LEARNING LINGUISTICALLY VALID PRONUNCIATIONS FROM ACOUSTIC DATA

Françoise Beaufays, Ananth Sankar, Shaun Williams, Mitch Weintraub

Nuance Communications
1380 Willow Road
Menlo Park, CA 94025
{francoise, sankar, mw}@nuance.com

ABSTRACT

We describe an algorithm to learn word pronunciations from acoustic data. The algorithm jointly optimizes the pronunciation of a word using (a) the acoustic match of this pronunciation to the observed data, and (b) how "linguistically reasonable" the pronunciation is. Variations of word pronunciations in the recognition dictionary (which was created by linguists), are used to train a model of whether new hypothesized pronunciations are reasonable or not. The algorithm is well-suited for proper name pronunciation learning. Experiments on a corporate name dialing database show 40% error rate reduction with respect to a letter-to-phone pronunciation engine.

1. INTRODUCTION

An important component in automatic speech recognition (ASR) systems is the dictionary. In most speaker-independent systems, the dictionary reflects generic word pronunciations, while phonetic variations due to accent, speaking style and other factors are left to the acoustic models to cover. With this approach, it is straightforward to build a dictionary that has a good coverage for most common names, verbs, adjectives, etc. For example, the average number of pronunciations per word is typically less than two. In the event that the application requires words that are not in the dictionary, a letter-to-phone engine (*e.g.* [1]) can be used to estimate the most likely pronunciations of the new words.

The automatic generation of pronunciations is harder for proper names than for common words where spelling provides a much better indicator of phonetic realizations. For example, the pronunciation of a person's name may change to adjust to a new sociological environment (e.g. the absence of a phone in the new language in case of immigration), but pronunciations can also retain some phonetic aspects that are dialectal or historical and don't conform to the current phonological rules of the name bearer's language.

In addition, proper names may require several pronunciations to accomodate for the realizations of speakers with different linguistic backgrounds. For example, an American speaker is likely to pronounce the first author's name as [b u f e] (CPA symbols), a French person will typically say [b o f e], and a french-speaking Belgian will use [b o f A i], extrapolating from walloon dialects. Likewise the second author's name may be pronounced as [s a n k *r] or [S a n k *r] depending on the speaker's exposure to Indian names. This makes the task of predicting the phonetic realizations of proper names from their spelling difficult. Many times the actual pronunciation of a person's name cannot be determined without access to audio samples of the name.

Data-driven pronunciation methods aim at learning pronunciations from acoustic data. They typically proceed by applying a set of phone transformation rules to some training material, and by learning from force-alignments the frequency of each rule. The rules may be predefined [2], or learned from phonetically hand-transcribed data [3, 4, 5, 6]. The same rules are then applied to expand the pronunciations of all the words in the dictionary, and a word-level probability is attached to each pronunciation variant to reflect the likelihoods of the rules that generated it. This family of approaches is well-suited to the learning of systematic variations, *e.g.* spontaneous vs. read speech, but it does not focus on capturing the actual pronunciations of specific words as they appear in the training material. Another approach [3], based on recognizing the training data with a phone loop, is more suited to our task, however it does not easily allow the use of a linguistic model as we will describe below.

The method we propose is one of successive refinements. We start by applying a letter-to-phone engine to the names for which we need pronunciations. We then alter these initial phone sequences to improve their match to a set of audio training samples. In order to limit the number of pronunciations per word, and to avoid word confusability during recognition, we restrain the generation of pronunciation variants to regions of bad acoustic match, where pronunciation modifications are deemed most necessary. Phone transformations are suggested to increase the overall match of the pronunciations to the training data, given a set of acoustic models.

If the task of choosing new pronunciations is left solely to the acoustic models, pronunciations that "do not make sense" may be generated. For example, the phone sequence [k i t *r] may be proposed for "Peter" because "p" and "k" are two unvoiced stop-consonants and are acoustically confusable. Likewise, an unvoiced fricative could be inserted in front of a name to match some breath noise. Such "incorrect" pronunciations are undesirable: they typically don't generalize to other acoustic samples, they increase name confusability, and they may cause a drift in the acoustic models in case of retraining or acoustic adaptation with the learned dictionary.

To avoid generating such phone sequences, we express the pronunciation learning process as a joint optimization problem that simultaneously optimizes the acoustic match of the proposed phone sequence to the acoustic data and its linguistic probability given some initial pronunciation and a linguistic model. This pronunciation model is derived in a data-driven fashion from an existing ASR dictionary.

