Improving Phonotactic Language Recognition with Acoustic Adaptation

Wade Shen and Douglas Reynolds
MIT Lincoln Laboratory
244 Wood Street
Lexington, MA 02420, USA
{swade,dar}@ll.mit.edu

Abstract

In recent evaluations of automatic language recognition systems, phonotactic approaches have proven highly effective [1][2]. However, as most of these systems rely on underlying ASR techniques to derive a phonetic tokenization, these techniques are potentially susceptible to acoustic variability from non-language sources (i.e. gender, speaker, channel, etc.). In this paper we apply techniques from ASR research to normalize and adapt HMM-based phonetic models to improve phonotactic language recognition performance. Experiments we conducted with these techniques show an EER reduction of 29% over traditional PRLM-based approaches.

Index Terms: language recognition, LID, ASR, Adaptation

1. Introduction

The problem of language recognition from speech lends itself to a variety of modeling approaches at different levels of the linguistic hierarchy [4]. The best systems, as evaluated by NIST [5] in recent years, have made use of multiple techniques that exploit both acoustic and phonotactic information.

For the automatic language recognition problem, all of these techniques rely on language-specific differences in the underlying speech for discrimination between languages. However, differences in both the acoustic and phonotactic characteristics of real speech can arise from a variety of non-language sources. While some of these sources may be linguistic (e.g. word usage, etc.), others may result from paralinguistic factors (e.g. speaker, gender, language, channel, etc.).

Non-language sources of variability can limit the performance of current modeling techniques. The modeling techniques used by phonotactic systems are subject to non-language variability from the underlying phone recognizer used to tokenize the speech signal. Ideally, two speech samples that differ only in channel or speaker effects would result in the same sequence of phone tokens, although, as most of these tokenizers are HMM-based, the resulting token sequences can be highly affected by speaker and channel differences [6][7].

In this paper we explore methods aimed at eliminating or normalizing non-linguistic sources of variability in phonotactic language recognition systems in hopes these techniques will allow a standard PRLM (Phone Recognition followed by Language Modeling) system to better model language differences. Specifically, we explore the use of Speaker Adaptation and Vocal Tract Length Normalization (VTLN) techniques borrowed from research in Automatic Speech Recognition (ASR). Using these techniques we conducted a series of experiments on the LRE 2005 corpus comparing our new adaptation framework to our standard PRLM-based baseline.

2. Language Recognition System

A standard phone recognition followed by language modeling (PRLM) approach is used for our language recognition system. While using multiple phone recognizers in parallel (P-PRLM) has also been effective, the aim of this paper is to explore the language recognition gains that can be achieved with a single phone tokenizer.

2.1. HMM-based Phone Recognition

For phone recognition, we use a standard monophone HMM-based system. Each phone is modeled with a three-state, left-to-right network (no skips, single-state inter-word silence). The basic HMM model is defined by equation 1:

\[ P(O|\lambda) = \sum_{\forall s} \prod_{t=1}^{T} P(o_t|s_t, \lambda) \times P(s_t|s_{t-1}, \lambda) \]  

where \( \lambda \) are parameters of the model and \( O = (o_1, \ldots, o_T) \) is a sequence of observation vectors (PLP-derived cepstra + derivative and acceleration) of length \( T \) generated by paths \( s \) is a hidden sequence of states through the HMM(s).

State-dependent observation probabilities \( P(o_t|s_t, \lambda) \) are modeled as a continuous density mixture of Gaussians:

\[ P(o_t|s_t) = \sum_{m=1}^{M} \mathcal{N}(\mu_{mj}, \Sigma_{mj}) \]  

Both observation and state transition probabilities, \( P(s_t|s_{t-1}) \), are trained using the Baum-Welch algorithm. Details of the phone recognizers used in this paper are given in Section 5.

During phone recognition, the goal is to find the sequence of phones \( W = (w_1, \ldots, w_N) \) that maximizes the likelihood of an input observation sequence \( O \). In addition to the single best sequence of phones, a lattice encoding many alternative sequences can also be produced. Note, unlike the standard phone or word recognition task, we use no grammar constraints during decoding since the aim for language recognition is to model and classify the language-dependent phonotactic distributions learned from an open-loop phone token stream.

