# Two Uses for Syllables in a Speech Recognition System

Robert Arthur Blokland

Thesis submitted for the degree of Ph D

University of Edinburgh

1993

## Abstract

Many acoustic and phonetic studies attest to the value of the syllable as a unit of linguistic description. Phenomena of coarticulation and assimilation often occur within a syllable, and the acoustic realisation of some phonemes also correlates with their position in a syllable. For example, stops are more often released at the beginning of a syllable than at the end, and the clear allophone of /l/ is also more often found at the beginning than the dark allophone. Despite the prevalence of such studies, there have been relatively few attempts to apply this knowledge in a speech recognition system. This is what is attempted here.

The matter is investigated by modifying a speech recogniser that was built at the Centre for Speech Technology Research (CSTR. The recogniser is not based on syllables. It is a modular system, with a separate front end and back end. The front end uses hidden Markov models to produce a phoneme lattice, and the back end uses a dynamic programming algorithm to construct words out of the lattice. Syllable information is incorporated in both the front and the back end. Because the system is modular, the effects of incorporating syllables can be studied independently.

The experiments in the front end centre on the choice of a set of allophones that correlate well with their position in a syllable. Segmentation is then constrained to produce only those segment sequences that form valid syllables. The experiments in the back end centre on specialising the confusion matrix for syllable position. The confusion matrix contains statistics about the frequency with which the front end confuses one phoneme for another, and it is used to guide the back end in constructing words out of phonemes. The front end's confusions depend on the position of the phoneme in a syllable, and the experiments aim to increase the back end's intelligence by taking this into account.

The results, as we might expect, depend on the quantity of training data. Even with limited training data, however, there are promising indications that syllables have a role to play in future recognition systems.

# Declaration

This thesis was composed by me and describes original work which was executed by me.

# Acknowledgements

I thank my supervisor Steve Isard for his enthusiasm and encouragement, his patience during several barren years, and his forbearance in the penultimate year during a foolish and, I am happy to say, temporary change of topic.

I thank Professor Mervyn Jack, the Director of the Centre For Speech Technology Research, for making available office space and computers while I completed this thesis.

I thank Fergus McInnes, who provided the front-end suite of programs. The front end is comprehensive in scope, and conscientiously documented. I found it a joy to use. The occasional shortcoming Fergus was quick to fix. Most of all I appreciate his unfailing willingness to answer my questions.

I thank David McKelvie who provided the back-end suite of programs. David became actively interested in how I was hacking his system, and made many changes himself beyond the call of duty.

# Contents

# List of figures

# List of tables

## Abbreviations

| Abbreviation | Meaning | Introduced in |
|---|---|---|
| apu | acoustic-phonetic unit | chapter 3 |
| ATR | Advanced Telecommunications Research | |
| | (speech database for) | chapter 5 |
| CSTR | Centre for Speech Technology Research | |
| cyt | cytology | |
| | (speech database) | chapter 1 |
| epd | end-point difference | chapter 5 |
| HMM | hidden Markov model | |
| indel | insertion and deletion | chapter 6 |
| lexax | lexical access module | chapter 3 |
| MRPA | machine readable phonetic alphabet | appendix A |
| ms | millisecond(s) | |
| O-N-C | onset-nucleus-coda | |
| | (confusion matrices for) | chapter 6 |
| rp | received pronunciation | chapter 1 |
| VQ | vector quantisation | chapter 3 |

# Chapter 1

# Introduction

## 1.1 Introduction

This thesis addresses a problem which occurs when a speech recognition system chooses phonemes as its unit of recognition, which is that the realisation of phonemes are affected by neighbouring phonemes, to such an extent that it is sometimes difficult to recognise them. In the words *rim* and *trim*, for example, the sounds represented by the letter *r* are very different, at least for some accents of English. Such phonetic variation can be dealt with in various ways, and this thesis describes one of them, namely, the use of syllables as a predictor of these effects.

The rest of this chapter is organised as follows. The next section is a brief introduction to continuous speech recognition, to introduce the reader to the technical terminology used in the thesis. This is followed by a description of the problem of phonetic variation, with an indication of how the problem has been solved in the past. A short section describes how this thesis will tackle the problem (namely, the use of syllables), and gives justifications for why this approach was chosen.

## 1.2 Continuous speech recognisers

Continuous speech recognisers work on speech that is spoken more or less fluently. They contrast with isolated word recognisers, which can recognise words only when they are spoken one word at a time. The two kinds of system use different techniques for recognition. Isolated word recognisers often have stored patterns for the words that are going to be recognised; they are sometimes called whole-word recognisers. A continuous speech recogniser does not usually use whole-word recognition[1]. This is for a number of reasons. One is that in continuous speech there are no breaks between the words, and the system needs to discover where the words begin and end. The simple method of matching input against stored word patterns, which isolated word systems use, is therefore usually deemed unsuitable.

Another reason is that continuous speech recognisers usually have larger vocabularies. These consist of at least a few hundred words, and often several thousand. When the number of words is larger than a thousand or so, the patterns become too similar and it becomes hard to distinguish between them (Waibel, 1988). Another problem of large vocabularies is storage space. For example, the continuous speech recogniser Tangora, which is described in the next chapter, has a vocabulary of 20,000 words. The storage requirement for this number of word models is about one gigabyte (Lee, 1988, p84). A third problem is that it is difficult to find enough training data for vocabularies of this size.

A third reason why continuous speech recognisers don't use whole-word patterns is that the pronunciation of words is much more variable in continuous speech than in isolated words. Examples of this will be given in a later section.

For these reasons continuous speech recognisers use a unit of recognition that is smaller than the word. This unit is usually the *phoneme*. A phoneme, roughly speaking, are the sounds that make up a word, like the *k*, *a* and *t* sounds in the word *cat*. Instead of stored patterns for words, continuous speech systems

[1](Bridle & Brown, 1979) is one of the exceptions.

have stored patterns for phonemes. There is a heavy price to pay in using this smaller unit: it is very confusable. Lee cites Paul (Paul & Martin, 1988), who reports a tenfold increase in error rate when word models are replaced by phoneme models in an isolated word recogniser. However, some of the loss of recognition accuracy at this pattern-matching level can be made up at later stages of processing, as we will see.

The operation of a typical continuous speech recognition system is now described. As usual, phonemes are written between slashes; thus /k a t/ for the phonemes in *cat*. In this thesis phonemes are written in a *machine readable phonemic alphabet* or MRPA. Appendix A lists the 44 phonemes found in the British English accent of *received pronunciation* or RP, and their MRPA symbols. RP is the main accent with which this thesis is concerned.

The stored phoneme patterns of a continuous speech recogniser need to be trained. The training material is typically a set of sentences, which have been segmented into their constituent phonemes. The system is trained on many examples of each phoneme (thousands if possible, but not fewer than about ten), and the result, for RP, is a set of 44 patterns, one for each phoneme[2].

During recognition the system is presented with a phrase or a sentence. The system discovers where the phonemes begin and end by trial and error, by 'sliding' the patterns across the utterance, as follows. All the patterns are matched against the beginning of the utterance, and the best $n$ of these are kept; $n$ is usually an adjustable parameter. This yields $n$ possible places from which to match the second phoneme. All the patterns are now tried at each of the $n$ places, and the best $n$ of each are kept. There are then $n \times n$ places from which to start matching the third phoneme. If this were to continue until the end of the utterance, we would soon end up with an impractical number of possibilities. This is avoided by calculating cumulative scores, and keeping only the best $m$ paths at each stage. By using a dynamic programming algorithm, no paths that are better than the $m$ best ones are lost this way (dynamic programming is described in chapter 3).

[2]More complicated arrangements will be mentioned later.

The result at the end is a *phoneme lattice.* Figure 1.1 illustrates one for the sentence *The features suggest an acute inflammatory process.* The sentence is spanned by an unbroken chain of non-overlapping segments, each of which has several phoneme hypotheses. The figure shows only the top-scoring five hypotheses for each segment. There are three kinds of errors in the lattice. The sentence as spoken contains 36 segments. The lattice is spanned by 45 segments. This oversegmentation is a typical problem, and the extra segments are called insertion errors. A second kind of error is deletion errors, caused by missing segments. The third kind are substitution errors, which arise when the phoneme hypotheses of a segment do not match the transcription. All these errors need to be repaired before the phonemes can be assembled into words. This is usually done by the *lexical module*, also called lexical access or lexical lookup.

The part of the recogniser described so far, which digitises the speech, encodes it, and recognises the phonemes, is called the *front end.* Lexical access is the first stage of the rest of the system, called the *back end.* The task of lexical access is to form words out of the hypotheses in the phoneme lattice. It uses a lexicon to do this. The system can only recognise the words that are in this lexicon, and its size is anything from several hundred to many thousands of words. The lexicon relates phoneme strings to words. Under the head /k a t/ is found the corresponding word *cat*, under /k ei l/ is found *kale*, and under the head /dh e@/ are found the homonyms *their* and *there.*

Lexical access repairs the three kinds of errors mentioned — substitution, insertion and deletion errors — while it is matching the phonemes against the various head entries. Table 1.1 shows how the lattice of figure 1.1 is turned into the sentence *The features suggest an acute inflammatory process.* A substitution error is corrected by substituting it with the correct phoneme. In the table even correct phonemes are shown as obtained by substitution; these are identity substitutions. Deletion errors are repaired by inserting the required segment, and insertion errors are repaired by deleting the offending segment. In the table *ins* and *del* refer to the repair and not to the error: an *ins* repairs a

