# Between-Word Coarticulation Modeling for Continuous Speech Recognition

Mei-Yuh Hwang, Hsiao-Wuen Hon, Kai-Fu Lee

April 22, 1989
CMU-CS-89-141

School of Computer Science
Carnegie-Mellon University
Pittsburgh, PA 15213

## Abstract

Between-word coarticulation is a major source of phonetic variability in continuous speech. Yet, previous systems have ignored this important effect. In this paper, we extend the current generalized triphone modeling in the SPHINX system to incorporate between-word triphones. We report error rate reduction of 16%-20% by modeling between-word coarticulation for different test sets and grammars on the Resource Management Task.

This research was sponsored by Defense Advanced Research Projects Agency Contract N00039-85-C-0163. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency, or the US Government.

# Table of Contents

# 1. Introduction

In large-vocabulary continuous speech recognition, subword units must be used for practical reasons. Context-dependent phoneme models, such as triphones [Schwartz 85] or generalized triphones [Lee 89], have become a very successful class of subword units. These phoneme-sized models take into account the neighboring phonetic contexts, which strongly affect the realization of a phoneme. However, previous approaches have only considered *within-word* coarticulation, and have ignored *between-word* coarticulation, which is very important in continuous speech. This paper describes the addition of between-word coarticulation modeling to the SPHINX system [Lee 88].[1]

Triphone models are a powerful subword modeling technique because they account for the left and right phonetic contexts, which are the principal causes of phonetic variability. However, previous triphone models considered only within-word context. For example, in the word SPEECH (`/s b iy ch/`), both left and right contexts for `/b/` and `/iy/` are known, while the left context for `/s/` and the right context for `/ch/` were a special symbol for "word boundary". Yet in continuous speech, phonemes at the boundary of two consecutive words are more or less affected by each other. This is especially true for short function words like *the* or *a*. A simple extension of triphones to model between-word coarticulation is problematic because the number of triphone models grows sharply when between-word triphones are considered. For example, there are 2381 within-word triphones in our 997-word task. But there are 7549 triphones in our training data when between-word triphones are also considered (If all pairs of words are considered, there are 21,946 triphones. Even with the constraint of a grammar, there are 12,051 triphones.) For SPHINX, 7549 models means over 17,000,000 parameters, which we cannot hope to train with the limited training set.

Instead, we use generalized triphones to model between-word coarticulation. Generalized triphone models were introduced to combine similar triphone contexts, and to reduce the large number of triphone models. We first generate 7549 triphone models that accounted for both within-word and between-word triphones. These 7549 models are then clustered into 1100 generalized triphone models according to their similarity. Few program modifications were needed for training, since the between-word context is always known. However, care must be taken for single- and double-phone words while constructing sentence models in the training program. On the other hand, there are also some issues which cannot be ignored in the recognition programs. First of all, each word now has multiple beginning and ending triphones, depending on which words can connect to and from the word. Secondly, while connecting two words, we must ensure that we use the correct triphones to connect them.

We incorporated the above changes into SPHINX, and evaluated them using the 997-word

[1]We understand that SRI and Lincoln Lab independently developed between-word coarticulation modeling at the same time.

DARPA resource management task on the speaker-independent database. We report a 16% ~ 20% reduction in error rate for various test sets. This result demonstrates the importance of between-word coarticulation modeling.

In this paper, we will begin with a spectrogram demonstration of how phonemes at word boundaries are affected by neighboring words. Then, we will explain the training and recognition modifications needed to extend SPHINX from within-word triphones to between-word triphones in detail. Next, we present results and comparisons with previous SPHINX versions and other recognizers. Finally, we conclude this paper with a summary.

# 2. Word Boundary Coarticulatory Effects

Before starting on techniques, let's take a view of how speech variates while crossing word boundaries. The spectrograms shown in Figure 2-1 give us a vivid example. The high frequency of the word YELLED pulls a bit up its preceeding word WE, while the low frequency of WERE pulls down the spectrogram of the same word. What this example suggests us is that although it is the same word WE, nevertheless we should use two different /i/ phones for these two cases since their right context phones are different.

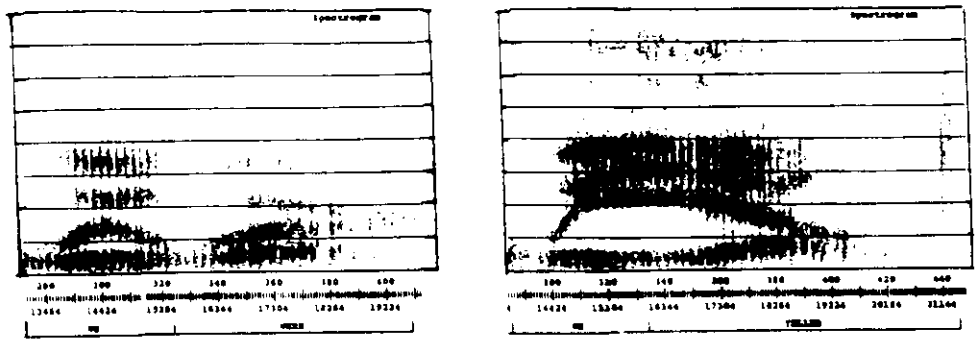

**Figure 2-1:** The spectrograms of WE YELLED and WE WERE.

Even with the same left and right context phones, a phone may still have significantly different realizations at different word positions (the beginning, middle, or ending part). For example, in Figure 2-2, the /t/ phone in THAT ROCK has the same contexts as the /t/ phone in THEATRICAL. However, the /t/ at the end of THAT is almost extinct while the /t/ in the middle of THEATRICAL sounds like /ch/. In fact, our experiments showed that only about 37% of the same triphones but at different word positions are similar. This implies the coarticulations within and between a word are mostly different.

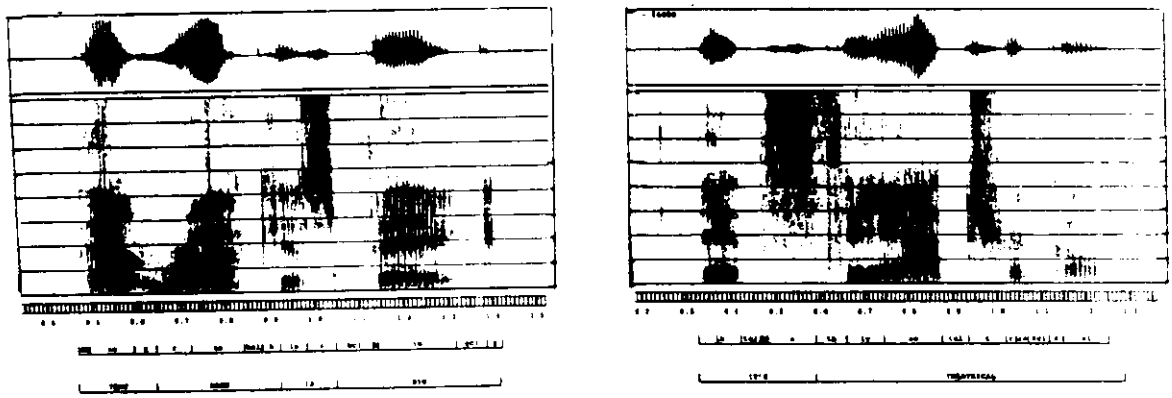

**Figure 2-2:** The spectrograms of THAT ROCK and THEATRICAL.

Singe-phone words are even harder to be modeled. They are so short that their boundaries

with neighboring words are unclear as shown in Figure 2-3, which is the spectrogram of THAT IS A WORD. We think it may be a good idea to model single-phone words separately.

