Improved Probability Estimation with Neural Network Models

Wei Wei, Etienne Barnard and Mark Fanty

Center for Spoken Language Understanding
Oregon Graduate Institute of Science and Technology
20000 N.W.Walker Road, Portland, OR 97291-1000

ABSTRACT

Neural network classifiers can provide outputs that estimate Bayesian posterior probabilities under the assumptions that an infinite amount of training data are available, the network is sufficiently complex and the training can reach the global minimum. In practice, however, the number of training tokens is limited and may not accurately reflect the prior class probabilities and true likelihood distributions. Additionally, computational constraints place a limit on the complexity of the network. Consequently, practical networks often fall far short of being ideal estimators.

We address this problem and propose a new method of improved probability estimation by combining neural network models with empirical probability estimation methods. We use a histogram-based estimation method to remap the network outputs to match the data and thereby improve the accuracy of the probability estimates. Our current experiments on the OGI Census Year corpus resulted in a 20.6% reduction in recognition errors at the utterance level.

1. Introduction

Traditionally neural network outputs are expected to have binary values that should always be near zero or one. Unless the value of a "correct" network output is greater than 0.5, the classification decisions are often considered incorrect and it is thought that more training is still required. However, this might not always be true, because the output values of the neural networks often approximate Bayesian posterior probabilities [5]. If density functions of classes overlap, the network outputs need not be close to 0 or 1: they can have values ranging from 0.0 to 1.0. Therefore applying extra training using those patterns that fail to generate outputs close to the desired values will be counter productive, because it alters the distributions and makes the network less likely to generate the correct Bayesian posterior probabilities [2].

Unlike conventional Bayesian classifiers, which derive the Bayesian posterior probabilities from likelihood indirectly by Bayes Rule, neural network classifiers can provide outputs that estimate Bayesian posterior probabilities directly. The latter is however true only under the assumptions that (1)an infinite amount of training data is available, (2) the network is sufficiently complex and (3) the training error can reach the global minimum [5]. In practice, however, these assumptions are not satisfied and therefore neural network outputs may fail to approximate Bayesian posterior probabilities well.

2. Combining Probability Estimation Methods with Neural Networks

We can measure the accuracy of a network's probability esimates by making a histogram of the frequency with which a network output matches the desired output in a given data set, with histogram buckets determined by the output values of the network. Our experimental results (shown in Section 4) confirm that the network outputs do not always estimate probabilities accurately. Earlier simulation results [5] also demonstrate that the neural network classifiers provide outputs that estimate Bayesian posterior probabilities, with the estimation accuracy influenced by the network complexity, the number of training data, and the degree to which training data reflect true likelihood distributions and the prior class probabilities.

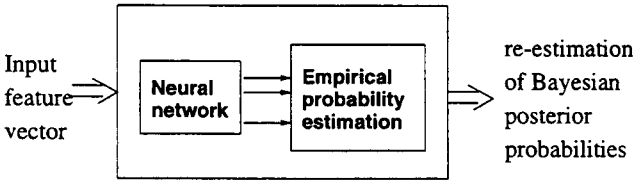

Figure 1: A conceptual posterior probability estimation model based on neural networks

To solve this problem, we propose a new approach to improve probability estimates by combining neural network models with empirical probability estimation methods. Figure 1 shows our conceptual posterior probability estimation model based on neural networks. We use neural network outputs as the first estimates of Bayesian posterior probabilities.

Neural networks are flexible in modeling regularities in the data, but an over-flexible network can be misled by stray correlations within the data into "discovering" improbable structure [4]. Our empirical probability estimation based on histograms is to remap the network outputs to match the data and thereby improve the accuracy of the probability estimation. The strategy is to find a simple function for each unit in the output layer of the network; this function maps that output value of the unit to a more accurate estimate of the true probability.

3. Empirical Probability Estimation Models based on MLPs

Figure 2 shows our empirical probability estimation model based on a multilayer perceptron (MLP) neural network. We remap the outputs of neural networks using histograms.

