Acoustic Data-driven Pronunciation Lexicon for Large Vocabulary Speech Recognition

Citation for published version:
Lu, L, Ghoshal, A & Renals, S 2013, Acoustic Data-driven Pronunciation Lexicon for Large Vocabulary Speech Recognition. in Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on. Institute of Electrical and Electronics Engineers (IEEE). https://doi.org/10.1109/ASRU.2013.6707759

Digital Object Identifier (DOI):
10.1109/ASRU.2013.6707759

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.
ACOUSTIC DATA-DRIVEN PRONUNCIATION LEXICON FOR LARGE VOCABULARY SPEECH RECOGNITION

Liang Lu, Arnab Ghoshal, and Steve Renals

Centre for Speech Technology Research, University of Edinburgh, Edinburgh, EH8 9AB, UK
{liang.lu, a.ghoshal, s.renals}@ed.ac.uk

ABSTRACT
Speech recognition systems normally use handcrafted pronunciation lexicons designed by linguistic experts. Building and maintaining such a lexicon is expensive and time consuming. This paper concerns automatically learning a pronunciation lexicon for speech recognition. We assume the availability of a small seed lexicon and then learn the pronunciations of new words directly from speech that is transcribed at word-level. We present two implementations for refining the putative pronunciations of new words based on acoustic evidence. The first one is an expectation maximization (EM) algorithm based on weighted finite state transducers (WFSTS) and the other is its Viterbi approximation. We carried out experiments on the Switchboard corpus of conversational telephone speech. The expert lexicon has a size of more than 30,000 words, from which we randomly selected 5,000 words to form the seed lexicon. By using the proposed lexicon learning method, we have significantly improved the accuracy compared with a lexicon learned using a grapheme-to-phoneme transformation, and have obtained a word error rate that approaches that achieved using a fully handcrafted lexicon.

Index Terms— Lexical modelling, Probabilistic pronunciation model, Automatic speech recognition.

1. INTRODUCTION
Training a speech recognition system relies on three principal resources: transcribed acoustic training data; text data for language model estimation; and a pronunciation lexicon that maps a word to one or more phonemic transcriptions. Obtaining such resources usually requires significant manual intervention, making the development of a speech recogniser for a new language, or a new domain, a costly process. The development of speech recognition systems for languages or domains with limited resources has become a major research focus in the past few years. Encouraging results have been reported for acoustic modelling using very limited amounts of transcribed audio for a new target language, e.g. by leveraging the acoustic data of other languages using multi-layer perceptrons [1, 2] or subspace Gaussian mixture models [3, 4].

The pronunciation lexicon usually has a large, finite vocabulary and is handcrafted by linguistic expert. Building a lexicon is normally time consuming and expensive: moreover, the expert pronunciation lexicon is usually fixed during the training and application of an ASR system. Updating the lexicon to cover additional words is not a trivial task. Automatically learning the pronunciation lexicon has been pursued for more than a decade, with the original focus being on the learning of pronunciation variations or alternative pronunciations for some words [5, 6, 7, 8]. More recently, McGraw, et al. [9] proposed a stochastic pronunciation mixture model framework to automatically update the pronunciation weights of the words in the lexicon. Such approaches assume the availability of a high-quality initial pronunciation dictionary and aim to add suitable pronunciation variations of words beyond the canonical pronunciations present in the dictionary.

There have been previous attempts to move beyond phonemic baseforms and jointly learn the inventory of subword units and the pronunciation lexicon. For instance, Bacchiani and Ostendorf [10] proposed an iterative acoustic segmentation and clustering approach to build a sub-word inventory from the acoustics and then automatically construct a dictionary based on those sub-word units. Later on, Singh, et al. [11] presented an expectation-maximisation (EM) algorithm for this purpose and demonstrated some promising results on the relatively small resource management (RM) corpus.

While learning the entire lexicon from scratch is challenging for large vocabulary speech recognition task, a more practical technique is to enlarge an expert phonemic lexicon by learning the pronunciations of additional words and update the acoustic model based on this updated lexicon. This approach was used in learning pronunciations of names [12, 13], or learning pronunciations of all types of words starting from a small seed lexicon [14, 15]. The work reported in this paper follows this general approach. We start with a seed lexicon and use a grapheme-to-phoneme (G2P) converter [16] to generate multiple pronunciations for new words. We present a WFST-based EM algorithm to estimate the weights of these alternative pronunciations based on acoustic evidence, and compare with its Viterbi approximation. On the 300-hour Switchboard task we start with a random 5000-word subset of the 30,000-word expert lexicon and show that the proposed method is able to learn the pronunciations of the remaining words with only a small reduction in accuracy.

