In: ICASSP-91, Toronto, Canada, 1991.

A STATISTICAL MODEL FOR GENERATING PRONUNCIATION NETWORKS

Michael D. Riley

AT&T Bell Laboratories
Murray Hill, NJ

ABSTRACT

We describe how to predict detailed phonetic pronunciations from a coarse phonemic transcription. The phonemic base forms, obtainable from orthographic text by dictionary lookup and other means, do not specify fine phonetic detail such as flapping, glottal stop insertion, or the formation of syllabic nasals and liquids. These phenomena depend on the phonemic context (often spanning word boundaries), stress environment, speaking rate, and dialect. We describe a procedure that builds decision trees, trained on the TIMIT database, using some of these features to predict pronunciation alternatives. The resulting phonetic network predicts the correct pronunciation of a phoneme on test data from the same corpus 83contains the correct phone in the top 5 guesses 99and has a conditional entropy of .8 bits.

1. INTRODUCTION

This paper is about how to predict, in context, the likely pronunciations of words. If we start with some text, e.g., `'I like butter'` , we can tell something about its pronunciation by simply looking up the words in a pronouncing dictionary and concatentating the *phonemes* found, e.g., `/ay l ay k b uh t er/` . In this paper, we use the *ARPABET* symbols for specifying phonemes [1].

The phonemes are a set of base forms for representing the sounds in a word. Replacing one phoneme in a word with another is usually drastic enough to turn that word into a different word (or a non-word). It is, by definition, drastic enough to do so for some word.

There is finer kind of variation, however, that is not indicated at the phonemic level, so-called *allophonic variation*. For example, the `/t/` in `'butter'` may be pronounced as a flap, `[dx]`, or as a released t, `[tcl t]` . In this paper, we use the TIMITBET symbols, a superset of the ARPABET symbols, for specifying phones [2]. Which allophone of the phoneme `/t/` will occur in this word depends, in part, on the speaker's dialect and speaking rate. In another example, the phoneme `/k/` in `'like'` may be released or not (i.e., have noticable burst and aspiration). This will depend on the context in which the word is found, e.g., whether it is followed by another stop consonant.

Thus, phones are acoustically distinct realizations of phonemes. The choice of a phone set is a matter of judgment, but phoneticians have traditionally agreed that certain kinds of variation are worth noting based on their acoustic prominence and regularity.

Deriving a pronunciation can thus be divided into two steps – mapping from orthography to phonemes, and then mapping from phonemes to phones. The problem of mapping orthographic text onto its phonemic representation will not be discussed further, except to say it is not entirely a simple dictionary lookup, e.g., unknown words and some homonyms (like *read* , which depending on the tense can be pronounced as `/r iy d/` or `/r eh d/`), require special treatment. We use the Bell Labs text-to-speech system to provide the transformation from orthography to phonemes [3]. The focus of this paper is how to predict allophonic variation, i.e., mapping from phonemes to phones.

To get a measure of the difficulty of this problem, consider the TIMIT phonetically-labelled database. This is a hand-labelled corpus of 6300 phonetically-rich utterances by 630 speakers created at TI and MIT [2], and includes both phonetic and orthographic transcriptions of the utterances. From the orthographic transcriptions, we have derived phonemic transcriptions and aligned them with the phonetic transcriptions. From this, we have estimated the conditional entropy of a phone given the matching phoneme to be about 1.5 bits (in Section 6 we describe how we treat phone insertions).

Significantly, this estimate does not include any contextual information. Knowing what the neighboring phonemes are, what the stress environment is, and where the word boundaries are, will help considerably in predicting how a phoneme is realized as a phone. Our goal, in part, is to use this kind of information to reduce the 1.5 bits of uncertainty.

This paper will describe a method that will reduce the uncertainty from 1.5 bits to about .8 bits. In another way of measuring performance, this method predicts the correct phone from phonemic context about 83% of the time and the correct phone lies in our top five guesses 99% of the time. In comparison, if we only use the matching phoneme and no contextual information, we able to predict the correct phone only 69% of the time and we must look at the top ten guesses to find the correct phone 99% of the time.

We can never remove all uncertainty when predicting just from the phoneme string. In the `'butter'` example, flapping the `/t/` is the most likely outcome (for American speakers), but other allophones of `/t/` can also occur. Therefore, when we predict a realization, we will allow for alternatives and estimate their likelihoods. For example, in a task like TIMIT, our method predicts

CH2977-7/91/0000-0737 $1.00 © 1991 IEEE

the /t/ in butter will flap about 73% of the time with released t, [tcl t], the second most likely outcome. Thus, our predicted realization of a phoneme string is a network of phonetic alternatives (see Figure 1).