2. PRONUNCIATION LEARNING ALGORITHM

2.1. High-Level Description of the Algorithm

An initial dictionary is first produced by applying a letter-to-phone pronunciation engine to all the names in the grammar. Training utterances are collected and orthographically transcribed. The following 5 steps are then applied to each utterance.

STEP 1: Force-Alignment: Each training waveform is aligned to its orthographic transcription. If many pronunciations are available for a transcribed word, the one chosen by the force-alignment is marked as the "original pronunciation". Phone-level time boundaries are indicated.

STEP 2: Identification of the Region of Worst Acoustic Match: The posterior probability of each phone in the force-alignment is computed over the time segment it was aligned to. The worst matching region is identified as that whose phone has the lowest probability. This one phone is considered as candidate for modification.

STEP 3: Suggestion of Alternative Pronunciations: A series of alternative phone sequences are proposed. All the phones in the original pronunciations are kept unchanged, besides the worst matching phone from STEP 2. That phone is successively deleted, replaced by each phone from the phone set, and preceeded or followed by the insertion of each phone. The original pronunciation is also kept in the list. In addition, a series of frequent monophone to biphone (and vice-versa) transformations are also allowed (*e.g.* "*r" → "E r" or "A r").

STEP 4: Pronunciation Scoring: The original waveform is re-aligned to each alternative pronunciation, and a score is computed for each alignment. This score depends on the likelihood of the alignment and on some linguistic factors. Its exact expression will be detailed in Section 2.2. The alignment of highest score indicates the best pronunciation.

STEP 5: Dictionary Update: The original (STEP 1) and best scoring (STEP 4) pronunciations are inserted in the new dictionary.

Pronunciations listed in the initial dictionary and not "exercised" by STEP 1 of the algorithm are discarded. The 5 steps can be repeated any number of times.

2.2. Pronunciation Scoring

This section describes how pronunciation scores are computed in STEP 4 of the above algorithm. Let $\mathcal{X}$ be the sequence of acoustic observations corresponding to the waveforms. Let $\mathcal{A} = [a_1 a_2 ... a_n]$ be the phone sequence that was aligned to the waveform, and $\mathcal{B}_i$ the phone sequences corresponding to the alternative pronunciations. We want to find a phone sequence $\mathcal{B}^*$ that is (1) a good match to the observations $\mathcal{X}$, and (2) linguistically valid, given that the original pronunciation for the word was $\mathcal{A}$. In other words, we want to find the phone sequences $\mathcal{B}^*$ whose probability is highest given $\mathcal{A}$ and $\mathcal{X}$:

$$\mathcal{B}^* = \arg\max_{B_i} P(\mathcal{B}_i|\mathcal{A},\mathcal{X}). \quad (1)$$

Since $\mathcal{A}$ and $\mathcal{X}$ are givens for the optimization problem, we assume them to be independent. This is an assumption to the extend that $\mathcal{A}$ was selected among several pronunciations in the original dictionary to best match $\mathcal{X}$ (STEP 1).

Inverting the probability from Eq. 1, making use of the independence assumption, and regrouping terms, we get:

$$\begin{aligned} P(\mathcal{B}_i|\mathcal{A},\mathcal{X}) &= \frac{P(\mathcal{A},\mathcal{X}|\mathcal{B}_i)P(\mathcal{B}_i)}{P(\mathcal{A},\mathcal{X})} \\ &= \frac{P(\mathcal{A}|\mathcal{B}_i)P(\mathcal{X}|\mathcal{B}_i)P(\mathcal{B}_i)}{P(\mathcal{X}|\mathcal{A})P(\mathcal{A})} \\ &= \frac{P(\mathcal{X}|\mathcal{B}_i)}{P(\mathcal{X}|\mathcal{A})}P(\mathcal{B}_i|\mathcal{A}). \end{aligned} \quad (2)$$

Eq. 2 expresses the probability of the phone sequence $\mathcal{B}_i$ as a product of two terms: a likelihood ratio, which constitutes the acoustic part of the model and that simply states that the desired pronunciation should have a high likelihood compared to the original pronunciation, and a pronunciation conditional term which constitutes the linguistic part of the model. This last term, $P(\mathcal{B}_i|\mathcal{A})$, is the object of the next section.