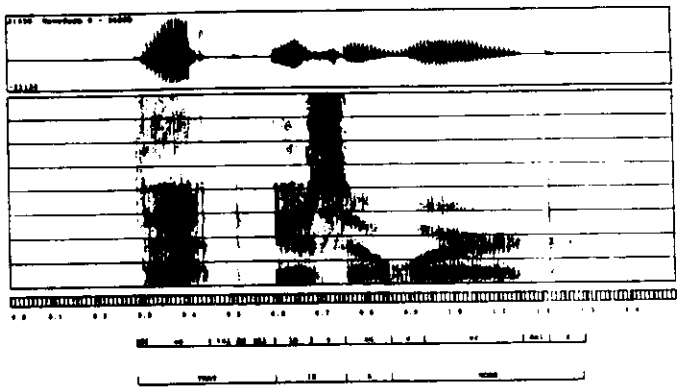

**Figure 2-3:** The spectrograms of THAT IS A WORD.

To take into account all the phonetic variability in (1) within-word triphones, (2) between-word triphones, and (3) phonetic locations, we come up with four types of triphones. The first type is the usual triphone which is embedded within a word. For instance, the /t/ phone in THEATRICAL. From now on, we will use the notation P(L,R) to represent a triphone P with a left context phone L and a right context phone R. SIL is used to stand for the model of silence. So, this exemplary /t/ will be written as TD(AE,R).

The second type of triphone is a triphone which appears at the beginning of a word. For example, the /r/ phone in THAT ROCK. This type of triphone will be postfixed with a letter b. So the exemplary /r/ is written as R(TD,AA)b.

The third type is a triphone appearing at the end of a word and is postfixed with a letter e. For example, the /t/ phone in THAT ROCK is denoted by TD(AE,R)e.

The last type of triphone is the only phone of a single-phone word with its left and right context phones specified. For example, in THAT IS A WORD, the /a/ phone is represented by AX(Z,W)s. Note that this type of triphone is postfixed with a letter s.

To sum up, while the old SPHINX has only the first type of triphones, the new SPHINX has four types of triphones in order to incorporate word boundary coarticulation modeling.

# 3. Between-Word Triphone Modeling in SPHINX System

## 3.1 The Framework of SPHINX

SPHINX is a large-vocabulary, speaker-independent, continuous-speech recognition system based on discrete phonetic Hidden Markov Models (HMMs) [Lee 88].

The system is divided into three parts: signal processing, training and recognition as shown in Figure 3-1. In this subsection, we will only introduce briefly the signal processing part. The trainer and recognizer will be explored in the subsequent subsections.

The input speech is sampled at 16KHz, and pre-emphasized with a filter of $1 - 0.9Z^{-1}$. Then a Hamming window with a width of 20 msec is applied every 10 msec. Autocorrelation analysis with order 14 is followed by LPC analysis with order 14. Finally, 12 LPC-derived cepstral coefficients are computed from the LPC coefficients, and these LPC cepstral coefficients are transformed to a mel-scale using a bilinear transform.

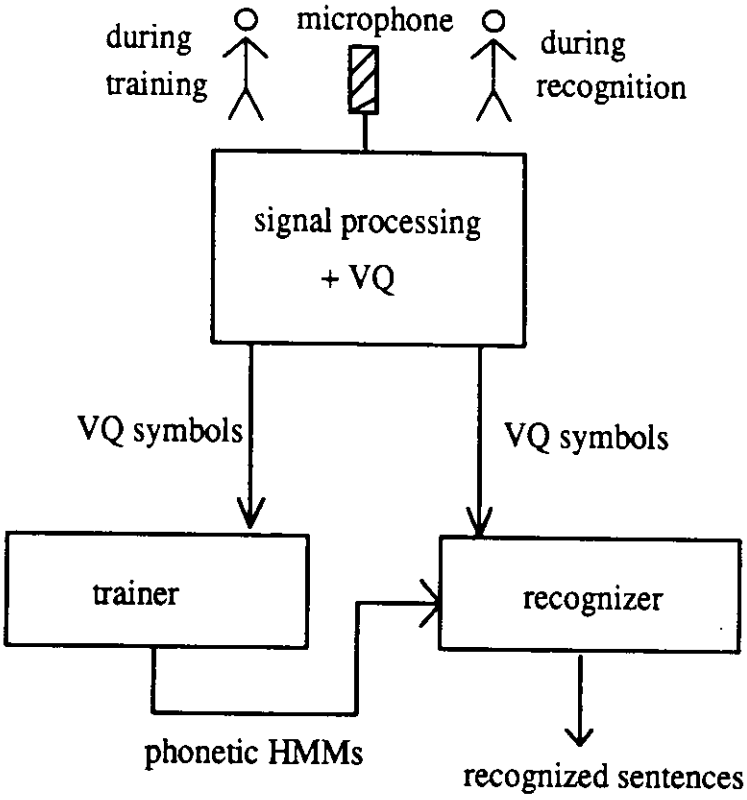


**Figure 3-1:** The framework of SPHINX.

These 12 coefficients are vector quantized (VQ) into a codebook of 256 prototype vectors. In order to incorporate additional speech parameters, we created two additional codebooks. One codebook is vector quantized from *differential coefficients*. The differential coefficient of frame

*n* is the difference between the coefficient of frame *n*+2 and frame *n*-2. This 40 msec. difference captures the slope of the spectral envelope. The other codebook is vector quantized from *energy* and *differential energy* values.

Each phone is modeled by an HMM. The topologies of HMMs are all the same and shown in Figure 3-2. Each phonetic HMM contains three discrete output distributions of VQ symbols. Each of the 12 transitions is tied to one of the distributions. A distribution is the joint density of the three codebook pdf's, which are assumed to be independent. The use of multiple codebooks was introduced by Gupta, *et al.* [Gupta 87].

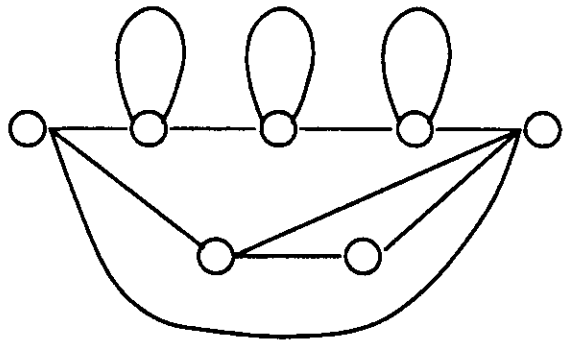

**Figure 3-2:** The topology of HMMs used in SPHINX.

The following two subsections will concentrate on the training and recognition parts respectively. The words `WHAT+S MARS STATUS` will be used extensively as examples. Context-independent phones, such as `/t/` and `/iy/`, will sometimes be called *monophones*.