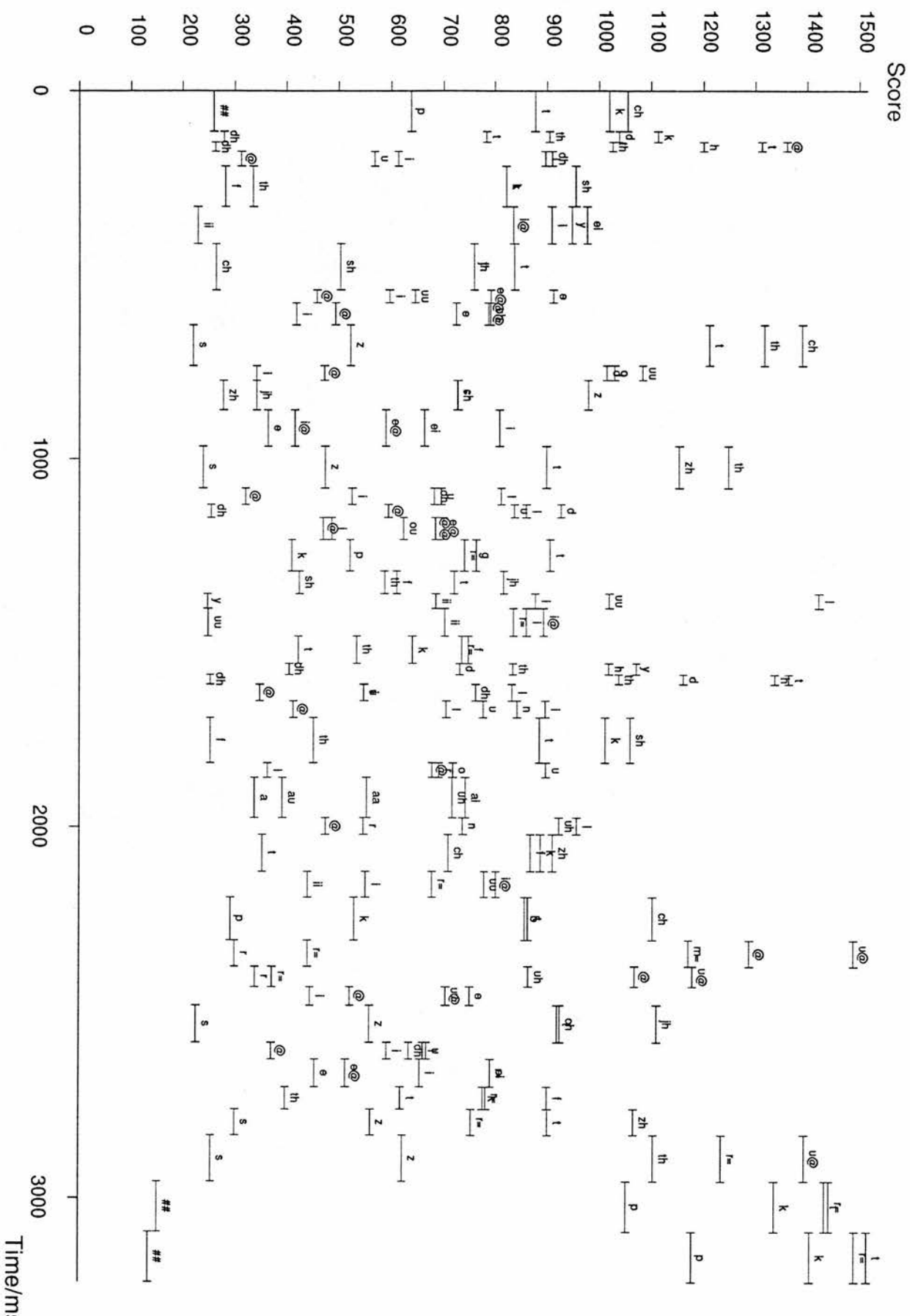



Figure 1.1: Phoneme lattice for the sentence *The features suggest an acute inflammatory process.* Phoneme hypotheses are shown to a depth of five. Those with a better score are nearer the bottom of the diagram.

**Phoneme lattice:**

```
## dh dh @ f ii ch @ [i @] s [i ...uh] [zh jh] e s @ [dh ...n]
@ k sh y uu t dh dh [@ i] [@ i u n] f l a @
t [ii i] p r r [i ...ou] s @ e th s s ## ##
```

**Operations:**

| | | |
|---|---|---|
| 1 | the | *##.del* dh.dh dh.*del* @.@ |
| 2 | features | f.f ii.ii ch.ch @.*del* @.@ z.*ins* |
| 3 | suggest | s.s uh.@ jh.jh e.e s.s t.*ins* |
| 4 | an | @.@ n.n |
| 5 | acute | @.@ k.k sh.*del* y.y uu.uu t.t |
| 6 | inflammatory | dh.*del* dh.*del* i.i n.n f.f l.l a.a m.*ins* @.@ t.t r.*ins* i.i |
| 7 | process | p.p r.r r.*del* ou.ou s.s @.*del* e.e th.*del* s.*del* s.s *##.del* *##.del* |

Table 1.1: Substitutions, insertions and deletions performed by lexical access to turn the phoneme lattice of figure 1.1 into the sentence *The features suggest an acute inflammatory process.* The phonemes in square brackets are alternatives for the same segment, with the best-scoring ones first. Alternatives are shown only where they are used in the operations.

missing segment and a *del* removes an unwanted segment. Twelve deletions, four insertions, and 29 substitutions (most of which are identity substitutions) are performed.

The result of lexical access is a *word lattice*, which, like a phoneme lattice, has multiple paths through it. The word lattice can be cleaned up in a *syntax module*, which tries to construct grammatical phrases. This is done on the basis of a grammar, which determines which sequences of words are grammatical and which not. It is common to provide the system with a grammar of a subset of the language, one that is specific to the domain of application. An example of such a domain is that of a travel agent. A grammar for this domain would allow sentences like 'How much is the fare to London' or 'I want to fly to Nepal', but not sentences like 'Wheat production has once again exceeded the targets of the Five Year Plan'. Domain specific grammars like this are the main reason why continuous speech recognition systems work at all: 'System performances were

found to be more closely associated with the branching factor[3] of the language than with any other system variable.' (Lea & Shoup, 1979, p 21).

As described here a continuous speech recogniser falls in two halves, the front end and the back end. The front end generates phoneme hypotheses, and the back end constructs words and phrases out of them. It is a *modular* system. A modular speech recognition system is one in which some or all of the different stages are run in succession. An earlier stage produces its output for the next stage to use as input, and the later stages do not affect the processing of the earlier stages. Most speech recognition systems are not modular, but *integrated.* In an integrated system the later stages interact with the earlier stages in such a way as to affect their processing. For example, it is common to couple the phoneme recogniser with the lexical module in such a way that the phoneme recogniser is constrained to produce only those phoneme sequences that form words. In a modular system the phoneme recogniser works in an unconstrained way, and initially produces many more phoneme hypotheses, which are then later weeded out by the lexical module.

We shall return to the matter of modular and integrated systems in chapter 3.

## 1.3 Acoustic-phonetic variation

When a word is spoken on different occasions, the signal is different each time. Some of these differences are due to the differences between speakers, or, with a single speaker, due to such factors as health, mood, and background noise. These factors are the source of non-linguistic variations. They arise from causes that lie outside the utterance. Linguistic variations are predictable from the words that were spoken. They are usually classified as acoustic, phonetic, phonemic and phonological, according to their origin. I shall simply call them *acoustic-phonetic*, to indicate that sometimes the differences are perceptible and

[3][My footnote] Branching factor is closely related to *perplexity*, which is defined in appendix B. The quotation comes from the authors' review of the ARPA SUR systems, which are surveyed in chapter 2.

sometimes not.

Acoustic-phonetic variations include the following phenomena.

### Word reduction

Consider the following pronunciations of the word *America.*

/a m e r i k @/
/@ m e r i k @/

In the second pronunciation the first vowel has been reduced to *schwa.* Further reductions are possible, e.g to

/@ m e r @ k @/

and even to

/@ m e r k @/

where one of the phonemes is deleted entirely. Other examples of phoneme deletion are /s p oo t/ for *support* and /l aa s n ai t/ for *last night.*

### Coarticulation

In the words *seen* and *soon*, the quality of the /s/ is different. In the second case it is spoken with the lips rounded, which are in this position in anticipation of the following vowel. This makes a difference to the acoustics. Another example is *kit* and *cot*, for which the /k/ is coloured differently by the following vowel; broadly speaking, the mouth is more open during the pronunciation of *cot* than *kit.* Coarticulation occurs when the gestures used to articulate one phoneme persist in the articulation of the following phoneme, or when the postures for a following phoneme are adopted in advance during the articulation of its predecessor. It is a widespread phenomenon, and affects both vowels and consonants.

### Assimilation

Consider the following two ways of pronouncing the word *actually.*

/a k t y uu l i/
/a k ch uu l i/

In the second case the adjacent phonemes /t/ and /y/ have become assimilated into a single phoneme /ch/. Another example is *newspaper*, which becomes /n y uu s p ei p @/: the /z/ at the end of the word *news* has become devoiced because the following phoneme /p/ is voiceless. Assimilation can occur also across word boundaries, as when *did you* becomes /d i jh uu/, *have to* becomes /h a f t uu/, and *ten minutes* becomes /t e m i n i t s/.

Some of the articulation processes mentioned can occur in combination. When *bandwidth* becomes /b a m w i d th/ for example, the /d/ is deleted and the /n/ is assimilated to the following /w/ to become /m/. Another example is *handbag*, which after deletion and assimilation becomes /h a m b a g/.

## 1.4 The syllable as conditioning environment

The examples given above give the impression that the acoustic-phonetic variations that a phoneme undergoes are caused by the phoneme's immediate neighbours. While the neighbours are a large influence on the form the phoneme takes, they are not the only influence. The realisation of a phoneme also depends on the position of the phoneme in the syllable; see, for instance Gimson (Gimson, 1980). Syllables will be discussed in detail later (see 4); for the moment let us agree that syllables are parts of words, as in the examples that follow. There are two sylables in a word like *de-tail*, three in a word like *syl-la-ble*, and four in a word like *un-re-qui-ted.* Syllables also influence the form of a phoneme. It is well known, for example, that the clear allophone of the phoneme /l/ occurs more often at the beginning of a syllable, while the dark allophone occurs more often at the end.

Another example is the stop allophones. Stops at the beginning of a syllable tend to be released, while those at the end are released less frequently. Table 1.2

| | Released | | Unreleased | | Total |
|---|---|---|---|---|---|
| | No | % | No | % | No |
| Initial clusters | 958 | 69.3 | 49 | 17.6 | 1007 |
| Final clusters | 425 | 30.7 | 230 | 82.4 | 655 |
| Total | 1383 | 100 | 279 | 100 | 1662 |

Table 1.2: Statistics concerning stops in the ATR database.

shows this effect. 200 sentences[4] at CSTR were syllabified, and the distribution of released and unreleased stops was calculated. We see that a released stop is more than twice as likely to come from a syllable-initial cluster than a syllable-final one, and that an unreleased stop is more than four times as likely to be in the syllable-final cluster than in the syllable-initial cluster.

There have been many studies that show that English speech exhibits statistical regularities which can be conveniently stated in terms of syllables. Examples are (Rakerd *et al.*, 1987), (Campbell, 1988), and (Randolph, 1989).

There have also been many studies to show that humans use information like stop releases to decide how a word is syllabified, such as (Lehiste, 1960), (Christie, 1974) and (Nakatani & Dukes, 1977). Christie showed that the presence of aspiration in the /t/ of the nonsense word /asta/ led to the perception of /as-ta/, while its absence led to the perception of /a-sta/. The period of silence in the /t/ also had an effect, but a smaller one. As the silence became longer, there was a greater likelihood of perceiving /as-ta/. Nakatani and Dukes used diphones, and discovered that glottalisation of vowels is a strong cue for a boundary before them, and, like Christie, that aspiration in a stop indicates a boundary before it.