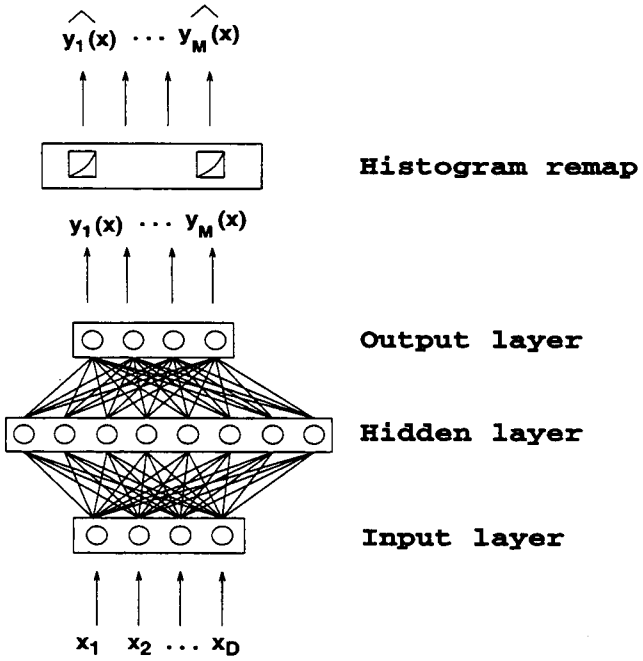

Figure 2: An empirical probability estimation model based on an MLP neural network

We made a histogram of the frequency with which a network output matches the desired output in a given data set, with histogram buckets determined by the output value of the network. For example, if we put all outputs with a value between 0.3 and 0.4 in one bucket, ideally the frequency of correct matches for this bucket will be about 30 to 40 percent of the total hits for this bucket. That is, if the neural network provides outputs that accurately estimate posterior probabilities, the histograms should be close to a diagonal line, when plotted against network outputs.

Because different classes have different density distributions of patterns, if we use an identical bucket set for all classes we may leave some buckets containing no histogram values. This type of histogram is not smooth enough for remapping. We solve this problem by dynamically adjusting the buckets of a class based on the output distribution of this class. Because an ideal histogram is a diagonal line, we assume that a histogram curve that is close to this diagonal line may be effective in increasing the estimation accuracy through remapping. In our approach, we developed a non-parametric smoothing method to dynamically adjust the buckets of each class based on the output distributions of this class until we get a bucket set that gives a monotonically increasing histogram distribution.

Figure 3 shows an example histogram with bucket-adjustment by a solid line. It was computed from a neural network trained on the OGI Census Year corpus, using a cross-validation data set. The dashed line is a diagonal line, which shows the histogram of an ideal estimator. From these figures, we can see that these two histograms are quite different from each other, which confirms that practical networks often fall short of being ideal estimators.

4. Experiments and Evaluations

4.1. Experimental Setup

Our experimental task is recognition of the OGI Census Year corpus. This corpus was collected and created by CSLU, as part of a study to determine the feasibility of using an automated spoken questionnaire to collect information for the Year 2000 United States Census [1]. The goal of the study was to develop and evaluate a telephone questionnaire that automatically captures and recognizes the following information: (1) full name, (2) sex, (3) birth date, (4) marital status, (5) Hispanic origin, and (6) race. The OGI Census Year corpus comes from the birth-date database.

We separate this data set into three parts by the ratios 3:1:1, used as training data, cross-validation data and test data, respectively.

The MLP neural network built for the OGI Census Year speech recognition system is a three-layer perceptron neural network consisting of 56 input nodes, 45 hidden nodes and 123 output nodes that correspond to 123 biphone classes. It is trained with stochastic back propagation algorithms, using a cross-entropy cost function. We use an empirical probability estimation model (shown in Figure 2) based on this MLP neural network as the phoneme probability estimator instead of an MLP neural network.

Below are our experimental results and their methodological analysis.

4.2. Selecting a data set for remapping use

We use (1) the cross-validation data set and (2) the combination of the training data set and the cross-validation data set as the remapping data set. Our experimental results show that a remapping based on the cross-validation data set is as good as that based on the combination of the training data set and the cross-validation data set, and even better if the computational cost is considered. A possible explanation for this result is that the patterns in the training data set have been learned during training and are ultimately reflected by the parameter values of the neural network, but the cross-validation set is used to stop overfitting and gives no additional contribution to the parameter values via pattern-learning. Therefore, the patterns in the cross-validation data set still contain new information that will help in improving the estimation accuracy and generalization of the neural-network-based estimator by the remapping process.