## 3.2 SPHINX Training Procedure and its Modifications

There are three inputs to the trainer as shown in Figure 3-3. The first input is a phone file which contains all those phones known to the system. The second input is a pronunciation dictionary which specifies how each word is pronounced in terms of those phones in the phone file. The final input consists of pairs of vector-quantized speech and the corresponding sentence text. Note that during training, the correct sentence texts are always known.

For each input sentence, the trainer first builds a corresponding sentence HMM. Then it runs the forward-backward algorithm [Bahl 83] on the sentence HMM using the associated input speech. After running forward-backward on the whole set of training sentences, it re-estimates the HMM parameters using the Baum-Welch reestimation formulas. The entire reestimation procedure is repeated several times to get better estimated parameters. If the training set is not rich enough, some phones may occur too seldom or not at all. When this happens, an interpolation with context-independent phones is used [Jelinek 80, Lee 88]. The set of phonetic HMMs generated by the trainer is tested in the recognizer.

The following two subsections will discuss how to generate the phone file and

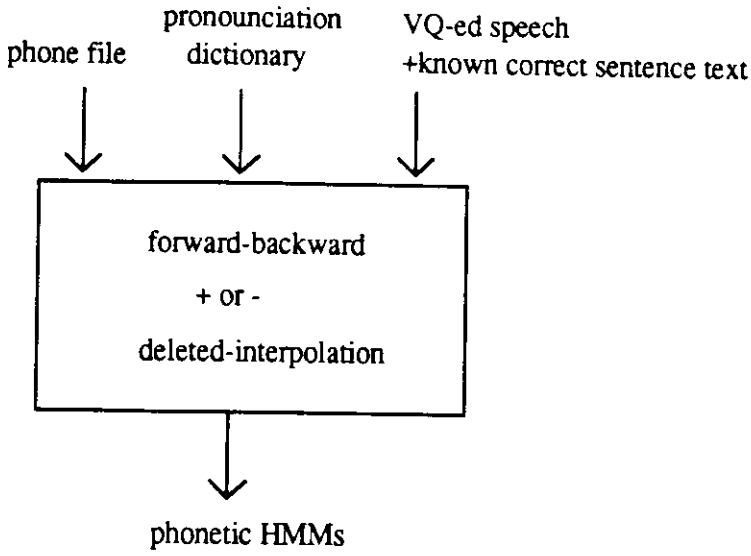


**Figure 3-3:** The SPHINX training process.

pronunciation dictionary, and how to construct a sentence HMM. We won't talk about the reestimation algorithm. Interested readers are encouraged to refer to related materials.

### 3.2.1 The Phone File and Pronunciation Dictionary

When only within-word triphones are considered, there is no left-context phone for the first phone of a word; nor is there a right-context phone for the last phone of a word. Instead, a special symbol, #, is used as word boundary. So the pronunciation of the word `SPEECH` becomes `/s(#,b) b(s,iy) iy(b,ch) ch(iy,#)/`. The monophone dictionary described in [Lee 88] is used to produce the Within-Word Triphone (WWT) dictionary. There are a total of 2381 within-word triphones in the Resource Management Task.

For the between-word triphone modeling job, we generate all of the within- and between-word triphones (e.g., the four types of triphones mentioned in Section 2) occurring in our training data. There are 7549 triphones in total, which means `7549 * (12 + 9 * 256) = 17,483,848` parameters in SPHINX.

We also, for each word, find which words can be connected to and from it in the training data. Then according to this connectivity and the monophone pronunciation dictionary, we generate a Between-Word Triphone (BWT) pronunciation dictionary which for each word specifies *multiple* beginning and ending triphones, in addition to the middle within-word triphones. Figure 3-4 shows parts of the monophone dictionary, the within-word triphone dictionary, and the between-word triphone dictionary. In the BWT dictionary, beginning and ending triphones are enclosed separately by brackets. Ellipses are used in the figure to save

space.

```
A                AX
MARS             M AA R S
THE              DH AX
STATUS           S D AE DX AX S
```

(a) Parts of the monophone dictionary.

```
A        AX(#,#)
MARS     M(#,AA) AA(M,R) R(AA,S) S(R,#)
THE      DH(#,AX) AX(DH,#)
STATUS   S(#,D) D(S,AE) AE(D,DX) DX(AE,AX) AX(DX,S) S(AX,#)
```

(b) Parts of the WWT dictionary before generalization.

```
A        < AX(DD,B)s AX(DD,S)s AX(DD,K)s AX(DD,CH)s
           AX(DD,G)s AX(DD,HH)s AX(DD,L)s AX(DD,M)s
           AX(DD,P)s AX(DD,SH)s AX(DD,SIL)s .... >
MARS     < M(V,AA)b M(N,AA)b M(SIL,AA)b M(K,AA)b
           M(AX,AA)b > AA(M,R) R(AA,S) < S(R,SIL)e
           S(R,S)e S(R,T)e >
THE      < DH(DD,AX)b DH(L,AX)b DH(R,AX)b DH(T,AX)b
           DH(KD,AX)b DH(OW,AX)b DH(N,AX)b  .... >
         < AX(DH,EY)e AX(DH,AX)e AX(DH,AE)e AX(DH,AA)e
           AX(DH,B)e AX(DH,S)e AX(DH,K)e  .... >
STATUS   < S(L,D)b S(N,D)b S(SIL,D)b S(AX,D)b > D(S,AE)
           AE(D,DX) DX(AE,AX) AX(DX,S) < S(AX,AX)e
           S(AX,SIL)e >
```

(c) Parts of the BWT dictionary before generalization.

**Figure 3-4:** Various pronunciation dictionaries.

### 3.2.2 The Construction of Sentence HMMs

During training, a sentence model is constructed for each input sentence text. Sentence models are built by concatenating their component word models. A word model is in turn the serial concatenation of its component phones. Finally each phone is instantiated by its corresponding HMM.

When only within-word triphones are considered, the formation of word models from phonetic HMMs and the formation of sentence models from word models are made with the usual serial connections. Take the sentence WHAT+S MARS STATUS as an example. Its sentence model construction is shown in Figure 3-5. The skip over silence allows words to be spoken continuously without pause.