Work on syllables in the field of speech technology seems to have been of two kinds. One of them detects, or otherwise segments the speech into, syllables as a preparation for more robust phoneme recognition (e.g (Mermelstein, 1975), (Nakagawa & Jilan, 1986), (Mertens, 1987), and (Green *et al.*, 1990)). The other kind of work studies the properties of lexicons and the distribution of allophones in them, to show that syllables are a useful unit for speech recognisers (e.g (Church, 1983), (Waibel, 1988) and (Randolph, 1989)). We shall look at

[4]The ATR database, to be introduced in section 5.2.

this work in chapter 4.

This thesis is a development of the above work, particularly Church and Randolph. The aim is to exploit the conditioning effect of syllables in a speech recognition system. In particular, it aims to show that statistically based speech recognition will benefit from the use of an explicit syllable level of modelling between phonemes and whole words. Appeal will be made to two kinds of regularities.

**Allophone distribution** Many systems (e.g (Lee, 1988; McInnes *et al.*, 1990)) improve their bottom-up phoneme recognition by creating different models for the acoustically distinguishable allophones of a phoneme, such as clear and dark /l/. The distributions of some allophones correlate well with their position in a syllable. Many speech recognition systems (although not all; see for example (Russell *et al.*, 1990a)) ignore this information. When they recognise a particular allophone, it is taken as evidence only for the presence of the corresponding phoneme. We will make use of the distributional evidence also.

**Phoneme confusions** It is common for speakers to pronounce phonemes indistinctly or to omit them altogether. Usually this phenomenon is modelled by considering only the left and right neighbours of a phoneme. However, the position of the phoneme in the syllable is relevant as well. For example, phoneme deletion is more common at the end of a syllable than at the beginning. Some systems (e.g (Lee, 1988)) take account of this fact, but only at the end of words. We will do so at word-internal syllable-boundaries as well.

We will be testing these ideas on the speech recogniser at CSTR (McInnes *et al.*, 1991), which is based on hidden Markov models. The benefits of the proposed work will be looked for in two places: in improved scores in the phoneme lattice and in improved word scores at the lexical level.

## 1.5 Concluding remarks

We know that the realisation of phonemes in continuous speech varies greatly. Some of the variation has been described as phenomena of reduction, coarticulation and assimilation. The effects of these phenomena may be called acoustic-phonetic variations. Recognisers of continuous speech need to take these variations into account. Many of the variations can be conveniently described by reference to the syllable. The thesis will take account of the variation in syllable terms. There will be two points of focus: the use of syllables in the front end, and introducing syllables as an extra level of organisation in the back end.

In the next chapter a survey of continuous speech recognisers is given, which outlines the development of the field from attention to linguistic units like syllables in the early days, through their virtual eclipse in more recent times, to a return to more linguistically motivated work today.

# Chapter 2

# Some Trends in Speech Recognition

## 2.1 Introduction

There has been a great change in the way speech recognisers are built, from the early decoding systems of the fifties, through the artificial intelligence influenced years of ARPA SUR, to the hidden Markov systems of today. This chapter picks out some of the landmarks.

## 2.2 ARPA speech understanding research

The Advanced Research Projects Agency of the US Department of Defense initiated a five-year research effort into speech understanding in 1971. The aim of the research was to build systems that would 'accept connected speech from many cooperative speakers, in a quiet room, using a good microphone, with slight tuning for new speakers, accepting 1000 words using an artificial syntax in a constraining task, yielding less than 10% semantic error' (Klatt, 1977). This initiative resulted in four systems, which are collectively known as ARPA SUR. The following overview is based on (Reddy, 1975), (Klatt, 1977), (Lowerre & Reddy, 1980) and (Lea & Shoup, 1979).

The four systems were concerned with speech *understanding*, as opposed

to the speech *recognition* task with which this thesis is concerned. Speech understanding has a slightly different focus from speech recognition. In speech understanding the aim is to produce an appropriate response to the spoken input. In speech recognition the aim is to produce a faithful transcription of the input. In understanding a complete transcription is not always necessary. Suppose the application is a telephone-ordering service, and the caller says "I'd like to order three bath towels in midnight blue." The important words in that sentence are ... *three bath towels* ... *midnight blue.* The order will be filled correctly even if the other words *I'd like to order* ... *in* ... are garbled or missing. In speech recognition, on the other hand, the correct transcription of every word is wanted. This makes recognition a harder task than understanding, in some respects. In practice, however, there is little difference in the techniques used in the different kinds of systems, and the material below on the ARPA SUR systems is fully applicable also to speech recognition.

The four ARPA systems were demonstrated in September 1976. Only two were worked on for the full five years: Hearsay-II, developed at Carnegie-Mellon University (CMU), and HWIM, developed at Bolt, Beranek and Newman (BBN). A third system, Harpy, was developed as part of a PhD thesis at CMU in the record space of a year. A fourth project at the Stanford Research Institute and System Development Corporation (SDC) was not completed for non-technical reasons.

The three successful projects are famous, and although they were aimed at speech understanding, almost every system for continuous speech recognition since then has taken them as a starting point.

All systems had an initial segmentation into broad manner of articulation classes, followed by refined categorisation (Lea & Shoup, 1979, p 78).

### 2.2.1 Harpy

Harpy is claimed to be the first demonstrated continuous speech understanding system with a vocabulary of over 1000 words (Lowerre, 1976, p 332). It is the only system which met the ARPA design goals. The application for which it was

developed is document retrieval. An example sentence is "How many articles on psychology are there?". It performed successfully mainly because of the small set of sentences it could recognise, and not for any interesting reason of system design.

The approach is essentially incremental template matching for whole sentences. The system contains a network of states in which the whole repertoire of sentences it can recognise is encoded. A state corresponds to a 'segment', which is a short passage of acoustically similar material, according to some measure of similarity. Segments correspond roughly to allophones, and there were 96 of them. The segments are encoded as autocorrelation values and LPC coefficients. Each state contains an idealised template for a segment, and also bears phonetic, lexical and durational information. The network also contains word-juncture information. A state is linked to several following states. An unknown utterance is processed by matching its segments against states on a path, starting at the first state. The matching is done using the Itakura distance measure (Saito & Itakura, 1966). Network traversal is constrained by beam search.

The vocabulary was 1011 words, and the syntax was restricted English with a low number of function words (Klatt, 1977, p 118). The number of hypotheses per input word was 33.

The system recognised 42% of phonetic segments in top choice, and 65% in top three. This is the worst performance of all the systems at this level. Even the SDC system recognised phonetic segments correctly 50% of the time. At the sentence level, of course, Harpy performed best. For five speakers, male and female it achieved 90% sentence accuracy, and 94.3% word accuracy.

The success of Harpy encouraged speech work using dynamic time warping and hidden Markov models, which have dominated practical system building, including the CSTR system that is used in this thesis.

### 2.2.2 Hearsay-II

Hearsay-II was developed from the earlier Hearsay-I, for the same application as Harpy. It is the darling of the ARPA systems in the Artificial Intelligence community, because of its blackboard model, which has found widespread discussion (if not use) there. Hearsay was written in the SAIL language, which was used for several AI systems. The blackboard is a central repository of information, around which cluster parts of the program called *knowledge sources* (KSs). The operation of the model is well described by Reddy, (1975).

> The blackboard model conceives of each KS as an information gathering and dispensing process. When a KS generates a hypothesis about the utterance that might be useful for others, it broadcasts the hypothesis by writing it on the 'blackboard' - a structurally uniform global data base. The hypothesis-and-test paradigm ...serves as the basic medium of communication among KSs. The way KSs communicate and cooperate with each other is to validate or reject each other's hypotheses. The KSs are treated uniformly by the system and are independent (i.e anonymous to each other) and therefore relatively easy to modify and replace. The activation of a KS is data driven, based on the occurrence of patterns on the blackboard which match the templates specified by the KS. ...The blackboard consists of a uniform multilevel network ...and permits generation and linkage of alternative hypotheses at all levels. A higher level KS can generate hypotheses at a lower level and vice versa. It is not necessary for the acoustic processing to be bottom-up and the language model to be top-down.

The interaction between knowledge sources can be uncontrolled or controlled, and if controlled, then controlled according to various schemes. After initial attempts at uncontrolled interaction, Hearsay settled for a tightly controlled one. The blackboard had speech represented at various levels. At the lowest level the signal was represented as LPC coefficients. Another level was

a broad or midclass syllable type, such as 'STOP FRONT-VOWEL FRICATIVE' for the syllable *bif*. The syllables were obtained from the signal by template matching, using the Itakura distance. Sentences were predicted by a syntactic knowledge source, and verified by a lower level knowledge source.

The vocabulary was 1200 words, and the syntax restricted English. The number of hypotheses per input word was 46. Hearsay understood 74 out of 100 sentences in the standard ARPA test.

The uncontrolled nature of the interaction between Hearsay's different components (the knowledge sources) have often been quoted as its lack of success. See, for example, (Thompson, 1984).

### 2.2.3 HWIM (Hear what I mean)

Poorest performing of the big three ARPA systems, it was developed to demonstrate a 'travel budget management assistant', which through dialogue allows the user to plan trips. Example sentence: "What is the plane fare to Ottowa?" The response consisted of synthesised speech. This is a pioneering example of the use of both speech input and output in one system.

The signal was represented as frame vectors, which encoded formant information and other acoustic features. These were constructed into 'phonetic units', of which there were 71 different kinds. The correct phonetic unit was recognised 52% of the time, and 80% of the time in the top three. Words were originated either bottom-up or top-down. In the bottom-up case, available phonetic units trigger the construction of seed words. They are verified by synthesising the word out of signal parameters, and comparing against the input pattern. In the top-down case words were predicted from the syntax of expected sentences. These top-down hypotheses are verified through analysis-by-synthesis as before.

The vocabulary was 1000 words, stored as a 'lexical decoding network', a tree structure. The words are spelled in phonetic units, with several versions for continuous speech phonology. The syntax was English-like, and stored as an augmented transition network (ATN). The system generated 196 hypotheses

per input word, and recognised 44 out of 100 sentences in the standard ARPA test.

### 2.2.4 Assessment of ARPA SUR

Lea and Shoup sum up the contribution of these systems as follows (1979, p 21).

