided in the following sections. The methods all consider some kind of scaling which is performed on a class by class basis[4].

2.1 Prior Scaling

A method of scaling weight updates on a class by class basis according to the prior class probabilities is proposed in this section. Consider gradient descent weight updates for each pattern: $w^l_{ki}(\mathrm{new}) = w^l_{ki}(\mathrm{old}) + \Delta w^l_{ki}(p)$ where $\Delta w^l_{ki}(p) = -\eta \frac{\partial E(p)}{\partial w^l_{ki}}$, p is the pattern index, and w_{ki} is the weight between neuron k in layer l and neuron i in layer $l-1$. Scaling the weight updates on a pattern by pattern basis is considered such that the total expected update for patterns belonging to each class is equal (i.e. independent of the number of patterns in the class):

$$\left\langle \sum_{p=1}^{N_p} |s_x \Delta w^l_{ki}(p)|_{p_c=x} \right\rangle = c_1 \, , \forall x \in X \tag{1}$$

[3] For the data in general. Others have considered the case of different class probabilities between the training and test sets, e.g. [23].

[4] Anand et al. [2] have also presented an algorithm related to unequal prior class probabilities. However, their algorithm aims only to improve convergence speed. Additionally, their algorithm is only for two class problems and batch update.

where p_c is the target classification of pattern p, c_1 is a constant, s_x is a scaling factor, x ranges over all classes X, $<>$ denotes expectation, and the $p_c = x$ subscript indicates that the sum is only over the patterns in a particular class x. This effectively scales the updates for lower frequency classes so that they are higher – the aim is to account for the fact that lower frequency classes tend to be "ignored" in certain situations. We assume that the expected weight update for individual patterns in each class is equal:

$$\left\langle |\Delta w_{ki}^{l}(p)|_{p_c=x} \right\rangle = c_2, \forall x \in X \tag{2}$$

where c_2 is a constant not related to c_1. The scaling factor required is therefore:

$$s_x = \frac{1}{p_x N_c} \tag{3}$$

where s_x is the scaling factor for all weight updates associated with a pattern belonging to class x, N_c is the number of classes, and p_x is the prior probability of class x.

Scaling as defined above invalidates the Bayesian *a posteriori* probability proofs (for example, scaling a class by two can be compared with duplicating every pattern in the data for that class – causing changes in probability distributions), i.e. there is no reason to expect that the scaling strategy will be optimal. This, and the empirical result that the scaling may improve performance, leads to the hypothesis that there may be a point between no prior scaling and prior scaling as defined above which produces performance better than either of the two extremes. The following scaling rule can be used to select a degree of scaling between the two extremes:

$$s'_x = 1 - c_s + \frac{c_s}{p_x N_c} \tag{4}$$

where $0 \leq c_s \leq 1$ is a constant specifying the amount of prior scaling to use. $c_s = 0$ corresponds to no scaling according to prior probabilities, and $c_s = 1$ corresponds to scaling as above. Prior scaling in this form can be expressed as training with the following alternative cost function[5]:

Definition 1.

$$E = \frac{1}{2} \sum_{k=1}^{N_p} \sum_{j=1}^{N_c} s'_x (d_{kj} - y_{kj})^2 \tag{5}$$

[5] A cost function with similar motivation, the "classification figure-of-merit" (CFM) proposed by Hampshire and Waibel [13], has been suggested as a possible improvement when prior class probabilities vary [3]. In [13], the CFM cost function leads to networks which make different errors to those trained with the MSE criterion, and can therefore be useful for improving performance by combining classifiers trained with the CFM and the MSE. However, networks trained with the CFM criterion do not result in higher classification performance than networks trained with the MSE criterion for the experiments reported in [13].

where the network has one output for each of the N_c classes, N_p is the number of patterns, d is the desired or target output, y is the predicted output, and x is the class of pattern k.