4.3. Remapping based on histograms

Our strategy is to find a simple function for each unit in the network's output layer that maps the unit's output values to a more accurate estimate of the true probability. This remapping is based on the histogram values described above, computed on a cross-validation data set. We have tried several methods of remapping based on histograms, including doing regression on histogram data and then remapping using the regressive data. Finding new fitting methods is still an active research topic. So far, among those remapping methods, we have had our best results fitting the histogram of each class with two lines in the following two areas:

1. In the interval $[0.0, 0.1]$, we use linear fitting based on logarithmic scale – we get two parameters a and b;
2. In the interval $(0.1, 1.0]$, we use linear fitting – we get one parameter c.

The remapping requires only three parameter values (a, b and c) per class. After we get an output value y_i (in Figure 2) of the neural network, we can use the following transformation function to get its remapped value $\widehat{y_i}$ (in Figure 2):

$$\widehat{y_i} = a * y_i^b, \qquad \text{if} \quad y_i \leq 0.1;$$
$$\widehat{y_i} = c * (y_i - 0.1) + a * 0.1^b, \qquad \text{otherwise} \qquad (1)$$

Because each class requires only three parameter values for remapping and the transformation function is very simple, applying remapping during recognition has a negligible effect on the total computation time. We do remapping on the classes that have smooth histograms. For example, we can set up a threshold k, we do remapping on the classes whose histograms contain more than k data points; otherwise, we do not. If k is large, then the number of the remapping classes is small. There is an optimal k that gives good

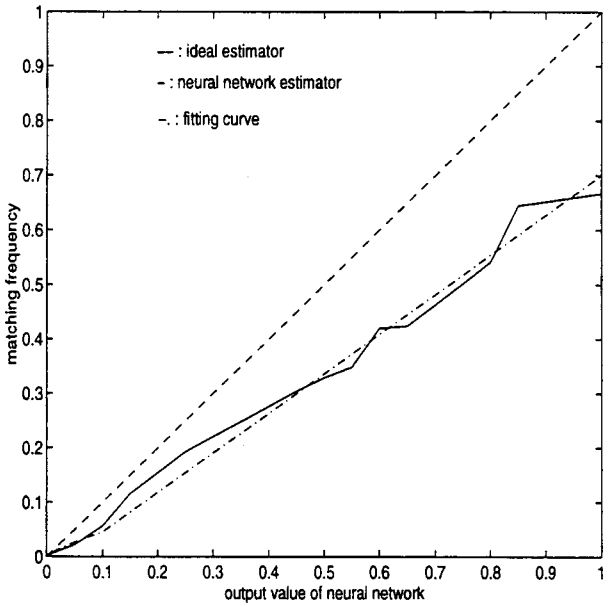

Figure 3: An example histogram

balance. The dashed and dotted curves in Figure 3 show the fitting curve for one output of our network.

4.4. Estimating Bayesian Posterior Probabilities

The Bayesian posterior probabilities are estimated using a weighted linear combination of both the network outputs (y_i) and the remapped outputs ($\widehat{y_i}$).

Because the network outputs and the remapped network outputs may contain different information coming from the training data set and the combination of the training data set and a cross-validation data set, separately, we weight them to get new estimates based on the network outputs and their remapped values.

If we use α ($0 \leq \alpha \leq 1$) to represent the weight of a network output y_i, then the weight of its corresponding remapped output $\widehat{y_i}$ is $1 - \alpha$. Therefore the output value $\widehat{o_i}$ of our empirical probability estimation model with neural networks can be calculated as:

$$\widehat{o_i} = \alpha * y_i + (1 - \alpha) * \widehat{y_i}(y_i), \qquad 0 \leq \alpha \leq 1 \qquad (2)$$

Let $\widehat{O}\{\widehat{o_i} : i = 1, \cdots, M\}$ denote the estimated output vector. We tune weight α to get different empirical probability estimator. This weighting process can be shown in Figure 4.