2. PROBABILISTIC PRONUNCIATION MODEL
Given the acoustic observations \( \mathbf{O} \), the optimal word sequence \( \hat{\mathbf{W}} \) is obtained from a conventional ASR engine as

\[
\hat{\mathbf{W}} = \arg \max_{\mathbf{W}} \ p(\mathbf{O}|\mathcal{M}, \mathbf{W})P(\mathbf{W}),
\]

where \( \mathcal{M} \) denotes the acoustic model parameters, and \( P(\mathbf{W}) \) is the prior probability for word sequence \( \mathbf{W} \), normally obtained from the language model. In order to improve the modelling accuracy as well as to generalise to unseen words in the training dataset, the acoustic model \( \mathcal{M} \) usually operates at the level of sub-word units (typically context-dependent phonemes) rather than words. The mapping from each word to its corresponding phonemic transcription is usually defined by a handcrafted pronunciation lexicon in which most of the words have a single canonical pronunciation.

This work was funded by the EPSRC Programme Grant EP/I031022/1 (Natural Speech Technology).
When the dictionary contains multiple pronunciations per word corresponding to pronunciation weights that form a valid probability distribution, the decision rule (1) may be written as:

$$W = \arg \max \mathbf{W} P(W) \prod_{B \in \Psi_W} p(O|\mathcal{M}, B) P(B|W),$$  \hspace{1cm} (2)

where $B = \{b_1, \ldots, b_n\}$ denotes a valid pronunciation sequence for the word transcription $W = \{w_1, \ldots, w_n\}$, and $P(B|W)$ denotes its probability. $b_r$ is the pronunciation of word $w_r$. $\Psi_W$ denotes the set of all possible pronunciation sequences of $W$. This is same as the formulation used in [12, 13], and in [9] where the dictionary is viewed as containing all possible phonetic realisations of a word and the model is referred to as a pronunciation mixture model.

The pronunciation of words may depend on the surrounding words. However a reasonable simplification is to ignore this effect and express $P(B|\mathbf{W})$ as the product of the pronunciation probabilities of each word:

$$P(B|\mathbf{W}) = P(b_1|w_1) \cdots \prod_{i=1}^{n} P(b_i|w_i).$$  \hspace{1cm} (3)

Like in [9], we assume that each word may have multiple surface pronunciations with a corresponding pronunciation weight. More formally, it may be expressed as

$$P(b_i = p_j|w_i) = \theta_{ij}, \quad j = 1, \ldots, J_i$$

subject to: $$\sum_j \theta_{ij} = 1.$$  \hspace{1cm} (4)

where $J_i$ is the number of alternate pronunciations of $w_i$, and $p_j$ denotes one of those surface pronunciations with a weight $\theta_{ij}$.

### 2.1. Pronunciation weight estimation using EM

Given a pronunciation dictionary, an acoustic model $\mathcal{M}$, and some transcribed acoustic data, the pronunciation weights $\theta_{ij}$ can be updated using the EM algorithm [11, 12, 13, 9]. If $O_r$ represents the acoustic observations and $W_r$ the corresponding word-level transcription for $r = 1, \ldots, R$ training utterances, then the pronunciation weight $\theta_{ij}$ for a particular word and pronunciation pair $(w_i, p_j)$ can be optimized using the following EM auxiliary function:

$$Q(\theta_{ij}) = \sum_{r=1}^{R} \sum_{B_r \in \Psi_{W_r}} P(B_r|O_r, M, W_r) \log P(B_r|W_r) + k$$

$$= \sum_{r=1}^{R} \sum_{B_r \in \Psi_{W_r}} P(B_r|O_r, M, W_r) C_{ijr}|B_r \log \theta_{ij} + k$$

where $C_{ijr}|B_r$ denotes the number of times that $(w_i, p_j)$ appears in the pronunciation sequence $B_r$. $\Psi_{W_r}$ represents the subset of pronunciation sequences that contain $(w_i, p_j)$, and $k$ is a constant which does not depend on $\theta_{ij}$. The posterior probability of the pronunciation sequence $B_r$ can be computed according to Bayes’ rule:

$$P(B_r|O_r, M, W_r) = \frac{p(O_r|B_r, M) P(B_r|W_r)}{\sum_{B_r \in \Psi_{W_r}} p(O_r|B_r, M) P(B_r|W_r)}$$

where $P(B_r|W_r)$ is calculated using the old estimate of the pronunciation weights. Using this, the auxiliary function $Q(\theta_{ij})$ may be rewritten as

$$Q(\theta_{ij}) = \sum_{r=1}^{R} \lambda_{ijr} \log \theta_{ij} + k$$  \hspace{1cm} (7)

where $\lambda_{ijr}$ represents the following term:

$$\lambda_{ijr} = \frac{\sum_{B_r \in \Psi_{W_r}} p(O_r|B_r, M) P(B_r|W_r) C_{ijr}|B_r}{\sum_{B_r \in \Psi_{W_r}} p(O_r|B_r, M) P(B_r|W_r)}.$$

Maximising this under the constraint that $\sum_j \theta_{ij} = 1$, the new value of $\theta_{ij}$ is obtained as:

$$\hat{\theta}_{ij} = \frac{\sum_{r=1}^{R} \lambda_{ijr}}{\sum_{r=1}^{R} \sum_j \lambda_{ijr}}$$  \hspace{1cm} (9)

However, the computation of $\lambda_{ijr}$ may be excessive, since the size of the pronunciation sequence space $\Psi_{W_r}$ is exponential in the length of $W_r$, which prohibits an exhaustive enumeration of all possible sequences. To reduce the computational requirement, [13] and [9] propose approximating $\Psi_{W_r}$ with an N-best list of alternate pronunciation sequences generated by a G2P model, and rescoring this N-best list with the acoustic model. However, this is still costly due to the repeated evaluation of $p(O_r|B_r, M)$ for each pronunciation sequence $B_r$ on the N-best list. Moreover, it may introduce a loss in accuracy since the search space is reduced, especially for shallower N-best lists.

Instead of N-best pronunciation sequences, we propose using speech recognition lattices. Lattices are an efficient representation of an exponential number of alternatives using linear space. Moreover, lattices only contain pronunciation variants that have sufficiently high likelihood given the acoustics and hence we do not need to rescore very unlikely pronunciation variants as in the N-best list approach. It is important to note that since we do not do an unconstrained decoding [see section 2.2], the concern about learning "linguistically incorrect pronunciations" expressed in [13] is not applicable here. Another motivation for using lattices is that efficient polynomial time algorithms exist for computing posterior probabilities and the value of $C_{ijr}|B_r$ over lattices. In the following subsection we present a WFST-based formulation to do so.

### 2.2. WFST-based pronunciation weight estimation

The space of all possible pronunciation sequences for a word sequence $W_r$ may be represented as a weighted finite state transducer:

$$\mathcal{P}_r = \min(\det(\mathcal{L} \circ \mathcal{G}_r)),$$

where $\mathcal{L}$ is the lexicon transducer that maps the words to their corresponding pronunciations; $\mathcal{G}_r$ is a linear acceptor representing $W_r$; $\det$ and $\circ$ denote the determination and minimisation operations respectively; and $\circ$ is the FST composition operation [18]. The pronunciation variants in $\mathcal{P}_r$ are scored by running a recogniser over the decoding graph $D_r = \min(\det(\mathcal{H} \circ \mathcal{C} \circ \mathcal{P}_r))$, where $\mathcal{C}$ is the context-dependency transducer and $\mathcal{H}$ represents the HMM set. The hypothesis space of the recogniser containing the likelihoods of the pronunciation variants is obtained as a phone lattice [19]. The arcs of this lattice contain both acoustic likelihoods and pronunciation weights, and the lattice is further converted such that the arc costs form a log semiring [18]. This lattice is, in practise, the $\Psi_{W_r}$ in equation (8).

The denominator of (8) can be computed by using the OpenFST [20] tool $\text{fstshortestdistance}$ on $\Psi_{W_r}$, which computes the summation of the weights of all the paths since $\Psi_{W_r}$ is in log semiring. Its computational complexity is linear in the number of states.

\[1\] Although not relevant to the current work, such non-canonical pronunciations may not be a limitation as long as they are consistent [17].
and arcs for the class of acyclic transducers [18] that $Ψ_{W}$ belongs to. To compute the numerator of equation (8), we borrow the idea of path counting transducer used in language modelling [21]. We construct a path counting transducer for each pronunciation to select the paths in $Ψ_{W}$ containing the pronunciation and to accumulate their weights using WFST composition. This is illustrated in Figure 1.

