ADAPTIVE TRANSITION BIAS FOR ROBUST LOW COMPLEXITY SPEECH RECOGNITION

Konstantinos Koumpis and Søren Kamaric Riis

Nokia Mobile Phones
Digital Signal Processing Group
Copenhagen, Denmark
soren.riis@nokia.com

ABSTRACT

The basis for all methods described in this paper is the application of an adaptive transition bias to the sequences of phoneme models that represent spoken utterances. This offers significantly improved accuracy in phoneme based speaker independent recognition, while adding very little overhead to the overall system complexity. The algorithms were tested using the low complexity hybrid recognizer denoted Hidden Neural Networks (HNN) on US English and Japanese speaker independent name dialing tasks. Experimental results show that our approach provides a relative error rate reduction of up to 47% over the baseline system.

1. INTRODUCTION

As speech recognition technologies are being transferred to end-user applications, the issue of robustness to environmental noise and speaker variability is becoming increasingly important. This is due to the fact that the performance of most state-of-the-art speech recognition systems tend to deteriorate seriously in mismatched conditions, *e.g.*, when the signal-to-noise ratio (SNR) is low or when the native language or dialect of the speaker is not well represented by the system. Many noise robustness techniques work by characterizing the differences between data used for training and testing the system [6]. Based on some formal statistical measure these methods can be used for adapting either the speech derived input features or the model-based representations of speech so as to better match the testing conditions.

Several adaptation approaches such as Maximum A Posteriori (MAP) [3], Maximum Likelihood Linear Regression (MLLR) [10] and EigenVoices [9] have high requirements for memory and computational resources. Although the computational power of embedded portable devices is rapidly increasing with time, the number of applications required to run simultaneously increases as well. Thus, complexity and memory requirements of any application running on a portable device will always be an issue.

The transition bias method presented in this paper adaptively changes the duration characteristics of acoustic phoneme models during recognition. As illustrated in Figure 1, the transition bias is a single positive parameter used as a transition probability between *all* phoneme models. This typically turns the inter-phoneme model transition probabilities into 'non-probabilistic' scores. However, it is easy to show that a probabilistic normalization of the transition probabilities is not needed during decoding [11]. The adaptive transition bias approach is computationally very simple, but was nevertheless found to give a relative error rate reduction of up to 47% over the baseline system. The basic idea in this approach is to take advantage of the strong correlation between the transition probabilities between phoneme models and the duration of the acoustic signal. The mechanical equivalent of the transition bias is a spring (model sequence) that is stretched by a force (transition bias) in order to reach a certain length.

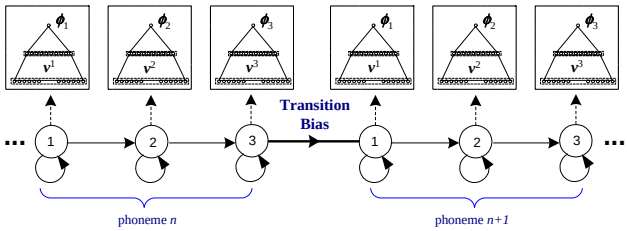

Fig. 1. *The use of a transition bias between phoneme models for duration modeling.*

An analogous technique for duration modeling called phone-deletion penalty was reported in [5], in which the score of the active hypothesis was normalized according to the *actual* number of phonemes in the *active hypothesis*. This differs from our approach, in which the normalization is related to the *estimated* number of phonemes in the *utterance* currently being decoded.

The rest of the paper is organized as follows: in section 2 we describe the setup of the recognizer. The transition bias and its estimation is presented in sections 3 and 4, along with the experimental results in section 5. The paper is concluded in section 6.

2. SETUP OF THE BASELINE HNN SYSTEM

The Hidden Neural Network (HNN) is a simple and intuitive extension of the standard Hidden Markov Model (HMM), in which the usual probability parameters of an HMM are *replaced* by small *state specific* multi-layer perceptron (MLP) neural networks [8, 11]. In this work the Gaussian mixture function commonly used in HMMs is replaced by a match network estimating the score that the current observation matches a given state. Only the current feature vector was used as input to the match networks.

In [12] it was shown that the HNN can outperform a conventional HMM based speech recognition system which requires at least 14 times more memory for storing phoneme models. Thus, a HNN system requiring 6KB phoneme model memory was shown to obtain results comparable to that of an HMM based system requiring 160KB phoneme model memory. Due to the low number of parameters in the HNN it also has a significantly lower computational requirement for real-time decoding.