Rephrasing Eq. 2 in the log domain, and introducing an additional parameter, γ, we get

$$\log P(\mathcal{B}_i|\mathcal{A},\mathcal{X}) = \gamma \log \frac{P(\mathcal{X}|\mathcal{B}_i)}{P(\mathcal{X}|\mathcal{A})} + (1-\gamma)P(\mathcal{B}_i|\mathcal{A}). \quad (3)$$

The log conditional $P(\mathcal{B}_i|\mathcal{A},\mathcal{X})$ is the score we want to maximize in STEP 4. The parameter γ, which we'll refer to as the "acoustic weight", introduces some flexibility in balancing the contributions of the acoustic and linguistic parts of the model.

2.3. Pronunciation Model

Intuitively, the probability $P(\mathcal{B}|\mathcal{A})$ is a pronunciation transformation model. Assume that the speaker said "Peter", and that the waveform was aligned to $\mathcal{A}$ = [p i ! *r]. We would argue that the alternative pronunciation $\mathcal{B}_1$ = [p i t *r] is "reasonable", whereas $\mathcal{B}_2$ = [k i ! *r] is not. In this case, we would want P($\mathcal{B}_1|\mathcal{A}$) to be large, and P($\mathcal{B}_2|\mathcal{A}$) to be equal to zero.

The pronunciation transformation model therefore requires some linguistic knowledge to assess which phone transformations are reasonable and which are not. We chose to extract this information in a data-driven manner from the system dictionary. Any dictionary word that has more than one pronunciation provides information about one or more phone transformations that have been validaded by a linguist. Such transformations can thus contribute to our model.

More specifically, for each word in the dictionary with two pronunciations, $\mathcal{R}$ and $\mathcal{S}$, we perform a dynamic programming alignment of the two phone sequences, using a linguistically motivated phone distance metric similar to that used in [7]. Counts are then accumulated over the phone correspondances between the two aligned strings. Two types of counts are kept: (1) context-independent counts of the type $C(r_i, s_j)$ if phone r_i in $\mathcal{R}$ was aligned to phone s_j in $\mathcal{S}$, and context-dependent counts of the form $C(r_{i-1}\ [r_i]\ r_{i+1},\ s_{j-1}\ [s_j]\ s_{j+1})$. In this last case, we impose the constraints $r_{i-1} = s_{j-1}$ and $r_{i+1} = s_{j+1}$, that is, counts are accumulated only for events whose context is "stable" (identical in both sequences). This is in agreement with the decision taken in STEP 3 of the algorithm, *i.e.* that a single phone is modified at a time, assuming that its context is kept unchanged. In a slight relaxatin of the above, we also allow r_i or s_j (but not both) to represent a sequence of 2 phones. This allows transformations such as [E r] ↔ [*r] to be considered in the model.

The counts accumulated as described above are then transformed into context-dependent and independent phone transformation probabilities. A smoothed context-dependent probability of the form $P(a[b]c \mid a[d]c)$ can then be estimated as

$$P(a[b]c \mid a[d]c) = \alpha P(a[b]c \mid a[d]c) + (1-\alpha)P(b \mid d), \quad (4)$$

where the smoothing constant α can be expressed as a function of the counts

$$\alpha = \frac{C(a[d]c)}{C(a[d]c) + \text{const.}}. \quad (5)$$

3. EXPERIMENTS

3.1. Recognizer

The recognition system used in this section is based on standard 3-state triphone hidden Markov models (HMM), with a Genone-based state clustering mechanism [8]. The front-end is based on 27-dimensional mel-filterbank cepstral coefficient feature vectors, with cepstral mean subtraction and standard noise reduction.

3.2. Database

The database used in these experiments was collected from a live name dialer implemented in a large company of the San Francisco Bay Area. Each token in the database consists of a first name followed by a last name. Because of the company location, a relatively large portion of the names in the name list are from foreign origin, and many speakers are non-native speakers of English.

To avoid any speaker overlap between the training and test sets, we chose to learn pronunciations from male speakers, and to test them on female speakers exclusively. Three training sets and corresponding test sets were built, with 6, 10, and 15 repetitions of each name, respectively. The 6-repetition training set contains roughly 500 names repeated 6 times exactly by male speakers (roughly 3000 utterances). The 6-repetition test set consists of the same 500 names spoken 6 times each by female speakers. Similarly, the 10 and 15-repetition training and test sets contain roughly 300 and 200 names, respectively. The grammar consists of roughly 1600 name pairs.