However, this approach fails for pronunciations that are subsequences of other pronunciations. For instance, in Figure 1 if both $p_{1} = \{a b\}$ and $p_{2} = \{a \ b \ c\}$ are valid pronunciations for the word we are interested in, then the counts for $p_{1}$ will also include the paths that actually contain $p_{2}$. This is mostly remedied by simply having word-position markers on the phonemes. Examples are given in Table 1, where the suffix labels “S”, “B”, “I” and “E” represent word beginning, internal and end phones, respectively.

<table>
<thead>
<tr>
<th>words</th>
<th>pronunciation</th>
<th>type</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>ay</td>
<td>position-independent</td>
</tr>
<tr>
<td>able</td>
<td>ey b as l</td>
<td>position-dependent</td>
</tr>
<tr>
<td></td>
<td>ey b a as l</td>
<td>position-dependent</td>
</tr>
<tr>
<td></td>
<td>ey b a as l, I</td>
<td>position-dependent</td>
</tr>
</tbody>
</table>

Table 1. Pronunciation labels using position-dependent and position-independent phones, where “S” indicates a singleton phone, and “B”, “I” and “E” represent word beginning, internal and end phones, respectively.

In this case only the most likely path needs to be obtained from the decoder instead of the lattice. The value of $C_{ij}\hat{B}_{r}$ can be easily obtained by aligning the word with its corresponding pronunciation according to the lexicon. Again, we use position-dependent phone labels so that we can obtain the word boundary for the alignment.

In this case only the most likely path needs to be obtained from the decoder instead of the lattice. The value of $C_{ij}\hat{B}_{r}$ can be easily obtained by aligning the word with its corresponding pronunciation according to the lexicon. Again, we use position-dependent phone labels so that we can obtain the word boundary for the alignment.

In this case only the most likely path needs to be obtained from the decoder instead of the lattice. The value of $C_{ij}\hat{B}_{r}$ can be easily obtained by aligning the word with its corresponding pronunciation according to the lexicon. Again, we use position-dependent phone labels so that we can obtain the word boundary for the alignment.

In this case only the most likely path needs to be obtained from the decoder instead of the lattice. The value of $C_{ij}\hat{B}_{r}$ can be easily obtained by aligning the word with its corresponding pronunciation according to the lexicon. Again, we use position-dependent phone labels so that we can obtain the word boundary for the alignment.

In this case only the most likely path needs to be obtained from the decoder instead of the lattice. The value of $C_{ij}\hat{B}_{r}$ can be easily obtained by aligning the word with its corresponding pronunciation according to the lexicon. Again, we use position-dependent phone labels so that we can obtain the word boundary for the alignment.

In this case only the most likely path needs to be obtained from the decoder instead of the lattice. The value of $C_{ij}\hat{B}_{r}$ can be easily obtained by aligning the word with its corresponding pronunciation according to the lexicon. Again, we use position-dependent phone labels so that we can obtain the word boundary for the alignment.

In this case only the most likely path needs to be obtained from the decoder instead of the lattice. The value of $C_{ij}\hat{B}_{r}$ can be easily obtained by aligning the word with its corresponding pronunciation according to the lexicon. Again, we use position-dependent phone labels so that we can obtain the word boundary for the alignment.

In this case only the most likely path needs to be obtained from the decoder instead of the lattice. The value of $C_{ij}\hat{B}_{r}$ can be easily obtained by aligning the word with its corresponding pronunciation according to the lexicon. Again, we use position-dependent phone labels so that we can obtain the word boundary for the alignment.

In this case only the most likely path needs to be obtained from the decoder instead of the lattice. The value of $C_{ij}\hat{B}_{r}$ can be easily obtained by aligning the word with its corresponding pronunciation according to the lexicon. Again, we use position-dependent phone labels so that we can obtain the word boundary for the alignment.

In this case only the most likely path needs to be obtained from the decoder instead of the lattice. The value of $C_{ij}\hat{B}_{r}$ can be easily obtained by aligning the word with its corresponding pronunciation according to the lexicon. Again, we use position-dependent phone labels so that we can obtain the word boundary for the alignment.

In this case only the most likely path needs to be obtained from the decoder instead of the lattice. The value of $C_{ij}\hat{B}_{r}$ can be easily obtained by aligning the word with its corresponding pronunciation according to the lexicon. Again, we use position-dependent phone labels so that we can obtain the word boundary for the alignment.