> A primary contribution of the ARPA SUR project was its simplifying of the recognition task by constraining it markedly via syntactic, semantic, and task constraints. This is comparable to lexical constraints in a small-vocabulary isolated-word recogniser. Because continuous speech recognition is such a multiple-dimension problem, it can be constrained in many ways, and one question addressed by ARPA SUR work concerned which constraints were most effective in improving performance . System performances were found to be more closely associated with the branching factor of the language than with any other system variable. Harpy thus cannot be unequivocably appraised as 'winner', since its task was (on one measure) almost an order of magnitude easier than that undertaken by HWIM.

## 2.3 CSTR's RM2 system

The material in this section is based on personal knowledge.

RM2 was a continuous speech recognition system developed at Edinburgh University's then new Centre for Speech Technology Research, between 1985 and 1988. The approach was knowledge-based (there was almost no statistical training), and there was a uniform data structure processed by a chart parser. In a deliberate departure from the opportunistic control structure offered by a blackboard system, RM2's control flows in only one direction, from bottom to top.

The signal was divided into 5ms frames, and processed to produce acoustic parameters like formant values, energy values in frequency bands, voicing, etc.

Acoustic features were then found by threshold and similar tests on the signal parameters. The acoustic features were parsed into phonemes, under the control of rules like 'stop = silence followed by burst release'. The rules constituted a phoneme grammar. The grammar, stored as a set of rewrite rules, defined phonemes as strings of feature groups. Available phonemes triggered a search of a lexicon, matching phonemes left to right. Available words triggered the construction of word strings, under the control of a grammar that gave the probabilities of one word class's following another. A beam search found the best n strings, usually a screenful.

The vocabulary consisted of 5000 words, spelled phonemically and stored in a tree structure. The lexicon included reduced forms like /s l i s t @/ for *solicitor*. Interword assimilations for continuous speech could be produced on demand, and such forms greatly expanded the effective size of the lexicon.

### 2.3.1 Discussion

Despite the ten years that lay between them, RM2 did not perform as well as the ARPA SUR systems. The HEARSAY and RM2 systems were consciously based on the techniques of Artificial Intelligence (AI). Speech recognition was an active topic for research in AI then, mainly due to ARPA work, and a few famous AI techniques originated from there, like blackboards. The goal of AI is to make the computer exhibit intelligent behaviour. Recognising speech may well qualify as intelligent behaviour, but work in this area based on that approach, like HEARSAY and RM2 has been discouraging.

These days the major projects in speech recognition look less to AI than to statistical methods. Two such projects will be described below, TANGORA (Averbuch *et al.*, 1987) and SPHINX (Lee, 1988).

## 2.4 Tangora

Tangora is a class of recognition systems developed over a number of years at IBM Research at Yorktown Heights in New York State.

> The systems are named for Albert Tangora, listed by the Guinness Book of World Records as the fastest keyboard typist, who could sustain rates of 147 words per minute for one hour (Averbuch *et al.*, 1987, p 701).

The aim of the project is to develop a speech workstation at which documents can be created by voice. The task domain is business correspondence. Several versions of the system have been produced. The systems run on IBM PC machines with extra speech processing cards. The systems recognise isolated words, with degraded performance for continuous speech.

The approach taken is expressly non-linguistic, in favour of a statistical approach. Whereas systems like Hearsay incorporated the explicit expertise of linguists like phoneticians and phonologists, the IBM team took the attitude that linguistic information of the kind provided by experts is best obtained by training. An idea of this attitude can be obtained from the following quotation, which concerns a precursor of Tangora. It is from (Lea & Shoup, 1979, pp 30–1).

> Originally the acoustic processor performed phonemic classification of individual spectra and then the speech was segmented into phonemes for the final output. More recently, the phone segmentation and labelling was eliminated; instead, a sequence of centisecond labels from a 33-phone alphabet is outputted to the linguistic decoder. The advantages given for the centisecond-level models are that information related to phone length is made available in a form usable by the models in the linguistic decoder, that more of the important information is preserved, and that the segmentation and labelling decisions are delayed until decisions can be made by the linguistic decoder.

A phonetician would prefer to see a segmentation into phonemes, because that is the raw material of his science. For the IBM team, however, this smacks of vagueness, and if they can do without them, so much the better. Linguistic regularities should not be incorporated explicitly, but acquired through training.

The system works by first performing a fast acoustic match between the signal representation and the lexicon. This yields a list of candidates which are then subjected to more detailed matching. The linguistic decoder includes a statistical language model of the domain. Tangora-5, a version with a vocabulary of 5000 words, covered 92.5% of the task domain, i.e 92.5% of the words found in business correspondence are among the 5000 words in the lexicon. Tangora-20, with 20,000 words, covered 97.6% of the domain. The system is speaker dependent, and new speakers need to go through a period of training when they enrol. It is not necessary to train all 20,000 words. Training on 700 distinct words is enough for the Tangora-20 system. The error rates of Tangora-5 and Tangora-20 were 2.9% and 5.4% respectively, on isolated words.

## 2.5 Allerhand

An intriguing system that combines statistical and knowledge-based schemes was described by Allerhand (1987).

As we've seen in section 2.2.2, a knowledge-based system is one which recognises speech in stages, where each stage draws on a different 'domain of constraint', such as the acoustic-phonetic domain, the phonological domain, and the syntactic domain. Each stage, or component, produces data which it hands on to the next component. Thus the acoustic-phonetic component takes the digitised signal and passes an acoustic-phonetic description of it to the phonological component. The phonological component turns this into a phonological representation, and so on, until the last component produces a string of words. For 'domains of constraint' read 'fields of knowledge', whence the term 'knowledge-based'. We could also call this the divide and conquer strategy.

Allerhand outlines the problems of this approach as follows (1987, p10).

> Each domain contributes partial evidence, which leads to the generation of hypotheses, and every set of hypotheses is of course larger than it need be. In combination, the constraint domains generally compound the number of plausible hypotheses, leading to the char-

> acteristic combinatorial explosion. Under this effect the number of plausible hypotheses races ahead of the constraints which can be usefully applied to contain them, so that the search problem grows dynamically, producing excessive demands on execution time and memory space.

Allerhand's system was produced for his PhD thesis, and so only addresses part of the speech recognition problem, that of turning the signal into a sequence of broad class phonemes. There are nine broad classes, which contain a classification of the phonemes of the language. The problem is divided into two stages: classifying the time-domain vectors of the signal into acoustic symbols, and parsing the acoustic symbols into broad classes. The first uses vector-space pattern recognition and the second uses syntacting pattern recognition. The first is based on statistically trained, quantitative information, and the second is based on structural information obtained from observation. These are the two parts of Allerhand's hybrid system.

> The ideas presented ...are a tentative exploration of the ways in which a simple grammatical representation of the structural constraints inherent in speech and language can be used to improve the descriptive adequacy and the performance of pattern-matching speech recognition models. (Allerhand, 1987, p15).

## 2.6 SPHINX

At the time of its appearance SPHINX (Lee, 1988) was greatly superior in performance to any other system. It was developed by Kai-Fu Lee as part of his PhD at Carnegie-Mellon University, and the work was funded by DARPA, the successor to ARPA. As with ARPA, DARPA funded rival projects, and SPHINX was one of them. SPHINX is a speaker-independent system, and its results were better than the speaker-dependent results of some rival systems (Lee, 1988, p16). In the same place Lee goes on to say

> By utilising perceptual knowledge and stochastic modelling, by integrating knowledge and learning, and by fully utilising abundant training, the SPHINX Speech Recognition System has bridged the gap between speaker-dependent and speaker-independent systems.

The success of SPHINX is due to a number of factors, many of which became possible because Lee had a large amount of training data available to him. Table 2.1 lists some of the factors and the difference they made to the performance of the system[1].The first two of these factors concerns vector quantisation (VQ) of the speech signal. Vector quantisation is a way of encoding the digitised speech, and involves dividing the signal into fixed-length frames (20ms in the case of SPHINX), and translating the speech in each of the 20ms intervals by one of a fixed number of codes. The number of codes is usually 256, as it is here, and they are looked up in a *codebook* that has been trained in advance. The many-to-one mapping defined by the codebook introduces distortion, which can be reduced by using more than one codebook (Gupta *et al.*, 1987). Using three codebooks instead of one means each frame is encoded with three VQ symbols instead of one.

The next factor is the duration modelling of words. SPHINX models words and phonemes in a three-level HMM network (Hidden Markov models, or HMMs, are described in D). At the highest level the network consists of word models, arranged according to the grammar. The word models consist of their phonetic pronunciations, and each of these is a phone model in the usual way. While the phone models embody duration information for the phonemes themselves, when they are put together in a word model, they do not reflect the length of the word very well. Lee introduced a separate mechanism for modelling word durations, and obtained the results given in the table.

The next four factors in the table relate to the training of models. SPHINX is first trained on context-independent phones, i.e on phones independent of their setting. Next context-dependent models are trained on the basis of these. There

[1]The table refers to a later version of the system (Lee, 1989), and not the one developed for his PhD thesis.

are four kinds of context. The first kind of context is provided by function words, abbreviated *fn-word* in the table. As described in chapter 1, function words are greatly distorted in continuous speech. Because function words occurred frequently enough in his training data, Lee was able to train special models for phones that came only from these (a set of 42 functions words gave rise to 105 phones, and each of these was given a separate model (Lee, 1989, p150)). This specificity was next extended to *function phrases*. Lee observed that the phones in phrases like *is the*, *that are* and *of the* are even more distorted than in function words by themselves.

The third kind of context dependency is reflected in the *generalised triphone*. A triphone is a phoneme whose identity is determined not only by itself, but by its left and right neighbours. Thus *pro* and *tra* define two triphones for the phoneme /r/; a 45-phoneme set like SPHINX's gives rise to $45^3 = 91125$ triphones. Not all of these are legal for English, but even the number that remains is too large to be trained: it would be difficult to find enough training data for all of them. Lee's solution to this is the *generalised* triphone, which in effect is a class of triphones with similar contexts. Thus *pro* and *tra* might be collapsed into a single class, because in both cases the /r/ has a stop consonant on its left and a vowel on its right. A fourth kind of context-dependency was taken into account when the triphones were defined also across word boundaries.

The last factor in the table, *corrective training*, refers to the optimisation variable in the training of the hidden Markov models. The corrective training algorithm is an alternative to the usual forward-backward training algorithm. Both algorithms train models incrementally in an iterative procedure. During the iteration the forward-backward algorithm maximises the probability that the models will generate the training data, while the corrective training algorithm tries to maximise the recognition rate on the training data. Lee's implementation of it is an extension of the one described in (Bahl *et al.*, 1988).

