Kernel Density-based Acoustic Model with Cross-lingual Bottleneck Features for Resource Limited LVCSR

Van Hai Do\textsuperscript{1,2}, Xiong Xiao\textsuperscript{2}, Eng Siong Chng\textsuperscript{1,2}, Haizhou Li\textsuperscript{1,2,3}

\textsuperscript{1}School of Computer Engineering, Nanyang Technological University, Singapore
\textsuperscript{2}Temasek Laboratories@NTU, Nanyang Technological University, Singapore
\textsuperscript{3}Institute for Infocomm Research, A*STAR, Singapore

{dova0001, xiaoxiong, aseschng}@ntu.edu.sg, hii@i2r.a-star.edu.sg

Abstract

Conventional acoustic models, such as Gaussian mixture models (GMM) or deep neural networks (DNN), cannot be reliably estimated when there are very little speech training data, e.g. less than 1 hour. In this paper, we investigate the use of a non-parametric kernel density estimation method to predict the emission probability of HMM states. In addition, we introduce a discriminative score calibrator to improve the speech class posteriors generated by the kernel density for speech recognition task. Experimental results on the Wall Street Journal task show that the proposed acoustic model using cross-lingual bottleneck features significantly outperforms GMM and DNN models for limited training data case.

Index Terms: exemplar-based speech recognition, kernel density estimation, cross-lingual bottleneck feature, discriminative score calibration.

1. Introduction

Among the several thousands of spoken languages used today, few of them are studied by the speech recognition community [1]. One of the major hurdles of ASR system deployment in new languages is that ASR system relies on a large amount of training data for acoustic modeling. Usually, to build a reasonable acoustic model for a large- vocabulary continuous speech recognition (LVCSR) system, tens to hundreds of hours of training data are required, which makes a full fledged acoustic modeling process impractical for under-resourced languages. This motivates us to investigate methods to transfer well-trained acoustic models to under-resourced languages.

Various methods have been proposed for cross-lingual speech recognition such as universal phone set [2, 3], tandem approach [4–6], subspace GMMs (SGMMs) [7, 8], KL-HMM [9, 10], cross-lingual phone mapping [11–13] and its extension, context-dependent phone mapping [14–17]. Two concepts are generally used in these methods, i.e., more reliable estimation of the model parameters from a small amount of training data, and transfer of knowledge between languages. In addition, these methods usually use a parametric way of density estimation if feature distribution is required, e.g. in GMM and SGMM.

Exemplar-based methods are non-parametric techniques that use the training samples directly. Unlike parametric methods, such as GMM, exemplar-based methods, such as k-nearest neighbors (k-NN) [18] for classification and kernel density (or Parzen window) [18] for density estimation, do not assume a parametric form for the discriminant or density functions. This makes them attractive when the functional form or the distribution of decision boundary is unknown or difficult to learn. Although k-NN is very simple in theory and easy to implement, it is often among the best performing techniques in many machine learning tasks [19].

Recently, several studies apply exemplar-based methods for acoustic modelling [20–23]. In [20], the authors reported that using kernel density estimation produces better results than conventional GMM when less than 3 hours of speech training data is available. The k-NN method was also used in [23], a consistent improvement is achieved when applying smoothing across frames. Such results motivated us to apply exemplar-based methods to resource-limited LVCSR task, where the amount of training data is very little.

In this paper, we apply the idea of kernel density-based acoustic modeling in [20] to resource-limited speech recognition. However, we find that simply using the kernel density with conventional MFCC features yields worse results than GMM. Hence, we use cross-lingual bottleneck features [16, 24–26] as acoustic features and the performance is significantly improved. This shows that exemplar-based system can be employed when on good input features are used. We suspect that the simple Euclidean distance used in kernel density estimation is not robust against MFCC feature variations. A limitation of the kernel density method is that it tries to estimate the distribution of the classes rather than optimal decision boundary between classes. Hence, its performance is not optimal in terms of speech recognition accuracy. To address this issue, we introduce a discriminative calibration method on top of the kernel density estimation and the speech recognition performance is consistently improved.