3. ITERATIVE LEARNING OF LEXICON AND ACOUSTIC MODELS

We follow an iterative training schedule, where the pronunciation weights are estimated given the acoustic model and the acoustic model is re-estimated given the newly selected set of pronunciations. This alternating training schedule is commonly used in literature [11, 10, 14] instead of direct joint estimation, which is computationally cumbersome besides being unlikely to provide any accuracy gains. Starting from a seed lexicon, we train a grapheme-to-phoneme (G2P) model [16] to generate multiple alternative pronunciations of the words in the training set. The pronunciation weights are initialised to zero, and end phones, respectively. But the word position markers cannot account for the special case of homophones, i.e. when the subsequences are of the same length and hence identical even with position markers. It is possible to create a time-sensitive counting transducer2 but in this work we removed the utterances containing homophones from the training data. This removed around 15–30% of training data depending on the number of alternative pronunciations for each word.

2This can be implemented using a specialized Matcher class in OpenFST that uses word-timing information.
1. Train the G2P model
2. Generate the lexicon
3. Train the acoustic model
4. Update the lexicon
5. Update the acoustic model
6. Update the G2P model

Fig. 2. An iterative training scheme to learn the lexicon and acoustic models.

With multiple pronunciations per word, the EM estimation of the acoustic model $\hat{M}$ maximises the following objective:

$$\hat{M} = \arg \max_{M} \sum_{r=1}^{R} \sum_{B_r \in \Psi_{W_r}} \log p(O_r|M, B_r) P(B_r|W_r), \quad (14)$$

where the sufficient statistics are accumulated from all the possible pronunciation sequences $B_r \in \Psi_{W_r}$ for each utterance $W_r$. However, to simplify the implementation and to reduce the computational cost, the Viterbi approximation is used in practice where the sufficient statistics are accumulated only from the most likely pronunciation sequence:

$$\hat{M} = \arg \max_{M} \max_{B_r} \sum_{r=1}^{R} p(O_r|M, B_r) P(B_r|W_r). \quad (15)$$

To sum up, the recipe used in this paper is shown in Figure 2 and is detailed as follows:

1. Train the G2P model using the available initial seed lexicon.
2. For each word in the training transcriptions, generate up to $N$ pronunciations using the G2P model.
3. Train the acoustic model using the current lexicon.
4. Update the pronunciation weights using the algorithm described in Section 2 and prune those pronunciations whose weights are below a threshold $\theta_{ij} \leq T_1$.
5. Go to step 3 to re-train the acoustic model with the new lexicon until the maximum number of iterations is reached.
6. Go to step 1 to re-train the G2P model with those pronunciations whose weight is above the threshold $\theta_{ij} \geq T_2$ until the maximum number of iterations is reached.

In the experiments described below, we will show that only a few iterations of joint acoustic and lexicon model training is sufficient for the system to converge according to this recipe.

<table>
<thead>
<tr>
<th>Word</th>
<th>Initial Pronunciations</th>
<th>Updated Pronunciations</th>
</tr>
</thead>
<tbody>
<tr>
<td>abilities</td>
<td>ae b ih l ih t iy z 0.2</td>
<td>ax b ih l ih t iy z 1.0</td>
</tr>
<tr>
<td></td>
<td>ae ey b ih l ih t iy z 0.2</td>
<td>ax b ih l ih t iy z 0.2</td>
</tr>
<tr>
<td></td>
<td>ay ax b ih l ih t iy z 0.2</td>
<td>ax b ih l ih t iy z 0.2</td>
</tr>
<tr>
<td></td>
<td>s ax b ih l ih t iy z 0.2</td>
<td>ax b ih l ih t iy z 0.2</td>
</tr>
<tr>
<td>aboveboard</td>
<td>ax b ah ax v b ow r d 0.2</td>
<td>ax b ah ax v b ow r d 0.62</td>
</tr>
<tr>
<td></td>
<td>ax b ah v b ow r d 0.2</td>
<td>ax b ah v b ow r d 0.38</td>
</tr>
<tr>
<td></td>
<td>ax b ah v ey b ow r d 0.2</td>
<td>ax b ah v ey b ow r d 0.2</td>
</tr>
<tr>
<td></td>
<td>ax b ah v l b ow r d 0.2</td>
<td>ax b ah v l b ow r d 0.2</td>
</tr>
<tr>
<td>instance</td>
<td>ih n st ae n s 0.2</td>
<td>ih n st ax n s 0.72</td>
</tr>
<tr>
<td></td>
<td>ih n st ae n s ih 0.2</td>
<td>ih n st ih n s 0.28</td>
</tr>
<tr>
<td></td>
<td>ih n st ax n s 0.2</td>
<td>ih n st ih n s 0.2</td>
</tr>
<tr>
<td></td>
<td>ih n st en s 0.2</td>
<td>ih n st ih n s 0.2</td>
</tr>
<tr>
<td></td>
<td>ih n st ih n s 0.2</td>
<td>ih n st ih n s 0.2</td>
</tr>
</tbody>
</table>