3.3. Baseline

The baseline performance on the three test sets was measured with a standard statistical letter-to-phone pronunciation engine. This engine is trained from the same dictionary as the pronunciation model. It does not make use of acoustic data.

For ease of reference, we'll denote as BasePron the baseline pronunciation system, and LearnPron the proposed algorithm.

3.4. Performance

Table 1 below summarizes the recognition performance of the BasePron and LearnPron systems.

The PronLearn system is significantly more accurate than the BasePron one. The relative error rate improvement increases with the number of repetitions of each name, ranging from 36 to 50%. Only the setup with few (six) repetitions benefits from a second iteration of the algorithm (additional 9% relative improvement). This is probably because the algorithm was set up to learn only one word per sentence, so that in the 6-repetition experiment only 3 waveforms on average can contribute to the learning of a name. A second iteration gives a second chance to the algorithm to learn the name part (first or last) it did not learn in the first iteration. This advantage decreases with more repetitions.

Expt	BasePron	LearnPron	
		Iter #1	Iter #2
6 reps	11.9	7.6	6.9
10 reps	12.0	7.0	6.7
15 reps	10.9	5.4	5.6

Table 1. Error Rate on a Name Dialing task: Comparing the baseline letter-to-phone BasePron generator to the proposed data-driven LearnPron method. The error rates are expressed as percentages of incorrect full names (first and/or last name incorrect).

3.5. Acoustics vs Linguistics in the PronLearn Algorithm

We introduced in section 2.2, Eq. 3, a free parameter, the acoustic weight γ, to balance the contribution of the acoustic and linguistic parts of the pronunciation model. This section discusses the effect of γ on the learned dictionary. Figure 1 shows the effect of γ of the error rate for the 6-repetition experiment. The BasePron performance is indicated for reference.

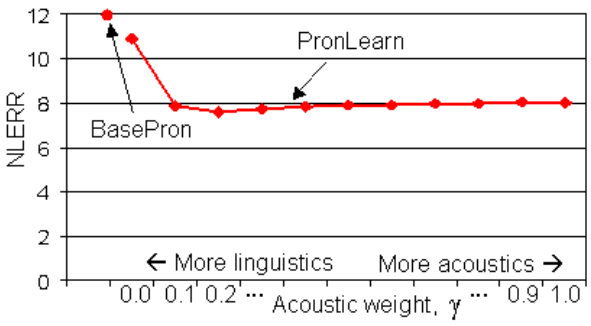

Fig. 1. Error Rate on a Name Dialing task as a function of the acoustic weight γ (see Eq. 3).

We observe the following. First, it is indispensable to include the acoustic part of the model in the equation: $\gamma = 0.0$ (left most point on the curve) results in only slightly lower error rate than the baseline (BasePron). The pronunciation model is word-independent and cannot capture the specificities of each name. Second, from an accuracy point of view it seems that any greater-than-zero value for γ is fine (though a smaller value might be slightly better). However. it should not be extrapolated that the pronunciation model is useless, its function is to reduce the number of pronunciations per word by constraining these to be linguistically valid. If γ is small, $1-\gamma$ is large, and $\log P(\mathcal{B}_i|\mathcal{A})$ will dominate Eq. 3. An alternative pronunciation that has a high likelihood ratio but a poor linguistic score will fall below the original pronunciation whose likelihood ratio is 1.0, and its linguistic probability is close to 1.0. In this case, no new pronunciation is generated for this token. In this experiment, varying γ reduced the dictionary from 4.7 ($\gamma = 1.0$) to 2.8 ($\gamma = 0.1$) average pronunciations per word, or 2.7 and 0.8 new pronunciations per word, respectively. As an

example, the name "Bash" has the following pronunciations in this experiment:

γ	Pronunciations
0.0	b a S
0.1	b a S, b a s
0.2	b a S, b a s, b aj a S, b e $\wedge$ S, h b a S

Table 2. Learned pronunciations for the last name "Bash", as a function of teh acoustic weight γ.

3.6. Data-driven Algorithm vs. Trained Linguist

Section 3.4 compared the performance of the data-driven method to that of a letter-to-phone method. One might wonder how well a trained linguist would have done on the same task.