From no remapping to remapping, we only remodeled the phoneme probability estimator using histogram remapping, with other experimental conditions unchanged. Therefore, the increase in recognition accuracy at the utterance level

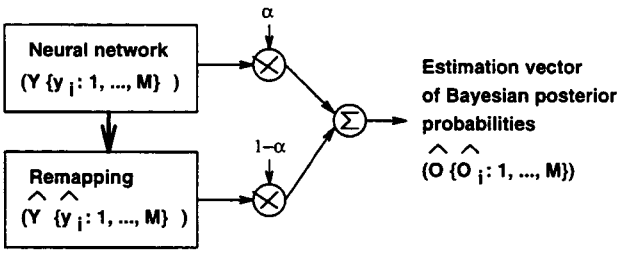

Figure 4: Bayesian posterior probability estimation based on weighting the network outputs and the remapped outputs

α	1.0	2/3	1/2	0.0
accuracy	94.7%	95.3%	95.0%	94.3%
error reduction	-	11.3%	5.7%	-

Table 1: Utterance-level recognition results on OGI Census Year corpus (doing evaluation on a cross-validation set).

reflects the improvement in the estimation accuracy of the Bayesian posterior probabilities.

We use a cross-validation data set (300 utterances) to do remapping and also use it to choose k and α. We get the optimal k which is 15. With this optimal value of k, almost half of the classes (49%) are not remapped. Table 1 shows the recognition results of the OGI Census Year corpus at the utterance level, including the recognition accuracies and the corresponding error reduction rates.

In Table 1, column 2 shows the result (94.7%) when we use network outputs directly as the estimation values of Bayesian posterior probabilities ($\alpha = 1.0$); column 3 shows the result (95.3%) when we set $\alpha = 2/3$; column 4 shows the result (95.0%) when we set $\alpha = 1/2$; column 5 shows the result (94.3%) when we use the remapped outputs only as the estimation value of Bayesian posterior probabilities ($\alpha = 0.0$). These results show that remapping can work effectively in increasing the speech recognition accuracy and that we can tune α to get an optimal probability estimator. The selection of α is related with the representative quality of neural networks (for example, generality) and the histogram effects (for example, smoothness of the histograms) of the remapping. From Table 1 we get the optimal α value which is 2/3.

Setting α to be 2/3, we then use the test data set (containing 734 utterances) to do evaluation; our experimental results are shown in Table 2.

As shown in Table 2, this optimal probability estimator ($\alpha = 2/3$) resulted in a 20.6% reduction in recognition errors at the utterance level. Using McNemar's test [3], the significance level of this improvement is 2.2%, which means that the reduction in the error rate is statistically significant. This result shows that remapping is effective in increasing the estimation accuracy of Bayesian posterior probabilities and thereby the speech recognition accuracy.

α	1.0	2/3
recognition accuracy	93.7%	95.0%
error reduction	-	20.6%

Table 2: Utterance-level recognition results on OGI Census Year corpus (doing evaluation on the test set).

5. Conclusions

In practice, speech data generally have different prior class probabilities and different likelihood distributions. The classes that have very few or even no samples are poorly trained during the neural network training process. Constrained by limited training data and network complexity, neural network outputs may not provide accurate estimation of Bayesian posterior probabilities. Unfortunately, we are also unable to remap those classes due to insufficient data. Our further work includes finding an effective way to deal with the remapping of the sparse classes that have few training samples and increasing the number of remapping classes. Compared to a single neural network estimator, a probability estimator combining neural-network-output estimation with empirical probability estimation is promising in increasing the estimation accuracy of Bayesian posterior probabilities and thereby improving speech recognition accuracy.

6. Acknowledgement

This research is supported in part by grants from the National Science Foundation, the Defense Advanced Research Projects Agency, and the member companies of CSLU.

7. REFERENCES

1. E. Barnard, R. Cole, M. Fanty, and P. Vermeulen. Real-world speech recognition with neural networks. *Proceedings of the International Symposium on Aerospace/Defense Sensing & Control and Dual-Use Photonics*, Apr. 17-21 1995. invited paper, Orlando, FL.
2. C. M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, 1995.
3. L. Gillick and S. J. Cox. Some statistical issues in the comparison of speech recognition algorithms. *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 532–535, 1989.
4. R. M. Neal. *Bayesian Learning for Neural Networks.* PhD thesis, University of Toronto, 1995.
5. M. D. Richard and R. P. Lippmann. Neural network classifiers estimate bayesian a posteriori probabilities. *Neural Computation*, (3):461–483, 1991.