There are different approaches toward predicting pronunciations. The traditional phonetician's approach has been to write rules that explicitly state the predicted behavior, e.g., 'a stop consonant is usually unexploded before another stop consonant' or 'a /t/ usually becomes a glottal stop before a nasal in the same word' [4].

Now that large, phonetically-labelled databases like TIMIT are available, it is possible to estimate the phoneme-to-phone mapping statistically [5,6,7]. This approach has several advantages. First, it readily permits assigning likelihoods to alternative pronunciations. Second, it permits the discovery of regularities perhaps overlooked by heuristic means. Finally, it allows predictors to be quickly retailored to new corpora - whether different tasks, different dialects, or even different languages.

2. PREDICTION MODEL

To predict from phonemes to phones, we take a phoneme string as input and produce phonetic realizations as output along with their likelihoods.

Let us make this idea precise. Let $\mathbf{x} = x_1 x_2 ... x_m$ be the string of phonemes of some sentence. So that we can mark both word boundaries and stress we augment the phoneme set to include /#/ as a word boundary marker and split each syllabic phoneme into an unstressed, a primary stressed, and a secondary stressed version. Further, let $\mathbf{y} = y_1 y_2 ... y_n$ be the string of corresponding phones. We include the phone symbol [-] to indicate that a phoneme may delete.

The most general form of our predictor is $\hat{P}(\mathbf{y}|\mathbf{x})$, where $\hat{P}$ estimates the probability that the phone sequence $\mathbf{y}$ is the realization of the phoneme sequence $\mathbf{x}$.

This specifies the probalitity of an entire phone sequence $\mathbf{y}$. For convenience, we want to decompose this into one phone prediction at a time. Since

$$P(\mathbf{y}|\mathbf{x}) = p_n(y_n|\mathbf{x}y_1...y_{n-1})p_{n-1}(y_{n-1}|\mathbf{x}y_1...y_{n-2})...p_1(y_1|\mathbf{x}), \quad (2.1)$$

we can restate the problem as finding a suitable predictor, $\hat{p}_k(y_k|\mathbf{x}\, y_1...y_{k-1})$, that estimates the probability that y_k is the kth phone in the realization, given the phoneme sequence $\mathbf{x}$ and the previous k-1 phones $y_1...y_{k-1}$.

Eq. 2.1 is more general than necessary since realistically the kth phone will depend only on a few neighboring phonemes and phones. Suppose that we can place the phoneme and phone strings into alignment. In fact, forming a good alignment betwen phonemes and phones is easy if deletions and insertions are permitted, using a phonetic feature distance measure and standard string alignment techniques [8]. Since we have augmented the phone set to include a deletion symbol, the only stumbling block to such an alignment would be if phones insert. For the moment, assume that they don't; we will come back to insertions later. Thus, under this assumption we can talk about the kth phoneme and its corresponding phone. We assume

$$p_k(y_k|\mathbf{x}\, y_1...y_{k-1}) = p(y_k|x_{k-r}...x_{k-1}x_k x_{k+1}...x_{k+r}y_1...y_{k-1}) \quad (2.2)$$

. In other words, p_k is stationary and depends only on the $\pm r$ neighboring phonemes.

If we assume the kth phone does not depend any of the previous phones, we have

$$p(y_k|x_{k-r}...x_{k-1}x_k x_{k+1}...x_{k+r}y_1...y_{k-1}) = p(y_k|x_{k-r}...x_{k-1}x_k x_{k+1}...x_{k+r}) \quad (2.3)$$

This is the assumption that phones are conditionally independent given the phonemic context. A less stringent assumption would be that the kth phone only depends on the immediately prior phone, In this case, we must estimate

$$p(y_k|x_{k-r}...x_{k-1}x_k x_{k+1}...x_{k+r}y_1...y_{k-1}) = p(y_k|x_{k-r}...x_{k-1}x_k x_{k+1}...x_{k+r}y_{k-1}) \quad (2.4)$$

This is the assumption that phones are conditionally 1st-order Markov given the phonemic context.

These last two models are the ones that we will explore – one that assumes an independence model of phones and the other that assumes a Markov model. We must also come back to the question of what to do when phones insert.