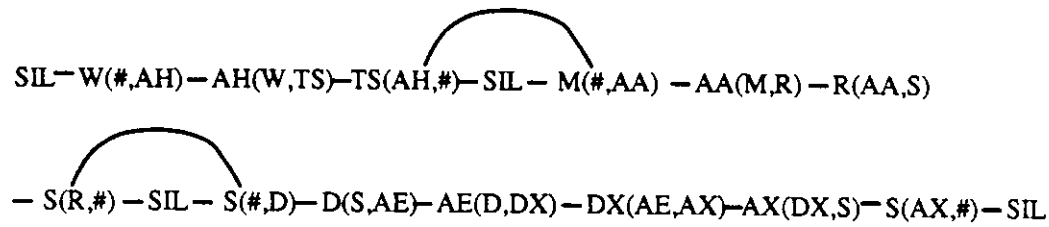


**Figure 3-5:** The sentence model of WHAT+S MARS STATUS when only within-word triphones are considered.

The construction of sentence HMMs in the between-word system is more complicated than that in the within-word system. The basic idea is still to allow each consecutive pair of words to have two ways of speaking. One is through a pause or silence by using the phones X(L,SIL), SIL and Y(SIL,R); this is similar to the above connection without skipping over SIL. However, the second way is to speak continuously, where we must take between-word contexts into account. That is, different pairs of words may need different triphones to connect to each other, instead of always using the same word-boundary phone X(L,#) and Y(#,R). Because these two ways of speaking use different ending and beginning triphones, a binary branch is created at each word boundary position. Thus the sentence model of the same sentence WHAT+S MARS STATUS becomes Figure 3-6 when between-word triphones are considered.

Unfortunately, the construction of sentence models is not so easy as it seems to be. Complications arise when a sentence contains single-phone or double-phone words. Examining Figure 3-6, we find it is always true that at the last phone of a word, there must be a binary branch to allow the two ways of speaking mentioned above. Then these two endpoints are connected to two different beginning phones of the next word. These two beginning phones are both models of the first phone of the next word but with different left contexts, one with SIL as its left-context phone, the other with the last phone of the previous word as its left-context phone. One of the important responsibilities of the second phone of the next word is to merge this branch before continuing to subsequent phones. Therefore in order for the branch to be merged, it would be better if the second word consisted of at least two monophones; one for the two beginning phones, the other for merging.

However, as Figure 3-7 points out, if the second word, THE, consists of exactly two monophones, then while the second monophone is merging the branch, it is also the last phone of the second word. That means another branch happens again since we wish to have two ways of

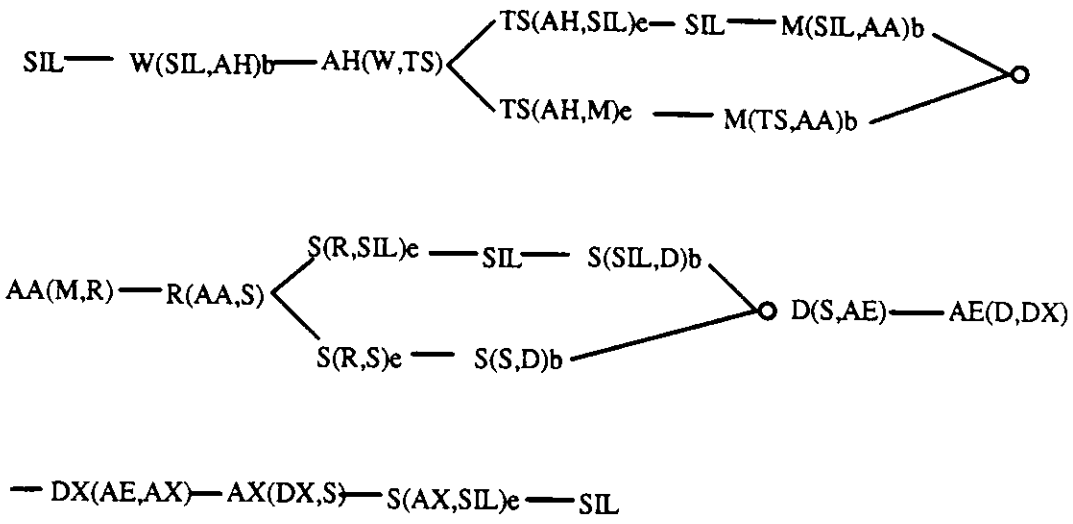


**Figure 3-6:** The sentence model of `WHAT+S MARS STATUS` with between-word triphones.

speaking at every word boundary. Therefore we need something to merge the first branch before the second branch starts.

Things get even worse when the second word is single-phone. Not only will there be no merge at the first word boundary position, but two further binary branches, continuing from the two endpoints of the first branch, are needed (see Figure 3-8). Programming gets more and more complex when many consecutive single- and double-phone words are concatenated. Although there is only one single-phone word (i.e., `A`) in the Resource Management Task, we cannot evade this problem if we want to apply our system in a general way. Moreover, there are many words in our task which consist of only two monophones.

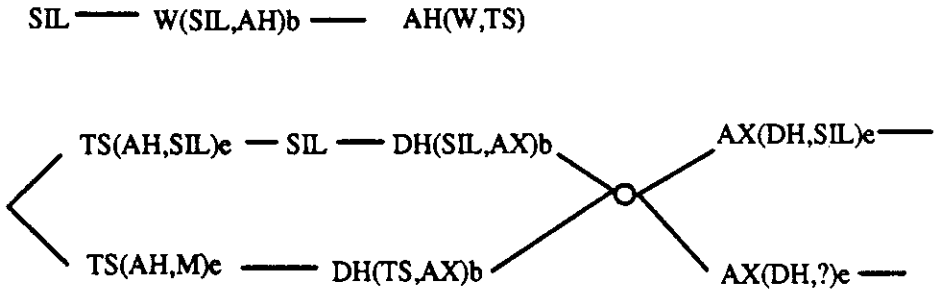


**Figure 3-7:** The branch condition of `WHAT+S THE....`

There are two possible solutions for the merging points illustrated as circles in the above figures. One way is to link all the final states of an HMM representing an ending phone of some

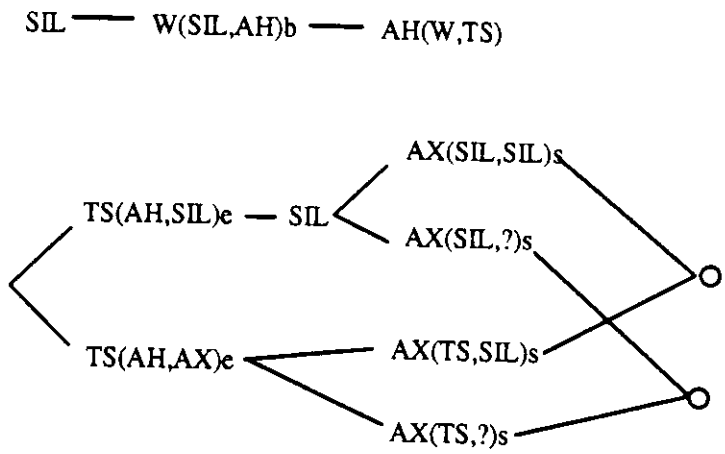


**Figure 3-8:** The branch condition of `WHAT+S A....`

word to all the initial states of another HMM corresponding to a beginning phone of the next word. However, for the sake of easy maintenance and debugging of the system, we divide the construction of sentence models into two levels; one is the *node* level; the other is the *state* level. At the node level, a node corresponds to an entire phonetic HMM. Merging points are considered as explicit extra nodes, called *dummy* nodes. At the state level, a phonetic node is expanded into 12 transitions and 7 states as shown in Figure 3-2. However, a dummy node contains only one state and no transition within itself.