2.1. Preprocessing

For each 10 ms, the preprocessor computed 13 Mel-Frequency Cepstral Coefficients (MFCC) and the corresponding first and second

K. Koumpis is currently with the University of Sheffield, UK. Email: k.koumpis@dcs.shef.ac.uk

order derivatives. A Recursive Cepstral Mean normalization approach similar to cepstral means normalization was applied in order to reduce the effects of low-frequent car noise [4]. In the recursive normalization method, all MFCCs were normalized to zero mean based on a recursively updated short-term mean estimate. Similarly, the log energy coefficient and its derivatives were normalized to unit variance based on a recursively updated short-term estimate of the variance.

2.2. Model Topology and Training

For each of the phonemes occurring in the transcriptions we used a left-to-right context independent phoneme model with three states and no skips. In this work the match network in each state was a very simple one-layer perceptron that simply passes a weighted sum of the elements in the input feature vector through a sigmoidal shaped nonlinear function. As the dimension of the feature vector is 39, each match network contains 40 weights (including a bias weight).

The US English recognizer was based on a set of 45 different phoneme models (including a silence model) and thus comprises a total of 5,670 parameters including the transition probabilities. The Japanese recognizer was constructed from a set of 25 different phoneme models (including a silence model). Hence, it comprises a total of 3,150 parameters. As each parameter can be quantized to 8 bits without loss in recognition accuracy [12], the US English model takes up about 6KB of memory whereas the Japanese model about 3KB of memory.

All HNN parameters (MLP weights and transition probabilities) were jointly trained by gradient descent to maximize the discriminative Conditional Maximum Likelihood (CML) criterion. CML is equivalent to Maximum Mutual Information [1] training, if the language model is assumed fixed during model training. For further details on HNN training please refer to [11].

2.3. Databases

The US English phoneme models were trained using a Nokia in-house database of more than 55,000 isolated words recorded in a car environment. The Japanese phoneme models were trained on a second in-house database of 8,170 connected speech utterances recorded in a clean laboratory environment.

For testing purposes two databases were used: A US English database of 3,150 utterances based on a 70 names vocabulary, and a Japanese one of 11,992 utterances based on a 120 names vocabulary. Both test databases were recorded in a clean environment, but noisy versions were created by mixing car noise at 5dB SNR.

3. THE TRANSITION BIAS

Due to the use of non-uniform transition probabilities between states, the HNN based recognizer tends to favor relatively short utterances over long ones. Therefore, by applying a transition bias larger than 1.0 during decoding we force the model to exit the current phoneme model faster in order to enter the following one. On the contrary, a transition bias less than 1.0 compels a model to stay longer in the same phoneme. This effect can be compared with a very crude form of phoneme duration modeling. As illustrated in Figure 2, a significant improvement in recognition performance can be obtained by setting the transition bias according to the number of phonemes in an utterance. The effects are particularly prominent under the presence of noise.

As the number of phonemes in the utterance to be decoded is *not* known a priori, one simple way to circumvent this problem is to use an average transition bias which is optimal for the entire recognition vocabulary. The average optimal bias can be selected by trial and error among the values that maximize recognition performance over a validation data set containing utterances from the

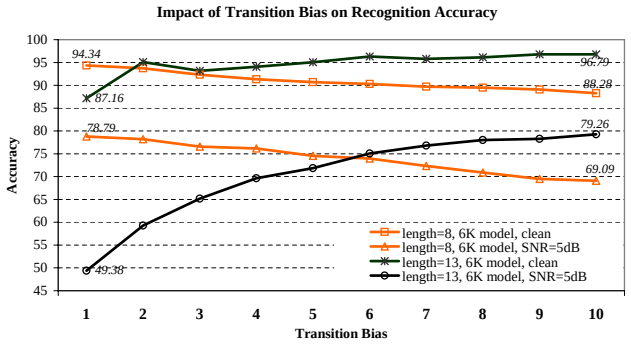

Fig. 2. *Recognition accuracy for different values of the transition bias. A large value increases the accuracy over long names (* e.g., *13 phonemes) by 30% absolute in the noisy environment. On the contrary, the same bias value can decrease the accuracy for rather short names (* e.g., *8 phonemes) by 10%.*

recognition vocabulary or a vocabulary related to the target application.

Although the use of an average optimal transition bias works well in most cases, some users might define very 'atypical' recognition dictionaries (*e.g.* in name dialing some users might use only short names and others only long names). For such users there might not be a gain in recognition accuracy by using a transition bias which was found to be optimal on average for some particular validation data set. Furthermore, the quality of the bias estimated from a validation set depends heavily on the available validation data and the estimation procedure needs to be carried out for each new language to be supported. It is therefore desirable to set the optimal transition bias according to a speaker and language independent criterion such as the estimated number of phonemes in the utterance.