3. CLASSIFICATION TREES

We now discuss the question of how, in general, we will estimate the phoneme–to–phone mapping probabilities specified in the previous section. The simplest procedure would be to collect n-gram statistics on the training data. A bi–phonemic or possibly tri–phonemic context would be the largest possible with available training data if we want statistically reliable estimates.

We believe that a straight-forward n-gram statistics on the phonemes are probably not ideal for this problem since the contextual effects that we are trying to model often depend on a whole class of phonemes in a given position, e.g., whether the preceding phoneme is a vowel or not. A procedure that had all vowels in that position clustered into one class for that case would produce a more compact description, would be more easily estimated, and would allow a wider effective context to be examined.

Thus intuitively we would like a procedure that pools together contexts that behave similarly, but splits apart ones that differ. An attractive choice from this point of view is a statistically-generated decision tree with each branch labelled with some subset of phonemes for a particular position. The tree is generated by spliting nodes that statistical tests, based on available data, indicate improve prediction, but terminating nodes otherwise.

An excellent description of the theory and implementation of tree-based statistical models can be found in *Classification and Regression Trees* [9]. The interesting questions for generating a decision tree from data – how to decide which splits to take and when to label a node terminal and not expand it further – are discussed in these references along with the widely-adopted solutions.

Suffice it to say here the result is a binary decision tree whose branches are labelled with binary cuts on the continuous features and with binary partitions on the categorical features and whose terminal nodes are labelled with continuous predictions (*regression tree*) or categorical predictions (*classification tree*). By a continuous feature or prediction we mean a real-valued, linearly-ordered variable (e.g., the duration of a phone, or the number of phonemes in a word); by a categorical feature or prediction we

mean an element of an unordered, finite set. (e.g., the phoneme set).

4. BASELINE MODEL

In Section 2 we developed two models for predicting the probability that a particular phone is the realization of a phoneme. One used the phoneme context as the predictor input. The other used that plus the previous phone as input. In this section, we describe the implementation of the first model. We still exclude the treatment of insertions at this point. We will call this our *baseline model.* In later sections, we will describe refinements to this model.

In the exposition in Section 2, we combined word boundary and stress information into the phoneme set itself. When we actually input the features into the tree classification procedure we have found it more convenient to keep them separate.

We include $\pm r$ phonemes around the phoneme that is to be realized (typically, $r = 3$). This is irrespective of word boundaries. We pad with blank symbols at sentence start and end.

Since there are 40 different phonemes, if we directly input each phoneme into the tree classification routine, 2^{40} possible splits would have to be considered per phoneme position at each node, since, by default, all possible binary partitions are considered. This is clearly intractable, so instead we encode each phoneme as a feature vector. A manageable choice is to encode each phoneme as a four element vector: `(consonant-manner, consonant-place, vowel-manner, vowel-place)`. Each component can take one of about a dozen values and includes `'n/a'` for 'not applicable'. For example, `/s/` is encoded as `(voiceless-fricative, palatal, n/a, n/a)` and `/iy/` is encoded as `(n/a, n/a, y-diphthong, high-front)`

If the phoneme to be realized is syllabic, then we also input whether it has primary or secondary stress or is unstressed. We use stress as predicted by the Bell Labs text-to-speech system; this is essentially lexical stress with function words de–accented. If the phoneme is not syllabic, we input both the stress of the first syllabic segment to the left and to the right if present within the same word (and use `'n/a's'` if not).

To encode word boundaries, we input the number of phonemes from the beginning and end of the current word to the phoneme that is being realized.

We do not input the syllabification directly since we do not have that information readily available. But, because we typically use a wide phonemic context, the syllabification is often implicitly present. If we had the syllabification, however, we would include it since it might help in some cases. We note, nonetheless, that Randolph[7], who included the syllabification in a tree classifier of TIMIT stop allophones, achieved classification rates nearly identical to what we achieve on that data using the feature set describe here.

Our output set is simply a direct encoding of the phone set plus the symbol `[-]` if the phoneme deletes. Computation time grows only linearly with the number of *output* classes so this direct encoding presents no problem similar to the exponential growth found with size of the input feature classes.