When using prior scaling as defined in this section, the individual s'_x values can be large for classes with low prior probability. This may lead to the requirement of decreasing the learning rate in order to prevent the relatively large weight updates interfering with the gradient descent process. Comparing the use of prior scaling and not using prior scaling then becomes problematic because the optimal learning rate is different for each case. An alternative is to normalize the s'_x values so that the maximum is 1. Another possibility is to present patterns repeatedly to the network instead of scaling weight updates, i.e. for a class with a scaling factor of 2 each pattern would be presented twice. This would have the advantage of reducing the range of weight updates in terms of magnitude, e.g. an update of magnitude x might be repeated twice rather than using a single update of magnitude $2x$. This may allow the use of a higher learning rate, and therefore reduce the number of epochs required. However, a disadvantage of repeating patterns is that the effective training set would be larger, resulting in longer training times for the same number of epochs. Such a technique could be done probabilistically, and this is the subject of the next technique.

2.2 Probabilistic Sampling

Yaeger et al. [33] have proposed a method called *frequency balancing* which is similar to the prior scaling method above. In frequency balancing, Yaeger et al. use all training samples in random order for each training epoch and allow each sample to be presented to the network a random number of times, which may be zero or more and is computed probabilistically. A balancing factor is included, which is analogous to the scaling factor above (c_s).

We introduce a very similar method here called *probabilistic sampling* whereby training patterns are chosen randomly in the following manner: the class is chosen randomly with the probability of choosing each class x, being $(1-c_s)p_x + \frac{c_s}{N_c}$. A training sample is then chosen randomly from among all training samples for the chosen class.

2.3 Post Scaling

Instead of scaling weight updates or altering the effective class frequencies, it is possible to train the network as usual and then scale the outputs of the network after training. For example, the network could be trained as usual and then the outputs scaled according to the prior probabilities in a similar fashion to the prior scaling method (using equation 3 or 4). Experiments with this technique alone show that it is not always as successful as prior scaling of the weight updates. This may be because the estimation of the lower frequency classes can be less accurate than that of the higher frequency classes [24] (the deviations of the network outputs from the true values in regions with a higher number of data points influence the squared error cost function more than the deviations in regions with a lower number of points [23]).

The post scaling technique introduced here can also be used to optimize a given criterion, e.g. the outputs may be scaled so that the probability of predicting each class matches the prior probabilities in the training set as closely as possible. Post scaling to minimize a different criterion is demonstrated in the results section. For the results in this chapter, the minimization is performed using a simple hill-climbing algorithm which adjusts a scaling factor associated with each of the outputs of the network.

2.4 Equalizing Class Membership

A simple method for alleviating difficulty with unequal prior class probabilities is to adjust (e.g. equalize) the number of patterns in each class, either by subsampling [24] (removing patterns from higher frequency classes), or by duplication (of patterns in lower frequency classes)[6]. For subsampling, patterns can be removed randomly, or heuristics may be used to remove patterns in regions of low ambiguity. Subsampling involves a loss of information which can be detrimental. Duplication involves a larger dataset and longer training times for the same number of training epochs (the convergence time may be longer or shorter).

3 Experimental Results

Results on an ECG classification problem are reported in this section after discussing the use of alternative performance measures. Results on a simple artificial problem are also included in the explanation section.

3.1 Performance Measures

When the interclass prior probabilities of the classes vary significantly, then the overall classification error may not be the most appropriate performance criterion. For example, a model may always predict the most common class and still provide relatively high performance. Statistics such as the Sensitivity, Positive Predictivity, and False Positive Rate can provide more meaningful results [1]. These are defined on a class by class basis as follows:

The **Sensitivity** of a class is the proportion of events labelled as that class which are correctly detected. For the two class confusion matrix shown in table 1 the sensitivity of class 1 is $\frac{c_{11}}{c_{11}+c_{12}}$.

The **Positive Predictivity** of a class is the proportion of events which were predicted to be the class and were labelled as that class. For the two class confusion matrix shown in table 1 the positive predictivity of class 1 is $\frac{c_{11}}{c_{11}+c_{21}}$.

[6] The heuristic of adding noise during training [22] could be useful here as with the other techniques in this chapter.

The **False Positive Rate** of a class is the proportion of all patterns for other classes which were incorrectly classified as that class. For the two class confusion matrix shown in table 1 the false positive rate of class 1 is $\frac{c_{21}}{c_{11}+c_{21}}$.

Class	1	2
1	c_{11}	c_{12}
2	c_{21}	c_{22}

Table 1. A sample confusion matrix which is used to illustrate sensitivity, positive predictivity, and false positive rate. Rows correspond to the desired classes and columns correspond to the predicted classes.