In a first experiment, the linguist took the list of 1600 in-grammar names, and wrote pronunciations for them, using his best judgement to decide how many pronunciations were needed for each word. We refer to this dictionary as "Lng". He then gathered a list of waveforms containing 6 repetitions of each name, all spoken by male speakers. The linguist then refined the Lng dictionary by listening to the waveforms. The resulting dictionary is refered to as "LngAud". Finally, a LearnPron dictionary was produced by applying the data-driven algorithm to the exact same audio data. The three dictionaries were evaluated against the female test sets.

Table 3 summarizes the resulting recognition error rates. Results with the BasePron dictionary are added for reference. Note that the BasePron and LearnPron numbers reported here are slightly different from the "6 reps" numbers in Table 1. This is because in this experiment *all* the in-grammar names were learned, as opposed to only some of them (those in the training and test sets) as in Table 1.

Expt	BasePron	Lng	LngAud	LearnPron
6 reps	11.9	9.8	7.9	7.6
10 reps	12.0	9.4	7.4	7.2
15 reps	10.9	8.6	6.7	5.9

Table 3. Error Rate on a Name Dialing task: Comparing a letter-to-phone BasePron gerenator, two hand-made dictionaries, Lng and LngAud, and the data-driven LearnPron method. The error rates are expressed in terms of percentages of incorrect full names (first and/or last name incorrect).

Comparing columns 1 and 2 shows that the linguist does roughly 20% better than the BasePron engine. With access to acoustic data, the linguist reduces the error rate by another 20%. The linguist attributes this additional gain to the facts that (1) with no audio he didn't know how to pronounce some foreign names, (2) with no audio he didn't know how others might pronounce some names. The LearnPron algorithm performs similarly to the linguist when given access to the same acoustic samples. This indicates that the data-driven algorithm was sucessful in capturing the pronunciation variants present in the data. It should be noted however that the PronLearn dictionary was slightly larger than the Lng and LngAud dictionaries, themselves being comparable in size.

4. CONCLUSIONS

We have described an algorithm that learns pronunciations from acoustic data while ensuring that the learned pronunciations are reasonable from a linguistic point of view. Our experiments showed that the algorithm reduces the error rate on a name dialing task by up to 40% with respect to a letter-to-phone pronunciation engine, with only 6 repetitions of each name. Varying a free parameter in the algorithm, we saw that (1) the acoustic part of the combined model is what gives recognition accuracy, (2) the linguistic part constraints the learned pronunciations to be linguistically valid. We also observed through these experiments that a trained linguist who writes pronunciations for proper names (especially foreign ones) does substancially better when he has access to acoustic samples of the names.

5. REFERENCES

[1] J. M. Lucassen and R. L. Mercer, "An Information Theoretic Approach to the Automatic Determination of Phonemic Baseforms.", *Proc. ICASSP*, pp. 42.5.1-42.5.4, 1984.

[2] G. Tajchman, E. Fosler, and D. Jurafsky, "Building Multiple Pronunciation Models for Novel Words Using Exploratory Computational Phonology", *Proc. Eurospeech*, 1995.

[3] M. Weintraub, E. Fosler, Ch. Galles, Y.-H. Kao, S. Khudanpur, M. Saraclar, S. Wegmann, "Automatic Learning of Word Pronunciation from Data", *1996 Large Vocabulary Continuous Speech Recognition Summer Research Workshop Technical Reports*, Johns Hopkins University.

[4] W. Byrne, M. Finke, S. Khudanpur, J. McDonough, H. Nock, M. Riley, M. Saraclar, Ch. Wooters, G. Zavaliagkos, "Pronunciation Modelling Using a Hand-Labelelled Corpus for Conversational Speech Recognition", *Proc. ICASSP*, 1998.

[5] M. Finke and A. Waibel, "Speaking Mode Dependent Pronunciation Modeling in Large Vocabulary Conversational Speech Recognition", *Proc. Eurospeech*, 1997.

[6] R. Bates and M. Ostendorf, "Modeling Pronunciation Variation in Conversational Speech Using Prosody", *Proc. ISCA Workshop*, 2002.

[7] R. Bates and M. Ostendorf, "Modeling Pronunciation Variation in Conversational Speech Using Syntax and Discourse", *Proc. Workshop on Prosody in Speech Recognition and Understanding*, pp. 17-22, 2001.

[8] V. Digalakis, P. Monaco, and H. Murveit, "Genones: Generalized Mixture Tying in Continuous Hidden Markov Model-Based Speech Recognition", *IEEE Trans. on Speech and Audio Processing*, pp. 281-289, 1996.