We now describe the results of this baseline model on the TIMIT database. The phonetic transcription of 3024 sentences from the TIMIT 'sx' an 'si' sentences were aligned with their phonemic transcription as predicted by the Bell Labs text-to-speech system from their orthographic transcription. For each of the resulting 100702 phonemes, the phonemic context was encoded as described in previous section. A classification tree was grown on this data and the tree size was chosen to minimize prediction error in a 5-fold cross-validation. The resulting tree had approximately 300 nodes.

This tree was then used to predict phonetic realizations of an independent 336 sentences from the TIMIT 'sx' and 'si' sentences. The result was 84.1% correct prediction and a conditional entropy of .77 bits.

5. MARKOV MODEL

To judge the 84.1% performance obtained by our baseline model, we have to look at the errors. They can be divided into two categories: those in which the prediction was, in fact, the most likely outcome, but the speaker of the test sentence used a less likely alternative pronunciation (e.g. he didn't flap the `/t/` in `'pretty'`), and those cases in which the model is imperfect and an implausible pronunciation is predicted. The first kind of uncertainty is inherent to the problem, the second kind is something we should try to fix.

Using the baseline model, we find the major latter kind of error occurs near a deletion. For example, `'are'`, phonemically `/aa r/`, is often realized as `[axr]` in fluent speech. In the training data this is modelled as `/aa/` → `[axr]` and `/r/` → `[-]`. But, the baseline model may predict the pronuncations `[aa -]` and `[axr r]` for `/aa r/`, which are unlikely for most TIMIT speakers. Similar problems occur with `/n/`, `/m/` and `/l/` in contexts where they are likely to syllabify.

The problem is that in these cases the realization of the previous phoneme strongly influences the realization of the current phoneme. This suggests we should use the second model outlined in Section 2 – the Markov model. The idea is that we augment the feature set with the previous phone that was output. During training, we use the actual phone uttered. During testing, we use dynamic programming to maximize Eq. 2.1 (with Eq. 2.1 & Eq. 2.4) over all phones.

We encode the previous phone with a scheme similar to that for phonemes, but add a few extra categories to fully specify all the phones.

The result is an improvement to 85.5% correct predictions on the TIMIT test set described in Section 5. Significantly, the implausible predictions like those described above near a deletion are judged to have very low probability with this newer model.

6. TREATMENT OF INSERTIONS

There are two ways to deal with the insertion of phones. The first way is add a second model that predicts the phone insertions. Consider a phone sequence $z_0 y_1 z_1 y_2 z_2 ... y_n z_n$ that is the realization of phoneme sequence $x_1 x_2 ... x_n$. We view phone y_i as the realization of phoneme x_i and view phone z_i as an insertion between phoneme y_i and y_{i+1}. In a realistic example, there will be only an occasional insertion, so most of the z_i insertions will marked as `[-]'s`. This scheme allows only one phone to insert after a phoneme; however, it is clear this can be generalized. In practice, contiguous insertions seldom occur. For example, for 100702 TIMIT phonemes, there were 13907 single insertions but only 352 multiple insertions.

The second way to deal with insertions is to augment the

output set to include phone pairs. For example, the phoneme /t/ can be realized as the pair of phones [tcl t]. The insertion tree approach accounts for this by treating one of these phones as an insertion. Instead, we might add [tcl+t] to our 'phone' set. The advantage of this approach is we can use the methods described in Section 4 and 5 without any need to predict insertions separately. The potential disadvantage is that we could conceivably need to square the size of our output phone set.

Fortunately, in practice, only a few phone pairs are found commonly. In particular, a stop consonant closure pairs with its release, and a glottal stop inserts before a vowel (e.g., phrase initially). We used the 37 most common phone pairs to augment our phone set. This set accounts for 95% of the insertion tokens in the TIMIT database.

A classification tree grown using this output set and the model of Section 5 predicts the realization of a phoneme correctly 83.3% of the time and has a conditional entropy of .82 bits. Note that the classification rate is lower here than in the previous models since this model must also predict phone insertions. Figure 1 shows an example of the output network for the sentence 'Don had your pretty red begonia in a little pot.'.

We can use dynamic programming to find efficiently the highest probability path through the network in Figure 1. In the cases where the outcome does not depend on the previous phone this simply means we are selecting the leftmost phone prediction displayed since it has the highest probability. When, however, it does depend on the previous phone, then we are dealing with a transition probability and must find which of several possible paths is best. The best result, in this case, is [d aa n hh ae dcl jh axr pcl p r ih dx iy r eh dcl b ix gcl g ow n y ax ix n ax l ih dx el pcl p aa tcl].