No single performance criterion can be labelled as the best for comparing algorithms or models because the best criterion to use is problem dependent. Here, we take the sensitivity as defined above, and create a single performance measure, the mean squared sensitivity error (MSSE). We define the MSSE as follows:

Definition 2.

$$\mathrm{MSSE} = \frac{1}{N_c}\sum_{i=1}^{N_c}(1 - S_i)^2 \tag{6}$$

where N_c = the number of classes and S_i = sensitivity of class i as defined earlier.

Sensitivities range from 0 (no examples of the class correctly classified) to 1 (all examples correctly classified). Thus, a lower MSSE corresponds to better performance. We choose this criterion because each class is given equal importance and the square causes lower individual sensitivities to be penalized more (e.g. for a two class problem, class sensitivities of 100% and 0% produce a higher MSSE than sensitivities of 50% and 50%). Note that this is only one possible criterion, and other criterion could be used in order to reflect different requirements, e.g. specific misclassification costs for each class. The post scaling heuristic can be used with any criterion (and doing so may be simpler than reformulating the neural network training algorithm for the new criterion).

3.2 ECG Classification Problem

This section presents results using the beforementioned techniques on an ECG classification problem. The database used is the MIT-BIH Arrhythmia database [21] – a common publicly available ECG database which contains a large number of ECG records that have been carefully annotated by experts. Detection of the following four beat types is considered: Normal (N), Premature Ventricular Contraction (PVC), Supraventricular Contraction (S), and Fusion (F) [21], i.e. there are four output classes. The four classes are denoted 1 (N), 2 (PVC), 3 (S), and 4 (F). An autoregressive model is calculated for a window of 200 samples centered over the peak of the R-wave of each beat. The inputs are the polar coordinates of each pole in the z-plane, i.e. frequency changes are reflected in the angular variation of the poles and damping is reflected in the magnitude variations. The model order was four corresponding to eight input variables. The prior probability of the classes (according to the training data) is (0.737, 0.191, 0.0529, 0.0196) corresponding to beat types (N, PVC, S, F).

MLPs with 20 hidden units were trained with stochastic backpropagation (update after each pattern) using an initial learning rate of 0.02 which was linearly reduced to zero over the training period of 500,000 updates. We used 5,000 points in each of the training, validation and test sets. The validation set was used for early stopping. The following algorithms were used – a) prior scaling with the degree of scaling, c_s, varied from 0 to 1, b) probabilistic sampling with the degree of scaling, c_s, varied from 0 to 1, c) as per a) and b) with the addition of post scaling, and d) equalizing the number of cases in each class by removing cases in more common classes. The post scaling attempted to minimize the MSSE on the training set[7]. 10 trials were performed for each case.

The median test set MSSE for d) was 0.195. The results for probabilistic sampling and probabilistic sampling plus post scaling are shown with box-whiskers plots[8] in figure 1. For probabilistic sampling, the best scaling results correspond to a degree of scaling in between no scaling and scaling according to the prior probabilities ($c_s \approx 0.8$). When c_s is larger, the sensitivity of class 1 drops significantly and results in higher false positive rates for the other classes. When c_s is lower, the sensitivity of classes 3 and 4 drops significantly. It can be seen that the addition of post scaling appears to almost always improve performance for this problem. The optimal degree of scaling, $c_s \approx 0.8$, is difficult to determine *a priori*. However, it can be seen that the addition of post scaling makes the selection of c_s far less critical ($c_s = 0.3$ to $c_s = 1.0$ result in similar performance). Figure 2 shows confusion matrices (in graphical form). Without scaling ($c_s = 0$),

[7] 400 steps were used for the hill climbing algorithm where each step corresponded to either multiplying or dividing an individual output scale factor by a constant which was reduced linearly over time from 1.5 to 1. The time taken was short compared to the overall training time.

[8] The distribution of results is often not Gaussian and alternative means of presenting results other than the mean and standard deviation can be more informative. Box-whiskers plots show the interquartile range (IQR) with a box and the median as a bar across the box. The whiskers extend from the ends of the box to the minimum and maximum values. The median and the IQR are simple statistics which are not as sensitive to outliers as the mean and the standard deviation [31]. The median is the value in the middle when arranging the distribution in order from the smallest to the largest value. If the data is divided into two equal groups about the median, then the IQR is the difference between the medians of these groups. The IQR contains 50% of the points.

it can be seen that classes 3 & 4 have low sensitivity. With scaling using $c_s = 1$ all classes are now recognized, however the sensitivity of class 1 is worse and the false positive rate of classes 3 & 4 is significantly worse.