The transition bias is applied before decoding and thus the optimal value of the bias should be computed before starting decoding. This implies that the entire utterance in principle must be spoken before decoding can start. For tasks such as name dialing this may not pose a problem, as the utterances are usually very short and the recognition result will therefore be ready with a very small lag. If a 'true' real-time decoding is required (*i.e.*, the result is available immediately after the speaker finishes speaking), the optimal bias is computed at the same time as decoding the concatenated phoneme models without a transition bias (set to 1.0). At the end of decoding, all scores are normalized by a contribution proportional to the optimal transition bias TB_{opt}:

$$\log \bar{S}_i = \log S_i + \log(TB_{opt}) N_p \qquad (1)$$

where S_i is the score and N_p is the number of phonemes in the ith hypothesis, respectively. If a Viterbi decoder is used, this normalization will give exactly the same change in the score as if the optimal transition bias was applied prior to decoding. This is due to the fact that the Viterbi algorithm only finds the single optimal path through the sequence of phoneme models corresponding to a word. Therefore, if the word contains N_p phonemes, the change in log-score by applying the optimal bias before decoding is exactly $log(TB_{opt})N_p$, as the transition bias enters the log-score in an additive way. For all-path forward decoding, on the other hand, the result is a sum over all possible paths through the model, and the expression (1) is thus only an approximation of the actual score.

4. ESTIMATION OF THE TRANSITION BIAS

Several strategies for estimating the length of the observed sequence are possible. This may be cast as a simple interpretation

of the speaking rate estimation or rate of speech (ROS) detection tasks. A description and performance comparison of three such methods, namely a free-order decoder, a speaker-specific ROS, and a MLP-based phoneme counter, follows:

4.1. Free-order Viterbi Decoder

In this approach each utterance is decoded using an unconstrained grammar, also known as a free-order or looped grammar. The Viterbi decoder gives a state segmentation of the utterance, which is then translated into a phoneme sequence. Although this phoneme sequence can be a fairly poor match to the true phoneme sequence in the utterance, for this task we only need the number of consecutive different phonemes in the segmentation. The disadvantage of this method is that it is rather computationally demanding, as it requires a decoding procedure (in contrast to those described in the following two sections).

4.2. Speaker-specific Rate of Speech estimator

For each speaker an estimate of the ROS is incrementally updated during the recognition process. The ROS detector measures the number of speech observations (usually 10 ms frames) per phoneme on average. This technique is based on robust endpoint detection and ideally requires knowledge of whether an utterance is correctly recognized or not. In a name dialing application the latter will not pose a serious problem, as the user is very likely to give feedback about the correctness of the recognition, *i.e.*, if a wrong name is recognized the user is very likely to cancel the call to the number associated with the misrecognized name. Based on the correctly recognized utterance, the current ROS estimate is updated as follows:

$$ROS(n) = \gamma ROS(n-1) + (1-\gamma)\frac{N_f(n-1)}{N_p(n-1)} \qquad (2)$$

where N_f is the number of speech observations (non-silence frames) in the nth correctly recognized utterance as estimated using a robust endpoint detector, N_p is the number of phonemes in the word corresponding to the nth correctly recognized utterance and γ is a weighting factor in the range [0.0;1.0]. A weighting factor close to 1.0 implies that the latest estimate of the ROS based on the last recognized utterance only contributes marginally to the running average. A weighting factor close to 0.0 implies that the ROS estimate is based almost entirely on the last recognized utterance. If it is not known whether the previous processed utterance was correctly recognized or not, N_p in (2) can be set to the number of phonemes in the highest scoring word for the previous utterance. Even though the previous utterance was not recognized correctly, the number of phonemes in the recognized word will typically be close to the number observed in the correct word.

From the current ROS estimate it is straightforward to find the number of phonemes in the utterance to be recognized:

$$\hat{N}_p(n) = \frac{\hat{N}_f(n)}{ROS(n-1)} \qquad (3)$$

The main assumptions in the above approach is that the speaker has a fairly constant speaking rate and that a single average ROS for all phonemes is sufficient for the purpose of estimating the number of phonemes in a word. If the speaker changes speaking style in an abrupt manner, the speaker-specific ROS based estimate might be highly inaccurate. Similarly, the phoneme count estimate can be poor for words that contain phonemes with a 'true' ROS far from the average ROS. For comparison, the Free-order Viterbi based estimator described above is less sensitive to speaking rate variation as well as the actual phonemes occurring in the vocabulary words. However, the Free-order Viterbi based method is significantly more complex than the speaker-specific ROS based estimator.