By adopting dummy nodes, we find a simple rule. While there is always only one link going into other nodes, there are always 2 links going into a dummy node. On the other hand, either dummy nodes or phonetic nodes may have 1 or 2 links going out of them. With this general rule, the construction of sentence models becomes much easier. Although sometimes dummy nodes may be unnecessarily introduced as in Figure 3-6, it hardly degenerates the trainer since each dummy node contains only one state and it does not need to be processed specially in the forward-backward and reestimation algorithms. Note that dummy nodes must be generated *after* the merged nodes in order to maintain the left-to-right temporal property of HMMs.

To appreciate the generality given by dummy nodes, let's consider the pseudo sentence `TO A TRACK`. The sentence model is shown in Figure 3-9. Whenever there is a branch, the two endpoints of the branch are labeled as *node 1* and *node 2*. If the next word is a single-phone, then the two triphones of the single-phone word with `SIL` as their right-context phone are considered first. They are labeled as *node 3* and *node 4*. If this single-phone word is the end of a sentence, then *node 1* and *node 2* will have no second successor nodes and the `SIL` after the first dummy node ends the sentence model. If this is not the end of a sentence, as in our example `TRACK` is not yet constructed, then after the first dummy node is built, the two triphones (i.e. `AX(SIL,T)` and `AX(UW,T)` in the example) of the single-phone word with the first monophone of the third word as their right context phone become the new *node 3* and *node 4*. Then a second dummy node is generated to merge them. The silence after the first dummy node

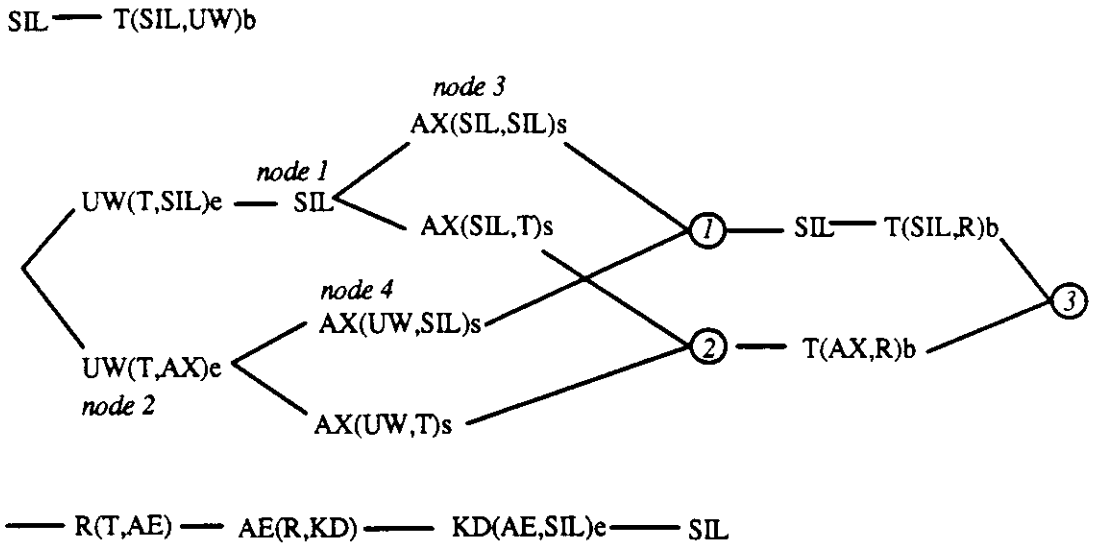


**Figure 3-9:** The sentence model of `TO A TRACK` with between-word triphones.

becomes *node 1* and the second dummy node becomes *node 2*. They are to be merged by the third word. From here on, the same construction procedure is repeated for the rest of the input sentence as though this were the first branch. With the help of dummy nodes, programming is now easier.

After getting the sentence HMM, we run the forward-backward algorithm on it using the corresponding input speech. After all of the training sentences have got their forward-backward counts (or probabilities), the HMM parameters are reestimated by the Baum-Welch formulas. The entire training process from the construction of sentence models to reestimation is run for several iterations (2 to 4 iterations in our experiments) to get better estimated parameters. Except for the construction of sentence models, the trainer remains almost unchanged. This demonstrates the maintenance and efficiency of our system.

### 3.2.3 Generalized Triphones

As discussed in earlier sections, 7549 triphones means over 17,000,000 parameters. With only 4000 training sentences or so, most parameters will be zeroes. Therefore we need some way to reduce the number of HMMs. The same clustering procedure introduced in [Lee 88] is used to merge similar triphones together. The outline of the clustering procedure is as follows:

1. An HMM is generated for every triphone context.
2. Clusters of triphones are created; initially, each cluster consists of one triphone.
3. Find the *most similar* pair of clusters which represent the same monophone, and merge them together.
4. For each pair of clusters, consider moving every element from one to the other.
   - Move the element if the resulting configuration is an improvement.
   - Repeat until no such moves are left.
5. Until some convergence criterion is met, go to step 3.

The *similarity* between two HMMs is determined by the amount of information lost when the two models are merged. We use entropy of an HMM to measure the amount of information it contains. The lost information is equal to the weighted difference between the merged entropy and the original entropies. Forward-backward counts (frequencies) are used as the weights. By preferring to merge models that do not appear frequently, each clustered or generalized model will be more trainable. The clustering result shows that 37% of the word-boundary (type 2, 3, or 4) triphones are merged into the same cluster as their within-word triphones.

After clustering, we map the BWT triphone file and dictionary generated in Section 3.2.1 to a new generalized triphone file and a new generalized dictionary according to the mapping produced by the clustering procedure. Then we re-train these generalized triphones as the first stage did[2], but use deleted-interpolation with context-independent phones after the last iteration of reestimation. Interpolation is needed because there are still many zero probabilities. These better-trained generalized models are tested in the final recognizer. Our best system currently has 1100 generalized BWT triphones.

The system mentioned in Section 4, with within-word triphones only, contains 1076 generalized phones. These generalized triphones are obtained by clustering the 2381 within-word triphones mentioned in section 3.2.1 and some *function-word dependent* phones.

## 3.3 SPHINX Recognition Procedure and its Modifications

After phonetic HMMs are obtained, they are supplied as input to the recognizer as shown in Figure 3-10. The speech uttered by a talker through a microphone is processed by the same signal processing subsystem and then as another input to the recognizer. The third input to the recognizer is a *language network* which contains the set of vocabulary specified in a task and all

[2]Remember to map into its corresponding generalized triphone before using a phone.

the connections between phones and words. The language network is pre-built before the recognizer is executed. Thus it is a static database for the recognizer to search during recognition.[3] The recognizer uses the Viterbi beam search [Schwartz 85, Ney 87] to find the optimal state sequence in the language network for each input speech. Except for the generation of the language network (the box named NetGen in Figure 3-10), the recognizer is almost intact to incorporate between-word triphone modeling.