The results for prior scaling and prior scaling combined with post scaling were very similar but slightly worse than the results with probabilistic sampling. The prior scaling results are not plotted in order to make the graph easier to follow, however the qualitative results are as follows: for low c_s, prior scaling and probabilistic sampling perform very similarly. However, for high c_s, probabilistic sampling has a clear advantage for this problem. This is perhaps just as expected – the relatively high variation in prior class probabilities leads to a high variation in weight update magnitudes across the classes when using high c_s. Results for all methods can be seen in table 2.

Method	Prior Scaling	Prior Scaling + Post Scaling	Probabilistic Sampling	Probabilistic Sampling + Post Scaling	Equalizing Membership
Average MSSE (for best c_s)	0.10 ($c_s = 0.8$)	0.096 ($c_s = 0.6$)	0.099 ($c_s = 0.8$)	0.089 ($c_s = 0.3$)	0.195
Average MSSE (over all c_s)	0.19	0.10	0.18	0.099	0.195

Table 2. Results for the various methods. We show the average results for the best selection of c_s and also an average across all selections of c_s. Note that selection of the optimal value of c_s is less critical when using post scaling in addition to either the prior scaling or probabilistic sampling methods.

4 Explanation

This section discusses why the techniques presented can be useful, limitations of the techniques, and how they relate to the theoretical result that MLPs approximate Bayesian *a posteriori* probabilities under certain conditions.

4.1 Convergence and Representation Issues

We first list four possible situations:

1. The proofs regarding estimation of Bayesian *a posteriori* probabilities assume networks with an infinite number of hidden nodes in order to obtain accurate approximation. For a given problem, it can be seen that a network which is too small will be unable to estimate the probabilities accurately due to limited resources.

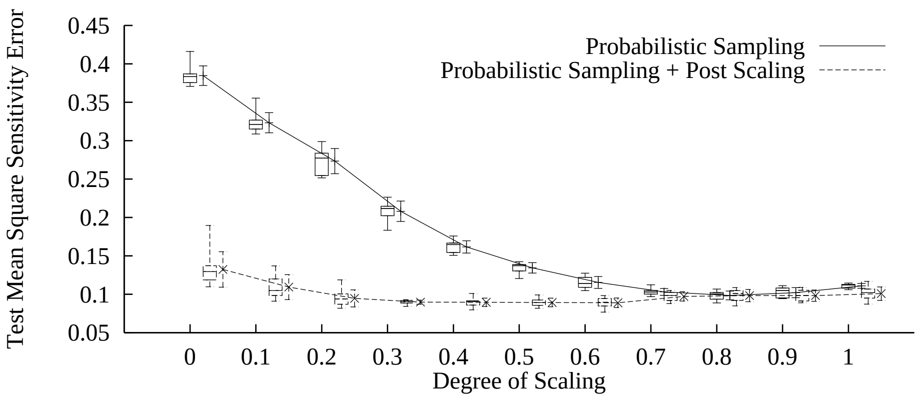

Fig. 1. Box-whiskers plots (on the left in each case) along with the usual mean plus and minus one standard deviation plots (on the right in each case) showing the test set MSSE for probabilistic sampling and for probabilistic sampling plus post scaling. Each result is derived from 10 trials with different starting conditions. The probabilistic sampling plus post scaling case is offset by 0.03 to aid viewing. It can be seen that the selection of the scaling degree for the best performance is not as critical when using the combination of probabilistic sampling and post scaling.

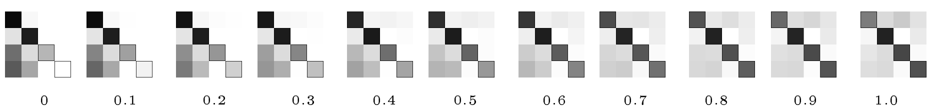

Fig. 2. Confusion matrices for the test set as the degree of prior scaling, c_s, is varied from 0 (left) to 1 (right). The columns correspond to the predicted classes and the rows correspond to the desired classes. The classes are (left to right and top to bottom) N, PVC, S, F. For each desired class, the predicted classes are shaded in proportion to the number of examples which are labelled as the desired class. White indicates no predictions. A general trend can be observed where classes S & F are recognized as normal when $c_s = 0$, and progressively more of the normal class examples are recognized as classes PVC, S, & F as c_s approaches 1.