The rest of this paper is organized as follows: Section 2 describes the kernel density based acoustic model and the discriminative score calibration. Section 3 presents the experimental settings and results. Finally, we conclude in Section 4.

2. Exemplar-based LVCSR system

2.1. System overview

There are three key components in the proposed acoustic model for resource limited LVCSR, i.e., the kernel density-based HMM state emission probability estimation, the use of cross-lingual bottleneck features for knowledge transfer between languages, and the proposed discriminative score calibration. Compared with conventional HMM/GMM or HMM/DNN system, the major difference here is that we replaced the GMM or DNN with kernel density to estimate the class posterior or likelihood of each HMM state.

Fig. 1 illustrates the proposed LVCSR system. There are
five steps to build the system.

1. Use the raw training features, \( \mathbf{x}_i \) (e.g. MFCC) of the resource limited target language to build a triphone-based HMM/GMM acoustic model. Generate frame level state label for training data using forced alignment.

2. Generate the bottleneck features \( o_i \) from \( \mathbf{x}_i \). The bottleneck feature extracting network (BN-DNN) is borrowed from a resource rich source language.

3. Use kernel density estimation and bottleneck features to estimate HMM state emission probability \( \hat{p}(o_i | s_j) \).

4. Apply discriminative score calibration to refine the likelihood scores generated from the kernel density estimation.

5. Plug in the state emission probability to a standard decoder such as Viterbi decoder for decoding.

In the following sections, we will present the key ingredients of the exemplar-based speech recognition system in details.

### 2.2. Kernel density estimation for speech classes

In this study, instead of using a GMM to model the feature distribution of a triphone tied states as in the conventional HMM/GMM acoustic model, we use kernel density estimation similar to the one used in [20]. Specifically, the likelihood of a feature vector for a speech class (i.e. a tied state) is estimated as follows:

\[
\hat{p}(o_i | s_j) = \frac{1}{Z_j N_j} \sum_{i=1}^{N_j} \exp \left( - \frac{||o_i - e_{ij}||^2}{\sigma} \right) \tag{1}
\]

where \( o_i \) is the feature vector at frame \( t \), \( e_{ij} \) is the \( i^{th} \) exemplar of class \( j \), \( N_j \) is the number of exemplars in class \( j \), and \( Z_j \) is a normalization term to make (1) a valid distribution. Note that here we use Euclidian distance between the test feature vector and the exemplars in the likelihood function. It is also possible to use other distance measures as well. From (1), the likelihood function is mathematically similar to a GMM with shared scalar variance for all dimensions and Gaussians. Effectively, we are putting a small Gaussian-shaped window at each training example and the summation of the windows becomes the feature density before normalization. As our final target is speech recognition rather than density estimation, the term we are interested in is actually the class posteriors. Hence, the normalization term \( Z_j \) will never need to be computed as it is the same for all classes due to the use of single \( \sigma \) in all classes. The parameter \( \sigma \) is used to control the scale of the Gaussians and hence the smoothness of the resulting distribution. If \( \sigma \) is too big, the resulting distribution will be very smooth and vice versa [27].

The likelihood function in (1) is a non-parametric way to estimate the distribution of the features, i.e. we do not assume any structure for the density and the model size grows with the training data. If there are infinite number of training data, the likelihood function will eventually approach the true feature distribution by using very small \( \sigma \). On the other hand, when there are very few training samples in a class, \( \sigma \) should be large such that the feature space of the class can be covered by a few Gaussian windows. Ideally, we should use class-dependent \( \sigma \), as different speech classes may have very different amounts of training samples. However, in this preliminary study, we only use a global \( \sigma \).

### 2.3. Cross-lingual bottleneck features

To develop speech recognition systems for resource limited languages, it is popular to borrow acoustic information from resource rich languages. Cross-lingual bottleneck features [16, 24–26] is one example, in which the bottleneck feature extracting neural networks is trained from a resource rich language and then used as the feature extractor of the resource limited language. Previous study has shown that cross-lingual bottleneck features are very useful for resource limited speech recognition. In this study, we also adopt it as the features for acoustic model. As we will show in the experimental section, the cross-lingual bottleneck features is actually critical for the success of the exemplar-based speech recognition system.

