Learning from Mistakes:
Expanding Pronunciation Lexicons using Word Recognition Errors

Sravana Reddy[1], *Evandro Gouvêa*

[1]Department of Computer Science, The University of Chicago, Chicago, IL, USA

sravana@cs.uchicago.edu, evandro.gouvea@alumni.cmu.edu

Abstract

We introduce the problem of learning pronunciations of out-of-vocabulary words from word recognition mistakes made by an automatic speech recognition (ASR) system. This question is especially relevant in cases where the ASR engine is a black box – meaning that the only acoustic cues about the speech data come from the word recognition outputs. This paper presents an expectation maximization approach to inferring pronunciations from ASR word recognition hypotheses, which outperforms pronunciation estimates of a state of the art grapheme-to-phoneme system.

Index Terms: speech recognition, pronunciation models, lexicon adaptation, out-of-vocabulary words

1. Introduction

Speech recognition systems often have to deal with out-of-vocabulary (OOV) words that are not represented in the pronunciation lexicon. One way of estimating pronunciations of OOVs is to use a grapheme-to-phoneme (g2p) model trained on the existing lexicon. The main disadvantage is that g2p systems tend to overfit to the lexicon, which is problematic when the OOV terms are proper nouns or loanwords whose pronunciations deviate from the standard grapheme-to-phoneme mappings.

An alternative is to directly estimate pronunciation from speech samples containing the OOV word. Previous works that take this approach make use of the phone lattice or the phoneme recognition output derived from the speech samples, sometimes constraining the pronunciations with a g2p model or language-specific rules [1, 2, 3, 4, 5, 6, 7].

In this paper, we ask whether pronunciations can be estimated from *word recognition mistakes*. When an ASR system is given an OOV word as input, it will obviously make an error – but its hypotheses will be phonetically similar to the original word, therefore providing cues to its pronunciation. For example, the proper name 'Marilyn' may be misrecognized as 'Maryland', 'Mary learns', 'parallel', 'Perelman', etc.

This question is mainly of interest in scenarios where we do not have access to the phone lattice or a phoneme recognizer, but are allowed to add words to the pronunciation lexicon, which is the case in many popular commercial speech recognition systems. For example, we envision a third-party application separate from the speech engine that takes the ASR recognition outputs that are flagged as errors by the user, and uses them to derive the word's pronunciation. This automation would save a non-expert user the time and difficulty of manually adding words to the pronunciation lexicon.

We describe an algorithm to find the pronunciation of an OOV word from a set of ASR recognition mistakes of one or more speech samples containing the word. The algorithm uses expectation maximization to maximize the phonetic similarity between the pronunciations of the ASR hypotheses and the OOV word's pronunciation. Our method is tested on a corpus of spoken proper names and is found to outperform g2p pronunciation estimates, measured both by word error rate in speech recognition on a held-out test set, and by phoneme error rate compared to gold standard phonetic transcriptions.

2. Learning from Mistakes

2.1. Generative Model

The approach we describe is similar to the pronunciation mixture model in [7], except that we do not have access to the acoustic models or phone lattices, only the word recognition mistakes. We focus on the task of isolated word recognition; however, this model can be easily extended to continuous speech with a few minor modifications.

A speech sample consisting of an OOV word is passed through an ASR decoder giving an n-best word recognition output. Each hypothesis in the n-best list may be a single word or a sequence of words. (Since the words are OOVs, every hypothesis will be a recognition mistake.)

To go from recognition mistakes to pronunciations, we start by describing below a generative story of the recognition output for a word w.

1. Pick a pronunciation baseform b according to the distribution θ, where $\theta_{b,w} = P(b|w)$.
2. Apply a phonetic confusion function from the word w and the selected baseform b to generate a phoneme sequence p with probability $P(p|b, w)$.
3. Generate a hypothesis word or word sequence e with probability $P(e|p, b, w) = 1$ using the pronunciation lexicon. (We assume e is generated deterministically from p).

Consider the name 'Marilyn', and one of the ASR recognition mistakes, 'Maryland'. The generative story describes a path from the word w to the recognition mistake e by picking a pronunciation baseform b for 'Marilyn' and perturbing it using phonetic confusions to form a phoneme sequence p which, according to the known lexicon, corresponds to 'Maryland'.

We have access to w and e. We assume access to a basic lexicon with pronunciations of the words in the system's vocabulary[1], which means we also know p (in this case, `M EH R`

Part of this work was performed while the authors were at Mitsubishi Electric Research Laboratories (MERL).