4.3. MLP-based Phoneme Counter

This method is based on a MLP classifier for phoneme boundary detection. Similar approaches for ROS detection and phoneme segmentation were reported in [14] and [13], respectively. The advantage of these methods is that they do not require a decoding procedure, and therefore typically are less computational expensive than *e.g.*, the Free-order Viterbi based estimator. In this work only the static MFCC and their first order derivatives were used as input to the MLP[1]. In the best performing configuration, four consecutive frames (resulting in a 104 dimensional vector) were used as input to the MLP classifier. The output layer had a single node determining whether the current input vector corresponds to a phoneme boundary or not. The hidden layer comprised 30 units, thus the MLP has 3,181 weights in total. The MLP was trained using a combination of batch mode gradient descent and a Gauss-Newton second order method.

When classifying frames to phoneme non-boundaries and boundaries, we would expect a very small prior probability of the latter ($\sim$5% in our training set). To obtain a good variety of boundary samples in the training set would therefore require huge numbers of training examples. To handle this poor balance in the training set, we artificially increased the proportion of boundary samples to 50% in the training and validation set, which contained a total of 47K input-ouput pairs. The phonetic segmentation for the training samples was obtained by a forced alignment on the US English test data. After training, the MLP output was scaled so as to match the original prior distribution of boundary frames in the training set [2].

The main disadvantage of the above method is the long (offline) training time. Furthermore, the threshold for boundary and non-boundary classification has to be carefully selected to avoid false boundary detections. In this work, false boundary detections were avoided by using four consecutive MLP outputs for determining whether or not a phoneme boundary occurred.

5. RESULTS

The number of phonemes detected by the above three methods was the basis for setting the transition bias. Two simple methods were used for this purpose:

Look-up table where the optimal bias is set according to the range of the number of phonemes detected. In particular, it is set to 1.0 for the range of [0;6), to 4.0 for [6;8), to 6.0 for [8;10) and to 10.0 for [10;$+\infty$). These ranges were selected after studying patterns as those shown in Figure 2.

Direct estimate where the optimal bias is simply set equal to the estimated number of phonemes in the utterance. In the rare case that no phoneme boundaries are detected the bias is set to 1.0.

Tables 1 and 2 show the error rate (ERR) of the baseline system and the system using an adaptive transition bias. Results are given for US English and Japanese name dialing tasks in both clean and noisy conditions (5dB SNR). The rows entitled 'Oracle' show the performance when assuming that the exact number of phonemes is known a priori. The rows entitled 'Average Optimal Transition Bias' show the performance when using an average transition bias estimated from a validation set based on the recognition vocabulary.

By studying Table 1 it is noticed that all three methods for estimating the transition bias adaptively, give a substantial decrease in

[1] A different feature extraction technique *e.g.*, one based on spectrogram representations [7] might deliver better results on phoneme counting.

Method	Clean	Noisy
Baseline (Transition Bias set to 1.0)	7.56%	26.41%
Average Optimal Transition Bias	6.30%	21.68%
Free-order Viterbi, Look-up table	6.44%	20.70%
Free-order Viterbi, Direct estimate	6.79%	20.63%
ROS Estimator, Look-up table	6.58%	20.80%
ROS Estimator, Direct estimate	6.83%	21.21%
MLP counter, Look-up table	7.14%	21.08%
MLP counter, Direct estimate	7.43%	21.84%
Oracle, Look-up table	5.84%	18.86%
Oracle, Direct estimate	6.67%	20.03%

Table 1. *Error rate on the US English name dialing task after applying the transition bias over the baseline system. The relative error rate reduction of the best performing configuration for clean and noisy conditions was 12.96% (22.75% for the Oracle) and 21.24% (28.59% for the Oracle) respectively.*

Method	Clean	Noisy
Baseline (Transition Bias set to 1.0)	4.91%	26.48%
Average Optimal Transition Bias	2.65%	16.85%
Free-order Viterbi, Look-up table	2.89%	18.63%
Free-order Viterbi, Direct estimate	2.65%	17.61%
ROS Estimator, Look-up table	2.73%	16.16%
ROS Estimator, Direct estimate	2.64%	16.05%
MLP counter, Look-up table	2.60%	18.51%
MLP counter, Direct estimate	2.88%	16.85%
Oracle, Look-up table	2.60%	15.90%
Oracle, Direct estimate	2.49%	15.70%