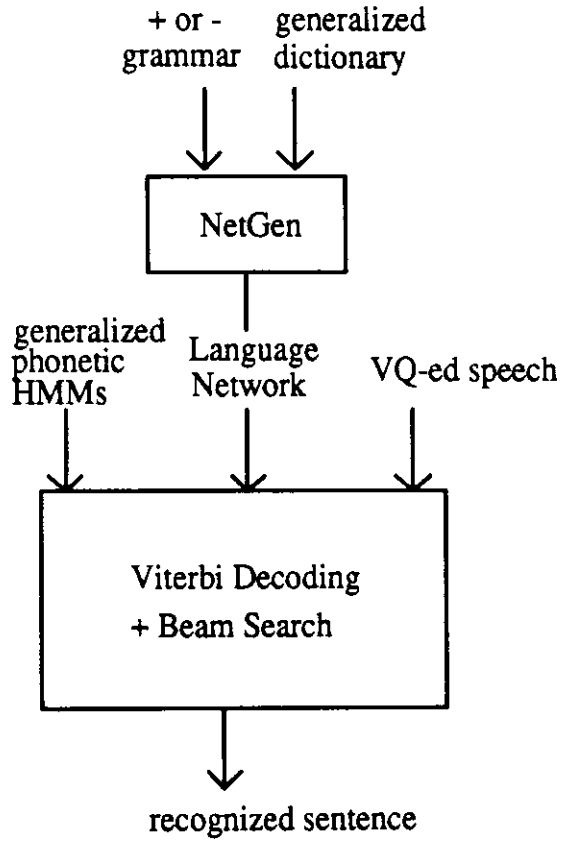


**Figure 3-10:** The SPHINX recognition procedure.

The following subsection will concentrate on the generation of the language network. The Viterbi beam search is explicated in listed references.

[3]The connections between phones and words can also be done dynamically during recognition.

### 3.3.1 The Language Network

For the recognizer to work, it must be told what the set of words we want it to recognize is and how the words are connected. These are exactly the purposes of the language network. The language network lists the set of vocabulary and makes explicit links between pairs of words with or without the constraint of the word-pair grammar used in the Resource Management Task. There are three issues to be concerned with while generating the language network: the connection within a word, the connection at word boundaries and unknown triphones. For a clearer explanation, the full triphone names will be used in the following paragraphs. Readers should keep in mind that as Section 3.2.2 says, generalized triphones, instead of full-context triphones are used in the recognizer. Therefore before any usage of a triphone in the recognizer, the phone should be mapped into its generalized triphone.

#### 3.3.1.1 The Connection Within a Word

As the construction of sentence models in the trainer, a word model is formed by concatenating its component phones specified in the *generalized* pronunciation dictionary created in Section 3.2.2. Each phonetic node is in turn instantiated by its corresponding HMM.

When only within-word triphones are considered, each word model has only one beginning and one ending triphone. Thus a word model is just the linear sequence of its component triphones as the example in Figure 3-11 illustrates.

M(#,AA) — AA(M,R) — R(AA,S) — S(R,#)

**Figure 3-11:** The word model of MARS with within-word triphones only.

However, when between-word triphones are added, words have multiple beginning and ending triphones since they may be connected from and to more than one word. The possible beginning and ending triphones of a word are depicted in pairs of brackets in the generalized dictionary, which is generated from the training sentences. The word model of MARS is shown in Figure 3-12.

#### 3.3.1.2 The Connection at Word Boundaries

Like the trainer, words are connected in two ways in the language network. When a system has only within-word triphones, the only ending phone of a word is connected to a SIL and the SIL is connected to the only beginning phone of another word, with the option to skip over the SIL.

When between-word triphones are taken into consideration, the connection through pause is done by linking all the ending triphones of a word with a SIL right context to a SIL node, and then linking the SIL node to all the beginning triphones of another word with a SIL left context.

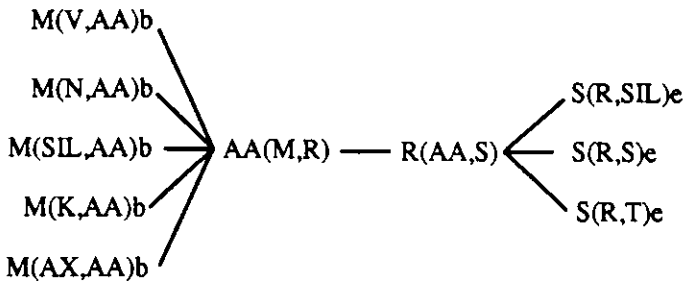


**Figure 3-12:** The word model of MARS with between-word triphones.

The number of beginning or ending triphones of a word with a particular left *or* right context is usually one. But for single-phone words the number of triphones with only one side of contexts specified is usually more than one, depending on how many words can be connected to and from it.

The second way of connecting two words is achieved directly by between-word triphones. It models between-word coarticulation. Be sure to use the correct between-word triphones (type 2, 3, or 4) to connect two words. Parts of the language network without and with between-word triphones are shown respectively in Figure 3-13 and Figure 3-14. Note that in the latter network, WHAT+S uses different /ts/s to connect to A and THE. Note also that there are more than two links between WHAT+S and A since A is a single-phone word. Once again, we remind readers that these *un*generalized triphones are used here just for explanation. In fact, it is generalized triphones that are used in the recognizers, both in the within-word triphone system and in the between-word triphone system.