[1] While we may not have access to the actual lexicon of a black-box ASR system, it can be approximated by a publicly available dictionary.

AH L AH N D). The phonetic confusion function can be estimated from rules or external data. The only unknown value is the pronunciation b of the OOV 'Marilyn' – or more accurately, the distribution θ – which we would like to discover. (Of course, this generative model abstracts away the internal ASR processes, since we are assuming only black box access.)

2.2. Algorithm Description

Let E_w be the union of the n-best recognition mistakes from all instances of w in the speech data.

Let $f_{e,b,w}$ parametrize the phonetic confusion function; $f_{e,b,w} = P(e|b,w)$. Then, the joint probability of a hypothesis and a reference word is

$$P(e,w) = \sum_b P(e,b,w) = \sum_b P(e|b,w)P(b|w)P(w)$$
$$= P(w) \sum_b f_{e,b,w}\theta_{b,w} \quad (1)$$

The log likelihood of the n-best recognition mistakes from all OOVs in the reference transcriptions W is given by

$$\sum_{w \in W} \sum_{e \in E_w} \log P(w) \sum_b f_{e,b,w}\theta_{b,w} \quad (2)$$

$P(w)$ is simply the number of spoken instances of w divided by the total number of utterances. Let us assume that we are given the phonetic transformation function f, and a candidate set of baseforms B. We estimate the parameter θ by expectation maximization as described below:

Expectation Step

$$P(b|e,w) = \frac{f_{e,b,w}\theta_{b,w}}{\sum_{p \in B} f_{e,p,w}\theta_{p,w}} \quad (3)$$

Maximization Step Update

$$\theta_{b,w} = \frac{1}{|E_w|} \sum_{e \in E_w} P(b|e,w) \quad (4)$$

At convergence, the pronunciation for w that is chosen to be added to the lexicon is the most probable b.

We denote our system by **LFM**, short for 'learning from mistakes'. LFM may also be used to find *pronunciation variants* of words that are already in the lexicon; however, due to the risk of increased error rate when using a lexicon with too many variants, we restrict ourselves to expanding the lexicon with a very small number of pronunciations for each OOV term.

2.3. Initializing Candidate Baseforms

There are several possible ways of initializing the parameter θ to constrain the space of candidate baseform pronunciations. In this paper, we use a g2p-based approach. The joint n-gram 'Sequitur' g2p model described and implemented by Bisani and Ney [8] is trained on the existing pronunciation lexicon. For every OOV w, we initialize $\theta_{b,w} = 0$ for baseforms b that are assigned a probability of 0 by the g2p model, and *uniformly over all* b for which the model assigns > 0 probability.

In order to ensure that the candidate set of baseforms with non-zero probability is not too restrictive, we train the g2p model with multigrams (sequences of 'graphones') of order 2, which is a relatively weak setting, but ensures wide coverage.

2.4. Defining Phonemic Confusions

It is important that the function f (mapping the underlying word w and its baseform b to the hypothesis e) be permissive enough that the observed e has a non-zero chance of being generated from b, but constrained enough to be a meaningful phonetic transformation. Assuming that e only depends[2] on b, this amounts to defining the phonetic change from b to the concatenation of the pronunciations of the words in e, denoted by p.

We define the transformation as a finite state transducer that allows insertions, deletions, and substitutions of phonemes. The probabilities are derived by aligning phoneme recognition results on TIMIT [9] with the reference phone transcriptions, and mapping them to the CMU phoneset. The probabilities are smoothed by add-one smoothing, and $f_{e,b,w}$ is computed to be the probability of all paths through the transducer with the input string b and output p.

3. Experiments

3.1. Data

The data comes from the CSLU Names corpus [10] of 24245 spoken personal names by 20184 different speakers. We use only those utterances that contain a single word – a first or a last name. We further remove all utterances where the transcriptions indicate that they contain 'unintelligibility' or background noise, giving a total of 20423 isolated-word utterances spanning 7771 unique names. The corpus was chosen for its large variety of isolated proper names, many of which do not occur in typical pronunciation lexicons.

One issue with this corpus is the variable number of utterances per name, ranging from 1 to 187, meaning that there is no way to make a uniform train-test split. Therefore, for each name that occurs in more than one utterance, we hold out a random 50% of the utterances for testing. Since many (5453) names occur in only one utterance, we evaluate our method by test-on-train for these cases. This produces a (partially overlapping) split of 12563 training and 13313 test utterances.