7. DISCUSSION OF RESULTS

The example pronunciation network in Figure 1 illustrates several features of this scheme. The first column in that figure gives the phoneme to realize. Pairs of probabilities and phones follow. For example, the initial phoneme, /d/, is predicted to realize as the phone [d], i.e., d release. with a 91% probability. Realizations with less than 10% probability are pruned from this figure.

On the other hand, the phoneme /d/ in 'had' is predicted to realize as [dcl jh] with 51% probability and as [dcl d] with 37% probability. This is an example where an alternative pronunciation is quite likely.

Some realizations depend on the realization of the previous phoneme. For example, the phoneme /y/ in 'your' will delete with 73% probability if the previous /d/ was realized as [dcl jh] but will appear with 90% probability as [y] if the /d/ was realized as [dcl d].

Finally note that several non-trivial transformations have been captured – the flapping of the /t/ in 'pretty', the realization of /uh l/ in 'little' as a syllabic l, and the combination of 'had' and 'your' giving rise to the affricate [jh].

9. REFERENCES

[1] Shoup, J. 1980. Phonological aspects of speech recognition. In *Trends in Speech Recognition*. W. Lea, ed. NY: Prentice-Hall. pp. 125-138.

[2] Fisher, W., Zue, V., Bernstein, D. and Pallet, D. 1987. An acoustic-phonetic data base. *J. Acoust. Soc. Am.* **81.** Suppl. 1.

[3] Coker, C.. 1985. A dictionary-intensive letter-to-sound program. *J. Acoust. Soc. Am.* **78**, Suppl. 1, S7.

[4] Ladefoged, P. 1982 *A Course in Phonetics*. New York: Harcourt, Brace, and Jovanovich.

[5] Chen, F. 1990. Identification of contextual factors for pronunciation networks. *Proc. ICASSP '90.* S14.9.

[6] Riley, M. 1989. Some applications of tree-based modelling to speech and language. *Proc. DARPA Speech and Natural Language Workshop.* Oct 1989, Cape Cod, MA, pp. 339-352.

[7] Randolph, M. 1990. A data-driven method for discover and predicting allophonic variation. *Proc. ICASSP '90.* S14.10.

[8] Kruskal, J. 1983. An overview of sequence comparison. In *Time Warps, String Edits, and Macromolecules: the Theory and Practice of Sequence Comparison.* D. Sankoff and J. Kruskal, eds. Reading, MA: Addison Wesley. pp. 1-44.

[9] Brieman, L., et. al. 1984. *Classification and Regression Trees.* Monterey, CA: Wadsworth & Brooks.

PHONEME	PHONE1	PHONE2	PHONE3	CONTEXT
d	0.91 d			
aa	0.92 aa			
n	0.98 n			
hh	0.74 hh	0.15 hv		
ae	0.73 ae	0.19 eh		
d	0.51 dcl jh	0.37 dcl jh		
y	0.90 y			(if d→dcl d)
	0.84 -	0.16 y		(if d→dcl jh)
uw	0.48 axr	0.29 er		
r	0.99 -			
p	0.99 pcl p			
r	0.99 r			
ih	0.86 ih			
t	0.73 dx	0.11 tcl t		
iy	0.90 iy			
r	0.99 r			
eh	0.87 eh			
d	0.80 dcl	0.15 dcl d		
b	0.93 bcl b			(if d→dcl d)
	0.96 b			(if d→dcl)
ih	0.58 ix	0.24 iy	0.15 ih	
g	0.85 gcl g	0.15 gcl		
ow	0.94 ow			
n	0.98 n			
y	0.90 y			
ax	0.70 ax	0.12 ah		
ih	0.46 ix	0.25 ih		
n	0.56 n	0.44 nx		
ax	0.53 ax	0.16 ix		
l	0.95 l			
ih	0.86 ih			
t	0.73 dx	0.11 tcl t		
uh	0.89 el			
l	0.98 -			
p	0.99 pcl p			
aa	0.92 aa			
t	0.57 tcl	0.26 tcl t		

Figure 1. *Pronunciation network for* 'Don had your pretty red begonia in a little pot.' *The first column gives the phoneme to realize. Pairs of probabilities and phones follow. For example, the initial phoneme, /d/, is predicted to realize as the phone [d], i.e., d release. with a 91% probability. Realizations with less than 10% probability are pruned from this figure.*