2.2. PRLM Language Recognition

Given the maximum likelihood token sequence \( W \) from the phone recognizer, the basic PRLM detection rule for the hy-
proposed language $L$ is to accept if

$$P_L(W|O) > \theta$$

The denominator is computed using other competing language models $P_L(W|O) = \sum_{\nu \neq L} P_L(W|O)$. The term $P_L(W|O)$ can be decomposed using Bayes’ rule:

$$P_L(W|O) = \frac{P(O|W) * P_L(W)}{P(O)}$$

For a single tokenizer, the language-independent terms $P(O|W)$ and $P(O)$ cancel out, leaving only the language-dependent terms $P_L(W)$ to be computed for a detection decision. In this paper, $P_L(W)$ is approximated by an n-gram language model

$$P_L(W) = \prod_{w \in W} P_L(w_i|w_{i-1}, \ldots, w_{i-n+1})$$

with the n-gram probabilities estimated by relative frequencies over training data for each language.

### 2.3. Lattice-based PRLM

An extension of the 1-best PRLM model, proposed in [8], incorporates the posterior probabilities of phone tokens from a phone lattice into the estimation and scoring of the language models. In this formulation, n-gram counts from the 1-best hypothesis are replaced with expected counts from a phone lattice generated by the decoding of a given utterance [3]:

$$\log P_L(W) \approx \sum_{w \in W} \log P_L(w_i|w_{i-1}, \ldots, w_{i-n+1})$$

All experiments reported use the lattice-based PRLM system.

### 3. Linear Transforms for Speaker Adaptation

As mentioned previously, nuisance variables, like speaker and channel, can affect the consistency and quality of the phone decoder thus degrading the performance of the resulting PRLM system. In this section, we describe the application of Maximum Likelihood Linear Regression (MLLR) methods, commonly used in ASR for speaker/channel adaptation, to our phone recognition system. In this framework, a linear transform of the HMM observation model parameters is estimated per test utterance to maximize its likelihood $P(O_T|\lambda)$ [9].

#### 3.1. MLLR (Mean Only)

In its simplest form, MLLR can be applied to the Gaussian mean parameters of the HMM observation model. A linear transform $W$ is applied so as to shift and rotate each Gaussian component of the HMM model, with covariance parameters left unaltered. Different transforms can be applied to individual Gaussians, or classes of Gaussians [10]. In subsequent experiments, we will estimate two transforms $W_{sil}$ and $W_{speech}$ for silence and speech classes respectively.

The MLLR transform applied to the Gaussian mean vector $\mu$ is

$$\hat{\mu} = A_r \mu + b_r = W_r \xi$$

where $\xi = [\mu_1 \mu_2 \ldots \mu_n]^T$, $n$ is the dimensionality of the observation features, and $W_r = [b_r \ A_r]$ is the transform for Gaussians of class $r$. $W_r$ is found using the EM algorithm [10].

#### 3.2. Constrained MLLR

In [11] a constrained variant of MLLR (CMLLR) was proposed. In this formulation, it is assumed that mean and covariance parameters are governed by one transforms as follows:

$$\hat{\mu} = A'_r \mu + b'_r$$

$$\Sigma = A'_r \Sigma A'_r^T$$

where $A'_r = A_r^{-1}$ and $b'_r = -A_r b_r$. The CMLLR parameters are estimated using a procedure similar to that used for mean-only MLLR parameter estimation [10]. The constrained transform $W_r = [b_r A_r]$ can be efficiently applied in the feature domain as $\hat{o}(t) = A_r o(t) + b_r = W_r \zeta$, where $\zeta = [1 \ o_1 \ o_2 \ \ldots \ o_n]^T$. CMLLR transforms can be applied when decoding or training as discussed in the following section.

#### 3.3. Speaker Adaptive Training

In ASR systems it is common to use CMLLR during training to arrive at speaker adapted (SAT) models [12]. Using CMLLR, we trained speaker adaptive monophone models for phone tokenization using the following procedure:

2. **Train Speaker Transforms** – From SI models and reference transcripts
3. **Train SAT Models** – Using CMLLR transforms applied to observations on a per speaker basis
4. **Iterate Transform/Model Training (2-3)**

For the experiments described in this paper, we trained our phone models using transcripts and audio from Switchboard-II, phase 4 (cellular) and TIMIT.