### 2.4. Discriminative score calibration

The class likelihood function in (1) has two limitations when used for speech recognition. First, although the likelihood function asymptotically approaches the true density when infinite training data is available and hence the classification based on it should obtain performance close to the Bayes classification performance, in the case of very few training data, it may not lead to good recognition performance as the density estimate is not optimal. Second, the dynamic range of the log likelihood and the log posterior generated from it may be very different from the conventional GMM system, making it necessary to tune language model scale and beam width every time we tune \( \sigma \) or change the distance function. Such tuning process is tedious and may not produce the best results. In this section, we propose a discriminative score calibration step to address these two limitations.

A neural network can be used to calibrate the scores optimally in the sense of frame classification accuracy. As illustrated in the blue box of Fig. 2, the input of the calibration network is the posterior vector generated by the kernel density estimation.
estimation using the Bayes rule:

\[
\hat{p}(s_j|\alpha_t) = \frac{\hat{p}(\alpha_t|s_j)p(s_j)}{\sum_{j=1}^{M} \hat{p}(\alpha_t|s_j)p(s_j)}
\]

(2)

where \(p(s_j)\) is the state prior which is estimated from the training data. \(M\) is the number of classes (triphone tied states) in the acoustic model. The number of inputs and outputs of the neural network is the same and equal to the number of states. The objective of the calibrator is to estimate a new posterior \(p'(s_j|\alpha_t)\) that maximizes the frame classification accuracy on a development set. In our preliminary study, we found that a two-layer neural network with soft-max output layer and no hidden layer is able to produce good results when very few training data is available.

Since the standard Viterbi decoder requires state likelihood scores, after state posteriors are calibrated, they are converted back to likelihood scores. In practice, scaled likelihoods are used as in Eq. 3 since the scaling factor \(p(\alpha_t)\) is a constant for all states and does not affect the classification decision.

\[
p'(\alpha_t|s_j) = \frac{p'(s_j|\alpha_t)p(\alpha_t)}{p(s_j)} \propto \frac{p'(s_j|\alpha_t)}{p(s_j)}
\]

(3)

### 3. Experiments

#### 3.1. Experimental procedures

We study the performance of the proposed speech recognition system on the WSJ task. The WSJ task has been chosen as the target under-resourced language as the effect of sufficient training data for it is well known, and we can hence clearly demonstrate the effect of the proposed work on small training sizes. We use different training data sizes. As described in Sec. 2.1 and Fig. 1, for each training size, an HMM/GMM system is first built with a different number of tied states. Specifically, we use 243, 243, 501 and 1005 triphone tied states for the cases of 7, 16, 55 and 220 minutes of training data, respectively. The training sets are randomly selected from the full 15 hours of S184 training set and are nested, i.e. the set of 7 minutes is a subset of the set of 16 minutes and so on. The use of triphone model even for 7 minutes is suggested by our previous study [14, 15, 17] which shows that triphone acoustic models produces better results than monophone models even in the case of very limited training data. The baseline HMM/GMM system provides both the state-tying decision tree and frame level state label for the building of the exemplar-based system. The test data is 166 clean utterances, or about 20 minutes of speech. For baseline GMM training and Viterbi decoding, the HTK toolkit [30] is used.

The raw features to train the acoustic models are 39 dimensional MFCC features, including 13 static features and its time derivatives. To reduce recording mismatch between the WSJ and the source language (Malay) that is used to train the cross-lingual bottleneck features, utterance-based mean and variance normalization (MVN) is applied to features of both languages.

The cross-lingual bottleneck feature network is trained from more than 100 hours of Malay read speech data [28]. As shown in Fig. 1, a 5-layer-DNN is used to generate cross-lingual bottleneck feature. 9 frames of 39-dimensional MFCCs are used to form 351 inputs for the DNN. 102 outputs representing 102 monophone states in the Malay source acoustic model are used to form 351 inputs for the DNN. 102 outputs represent 102 monophone states in the Malay source acoustic model i.e. 34 phones x 3 states/phone. 39 hidden units are used at the bottleneck layer while the two other hidden layers contain 2000 units. The network is first initialized by RBMs [29] and then refined by using cross-entropy criterion.