We also evaluate our method on the subset of the test data containing only those 7860 test utterances that do not overlap with the training data (i.e., excluding the single-utterance names).

3.2. Setup

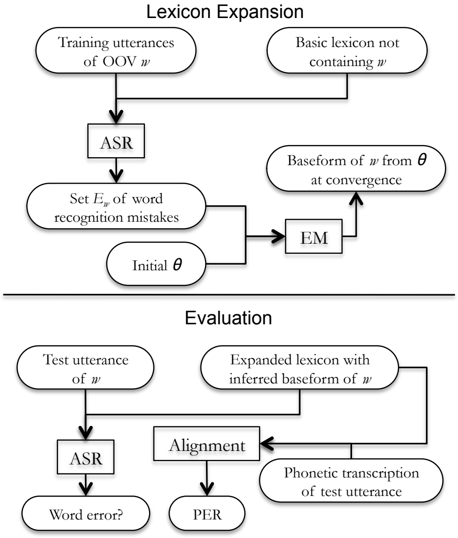
We use the HMM-based CMU Sphinx 3 recognition engine [11], with acoustic models trained on TIMIT. MFCC extraction is also carried out using Sphinx with the default parameters (13 cepstral coefficients with delta and delta-delta, 100 frames/sec., 0.025625 sec. window length).

Our *basic pronunciation lexicon* is the CMU pronunciation dictionary with the names in the speech corpus removed (i.e., every name in the corpus is an OOV with respect to the lexicon). The language model is a unigram model over the 7771 names in the speech corpus, backed off to include the 114455 words in the pronunciation lexicon.

Figure 1 summarizes the LFM methodology. In the **lexicon expansion step**, we generate n-best recognition outputs for each utterance in the *training data*. Since the utterances are

[2] This is not true when we are dealing with multiple-word utterances and interactions with the language model. However, since this paper focuses on isolated words with a flat language model, the assumption that e is conditionally independent of w holds.

Figure 1: Illustration of LFM methodology.


Lexicon Expansion
Training utterances of OOV w
Basic lexicon not containing w
ASR
Baseform of w from θ at convergence
Set E_w of word recognition mistakes
EM
Initial θ
Evaluation
Test utterance of w
Expanded lexicon with inferred baseform of w
Alignment
ASR
Phonetic transcription of test utterance
Word error?
PER

all OOVs, every hypothesis in the n-best outputs is a mistake, which may be a word or a short sequence of words.

We train a second-order g2p model using the Sequitur algorithm [8] as described in §2.3 to initialize θ for the names in the training set. This model produces an average of 218 candidate baseforms for each of the 7771 names.

The EM procedure described in §2 is used to find the single best pronunciation of each of the names, which is then added to the basic lexicon. The algorithm converges quickly (4 iterations when using a 0.1 log-likelihood convergence threshold).

In the **evaluation step**, we run the same decoder using the expanded lexicon on the *test set* utterances, constraining the language model to produce *single-word* utterances. We measure the word error rate (WER) of the output hypotheses.

We also evaluate the quality of the LFM-expanded lexicon by comparing its baseforms to the manual phonetic transcriptions[3] of the test utterances. We align the phonetic transcription to the inferred pronunciation of the reference word and compute the *baseform error rate* (denoted by BER), defined as the proportion of inferred baseforms that differ at all from the reference transcription, and the *phoneme error rate* (PER) – the percentage of phoneme insertions, deletions, or substitutions between the inferred baseforms and the reference.