As can be seen from the table, the greatest improvement in the word accuracy (nearly 20%) came from using three codebooks instead of one. It is SPHINX's 20ms frame size that makes multiple codebooks necessary. The CSTR

| Version | No grammar | Word pair |
|---|---|---|
| 1 codebook | 25.8% | 58.1% |
| 3 codebooks | 45.3% | 84.4% |
| +Duration | 49.6% | 83.8% |
| +Fn-word | 57.0% | 87.9% |
| +Fn-phrase | 59.2% | 88.4% |
| +Gen-triphone | 72.8% | 94.2% |
| +Between-word | 77.9% | 95.5% |
| +Corrective | 81.9% | 96.2% |

Table 2.1: Word accuracies of various versions of SPHINX. Reproduced from (Lee, 1989, p152). Word accuracy is the percentage of words correct, not counting insertions.

system, described in the next chapter, has a frame size four times smaller (5ms), and so suffers less VQ distortion. In fact, with only one codebook it has more VQ symbols per 20ms than SPHINX with multiple codebooks (4 vs 3).

The next largest improvement (more than 13%) comes from generalised triphones. Triphone models capture most of the assimilation effects mentioned in chapter 1. Lee was not the first the introduce triphone modelling; he cites (Bahl *et al.*, 1980) as the original proposal. Triphones lead to a large number of poorly trained models, and this is overcome by interpolating them with context-independent models. Lee's contribution was to note that many left and right contexts were sufficiently similar that they could be merged. He gives the example of the labial stops /b/ and /p/, which have similar effects on the following vowel (Lee, 1988, p88). The triphones with shared contexts were called generalised triphones, and the immediate benefit is in the larger number of training examples that become available. The extent to which triphones need to be collapsed to produce generalised triphones depends on the quantity of training data. One version of SPHINX had 500 generalised triphones (Lee, 1988, p96). We will see in the next chapter that generalised triphones are not a possibility in the CSTR recognition system.

We may note in passing two interesting points from table 2.1. One is that the use of a grammar makes a very big difference to recognition accuracy. Whereas the greatest improvement in word accuracy came from using three codebooks instead of one, as I've already drawn attention to, adding a grammar to the

one-codebook case makes a much bigger difference: more than 32% compared to nearly 20%. This fact has been noted widely (a reference to it was made in section 1.2), and we will meet it again in chapter 6. The grammar makes such a big difference that gains obtained without it become much smaller after it is added. This brings us to the other interesting point. Using a grammar can partly reverse improvements made without it. The addition of word duration modelling improved the word accuracy by about 4% when no grammar is used, but the accuracy went down slightly with the use of a grammar. We will see effects like this also in the CSTR system, in chapter 6.

## 2.7 Summary and discussion

The concern of artificial intelligence is to implement on a computer tasks for which no algorithm exists, or for which an existing algorithm is computationally impractical[2]. One way of achieving this is to represent the knowledge of experts at the task, and, usually, to run some kind of rules on the resulting information structures.

This kind of thinking was a heavy influence on the builders of the ARPA SUR systems. The task was speech recognition, the experts were phoneticians and phonologists, and the rules were grammatical rules which expressed such facts as conditions under which assimilation takes place. The schemes for this were not always implemented, and in the end the hopes were not realised. The best performing of the ARPA systems, Harpy, owed little to knowledge engineering and much to the rigorous control over the number of possibilities that could be generated by the acoustic matcher.

A swing away from knowledge engineering then took place, in favour of systems that relied on the training of statistical models. The various Tangora systems are the best example of this. These systems work better than early

[2]Every computer program embodies an algorithm, but sometimes only in the trivial sense of producing its output from its input. However, not every program is an algorithm *for the problem it is supposed to solve.* Chess-playing programs may be mentioned as an example. Such programs do not have an algorithm for the problem 'win every game'. We know this because such programs occasionally lose. These programs are an example of the case where an algorithm exists but is (currently) infeasible to run.

rule based efforts like Hearsay, HWIM and KEAL (Vaissière, 1989), even though from a theoretical linguistic point of view they are less sophisticated. The false conclusion has then often been drawn that statistics are a preferable alternative to linguistic theory in the design of speech recognisers. However, statistics and linguistic theory need not be mutually exclusive. The progression from the use of whole words as the basic unit of recognition in early statistical systems, to context-dependent allophones in systems such as SPHINX, would seem to vindicate the use of traditional linguistic categories where sufficient data can be gathered to characterise them statistically.

SPHINX's triphones can be seen as a response to the facts of assimilation. The use of syllables in the work proposed here is a response to the fact that acoustic-phonetic variation occurs on a scale larger than triphones. We have seen some encouraging figures that this approach is justified. At the end of the last chapter we saw that a released stop is more than twice as likely to be in the initial cluster of a syllable as in the final cluster. Such observations have of course often been made, but they have not often found effective expression in speech recognisers.

One person who strove to do so is Kenneth Church. It is appropriate here only to quote his proposals, in order to show the way forward. His work will be discussed in more detail in chapter 4. Church's PhD thesis advocated *using* allophonic information rather than ignoring it (Church, 1983). Previously, allophonic processes were regarded as obscuring the signal, and the resulting allophonic variations as a nuisance. In evidence Church quotes Klatt (Klatt, 1979):

> In most systems for sentence recognition, such modifications [the different realisations of a phoneme in different contexts] must be viewed as a kind of 'noise' that makes it more difficult to hypothesise lexical candidates given an input phonetic transcription.

Church argues, on the other hand, that allophonic information is useful:

> ...allophonic constraints provide an important source of *information* which should be exploited to the fullest possible extent by a

> speech recognition device. Despite the fact that allophonic variation can occasionally obscure certain cues, allophonic variation should not be viewed as a source of random noise. Allophonic variation is the result of very predictable processes. These processes provide important cues for the determination of syllable boundaries and stress assignment. This information will (often) compensate for whatever segmental cues may be occasionally obscured. (1983, p 16)

We will see in subsequent chapters how this information can be made use of.

# Chapter 3

# The CSTR Recognition System

## 3.1 Introduction

The syllable experiments were performed on a modified version of the CSTR speech recognition system. Describing that system is the purpose of this chapter. The system was produced at the Centre for Speech Technology Research (CSTR) in Edinburgh. It was developed at CSTR as part of the UK's Alvey Information Technology Initiative. The Alvey Initiative funded several large demonstrator projects, of which the CSTR system was one. The system was built mainly by Yasuo Ariki, Fergus McInnes, David McKelvie and Steve Hiller, with contributions also from others. The description below is based on personal experience, conversations with the above, and on (McInnes *et al.*, 1991).

The CSTR recognition system consists of an acoustic-phonetic *front end* and a lexical and syntactic *back end.* The front end processes the signal and produces acoustic-phonetic units (apus). Apus are phonemes and in some cases allophones of phonemes. The back end accepts apus and produces phrases or sentences. The two operate independently: the front end produces apus without regard to lexical and syntactic considerations, and the back end produces phrases without influencing the acoustic processing.

The front end itself also consists of separate stages: a signal processing

stage and a segmentation and classification stage. The signal processing stage takes the digitised speech signal and produces an acoustic representation of it. The segmentation and classification stage accepts this acoustic description and produces a lattice of scored apus.

This modularity of the system has advantages and disadvantages. The disadvantages follow from the denial to some components of information from other components. For example, in an integrated system in which the classifier has access to lexical information, the classifier need produce only those hypotheses that form words. This restriction on the number of hypotheses has two consequences. One is improved performance: there is a limit on the number of hypotheses the classifier can produce, and if these are lexically constrained then we are sure of only getting the good ones. The other is that an integrated system can use techniques which in a modular system are computationally intractable. An example is the use of triphones, which the modular system at CSTR cannot use, because there are too many combinations. This matter will be discussed later in this chapter.

The advantages of a modular system come from the fact that attention can be focused on one component at a time during development. The advantages include the following.

1. Variants of the front end can be constructed and tested without having to develop a corresponding vocabulary and language models for the back end. Similarly, different versions of the back end can be tried without each time repeating the front end processing.

2. The performance of the CSTR front end is measured in terms of the *entropy* of the apu lattice (this measure is described later). An entropy measure is more general than a measure based on utterance or word recognition rates. It is also more sensitive to small differences in front end performance, which allows statistically significant results to be obtained on smaller databases.

3. Causes of difficulty in the front end, such as an adverse choice of allo-

phones for a particular phoneme, can be discovered more easily from the apu lattices generated by the front end than from the words and phrases produced by an integrated system.

## 3.2 The CSTR Front End

As stated above, the front end consists of a signal processing stage that takes in the speech waveform and produces an acoustic description of it, and a segmentation and scoring stage that takes this acoustic description and produces a lattice of probability-scored apus.

### 3.2.1 Signal Processing

The signal processing stage includes end-point detection on the utterance, vector quantisation and computation of acoustic feature vectors from the waveform. The technical details are as follows (McInnes *et al.*, 1991).

> The input speech is passed through a lowpass filter with a cutoff frequency of 4.75kHz, and sampled to 16-bit precision at a frequency of 10kHz. The start and end points of the utterance are located by an algorithm based on that of Lamel et al (Lamel *et al.*, 1981), in which provisional start and end points are found using thresholds on signal magnitude (the sum of absolute sample values, taken over a 10ms frame), and these may be extended to include regions of high zero-crossing rate so as not to cut off initial or final weak fricatives or final stop bursts. The signal magnitude thresholds for locating the start and end points are adapted during non-speech intervals according to the background noise level (McInnes, 1988), pp 158–161.
>
> After preemphasis (with a factor of 0.97), a 14th-order linear predictive analysis is performed in a 20ms Hamming window every 5ms, and the first 10 cepstral coefficients are derived (Markel & Gray, 1986) ...

> The result of the processing so far is a vector of acoustic features (cepstral coefficients and optionally log formant frequencies) every 5ms. A concatenation operation and a discriminant transformation may be applied at this stage ... After this, a vector quantisation (VQ) operation is applied, in which each input feature vector is mapped to the nearest vector (as determined by a Euclidean distance calculation) from a codebook of 256 prototype vectors derived by a clustering analysis, and the sequence of vectors is replaced by a sequence of VQ *indices* which are integers in the range from 0 to 255. It is this sequence of VQ indices which forms the acoustic representation of the speech which is passed to the segmentation and classification component.

### 3.2.2 Segmenter and Classifier

Segmentation and classification are separate operations. Both use discrete-output hidden Markov models (HMMs) to represent the apus (HMMs are described in D). Each apu is represented by a three-state model, with no skip transitions. Segmentation uses a connected Viterbi algorithm (Forney, 1973) to find the globally best sequence of models that match the input. This sequence does not necessarily contain the best scoring segment at each point; it is the *sequence* that is optimal. The sequence of segments covers the input so that there are no overlaps or gaps. Only the start and stop times of the segments are kept, ready for the next stage.