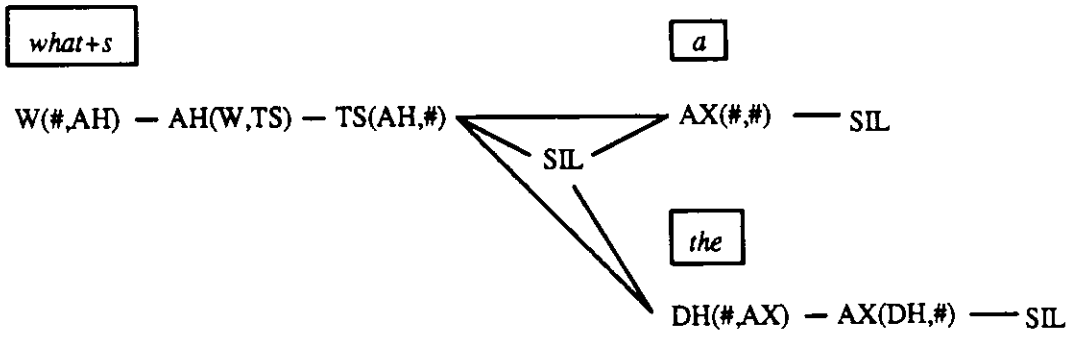


**Figure 3-13:** Parts of the language network with within-word triphones only.

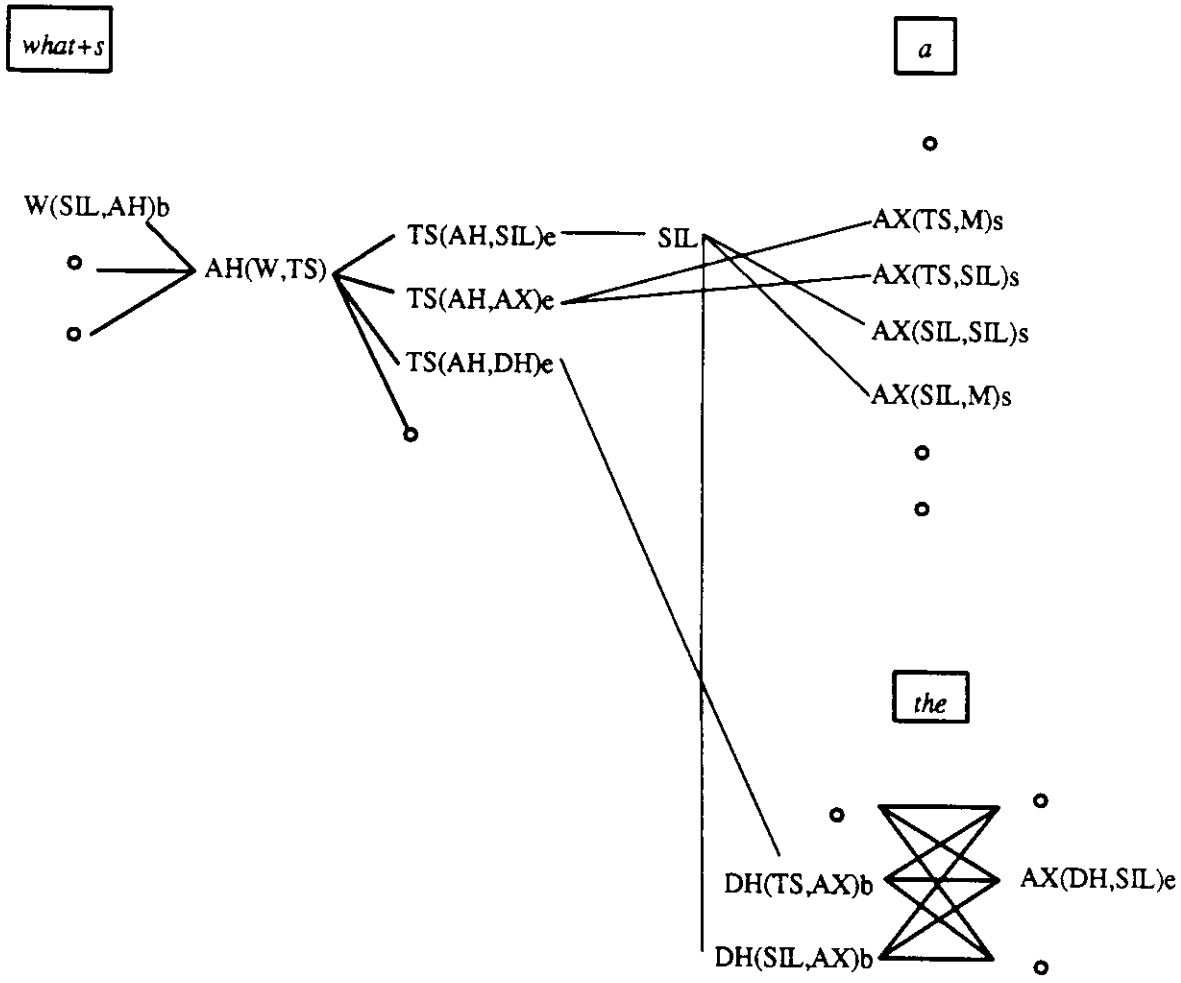


**Figure 3-14:** Parts of the language network with between-word triphones. Words now have multiple beginning and ending triphones.

### 3.3.1.3 Unknown Triphones

One possible problem for the BWT language network is that the 7549 triphones are constructed from the training data. Due to the incompleteness of training sentences, there exist other between-word triphones which are allowed by the grammar (or no grammar) but are not observed in the training data. In fact, as we pointed out earlier in Section 1, there are `12051 - 7549 = 4502` and `21946 - 7549 = 14397` such triphones for the word pair grammar and no grammar respectively. Hence the way of between-word triphone connection is not always possible. To relieve this problem, we use the word boundary triphone (that with # as contexts) in the within-word triphone system whenever that particular triphone does not occur in the training data. These word boundary triphones are called Word-Context Free (WCF) phones by Lincoln Lab. We have also done experiments by using context-independent phones when a triphone is not found. The recognition rates of both approaches are almost the same. Therefore it is better to use the 48 context-independent phones in SPHINX if space is an important factor.

However, the WCF approach is considered to be fairer to other systems and more appropriate when applied to future test sets. Both results will be discussed in Section 4.

### 3.3.2 Remarks

There is one subtle point which has been neglected. Examine Figure 3-15. Suppose word $w_2$ consists of a single phone, B. The last phone of word $w_1$ is A and the first phone of word $w_3$ is C. The word sequence $w_1$ to $w_2$ to $w_3$ is allowed by the word-pair grammar or no grammar. Let's put silence connection away temporarily and consider only the between-word triphone connection between two words. Block **a** of $w_1$'s ending triphones is connected to block **b.1** of $w_2$. Block **b.2** of $w_2$ is connected to block **c** of $w_3$'s beginning triphones. If there is no overlap between block **b.1** and block **b.2** (i.e., B (A, C) s does not exist in our phone file), then there will be no path for $w_1 \rightarrow w_2 \rightarrow w_3$ to go through. In this case, the best thing we can do is probably to allow a connection from block **a** to $w_2$'s WCF phone and a connection from $w_2$'s WCF phone to block **c**. The small circle in Figure 3-15 represents the WCF phone of $w_2$.

Currently, SPHINX does not take this situation into account. Although it may not contribute to more accuracy, we think it should be done according to our rationale.[4] Note also the links from **a** to **b.1** and from **b.2** to **c** cannot be eliminated since there may exist word sequences $w_1 \rightarrow w_2 \rightarrow w_i$ and $w_j \rightarrow w_2 \rightarrow w_3$ for some other words $w_i$ and $w_j$.