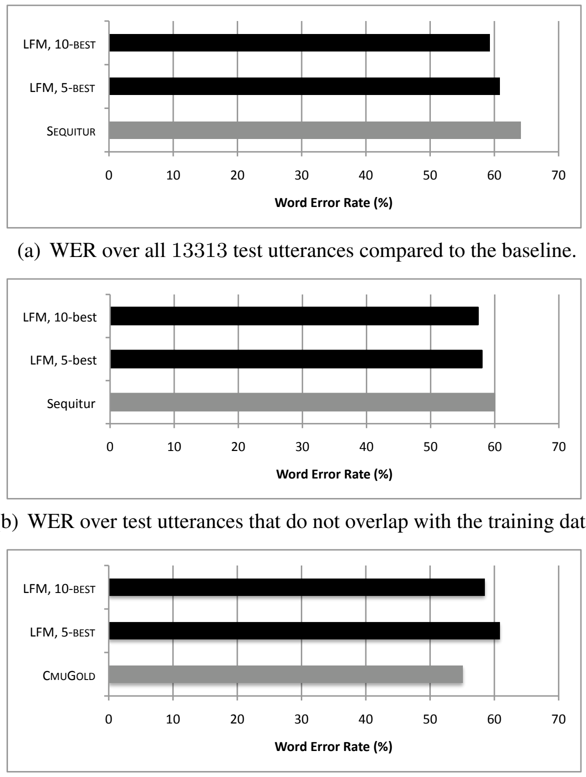
As a baseline, we evaluate the lexicon expanded with the best pronunciations hypothesized by the *Sequitur g2p* algorithm, using multigrams of order 6, trained on the CMU dictionary with the names removed. This model is one of the state of the art grapheme-to-phoneme systems, and is therefore an appropriate point of comparison for pronunciation estimation. We denote this baseline by SEQUITUR.

In addition, we define a 'gold standard' lexicon, denoted by CMUGOLD, using the CMU dictionary pronunciation of a name when available (4874 names). The CMUGOLD lexicon is tested alongside our algorithm on the subset of test utterances (10362 in total) that contain these names.

3.3. Results

Figure 2 shows the word error rate on the test data using the expanded lexicon derived from LFM with 5- and 10-best decoding in comparison to the baseline and the gold standard. Recall that because all the words in the data are OOVs by definition, the WER using the non-expanded basic pronunciation lexicon (with the names in the data artificially removed) is 100%.

Figure 2: Word error rate using LFM-expanded lexicon. "LFM, 5-best" and "LFM, 10-best" denote the lexicons produced by using the 5-best and 10-best word recognition hypotheses respectively to infer a pronunciation for each OOV.



(a) WER over all 13313 test utterances compared to the baseline.

(b) WER over test utterances that do not overlap with the training data.

(c) WER over only those 10362 test utterances containing the names in the CMU dictionary, compared to the gold standard lexicon CMUGOLD.

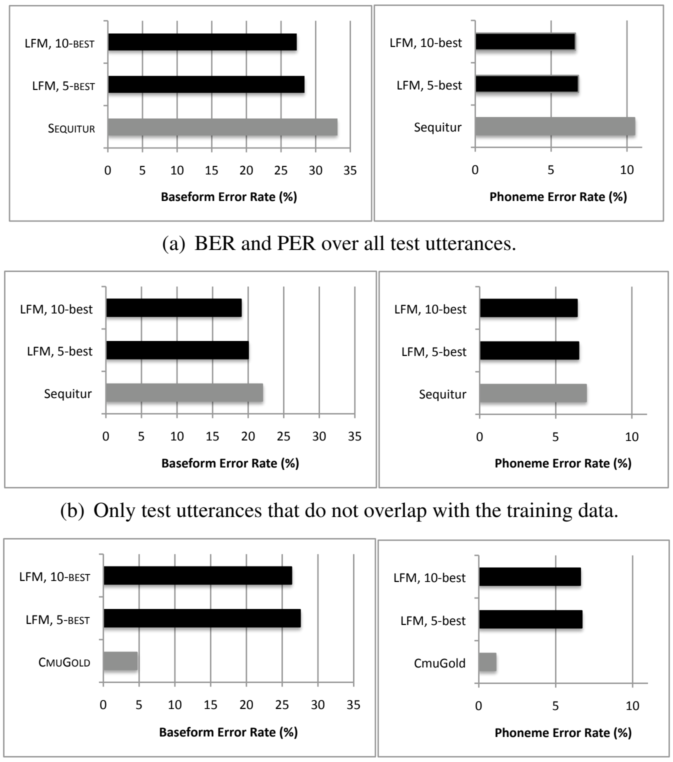
Evaluation against the phonetic transcriptions is measured by the baseform error rate and phoneme error rate over all the test utterances as shown in Figure 3. The close to 0 PER for CMUGOLD indicates that the phonetic transcriptions in the data are almost the same as the CMU dictionary pronunciations. The relative performance of LFM is similar to the speech recognition WER numbers. The phoneme error rate improves over SEQUITUR more noticeably than the baseform error rate.