#### 4. Vocal Tract Length Normalization

In addition to MLLR adaptation, we applied vocal tract length normalization (VTLN) to our phone tokenization process. VTLN attempts to warp features on a per speaker basis to compensate for vocal tract length differences typically associated with gender. Warping is typically implemented as a non-linear or piece-wise linear frequency mapping controlled by a single parameter $\alpha$. In this work, we use a piece-wise linear warping function.

During recognition, without supervision, the VTLN process involves finding a warp factor for a test message subject to:

$$\hat{\alpha} = \arg \max_{\alpha} \mathcal{L}(O|\lambda, \alpha)$$

Then warping observations $O$ with $\alpha$. The likelihood $\mathcal{L}$ can be estimated using the recognizer’s HMM models or by using a separate proxy model (typically a GMM, see [13][14]). For experiments conducted here, we use HMM-based likelihood estimation.

As reported in [15], application of VTLN during both training and decoding provides a significant improvement over applying VTLN during decoding alone. We used the following procedure when applying VTLN in both phone recognition training and decoding.

**Training Procedure**
- **Train Gender Independent (GI) Model**
- **Gender Normalized Training** – Init with GI Model
### 5. Experiments

In this section we present two sets of results using phone recognition with various combinations of the above adaptation/normalization procedures. First, to validate our implementation of these adaptation procedures and gauge underlying phone recognition accuracy, we conducted phone recognition experiments using both clean (TIMIT) and telephone (Switchboard) speech. Second we present a series of results showing the effect of these adaptations for language recognition on the 2005 NIST Language Recognition Evaluation (LRE) corpus.

#### 5.1. Phone Recognition using Adapted Monophones

We conducted a number of experiments to validate our adaptive training and decoding procedures using Switchboard II phase 4, cellular (SWB-CELL) and TIMIT data sets. In these experiments, we assessed the phone error rates of adapted, unadapted and VTLN monophone models.

The configuration of our SWB-CELL and TIMIT recognizers is shown in Tables 1 and 2 respectively. The TIMIT configuration follows protocols from [7] and speaker transforms were trained using all utterances from a given test set speaker. For SWB-CELL, individual sides were used to train speaker transforms. In both configurations, no language model is used (i.e. all phones are equiprobable at all times).

<table>
<thead>
<tr>
<th>Front End</th>
<th>PLP-13 + 1st and 2nd Delta</th>
</tr>
</thead>
<tbody>
<tr>
<td>Models</td>
<td>39 phone models, 3-states, 20g per state</td>
</tr>
<tr>
<td>Adaptation</td>
<td>SAT training, CMLLR and MLLR, VTLN</td>
</tr>
<tr>
<td>Training Data</td>
<td>3.5 hours (phonetically transcribed)</td>
</tr>
<tr>
<td>Phone Test Set</td>
<td>1.4 hours, 15k total word instances</td>
</tr>
</tbody>
</table>

Table 1: TIMIT Train/Test configuration

<table>
<thead>
<tr>
<th>Front End</th>
<th>PLP-13 + 1st and 2nd Delta</th>
</tr>
</thead>
<tbody>
<tr>
<td>Models</td>
<td>47 phone models, 3-states, 31g per state</td>
</tr>
<tr>
<td>Adaptation</td>
<td>SAT training, CMLLR and MLLR, VTLN</td>
</tr>
<tr>
<td>Training Data</td>
<td>23 hours (word transcription)</td>
</tr>
<tr>
<td>Phone Test Set</td>
<td>1.5 hours, 40k total word instances</td>
</tr>
</tbody>
</table>

Table 2: SWB-CELL Train/Test configuration

#### 5.2. Language Recognition

The 2005 LRE task was the detection of the presence of one of seven languages in speech utterances with nominal duration 30s, 10s and 3s. The speech was collected at OHSU using both domestic and international telephone lines. Details of the 2005 LRE corpus and NIST evaluation can be found in [5].