2. Training an MLP is NP-complete in general and it is well known that practical training algorithms used for MLPs often result in sub-optimal solutions (e.g. due to local minima). Often, a result of attaining a sub-optimal solution is that not all of the network resources are efficiently used. Experiments with a controlled task have indicated that the sub-optimal solutions often have smaller weights on average [17].
3. Weight decay [16] or weight elimination [30] are often used in MLP training and aim to minimize a cost function which penalizes large weights. These techniques tend to result in networks with smaller weights.
4. A commonly recommended technique with MLP classification is to set the training targets away from the bounds of the activation function (e.g. (-0.8, 0.8) instead of (-1, 1) for the tanh activation function) [14].

These four situations can all lead to a bias towards smaller weights, or "smoother" models[9]. The possibility of such a bias is not taken into account by the proofs regarding posterior probabilities, i.e. the difference between theory and practice may, in part, be explained by violation of the assumption that sufficient convergence is obtained.

When a network is biased towards a "smoother" solution, and accurate fitting of the optimal function is not possible, the result may be a tendency to "ignore" lower frequency classes[10], e.g. if a network has the choice of fitting either a high frequency class or a low frequency class then it can provide a lower MSE by fitting the high frequency class[11]. We demonstrate by example.

We generated artificial training data using the following distributions: class 1: $N(-5,1,2)+N(0,1,2)+N(5,1,2)$, class 2: $N(-2.5,0.25,0.5)+N(2.5,0.25,0.5)$, where $N(\mu,\sigma,x)$ is a normal distribution with mean μ, standard deviation σ, and is truncated to lie within $(\mu-x,\mu+x)$. We generated 500 training and test examples from these distributions with the probability of selection for classes (1,2) being (0.9,0.1), i.e. the training and test sets have nine times as many samples of class 1 as they do of class 2. Note that there is no overlap between the classes. Figure 3 shows typical output probability plots for training an MLP with 10 hidden nodes[12] with and without probabilistic sampling. 10 trials were performed in each case with very similar results (see table 3). It can be seen that the network "ignores" class two without the use of probabilistic sampling.

It should be noted that using conjugate gradient training for this simple problem results in relatively accurate estimation of both classes with standard training (alternate parameters with backpropagation may also be successful). Rather than arguing for either backpropagation or conjugate gradient here (neither training algorithm is expected to always find a global minima in general), we simply note that our experience and the experience of others [7, 18, 19, 27] suggests that conjugate gradient is not superior for many problems – i.e. backpropagation works better on one class of problems and conjugate gradient works better on another class. Conjugate gradient resulted in significantly worse performance when tested on the ECG problem. It should be noted that there are many options when implementing a conjugate gradient training algorithm and that poor performance may be attributed to the implementation used. We have used a modified implementation of the algorithm from Fletcher [9].

---

9 In general, smaller weights correspond to smoother functions, however this is not always true. For example, this is not the case when fitting the function $\mathrm{sech}(x)$ using two tanh sigmoids [8] (because $\mathrm{sech}(x) = \lim_{d\to 0}(\tanh(x+d) - \tanh(x))/d$, i.e. the weights become indefinitely large as the approximation improves).

10 In relation to the representational capacity (size of the network), Barnard and Botha [3] have observed that MLP networks have a tendency to guess higher probability classes when a network is too small to approximate the decision boundaries reasonably well.

11 Lyon and Yaeger [20] find that their frequency balancing technique reduces the effect of the prior class probabilities on the network and effectively forces the network to