4. EXPERIMENTS AND RESULTS

We performed experiments on the Switchboard corpus, where the training set contains about 300 hours of conversational telephone speech. The Hub-5 Eval 2000 data is used as the test set. We used the Kaldi speech recognition toolkit [22] to train conventional GMM-based acoustic models following the recipe described in [23]. 39-dimensional MFCC+$\Delta$+$\Delta\Delta$ features were used, with a context window of 7 frames to which linear discriminant analysis transformations were then applied to reduce the feature dimensionality to be 40, followed by a global semi-tied covariance matrix transform [24] to decorrelate the features. The expert lexicon was supplied by the Mississippi State transcriptions and it has more than 30,000 words, of which a random subset of 5,000 words is used as the seed lexicon.

4.1. Results: WFST-based training

We first evaluate WFST-based training for lexicon learning. The initial benchmarking and tuning experiments were performed using a 110 hour subset of the Switchboard corpus, which has a vocabulary size of about 20,000 words. The acoustic model has around 3,900 tied triphone states and 90,000 Gaussian components overall. The systems were trained using the maximum likelihood (ML) criterion without speaker adaptation. The baseline system using the expert lexicon achieves a recognition word error rate (WER) of 37.2%. It is not feasible to build a system using the seed lexicon solely, since most of the words in the training data are missing from the vocabulary. To tackle this problem, we used the toolkit described in [16] and trained a G2P model using the 5,000 entries in the seed lexicon. We then generated the 1-best pronunciations for the missing words, and built a baseline system which has a WER of 42.1%. This system provides a reasonable baseline for the lexicon learning experiments.

Following the recipe in Section 3, we learned the pronunciations from the training data using the WFST-based training approach. We first used the G2P model described above to generate $N = 5$ pronunciations for each word, with all pronunciations for a word initialised to have equal weights. With the pronunciation dictionary initialised in this way, we obtained 47.0% WER, which is much higher than the G2P baseline due to the large number of alternative pronunciations which cause confusions. We then performed the iterative lexicon and acoustic model learning algorithm, and pruned those pronunciations.
whose weights were below the threshold $T_1 = 0.1$. After the model converged, the system achieved 41.7% WER which is slightly better than the G2P baseline.

Given the output of the new lexicon, we then selected those pronunciations with weights above the threshold $T_2 = 0.4$, and used them together with the seed lexicon to form a new training set to update the G2P model. This improved the accuracy of the G2P model. Following another two iterations of lexicon and acoustic model learning, we obtained a significantly reduced WER of 39.5%. Detailed results are shown in Figure 3. We observed that the training converges after 2 iterations. This may be due to the fact that we set an aggressive pruning threshold, $T_1 = 0.1$. However, using a lower threshold did not improve the accuracy but did make training a bit slower. Table 2 shows a few examples of the pronunciations that learned by the WFST-based system. Words seen in the training data have around 1.2 pronunciations on average after the training converges. With a lower threshold value, the average number of pronunciations is larger, however, it does not reduce the word error rate since it causes more pronunciation confusions. Using a discriminative learning criterion to learn pronunciations may prove helpful in this case [25, 13].
is optimised for the ML-trained models.

5. CONCLUSION

This paper is about building pronunciation lexicon for speech recognition using transcribed acoustic data. This paper presents a WFST-based EM algorithm and its Viterbi approximation for estimating pronunciation weights using acoustic evidence. To demonstrate the effectiveness of the lexicon learning method, experiments were performed on the Switchboard corpus which contains around 300 hours of conversational telephone speech. The expert lexicon has about a 30,000 word vocabulary, from which randomly selected 5,000 words are our seed lexicon. By expanding the seed lexicon, we obtained a WER that approaches that obtained using the expert lexicon; a constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed lexicon. A constraint of this work is the requirement of a seed Informatio...