There is considerable improvement over the SEQUITUR baseline across all evaluations. While the actual pronunciation error rates are significantly higher than the CMU gold standard, the recognition accuracy is not much worse than CMUGOLD. This is because many of the cases where the inferred baseform does not match the transcription involves a minor difference like

[3] The transcriptions are in the Worldbet alphabet, which we map to the Arpabet system used in the CMU dictionary.

a similar-phoneme substitution ('AH' for 'IH', etc.), which does not affect recognition.

Figure 3: Baseform error rate and phoneme error rate of LFM-inferred pronunciations of names in test utterances evaluated against the phonetic transcriptions.



(a) BER and PER over all test utterances.

(b) Only test utterances that do not overlap with the training data.

(c) Only the test utterances containing the names in the CMU dictionary.

4. Discussion

4.1. Error Analysis

Unsurprisingly, the density of a word's 'phonetic neighborhood' – the number of words or short word sequences that are phonetically a small edit distance away from the word – affects the success of this approach. For example, very few words in the lexicon sound like 'Rutherford'. As a result, the ASR recognition mistakes are too dissimilar to the word's actual baseform to provide any useful acoustic information for our method to harness. On the other hand, we do very well on words like 'Marilyn' that have a denser phonetic neighborhood.

For a small number of names (75, less than 1%), the candidate baseforms proposed by the g2p model do not include the correct pronunciation, which is therefore never found by EM.

4.2. Future Work

As mentioned before, LFM can also be applied to discover pronunciation variants of existing lexical items, which could be particularly useful for pronunciation modeling of dialects and accented speech. We would also like to experiment with different datasets in English as well as other languages which require pronunciation lexicons. For wider relevance, LFM could be adapted to continuous speech rather than isolated words.

We would also like to explore ways of seeding θ that are independent of Sequitur. For example, one could generate candidate baseforms using more primitive but fast statistical g2p models, or with non-probabilistic grapheme-to-phoneme rules, or by bootstrapping from a few human annotations. Since the candidate space of baseforms generated from the basic Sequitur model was fairly large as mentioned earlier (218 baseforms per name), different ways of initializing θ will probably not hurt performance, but this needs to be confirmed by experiment.

An interesting future direction is to combine phoneme (or subword) and word recognition outputs to infer pronunciations. Such a problem has a different motivation than the one underlying this paper – rather than assuming that we are constrained to black box ASR output, the idea would be to use word recognition as a cue in addition to phone and other acoustic information when learning pronunciations from speech.

4.3. Conclusion

We have presented a method, applicable to black box speech recognition scenarios, that learns pronunciation baseforms of out-of-vocabulary words using ASR recognition mistakes, and outperforms a state of the art grapheme-to-phoneme system.

5. References

[1] C. Wooters and A. Stolcke, "Multiple-pronunciation lexical modeling in a speaker independent speech understanding system," in *Proceedings of ICSLP*, 1994.

[2] T. Sloboda and A. Waibel, "Dictionary learning for spontaneous speech recognition," in *Proceedings of ICSLP*, 1996.

[3] E. Fosler-Lussier, "Dynamic pronunciation models for automatic speech recognition," Ph.D. dissertation, University of California, Berkeley, 1999.

[4] B. Maison, "Automatic baseform generation from acoustic data," in *Proceedings of Eurospeech*, 2003.

[5] T.-P. Tan and L. Besacier, "Improving pronunciation modeling for non-native speech recognition," in *Proceedings of Interspeech*, 2008.

[6] D. Bansal, N. Nair, R. Singh, and B. Raj, "A joint decoding algorithm for multiple-example-based addition of words to a pronunciation lexicon," in *Proceedings of ICASSP*, 2009.

[7] I. Badr, I. McGraw, and J. Glass, "Learning new word pronunciations from spoken examples," in *Proceedings of Interspeech*, 2010.

[8] M. Bisani and H. Ney, "Joint-sequence models for grapheme-to-phoneme conversion," *Speech Communication*, vol. 50, pp. 434–451, 2008.

[9] J. S. Garofolo, et al, "TIMIT acoustic-phonetic continuous speech corpus," *Linguistic Data Consortium, Philadelphia*, 1993.

[10] Y. Muthusamy, R. Cole, and B. Oshika, "CSLU: Names Release 1.3," *Linguistic Data Consortium, Philadelphia*, 2006.

[11] Carnegie Mellon University, "CMU Sphinx open source toolkit for speech recognition," http://cmusphinx.sourceforge.net/.