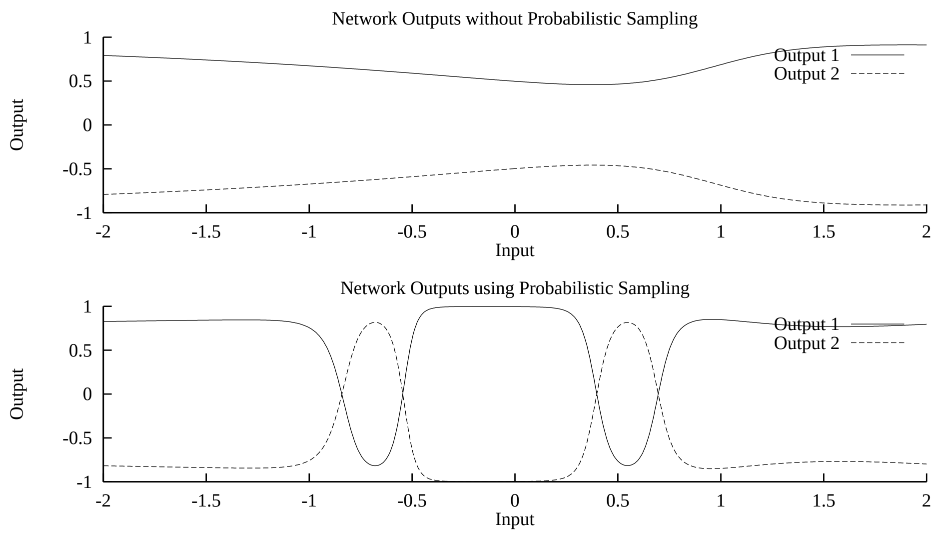

Fig. 3. Network outputs for the artificial problem with (below) and without (above) probabilistic sampling. It can be seen that the network "ignores" the lower frequency class without the use of probabilistic sampling. Note that the input has been normalized.

Classification Error	Mean	Standard Deviation
Standard Training	11.4	0.02
With Probabilistic Sampling	0.8	0.004

Table 3. Mean and standard deviation of the classification error for the artificial problem both with and without the use of probabilistic sampling.

4.2 Overlapping Distributions

Consider figure 4. If classes 1 and 2 have distributions differing only by translation (c_1 and c_2') then the decision threshold between these classes should be chosen at x_1. Equal percentages of each of these classes will be classified as the other class. Now, if the distribution of class 2 is as shown (c_2) then the decision threshold between the classes should be chosen at x_2. In this case, a higher percentage of class 2 will be classified as class 1 than the reverse. If it is desirable to maximize the class by class sensitivity then scaling such that the effective distribution of c_2 is c_2' might be appropriate. Similarly, class 3 (c_3) will be "ignored" without any scaling.

---

allocate more resources to the lower frequency classes.

[12] 500,000 stochastic training updates with backpropagation, initial learning rate 0.02 reduced linearly to zero.

Scaling on a class by class basis may be desired when i) the distribution of samples in the training set does not match the true distribution (e.g. it may be more expensive to collect samples of a particular class)[13], or ii) the distribution of the classes does not represent their relative importance, e.g. in a medical classification problem the cost of misclassifying a diseased case as normal may be much higher than the cost of classifying a normal case as a (possibly) diseased case [24]. The importance of each class may be independent of the class prior probabilities. Note that scaling such that lower frequency classes are made to be artificially more important can be useful when considering a higher level problem. For example, the training data from natural English words and phrases exhibit very non-uniform priors for different characters. Yaeger et al. [33] find that reducing the effect of these priors on the network using frequency balancing improves the performance of the higher level word recognition training.

Observations. a) There is no intrinsic problem if the distributions do not overlap. b) When distributions overlap, it is desirable to preprocess the data in a manner that results in reduced overlap. However, it is often not possible to obtain zero overlap (due to noise, for example).

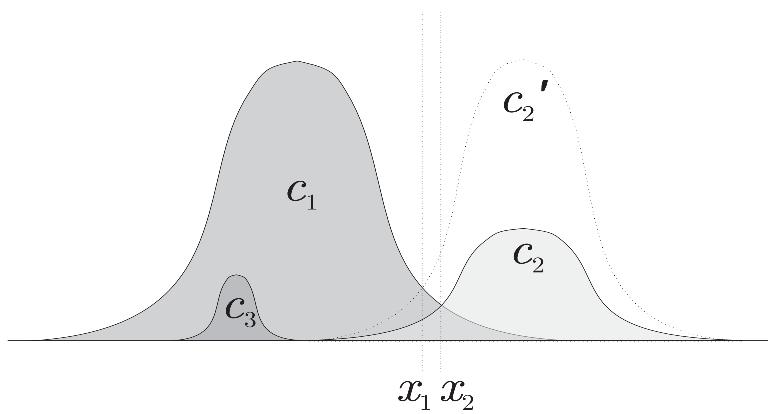

Fig. 4. Overlapping distributions.