For the kernel density estimation, Euclidian distance is used in (1) and the scaling factor \(\sigma\) is set to 1 empirically. The cross-lingual bottleneck features are normalized by corpus-based MVN to ensure that the weight of all feature dimensions is the same in the Euclidian distance. We use a 2-layer neural network for the discriminative score calibrator. Our preliminary experiments show that using 3-layer neural network do not bring any improvement over 2-layer neural network with the training data size used in this paper.

As we concentrate on reliable acoustic model training with limited training data, we assume the language model and pronunciation dictionary are available. The standard WSJ bigram LM and the 5k vocabulary are used in decoding. In the exemplar-based approach and hybrid models, for each HMM state, the probability of jumping to the next state is simply set to 0.5.

#### 3.2. Systems under comparison

In this paper, we compare the proposed exemplar-based system with two other cross-lingual systems as shown in Fig. 2. In the figure, there are four systems, all using cross-lingual bottleneck features. The first two systems are the proposed systems with and without discriminative score calibration. The third system is the conventional HMM/GMM system using bottleneck fea-
Table 1: WER (%) obtained by various systems at different training data sizes. Row 1 and 2 are results obtained by using only the English training data and MFCC features. Row 3-6 show results obtained by using cross-lingual bottleneck features. KD stands for kernel density used for acoustic modeling.

<table>
<thead>
<tr>
<th>No</th>
<th>Acoustic model</th>
<th>Training data (minutes)</th>
<th>7</th>
<th>16</th>
<th>35</th>
<th>220</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Monolingual (MFCC feature)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>HMM/GMM</td>
<td>30.9</td>
<td>23.1</td>
<td>14.8</td>
<td>9.1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>HMM/KD</td>
<td>33.7</td>
<td>26.2</td>
<td>15.5</td>
<td>11.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cross-lingual (cross-lingual bottleneck feature)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>HMM/GMM</td>
<td>24.6</td>
<td>18.5</td>
<td>11.3</td>
<td>10.1</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>HMM/KD</td>
<td>18.8</td>
<td>15.8</td>
<td>10.6</td>
<td>8.2</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>+ Calibration</td>
<td>15.8</td>
<td>13.3</td>
<td>9.9</td>
<td>7.9</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>HMM/DNN [16]</td>
<td>17.5</td>
<td>15.3</td>
<td>10.3</td>
<td>8.5</td>
<td></td>
</tr>
</tbody>
</table>

The results obtained by various systems are presented in Table 1. The first system is the conventional HMM/GMM system using MVN-processed MFCC features. As expected, the WER gets worse when less training data are used. For comparison, the WER obtained by using all the 15 hours of SIT4 training set is 7.9%.

The second row of Table 1 shows the results obtained by using kernel density estimation with MFCC features. Unfortunately, the kernel density produces worse results than the GMM baseline. We suspect that this is because the kernel density estimation of (1) uses simple Euclidian distance to measure distance between test features and exemplars. For comparison, GMM uses Gaussian dependent diagonal covariance matrix to scale the feature dimensions before computing the “distance”. For low level features such as MFCC, there are considerable nuisance variations, such as speaker variation, other than that due to speech classes. Hence, it could be possible that the benefit of kernel density estimation is offset by the weakness of Euclidian distance. This motivated us to use the more discriminative and invariant bottleneck features.

Next, let’s examine the results obtained by using the cross-lingual bottleneck features. The row 3 and 4 are the same as the row 1 and 2, except that MFCC features are replaced by bottleneck features. It is observed that using bottleneck features improves both GMM and exemplar-based systems. However, the improvement in the exemplar-based system is much larger than that in the GMM system. Now, the exemplar-based system consistently outperforms the GMM system in all training sizes. Note that at 220 minutes, the GMM with bottleneck features actually perform worse than the GMM with MFCC features. This could be due to that the gain of bottleneck features are offset by the loss due to the mismatch between the two corpora. However, for the exemplar-based system, the bottleneck features still produce better results than MFCC features.