A classification run now uses the same Viterbi algorithm to match each model against each segment, and obtains a probability score for each. The classification run would seem to repeat the work of the segmenter, but this is not strictly true. The best-scoring apu for each segment produced by the classifier is the same as the best-scoring apu produced by the segmenter; but the segmenter's second-best apu may not be for *this* segment, but for one that partly overlaps this one. The classifier therefore repeats the work of the segmenter, but only on the best-scoring segments. The purpose of the classification run is

to obtain correct scores for the lower-ranking apus for each segment.

For each segment, the classifier produces a scored match for each apu in the system. These log probability scores are derived from the Viterbi scoring algorithm, and normalised for segment duration by dividing by the length of the segment in 5ms frames. They are intended to be proportional to $P$(acoustic data|apu). The sum of the scores over all the apus is taken to be $Q$, the 'overall probability'. The scores are then normalised to sum to 1. An extra 'apu' called DELETE is now appended, and given a score $Q^d$, where $d$ is a constant. The probabilities of the other apus are adjusted by multiplying by $Q^x$, where $x$ is also a constant. If $x$ is made greater than $d$, then the probability of deleting the segment is greater than the scores for the apus if the overall score $Q$ is poor. The values of $x$ and $d$ are established by trial and error over several runs.

The DELETE apu is intended as a guide to the back end when it matches segment sequences against words in the lexicon; a segment with a high DELETE score will be more likely to be deleted than a segment of good quality. Back end operations are described in a later section.

### 3.2.3 Demi-diphones

It is now time to describe the mechanism that is used for modelling syllables at the front end. Syllable modelling is done using the context-modelling mechanism of the existing CSTR front end. The mechanism is used unmodified for the syllable work.

Considerable improvements in speech recognition performance have been obtained in recent years by modelling *triphones* (Lee, 1988; Russell *et al.*, 1990b; Wood & Pearce, 1990). A triphone is a phone as defined by the phones on its left and right. It does not actually include the left and right phones (the word 'triphone' is misleading in this respect). As we saw in chapter 2, the realisation of a phoneme is heavily affected by the surrounding phonemes, and triphone modelling recognises this effect by creating many models for each phoneme: one for every phonemic setting in which it is found. The improvement in performance is substantial. On a thousand-word system with no grammar, Lee (Lee,

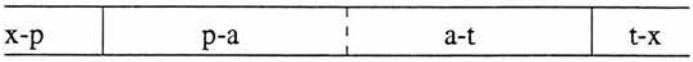
1989) reports an improvement in word accuracy from 59.2% to 72.8%[1].

Triphone modelling is only feasible in an integrated system, where the lexicon and the grammar can constrain the generation of triphone hypotheses. As we saw in section 2.6, the number of triphones is large. The CSTR classifier would need to generate this number of hypotheses for every segment. In an integrated system the number of hypotheses is much smaller, because in sequence they must form legal words in legal syntactic combinations. The CSTR front end, however, cannot restrict the number of hypotheses in this way, and the number is too large for practical computation.

The CSTR system does employ a more limited kind of context modelling, in the form of *demi-diphones* (McInnes *et al.*, 1991). A demi-diphone is a unit which extends from a phone boundary to a phone nucleus, and from a nucleus to a boundary. A left demi-diphone is specific to the phoneme on its left, and a right demi-diphone is specific to the phoneme on its right. For RP this gives $45^2 = 2025$ demi-diphones of each kind, making 4050 in all. Segmentation and classification works as follows. The signal is first segmented into demi-diphones. Left and right halves are then matched to produce segments of whole phonemes, which are annotated with the left and right contexts of the respective halves of the pair. The matching is done in accordance with a set of transition networks, which specify which left half goes with which right half, and their left and right contexts. The result looks like a triphone segmentation, but is not, because the left context and the right context do not together define the phone between them.

Figure 3.1 illustrates. It shows the phoneme /a/ in its context to form the sequence /p a t/. /a/ is made up of two demi-diphones, which are denoted 'p-a' and 'a-t' in the figure. The sequencer which matches the left and right halves takes due note of the contexts /p/ and /t/ when it produces the whole /a/. Note that if this were a full triphone segmenter, there would be a full 'p-a-t' triphone; the /a/ would specific to both left and right contexts. Although the

[1] Lee defines word accuracy as the percentage of words correct minus the percentage of insertions.




*left demi-diphone* *right demi-diphone*

Figure 3.1: Demi-diphones for the phoneme /a/ in 'pat'. 'x' stands for the unspecified neighbours of the /p/ and /t/ phonemes.

matched demi-diphone looks like a triphone (it has a left and right context), it is not, because the /a/ is not specific to both contexts. Its left half is specific only to /p/, and its right half is specific only to /t/.

Once a segmentation has been obtained, classification is performed using models that are appropriate to the segment's context. Besides the fact that the segments are not a true triphone segmentation, there is another reason why this scheme does not give the same performance as an integrated system. This is that although the segment halves obey their respective left and right contexts, the sequence of *phonemes* picked by the back end when it comes to construct words, need not obey these contexts. In the example we've just seen, the back end would see three segments /p/, /a/ and /t/. However, the /p/ segment will have other phoneme candidates as well, and the back end may prefer them to the /p/. It might construct the word *bat*, for example, and this would violate the left context of the first half of the /a/. Of course, the other segments also have other candidates, and the back end could construct *pad*, *beck*, or many other words. In an integrated system this situation cannot occur, because segmentation and classification happen under control of lexical access. The words that are constructed will contain phones that are fully determined by their contexts.

### 3.2.4 Syllable Networks

The demi-diphone mechanism will be used for our own purposes in this thesis, namely, to implement syllables. The demi-diphone transition networks specify which left demi-diphone goes with which right demi-diphone to produce a full phoneme. The networks are not in fact restricted to only two constituents; any number of constituents can be specified. We shall use the transition networks,

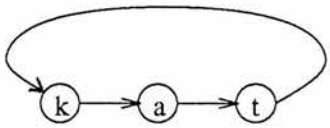



Figure 3.2: Network for the syllable 'cat'.

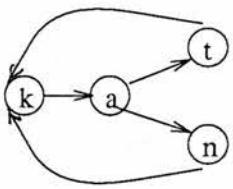



Figure 3.3: Combined network for the two syllables 'cat' and 'can'.

not to define phonemes in terms of demi-diphones, but to define syllables in terms of apus.

A network for the syllable *cat* is shown in figure 3.2. The network consists of three nodes, one for each phoneme. The nodes are connected by arcs, in sequence. The last node has an arc which connects it with the first node again. We shall see in a moment why this is so.

There is a network for every syllable, and for the sake of efficiency the networks of the different syllables are combined together. If the syllables *cat* and *can* are combined, for example, we get the network given in figure 3.3. The common stem /k a/ is collapsed into a single initial sub-network, after which the network branches to describe the different endings. Both branches loop back to the starting node /k/.

All the syllables that can occur are combined together to form an integrated network.Note the use of terminology: a *network* defines a single syllable. The networks of different syllables are combined to form an *integrated network.*

The syllable networks (i.e the networks that belong to an integrated network) are used at an internal stage of the segmenter. We shall call this stage the *sequencer.* The segments that the segmenter produces are matched against the networks so that the segments that are output form valid sequences of syllables. The network is applied continuously, and that is why the ends of syllables point to the beginning again. If the network illustrated in figure 3.3 were used, then the segmenter could produce only a sequence of *cat*s and *can*s, in whatever

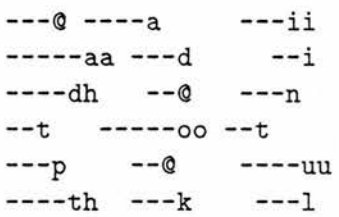



Figure 3.4: Multiple segments produced by an internal stage of the segmenter.

order is determined by the data.

The syllable-matching algorithm that is part of the segmenter works as follows. Figure 3.4 shows part of a segmentation of an utterance that is obtained by the segmenter before it has applied the syllable networks. The segments are shown non-contiguously to make the figure easier to read. Many paths can be traced out from the beginning, and the segmenter needs to choose the best one. Without syllable networks the best path is the one whose constituent segments have the highest score. The networks impose the additional requirement that the sequence must form a valid concatenation of syllables.

Matching the segments against the syllable networks is done under the control of a dynamic programming algorithm[2], which finds the globally optimal sequence of syllables that can be fitted to the segments. A beam search is used to restrict the number of possibilities that are generated internally. The output is a lattice of phonemes as before, with the difference that from left to right the phonemes now define a string of legal syllables. The process of matching the segments against a set of syllable networks to produce a cleaned-up segmentation, will be called *syllable-assisted segmentation* or *syllable sequencing.*

The syllable networks form the basis of the experiments to be performed in chapter 5.

[2]Dynamic programming is explained in appendix E.1.

## 3.3 Entropy as a measure of front end performance

In this section the use of entropy to measure front end performance is described. Entropy in general is described in appendix B. The idea of using entropy in a speech recogniser is due to Crowe (Crowe, 199?). The description below draws on personal discussions with him and with Fergus McInnes.

In speech recognisers which produce a single string of phonemes or words as output, the performance can be effectively measured as the percentage of phonemes or words spotted correctly. The CSTR system, however, does not output a single string of hypotheses, but a lattice of them. For such cases a different performance measure is needed, and the measure adopted is entropy.

In appendix B it is shown that the entropy $H$ of a symbol $s$ drawn from a distribution of many symbols is

$$H = -\sum_s p_s \log p_s$$

where $p_s$ is the probability of drawing the symbol. The logarithm is taken to base 2. The entropy indicates how much information is needed to specify the symbol, and it is measured in bits. With a phoneme lattice we are interested in the information needed to specify the sequence of apus in the correct answer, given the apus in the lattice. If the front end were 100% accurate, the apus in the lattice would be the same as the ones in the correct answer, and the information needed would be zero. In practice it is never zero, and the discrepancy between lattice and correct answer determines the information that the back end must supply in order to correctly identify the utterance.