4.3 Limitations

We note a couple of limitations with the heuristics considered herein:

1. *Local issues.* The heuristics presented counteract biases in the network, training algorithm and/or training data. There is no reason for these biases to

[13] It may be possible to obtain more accurate estimates of class probabilities using data that has class labels without input information. For example, word frequency information can be obtained from text databases and the frequency of various diseases can be obtained from health statistics [23].

be constant throughout the input space, e.g. scaling may be helpful in one region but detrimental in another.

2. *Nonlinear calibration.* There is no reason for the linear scaling heuristics used here to be optimal (in the sense that they best counteract the biases).

4.4 *A Posteriori* Proofs

Theoretically it is possible to show that the scaling techniques invalidate the *a posteriori* proofs – when performing scaling on a class by class basis the decision thresholds which are used to determine the winning class should be altered accordingly. This indicates another possible use of the prior scaling and probabilistic sampling techniques when the conditions given above do not exist. This use is related to the problem whereby lower frequency classes may be estimated less accurately than higher frequency classes (see section 2.3) – training may be performed with the heuristically altered problem (e.g. so that the class frequencies are effectively equal) and the outputs or decision thresholds can be altered accordingly.

5 Conclusions

In practice, training issues or characteristics of a given classification problem can mean that scaling the predicted class probabilities may improve performance in terms of overall classification error and/or in terms of an alternative criterion. We introduced algorithms which a) scale weight updates on a class by class basis according to the prior class probabilities, b) alter class frequencies probabilistically (very similar to the frequency balancing technique of Yaeger et al. [33]), and c) scale outputs after training in order to maximize a given performance criterion. For an electrocardiogram (ECG) classification problem, we found that the prior scaling, probabilistic sampling, and post scaling techniques provided better performance in comparison to a) no heuristics, and b) subsampling in order to equalize the number of cases in each class. The best performance for prior scaling and probabilistic sampling was obtained with a degree of scaling in between no scaling and scaling according to the prior probabilities. The optimal degree was difficult to determine *a priori*. However, it was found that the using prior scaling or probabilistic sampling in combination with post scaling made the selection of the optimal degree far less critical.

References

1. AAMI. Testing and reporting performance results of ventricular Arrhythmia detection algorithms. Association for the Advancement of Medical Instrumentation, Arlington, VA, 1987. ECAR–1987.
2. Rangachari Anand, Kishan G. Mehrotra, Chilukuri K. Mohan, and Sanjay Ranka. An improved algorithm for neural network classification of imbalanced training sets. *IEEE Transactions on Neural Networks*, 4(6):962–969, November 1993.