Table 2. *Error rate on the Japanese name dialing task after applying the transition bias over the baseline system. The relative error rate reduction of the best performing configuration for clean and noisy conditions was 47.05% (49.29% for the Oracle) and 38.97% (40.71% for the Oracle) respectively.*

ERR compared to the baseline, but also that the difference among the methods is less than 2% absolute. Interestingly, the estimator based methods give results that are very close to what can be obtained if the true number of phonemes is known a priori ('Oracle' row). As the ROS based estimator is the simplest in terms of computational complexity this method is the preferred solution for implementation on an embedded device. The results in Table 2 indicate that the methods generalize well for other languages. Thus, for Japanese the MLP-based estimator produces a ERR close to that of the ROS and Free-order Viterbi methods, even though the MLP was trained on US English data.

Further investigation of how to select the optimal bias from the phoneme count estimate is currently in progress. A simple extension of the methods described in this paper is to make use of a confidence measure for the phoneme count estimate to select the transition bias value. The confidence measure could also be used for dynamically splitting the vocabulary during decoding such that only words/utterances with a phoneme count similar to the estimated phoneme count are considered during decoding. The size of the active vocabulary should be set according to the confidence of the phoneme count estimate. Finally, it might be possible to introduce a more detailed 'phoneme duration modeling' by using separate transition biases for different phoneme models.

6. CONCLUSION

In this paper, we demonstrated how the adaptive transition bias scheme can significantly improve recognition accuracy of the parameter efficient HNN hybrid, especially under the presence of noise. The adaptive transition bias scheme can be considered as a very crude, but yet effective, duration modeling method and was shown to give a relative error reduction of up to 47% for a speaker independent name dialing task.

7. ACKNOWLEDGMENT

We wish to thank Morten With Pedersen for providing the classification software used in the MLP-based phoneme counter.

8. REFERENCES

[1] L. R. Bahl, P. F. Brown, P. V. de Souza, and R. L. Mercer. Maximum mutual information estimation of hidden Markov model parameters for speech recognition. In *Proceedings of ICASSP*, pages 49–52, 1986.

[2] H. A. Bourlard and N. Morgan. *Connnectionist Speech Recognition: A Hybrid Approach*. Kluwer Academic Publishers, Boston, MA, 1994.

[3] J. L. Gauvain and C. H. Lee. Maximum a-posteriori estimation for multivariate Gaussian mixture observations of Markov chains. *IEEE Transactions on Speech and Audio Processing*, 2:1291–298, 1994.

[4] J. Hakkinen, O. Vikki, J. Tian, and M. Vasilache. Development of robust speaker independent command word recognition for car hands-free applications. In *Proceedings of IEEE Workshop on Robust Methods for Speech Recognition in Adverse Conditions*, pages 139–142, 1999.

[5] M. Hochberg, S. Renals, T. Robinson, and D. Kershaw. Large vocabulary continuous speech recognition using a hybrid connectionist/HMM system. In *Proc. ICSLP*, pages 1499–1502, 1994.

[6] J.-C. Junqua and J.-P. Haton. *Robustness in Automatic Speech Recognition: Fundamentals and Applications*. Kluwer Academic Publishers, Boston, 1996.

[7] B. Kingsbury, N. Morgan, and S. Greenberg. Robust speech recognition using the modulation spectrogram. *Speech Communication*, 25:117–132, 1998.

[8] A. Krogh and S. Riis. Hidden Neural Networks. *Neural Computation*, 11(2):541–563, 1999.

[9] R. Kuhn, P. Ngyuen, J.-C. Junqua, R. Boman, N. Niedzielsky, S. Fincke, K. Field, and M. Contolini. Fast speaker adaptation using a priori knowledge. In *Proceedings of ICASSP*, pages 749–752, 1999.

[10] C. J. Leggeter and P. C. Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models. *Computer Speech and Language*, 9:171–185, 1995.

[11] S. Riis. *Hidden Markov Models and Neural Networks for Speech Recognition*. PhD thesis, Department of Mathematical Modelling, Technical University of Denmark, Lyngby, 1998.

[12] S. Riis and O. Vikki. Low complexity speaker independent command word recognition in car environments. In *Proceedings of ICASSP*, pages 1743–1746, 2000.

[13] J. Suh and Y. Lee. Phoneme segmentation of continuous speech using multilayer perceptron. In *Proceedings of ICSLP*, pages 1297–1300, 1996.

[14] J. P. Vershasselt and J.-P. Martens. A fast and reliable rate of speech detector. In *Proceedings of ICSLP*, pages 2258–2261, 1996.