The entropy that measures this information is based on the posterior (conditional) probability of the correct answer, given the lattice. This probability is not the same as the probability scores provided by the front end, which are in the lattice. In order to obtain the posterior probabilities we need to compare the lattice against the correct answer, and count the frequencies with which the correct answer is predicted. The comparison is not a straightforward left-to-right match between lattice and correct answer. Insertions and deletions may

be necessary, and the lattice may be based on a multiple segmentation, in which there is more than one segment for some of the correct answers. An alignment procedure is therefore needed to establish the correspondence between lattice and answer, and this procedure itself will produce more than one alignment for any one segmentation. All these difficulties lead to the following formula for the entropy of a phoneme lattice (the details are in (McInnes, 1993)).

$$H = -\log \frac{\sum_{correct\ alignments} Q(a)}{\sum_{all\ alignments} Q(a)}$$

where $Q(a)$ is the probability of an alignment between the lattice and the correct answer. It is proportional to the posterior probability

$P$(recognition and alignment a | apu sequence).

By Bayes' theorem this is proportional to

$P$(apu sequence)$P$(lattice and alignment | apu sequence).

The first factor is the prior probability of the apu sequence, that is, the product of the relative frequencies of the apus in the corpus from which the utterances are drawn. The second factor is the product of probabilities of apu-to-segment matches, of segment deletions, and of segment insertions. The probability of an apu-to-segment match is the bottom-up score provided by the front end, that is, $P$(acoustic data in the segment | apu). The probability of a segment deletion is the product of a similar bottom-up score, i.e the probability of the acoustic data in the segment given that there is no apu, and the probability that a deletion is needed here. This probability, and the insertion probability, are not bottom-up scores, and must be obtained separately.

If the lattice contains multiple segmentations, then different paths through it may contain different numbers of segments. Paths with fewer segments will have a higher probability than paths with more segments, since the probability for a path is the product of individual probabilities that are less than 1. This gives short paths an unfair advantage. To compensate for this, the probability for each segment is raised to the power of the segment's duration, expressed in

units of a standard duration. The standard duration is chosen as 80ms. If this compensation were applied to inserted segments as well, it would be nullified, since inserted segments have a duration of zero: raising the insertion probability to a power of zero would yield a factor of 1. Inserted segments therefore escape the length compensation. This makes a path with insertions somewhat more expensive than a path of the same length without insertions, because the former contains extra terms, which have not been adjusted for duration. It is easiest to see this in the log domain. In the log domain the score for the whole utterance is the *sum* of the apu scores. Each apu score is *multiplied* by a duration factor. The sum of the duration factors is proportional to the length of the utterance. Consider now two segmentations of the same utterance, one with $n$ segments and one with $n + 1$ segments. Let neither of them contain inserted segments. The score for the second segmentation is the sum of $n+1$ terms, but this is not unfair because the terms have been adjusted with duration factors which are slightly smaller than the ones for the first segmentation (the $n + 1$ factors still add up to the same total duration). If, however, the extra segment in the second segmentation is an inserted one, this is not true. Only $n$ terms will be adjusted for duration, the same as for the first segmentation. The $(n+1)$th term, which is in effect multiplied by a duration factor of 1, makes the second duration look longer. This bias against segmentations with insertions is unavoidable. We shall return to this matter in a later chapter.

The entropy is computed in two stages. The set of utterances on which the entropy is to be calculated is divided into two equal-sized sets: the *estimation set* and the *evaluation set*[3]. The estimation set is used to obtain alignments between the phoneme lattices and the correct answers. The insertions and deletions that were necessary to achieve these alignments are counted to obtain estimates for the insertion probabilities, and the probability of a deletion. The evaluation set is then used to compute the entropies. Alignments are constructed between the lattices and the correct answers, and the quantities $Q(a)$

[3]This is not to be confused with different sets of utterances for open and closed tests. To compute entropy scores for an open test, for example, the open test set is divided into an estimation set and an evaluation set.

referred to above are computed. Both per-utterance and per-apu entropies are calculated. The utterance entropies are used in comparisons between lattices produced by different versions of the front end, as described later in this thesis.

## 3.4 The Back end

In the CSTR system the construction of word strings is performed by the lexical access module. This module will be referred to as *lexax* in what follows. The construction of word strings involves both lexical lookup and parsing against a grammar, in an integrated operation. For ease of explanation the two steps are described separately, with lexical lookup first.

The task of lexical lookup is to match strings of phonemes in the input against words in the lexicon, in the presence of errors in the input. As explained in chapter 1, these errors are of three kinds, and need to be corrected in order for a match to succeed. A missing segment is repaired with an *insertion*, and a superfluous segment is repaired with a *deletion*. A mis-identification is repaired with a *substitution*. Since for each segment the front end produces a scored hypothesis for *all* the apus, so it might seem that the back end does not need to bother with substitutions. Suppose for example that the back end is given a sequence of segments whose top-scoring phonemes are /sh ou b @/, which it wants to match against the word *sober*. The first apu has been mis-identified, but we know that all the other apus are also present in the first segment (with worse scores), and in particular the apu /s/ is there. The substitution s.sh therefore seems unnecessary. However, this need not be so, because substituting /s/ for a better-scoring /sh/ might be cheaper than using a worse-scoring /s/ unchanged.

An example of the insertion, deletion and substitution repairs necessary to convert a phoneme lattice into a sequence of words was given in table 1.1.

| | |
|---|---|
| hard | /h aa d/ |
| heard | /h @@ d/ |
| hurt | /h @@ t/ |
| weird | /w i@ d/ |
| what | /w o t/ |
| word | /w @@ d/ |

Table 3.1: A six-word lexicon

### 3.4.1 Error correction

The matching operation will be described in more detail with the help of an example. Assume that the lexicon consists of the six words given in table 3.1. The input string is compared against each word, and a score is calculated for each comparison. The words are ranked according to their scores, and the top five or so (the number is an adjustable parameter) are put forward for further processing.

The score is calculated partly from the scores that were attached to the phonemes during the pattern matching stage. These are frequently called the *bottom-up* scores, because they come from an earlier stage of processing, which is visualised as being at the bottom. In the CSTR system these scores are negative logarithms of probabilities. Since a high score means a low probability, it follows that these scores can be interpreted as *costs*. Lexax will prefer phonemes with low costs to those with high costs.

If the input string is /h aa d/, so that it matches perfectly the phonemes of the lexicon entry for *hard*, then the probability that it was *hard* that was spoken is the product of the bottom-up probabilities. Because the logarithm of a product is the sum of the logarithms of the multiplicands, the score for this word is the sum of the bottom-up scores. In this simple case therefore we add the phoneme scores to get the score for the word.

Where corrections are necessary for a match to take place — that is, where the input string needs to be adjusted for substitution, insertion and deletion errors — lexax needs to take this into account. The corrections incur *penalties*. Suppose the input string is /w h @@/, and we are trying to match it against the word *hard*. Figure 3.5 illustrates. The first phoneme of the string, the /w/, is

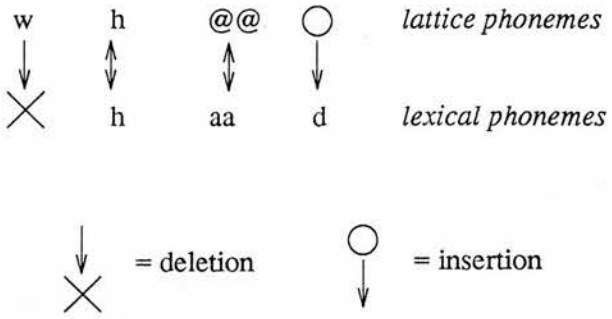



Figure 3.5: Matching operations performed by lexical access, to construct the word *hard*

an insertion and needs to be removed. We pay an insertion penalty by adding it to the score (a high score indicates a poor match). The second phoneme, the /h/, matches the first phoneme of the word, and no penalty applies. The phoneme score is added as usual. The third phoneme, the /@@/, does not match the /aa/ of *hard*. This is a substitution error, and a penalty is paid. The last phoneme of *hard* has nothing to match against. This is a deletion error, and again the appropriate penalty is paid. The resulting score is a sum of phoneme scores and penalties. It will be higher than the score in the earlier example, to indicate a poorer match.

The matching operation has been described as if the input is a single string of phonemes. In actual fact the front end produces a lattice rather than a string. This does not substantially alter the matching operation. Instead of one candidate per segment, there are many. They are all considered, in all combinations[4]. In the rest of this thesis I shall continue to talk as if lexax receives a simple phoneme string as input.

### 3.4.2 Finding the best match

The above match was achieved by means of a deletion, an equality, a substitution and an insertion. It can be achieved also using a different series of operations. Figure 3.6 shows how the same effect can be achieved with three substitutions. There are in general many ways a match can be achieved, and

[4]This is not strictly true. See section 3.4.2 below.

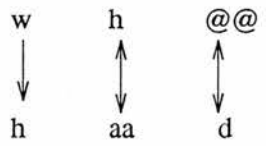



Figure 3.6: Substitutions performed by lexical access to construct the word *hard*.

their scores are not the same. This is true of the other words as well; there are several ways /w h aa/ can be matched against *hurt*, and against *weird*, and against *what*, etc. For each word we will have several corresponding scores. Which score shall we choose in each case? Since the score is used to decide which of these words is the most likely to have been spoken, the matter is an important one. A consistent choice is to use the lowest (best) score for each word.

To find the lowest score for a word, it is not necessary to perform all its matches first, and then choose the minimum. An algorithm called *dynamic time warping*, based on the principle of *dynamic programming*, allows the best match to be found with fewer operations. Dynamic programming is described in appendix E.1.

### 3.4.3 The confusion matrix

The substitution, insertion and deletion penalties which are paid when a phoneme sequence is matched against a word, come from a *confusion matrix*. This matrix is trained in a separate session, using a Viterbi algorithm. During the training the phoneme lattices produced by the front end are compared against hand transcriptions, and the number of times that an /e/ needs to be substituted for an /a/, or a /dh/ inserted, etc, are recorded in the matrix. The Viterbi algorithm, a close cousin of dynamic programming, is needed to find the cheapest combination of substitutions, insertions, and deletions to effect the match.

After the matrix has been trained, it can be used in normal recognition runs. If a run is performed on the same data on which the matrix was trained, it is a *closed test*. If it is performed on different data, it is an *open test*.

The confusion matrix is the chief instrument we use in making the back end

1. The price range is smaller than any of us expected.
2. They asked if I wanted to come along on the barge trip.
3. Amongst her friends she was considered beautiful.
4. The smell of the freshly ground coffee never fails to entice me into the shop.
5. I'm often perplexed by rapid advances in state of the art technology.

Table 3.2: The first five ATR sentences

responsive to syllables. In chapter 6 we shall train several confusion matrices, according to the position of the phonemes in a syllable.

### 3.4.4 Syntax

In the description of the matching procedure, it was assumed that, in matching phonemes against words from the lexicon, lexax could choose whichever words it wanted. This is in fact not so. In common with many speech recognisers (for example, SPHINX (Lee, 1988)), lexax is guided in its choice by a *grammar*. This procedure reduces the number of words that needs to be tried, and greatly improves the accuracy of the resulting word string.