3.3. Results using kernel density estimation

The results obtained by various systems are presented in Table 1. The first system is the conventional HMM/GMM system using MVN-processed MFCC features. As expected, the WER gets worse when less training data are used. For comparison, the WER obtained by using all the 15 hours of SIT4 training set is 7.9%.

The second row of Table 1 shows the results obtained by using kernel density estimation with MFCC features. Unfortunately, the kernel density produces worse results than the GMM baseline. We suspect that this is because the kernel density estimation of (1) uses simple Euclidian distance to measure distance between test features and exemplars. For comparison, GMM uses Gaussian dependent diagonal covariance matrix to scale the feature dimensions before computing the “distance”. For low level features such as MFCC, there are considerable nuisance variations, such as speaker variation, other than that due to speech classes. Hence, it could be possible that the benefit of kernel density estimation is offset by the weakness of Euclidian distance. This motivated us to use the more discriminative and invariant bottleneck features.

Next, let’s examine the results obtained by using the cross-lingual bottleneck features. The row 3 and 4 are the same as the row 1 and 2, except that MFCC features are replaced by bottleneck features. It is observed that using bottleneck features improves both GMM and exemplar-based systems. However, the improvement in the exemplar-based system is much larger than that in the GMM system. Now, the exemplar-based system consistently outperforms the GMM system in all training sizes. Note that at 220 minutes, the GMM with bottleneck features actually perform worse than the GMM with MFCC features. This could be due to that the gain of bottleneck features are offset by the loss due to the mismatch between the two corpora. However, for the exemplar-based system, the bottleneck features still produce better results than MFCC features.

3.4. Discriminative score calibration

The likelihood score of system 2 in Fig. 2 is further refined by the discriminative score calibration described in Sec. 2.4. A 2-layer neural network is used as the calibrator and trained to minimize the training frame classification error. The results are shown in the 5th row of Table 1. From the results, using the discriminative score calibration consistently outperforms the basic exemplar-based system using kernel density estimation. Furthermore, the score calibration seems to be more effective when there are fewer training data (e.g. 7 minutes) than when there are more training data (e.g. 220 minutes). This could be due to that when 7 minutes is used, the kernel density estimation is very poor, so it is more necessary to improve the scores using the calibrator. Fig. 3 illustrates the input to output weight matrix of the neural network score calibrator in the case of using 16 minutes of training data. We can see diagonal elements have much larger values than the off-diagonal elements. This shows that the calibrator is just fine tuning the posterior scores and the output scores has very high correlation with the input posterior. This is reasonable as the input and output are the posteriors of the same classes.

The last row of Table 1 shows our previous result [16] where cross-lingual bottleneck feature is used directly as the input of a neural network. For 7, 16, and 55 minutes of training data, we used 3-layer neural networks, while for 220 minutes of training data, we used a 5-layer neural network. The results are generally better than the cross-lingual HMM/GMM system, but worse than the discriminative score calibration system. Note that the score calibrator uses a much weaker neural network than that used in the hybrid system, yet it can produce consistently better results than the hybrid system. This shows that the kernel density estimation is extracting useful information for speech recognition.

4. Conclusion

In this paper, we studied an exemplar-based system for resource limited LVCSR. The system uses non-parametric kernel density estimation for HMM state emission probability, discriminative class posterior score calibration, and cross-lingual bottleneck features. When all these features are combined, the proposed system produces consistently better results than cross-lingual HMM/GMM and HMM/DNN systems, up to 220 minutes of training data. This shows that the proposed system has advantages over conventional system for small training sizes. In the future, we will focus on improving the proposed system by using better distance function. For example, we can use Mahalanobis distance \((o_i - e_j)^T A (o_i - e_j)\) to replace the Euclidian distance and learn the matrix A to minimize frame classification accuracy on development data.
5. References