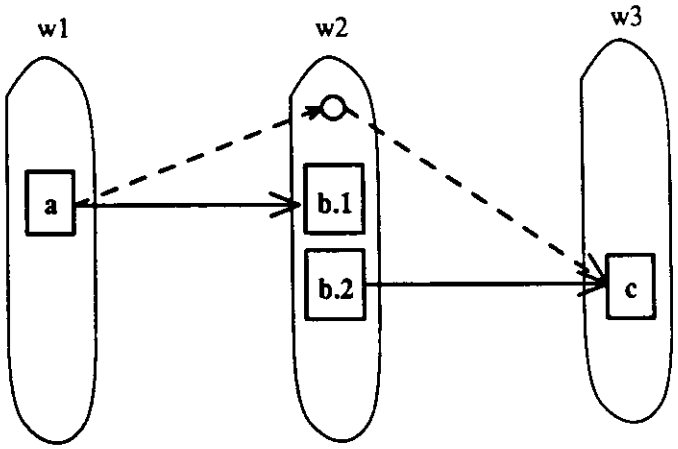


**Figure 3-15:** A bad situation when $w_2$ is single-phone and **b.1** $\cap$ **b.2** = $\varnothing$.

[4]Fortunately, this bad situation won't happen in a word consisting of more than one phone since **b.2** is always reachable from **b.1** within the word.

# 4. Results and Discussion

The algorithm described in Section 3 can be used in both recognition systems with and without a grammar. We applied SPHINX to the 997-word DARPA Resource Management task [Price 88], using 3990 training sentences (109 speakers) and two sets of test sentences: June88 with 300 sentences (12 speakers) and Feb89 with 300 sentences (12 speakers). Without a grammar, the perplexity is 997. With the word-pair grammar, which knows only about the legality of pairs of words, the perplexity is 60.

Our first experiment was run on the no grammar recognizer. With 1100 generalized between-word triphones and 4 iterations of Baum-Welch re-estimation, we obtain about 16% reduction in error rate.[5] During our experiments, we found that the usual 2 iterations of reestimation is not quite enough. It is probably because the learning problem is harder and takes a little longer to learn. The recognition results on June88 and Feb89 without a grammar are shown in Table 4-1, along with those of the within-word triphone system, which has a total of 1076 generalized triphones and function-word dependent phones. There are two between-word triphne systems. One uses WCF phones when between-word triphones are missing; the other uses context-independent phones.

| System Configuration | Accuracy on June88 Set | % Test Set Error Reduction |
|---|---|---|
| Within-word triphones | 66.2% | -- |
| + BWT (WCF) | 71.6% | 16.0% |
| + BWT (cxt-indep) | 71.8% | 16.6% |

| System Configuration | Accuracy on Feb89 Set | % Test Set Error Reduction |
|---|---|---|
| Within-word triphones | 68.7% | -- |
| + BWT (WCF) | 74.2% | 17.6% |
| + BWT (cxt-indep) | 74.8% | 19.5% |

**Table 4-1:** Results of different SPHINX versions without a grammar.

The same experiments are repeated on the word-pair grammar case and shown in Table 4-2.

SRI and Lincoln Lab have also independently developed between-word coarticulation

[5]An error could be a substitution, a deletion, or an insertion. Homonym confusions are considered correct recognition when no grammar is used, but are counted as substitution errors when a grammar is used.

| System Configuration | Accuracy on June88 Set | % Test Set Error Reduction |
|---|---|---|
| Within-word triphones | 91.0% | -- |
| + BWT (WCF) | 92.7% | 18.9% |
| + BWT (cxt-indep) | 92.6% | 17.8% |

| System Configuration | Accuracy on Feb89 Set | % Test Set Error Reduction |
|---|---|---|
| Within-word triphones | 90.8% | -- |
| + BWT (WCF) | 93.1% | 25.0% |
| + BWT (cxt-indep) | 93.5% | 29.3% |

**Table 4-2:** Results of different SPHINX versions with the word-pair grammar.

modeling. SRI does not use *generalized* triphones and thus adds a between-word triphone into their pronunciation dictionary only if that triphone occurs at least 15 times in the training set. Their tests on Tirmeval, which contains 150 sentences from 10 speakers, showed a 16.2% error reduction in the no grammar case and a 20.6% error reduction with the word-pair grammar[DARPA Workshop on Speech & Natural Language, February 1989]. Lincoln Lab, using continuous HMMs, showed 34.7% error reduction on the June88 speaker-*dependent* test set with the word-pair grammar. Yet the same experiment on the speaker-*independent* test set was worse than that without between-word coarticulation modeling. This may be due to the large increase in parameters.

# 5. Conclusion

Hidden Markov models with maximum-likelihood estimation are the predominant approach to automatic speech recognition today. However, the choice of a speech unit represented by an HMM is crucial. Previous studies have proved that triphone models are a practical and successful class of units. They model coarticulation effects caused by neighboring phonemes.

While previous studies considered within-word coarticulation effects, they have ignored between-word coarticulation, which is especially important in continuous speech. In order to take it into consideration, we add between-word triphone models into the SPHINX system. However, to control the growth of the number of parameters to be estimated, we use generalized triphones instead of the large number of complete triphones directly. The rough models of 7549 triphones are generated and trained first and then merged into 1100 clusters. These 1100 generalized triphones are then re-trained and used in the recognizer. We report 16% to 20% error reduction by adding between-word triphones only. The principal improvement comes from the improved modeling of triphones at word boundaries.

This work has shown that while HMM learning is very powerful, it is essential to use detailed models guided by speech knowledge, and to apply principled techniques to control the growth of the parameters. In the future, we will explore the modeling of other causes of phonetic variability.

## Acknowledgements

The authors would like to thank Raj Reddy for his encouragement and support. The authors would also like to thank Bob Weide for his helpful error analysis, and for providing the spectrograms.

# References


[Bahl 83] Bahl, L. R., Jelinek, F., Mercer, R.
A Maximum Likelihood Approach to Continuous Speech Recognition.
*IEEE Transactions on Patter Analysis and Machine Intelligence* PAMI-5(2):179-190, March, 1983.

[Gupta 87] Gupta, V.N., Lennig, M., Mermelstein, P.
Integration of Acoustic Information in a Large Vocabulary Word Recognizer.
In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 697-700. April, 1987.

[Jelinek 80] Jelinek, F., Mercer, R.L.
Interpolated Estimation of Markov Source Parameters from Sparse Data.
In E.S. Gelsema and L.N. Kanal (editor), *Pattern Recognition in Practice*, pages 381-397. North-Holland Publishing Company, Amsterdam, the Netherlands, 1980.

[Lee 88] Lee, K.F.
*Large-Vocabulary Speaker-Independent Continuous Speech Recognition: The SPHINX System.*
PhD thesis, Computer Science Department, Carnegie Mellon University, April, 1988.

[Lee 89] Lee, K.F.
*Automatic Speech Recognition: The Development of the SPHINX System.*
Kluwer Academic Publishers, Boston, 1989.

[Ney 87] Ney, H., Mergel, D., Noll, A., Paeseler, A.
A Data-Driven Organization of the Dynamic Programming Beam Search for Continuous Speech Recognition.
In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 833-836. April, 1987.

[Schwartz 85] Schwartz, R., Chow, Y., Kimball, O., Roucos, S., Krasner, M., Makhoul, J.
Context-Dependent Modeling for Acoustic-Phonetic Recognition of Continuous Speech.
In *IEEE International Conference on Acoustics, Speech, and Signal Processing*. April, 1985.