3. Etienne Barnard and Elizabeth C. Botha. Back-propagation uses prior information efficiently. *IEEE Transactions on Neural Networks*, 4(5):794–802, September 1993.
4. Etienne Barnard and David Casasent. A comparison between criterion functions for linear classifiers, with an application to neural nets. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(5):1030–1041, 1989.
5. Etienne Barnard, R.A. Cole, and L. Hou. Location and classification of plosive constants using expert knowledge and neural-net classifiers. *Journal of the Acoustical Society of America*, 84 Supp 1:S60, 1988.
6. H.A. Bourlard and N. Morgan. Links between Markov models and multilayer perceptrons. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 1*, volume 1, pages 502–510. Morgan Kaufmann, San Mateo, CA, 1989.
7. H.A. Bourlard and N. Morgan. *Connnectionist Speech Recognition: A Hybrid Approach.* Kluwer Academic Publishers, Boston, MA, 1994.
8. N. Scott Cardell, Wayne Joerding, and Ying Li. Why some feedforward networks cannot learn some polynomials. *Neural Computation*, 6(4):761–766, 1994.
9. R. Fletcher. *Practical Methods of Optimization, Second Edition.* John Wiley & Sons, 1987.
10. S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural Computation*, 4(1):1–58, 1992.
11. H. Gish. A probabilistic approach to the understanding and training of neural network classifiers. In *Proceedings of the IEEE Conference on Acoustics, Speech and Signal Processing*, pages 1361–1364. IEEE Press, 1990.
12. J.B. Hampshire and Barak Pearlmutter. Equivalence proofs for multilayer perceptron classifiers and the Bayesian discriminant function. In D. S. Touretzky, J. L. Elman, T. J. Sejnowski, and G. E. Hinton, editors, *Proceedings of the 1990 Connectionist Models Summer School.* Morgan Kaufmann, San Mateo, CA, 1990.
13. J.B. Hampshire and Alex H. Waibel. A novel objective function for improved phoneme recognition using time delay neural networks. In *International Joint Conference on Neural Networks*, pages 235–241, Washington, DC, June 1989.
14. S. Haykin. *Neural Networks, A Comprehensive Foundation.* Macmillan, New York, NY, 1994.
15. F. Kanaya and S. Miyake. Bayes statistical behavior and valid generalization of pattern classifying neural networks. *IEEE Transactions on Neural Networks*, 2(1):471, 1991.
16. A. Krogh and J.A. Hertz. A simple weight decay can improve generalization. In J.E. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Advances in Neural Information Processing Systems*, volume 4, pages 950–957. Morgan Kaufmann, San Mateo, CA, 1992.
17. Steve Lawrence, C. Lee Giles, and A.C. Tsoi. Lessons in neural network training: Overfitting may be harder than expected. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence, AAAI-97*, pages 540–545. AAAI Press, Menlo Park, California, 1997.
18. Y. Le Cun. Efficient learning and second order methods. Tutorial presented at Neural Information Processing Systems 5, 1993.
19. Y. Le Cun and Yoshua Bengio. Pattern recognition. In Michael A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 711–715. MIT Press, Cambridge, Massachusetts, 1995.
20. R. Lyon and L. Yaeger. On-line hand-printing recognition with neural networks. In *Fifth International Conference on Microelectronics for Neural Networks and Fuzzy Systems*, Lausanne, Switzerland, 1996. IEEE Computer Society Press.

21. MIT-BIH. MIT-BIH Arrhythmia database directory. Technical Report BMEC TR010 (Revised), Massachusetts Institute of Technology and Beth Israel Hospital, 1988.
22. Alan F. Murray and Peter J. Edwards. Enhanced MLP performance and fault tolerance resulting from synaptic weight noise during training. *IEEE Transactions on Neural Networks*, 5(5):792–802, 1994.
23. M.D. Richard and R.P. Lippmann. Neural network classifiers estimate Bayesian *a posteriori* probabilities. *Neural Computation*, 3(4):461–483, 1991.
24. B.D. Ripley. *Pattern Recognition and Neural Networks.* Cambridge University Press, Cambridge, UK, 1996.
25. Raúl Rojas. A short proof of the posterior probability property of classifier neural networks. *Neural Computation*, 8:41–43, 1996.
26. D.W. Ruck, S.K. Rogers, K. Kabrisky, M.E. Oxley, and B.W. Suter. The multilayer perceptron as an approximation to an optimal Bayes estimator. *IEEE Transactions on Neural Networks*, 1(4):296–298, 1990.
27. W. Schiffman, M. Joost, and R. Werner. Optimization of the backpropagation algorithm for training multilayer perceptrons. Technical report, University of Koblenz, 1994.
28. P.A. Shoemaker. A note on least-squares learning procedures and classification by neural network models. *IEEE Transactions on Neural Networks*, 2(1):158–160, 1991.
29. E. Wan. Neural network classification: A Bayesian interpretation. *IEEE Transactions on Neural Networks*, 1(4):303–305, 1990.
30. A.S. Weigend, D.E. Rumelhart, and B.A. Huberman. Generalization by weight-elimination with application to forecasting. In R. P. Lippmann, J.E. Moody, and D. S. Touretzky, editors, *Advances in Neural Information Processing Systems*, volume 3, pages 875–882. Morgan Kaufmann, San Mateo, CA, 1991.
31. N.A. Weiss and M.J. Hassett. *Introductory Statistics.* Addison-Wesley, Reading, Massachusetts, second edition, 1987.
32. H. White. Learning in artificial neural networks: A statistical perspective. *Neural Computation*, 1(4):425–464, 1989.
33. L. Yaeger, R. Lyon, and B. Webb. Effective training of a neural network character classifier for word recognition. In M.C. Mozer, M.I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, Cambridge, MA, 1997. MIT Press.