For the following experiments, we used the SWB-CELL monophone recognizer used in the MIT-LL LRE 2005 submission [3]. Tri-gram language models were trained using speech from 13 languages (7 targets plus 6 others) taken from the CALLFRIEND, CALLHOME, FISHER and MIXER corpora. For some experiments a backend classifier was applied that consisted of a LDA transform over the vector of 13 scores and a per-language diagonal Gaussian classifier, both trained on development data. Likelihood ratios are then computed between the target and non-target scores to produce the final detection score.

The results shown in Table 5 were obtained on 30s task (primary condition) with and without a backend classifier. All the adaptations explored here improve the performance of our PRLM system (25% relative with a backend, 29% without). The largest single improvement, CMLLR, results in a 15.3% relative EER gain.

#### Table 3: TIMIT phone recognition results

<table>
<thead>
<tr>
<th>Model Type</th>
<th>CMLLR</th>
<th>MLLR</th>
<th>VTLN</th>
<th>Phone Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>SI</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>33.5%</td>
</tr>
<tr>
<td>SAT</td>
<td>√</td>
<td>-</td>
<td>-</td>
<td>31.4%</td>
</tr>
<tr>
<td>SAT</td>
<td>-</td>
<td>√</td>
<td>-</td>
<td>29.2%</td>
</tr>
<tr>
<td>SAT</td>
<td>√</td>
<td>√</td>
<td>-</td>
<td>27.5%</td>
</tr>
</tbody>
</table>

#### Table 4: SWB-CELL phone recognition results

<table>
<thead>
<tr>
<th>Model Type</th>
<th>CMLLR</th>
<th>MLLR</th>
<th>VTLN</th>
<th>PER</th>
</tr>
</thead>
<tbody>
<tr>
<td>SI</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>73.6%</td>
</tr>
<tr>
<td>SAT</td>
<td>√</td>
<td>-</td>
<td>-</td>
<td>70.0%</td>
</tr>
<tr>
<td>SAT</td>
<td>√</td>
<td>√</td>
<td>-</td>
<td>69.6%</td>
</tr>
<tr>
<td>SAT</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>67.6%</td>
</tr>
</tbody>
</table>

#### Table 5: Language recognition results with different adaptation methods (LRE05 30s Primary)

<table>
<thead>
<tr>
<th>Model Type</th>
<th>CMLLR</th>
<th>MLLR</th>
<th>VTLN</th>
<th>EER w/o BE</th>
<th>EER w/BE</th>
</tr>
</thead>
<tbody>
<tr>
<td>SI</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>8.5%</td>
<td>7.1%</td>
</tr>
<tr>
<td>SAT</td>
<td>√</td>
<td>-</td>
<td>-</td>
<td>7.2%</td>
<td>5.5%</td>
</tr>
<tr>
<td>SAT</td>
<td>-</td>
<td>√</td>
<td>-</td>
<td>6.9%</td>
<td>5.3%</td>
</tr>
<tr>
<td>SAT</td>
<td>√</td>
<td>√</td>
<td>-</td>
<td>6.0%</td>
<td>5.5%</td>
</tr>
</tbody>
</table>

#### 6. Discussion

Both speaker adaptation and gender normalization improve language recognition performance. Interestingly, without an additional backend classifier, PRLM performance is well correlated with phone error rate in these experiments. This suggests that

---

1. English, Hindi, Japanese, Korean, Mandarin, Spanish, Tamil
adaptation may be correcting for some of the non-language-related variation in the token output of the phone recognizer. In doing so, the resulting phone sequences or lattices, may be more amenable to language modeling.

It is noteworthy that gains due to VTLN seem to interact with the backend classifier. As Figures 2 and 1 suggest, the VTLN system shows minimal gain when a backend classifier is applied. This is not true of non-VTLN systems (see Table 5). We speculate that systematic score differences result from VTL differences in non-VTLN PRLM systems, allowing the backend classifier to normalize VTL effects.

Other techniques that normalize non-language variation may also improve language recognition performance. It remains to be seen whether techniques like nuisance projection (applied at the token n-gram level) or front-end channel normalization techniques could also improve performance.

7. References