The simplest and most effective grammar is simply a list of all the utterances that are expected to be spoken. The ATR dataset, for example, consists of 200 sentences, and a simple grammar for this consists of just these sentences. Table 3.2 gives the first five of them. Suppose the dataset consisted of only these sentences. Guided by this grammar, lexax matches the initial phonemes of the input only against the words *the, they, amongst* and *i'm*. In the full dataset of 200 sentences there are 200 such words, not all of them distinct. Since the whole ATR lexicon contains about 1240 words, this is a considerable reduction in the number of possibilities. If now the first match yields the candidates *the* and *they*, then the grammar determines that the second word must be *price, asked* or *smell*, and so on through the remaining phonemes in the lattice.

The CSTR back end allows different kinds of grammar. A grammar which

**Phoneme string**
ii ch i z @ r oo @ s ou d k dh e m t ii ng
t @ z k r a r= ch th
**No grammar**
Itches are also camping scratch
**Bigram grammar**
Itches are always served at the back and to scratch
**Full grammar**
Itches are always so tempting to scratch

Table 3.3: The effect of different grammars on the ATR sentence *Itches are always so tempting to scratch.* Sentence 22.

contains all and only the word strings that were spoken, as illustrated above, is called *full syntax.* Another option is to run the system with no grammar at all, and this is called *zero syntax.* A third option is a *bigram grammar*, which consists of the word pairs in the dataset. A bigram grammar based on the first five ATR sentences would determine the same starting set of words, namely *the, they, amongst* and *i'm.* Following *the*, however, the possibilities then are *price, barge, smell, shop* and *art.* Other grammar options are available, some of which are based on word classes such as *determiner*, *count nouns*, and *colour adjectives.*

Table 3.3 shows the effect of different syntax options on an example sentence.

A bigram grammar is less restrictive than a full grammar, and yields higher error rates. The choice of grammar depends on the application. If the sentences that will be spoken are not known in advance, then obviously a full grammar is not appropriate. The experiments performed in this thesis were done using three kinds of grammar: zero, bigram and full. The experiments are described in chapter 6.

### 3.4.5 Implementation details

The details of the implementation of lexax are taken largely from (McInnes *et al.*, 1991). Lexax takes as its input a lattice of scored phonemes, matches

them against a lexicon and a grammar, and produces as its output a list of scored word strings. The matching is done using a two-level (word and sentence) dynamic programming algorithm which allows arbitrary insertions, deletions and substitutions in the match between phoneme lattice and lexicon. This procedure gives rise to an unacceptable number of word and sentence candidates, even in the presence of the grammar. Their number is kept down by imposing a limit, by means of *beam searching*: only those words, and those word strings, are kept which fall within a specified distance from the cheapest word and word string.

For efficiency the lexicon is tree structured. It is constructed by folding together common initial substrings of word pronunciations. Several versions of a word may be stored to represent common pronunciation variants.

The basic algorithm is implemented as a chart parsing operation. Chart parsing (Thompson & Ritchie, 1984) is a flexible mechanism for implementing different kinds of parsers. Lexax uses a modified version which incorporates the dynamic programming algorithm and lookup in a lexicon.

## 3.5 Modular and integrated systems

With its separate front end and back end, the CSTR system is a modular system. There are advantages and disadvantages to modular systems, as we have seen. The main disadvantage is that an integrated system outperforms a modular one. McInnes (September 1992) integrated the front end and the back end of the CSTR system, and compared the performance with the unintegrated system. In the integrated system the front end was constrained by the lexicon, and, optionally, a word-pair grammar. In the unintegrated system the front end produced the phoneme lattice unguided, and the phonemes were then matched against the lexicon, again with and without a word-pair grammar.

The phoneme models in either case were trained on 600 DARPA training utterances, and a standard phonemic lexicon was used with no variant pronunciations. The performance was measured on 25 evaluation utterances, using DARPA evaluation software, which measures word error rate. The word error

rate measures the operations necessary to turn the words obtained into the right answer. It is the sum of the substitutions, insertions and deletions of words divided by the total number of words[5]. For the integrated system the word error rate was 43.1% with no grammar, and 14.2% with the word-pair grammar. For the unintegrated system the word error rate was 59.9% with no grammar and 39.6% with the word-pair grammar. It is evident from these figures that an integrated system, which is able to apply top-down (lexical) constraints, performs much better than a modular one.

A disadvantage of integrated systems is the amount of retraining that is necessary when, say, a larger vocabulary needs to be used. Many integrated systems have unified statistical models with phonemes, words and phrases at different levels (the SPHINX system described in chapter 2 is an example). When such a system needs to accommodate new words, a large amount of retraining is necessary, because the models at the phoneme and grammar levels need to be retrained as well. A modular system does not have this disadvantage, but of course it does not perform as well. It would be useful if we could separate the front and back end sufficiently to give us the advantages of modularity, while retaining the ability to apply top down constraints to the front end. This possibility can be realised with the use of syllables.

When we constrain the front end to produce only those phonemes that form valid syllables, we are applying top down constraints without sacrificing the flexibility of a modular system. This is because the number of syllables is fixed, and is not affected by changing requirements in the vocabulary or grammar. Of course, using syllables as a source of constraint has a different effect from using words, and this matter is investigated in detail in chapter 5.

We might call a system that uses syllables at the front end a loosely-coupled system. A loosely-coupled system should have the flexibility of a modular system and the performance of an integrated system. It is not the aim in this thesis to spell out in detail how a loosely-coupled system might be designed.

[5] A similar measure is used to measure the performance of the CSTR back end, and examples can be found in section 6.2.

The intention is merely to explore the effects of having a syllable level as a top-down constraint in the front end, in order to provide information for future system-building efforts. A loosely-coupled system might have a separate level at which syllables are represented explicitly, so that phonemes will form syllables and syllables will form words. On the other hand, it might be better to have the syllables merely as a source of constraint, and retain phonemes until the lexical stage. That is the option which this thesis uses by default, but it is not meant to rule out other possibilities.

(This thesis also advocates the use of syllables as an extra level of organisation in the back end. This is a separate matter, which is independent of whether the system is modular, loosely-coupled, or integrated.)

## 3.6 Summary

The CSTR speech recogniser is a modular system, which makes it convenient to determine the effects when different parts of the system are to be improved. In particular the front end which produces the phoneme lattice and the back end which produces the word strings, are separate operations.

The front end uses hidden Markov models to segment and classify the input, and produces a phoneme lattice. It includes a mechanism for training apu models according to their context. This is the demi-diphone mechanism, which we shall use for the purpose of syllable sequencing in chapter 5. The performance of the front end is measured using entropy. It allows us to assess the quality of the phoneme lattice independently of the words that can be recognised from it. The entropy of the phoneme lattice is the negative log probability of finding a correct path in the phoneme lattice. To obtain this measure, the correct path must, of course, be known.

The back end, of which lexax is the main part, matches the phoneme lattice against words in the lexicon. In doing so, lexax must repair insertion, deletion and substitution errors in the lattice. A dynamic time warping algorithm is used to find the optimum way of doing this. The costs of repairing insertion, deletion and substitution errors are obtained from a confusion matrix. The

confusion matrix is trained in advance.

Lexax is optionally guided by a grammar, which determines which words from the lexicon are to be matched against the lattice. Different kinds of grammars can be devised for different purposes.

The experiments to be performed in the front end and the back end concern the use of the syllable. Before we turn to these experiments, we look at the status of the syllable in language and speech.

# Chapter 4

# Syllables and allophones

## 4.1 Introduction

In its Greek origins ($\varsigma\upsilon\lambda\lambda\alpha\beta\epsilon$) the word *syllable* means 'taking together', or a combination. According to the ancient Greeks the combination was that of a segment which could be sounded independently (roughly equivalent to a vowel) and segments which could not. Later Greek writers called the latter sounds $\sigma\upsilon\mu\phi\omega\nu\alpha$ (*symphona*; *consonantes* in Latin) because they produced a sound only when combined with vowels (Allen, 1973, p 32–3).

This chapter describes syllables in theory and in speech recognition practice.

## 4.2 Introduction to terms and concepts

Defining a syllable is not easy, and various definitions have been suggested. These are best considered after some preliminary terms have been defined, which is done in this section. The rival definitions of the syllable are considered in the next section. The material below comes from many sources, the main ones being (Abercrombie, 1967) and (Allen, 1973).

Although giving a definition is difficult, the intuitive understanding of syllables goes back to ancient times. Before speech can be written down, it needs to be broken down into convenient units, and most writing systems that mankind has devised use the syllable as this unit. (However, writing systems that use

| | *Onset* | *Nucleus* | *Coda* |
|---|---|---|---|
| oh | | oh | |
| go | g | o | |
| oat | | oa | t |
| goat | g | oa | t |

Table 4.1: Onset, nucleus and coda of a syllable

| *Syllable* | *Type* |
|---|---|
| oh | v |
| go | cv |
| oat | vc |
| goat | cvc |

Table 4.2: Syllable types

derivatives of the Greek alphabet, such as the Latin alphabet used in English, or the Cyrillic alphabet used in Russian, are not syllabic; they are phonemic). In these systems there is a sign for every syllable. The users of these systems have no difficulty dividing their speech into syllables.

Words consist of one or more syllables. Words with one syllable are *monosyllables*, and those with more than one are *polysyllables*. A syllable is divided into three stages: an *onset*, a *nucleus* and a *coda*. Consider the English monosyllables *oh, go, oat* and *goat* (the example is Abercrombie's (Abercrombie, 1967)). The three parts of the syllable are assigned to these words as given in table 4.1. We see that *oh* consists only of a nucleus, *go* of onset and nucleus, *oat* of nucleus and coda, and *goat* of onset, nucleus and coda. All these syllables have a nucleus.

In these examples the nucleus consists of a vowel (symbolised as V), and the onset and coda, when they are present, consists of a consonant (symbolised as C). *Go* is said to be a CV syllable. Table 4.1 can be summarised as shown in table 4.2.

English allows combinations of consonants at onset and coda, called *consonant clusters*. The word *spy* has the consonant cluster *sp* in its onset. It is a CCV syllable. *Oops*, a VCC syllable, has the consonant cluster *ps* in its coda. The word *streets* has a CCCVCC structure. The size of the cluster, and the consonants that make it up, are restricted, for English as in other languages.

The largest onset in English is CCC, which we have just seen in *streets*. The largest coda in English is CCCC, as in /s i k s th s/ *sixths*, and similar clusters in *exempts* and *glimpsed*[1].

The structure of consonant clusters is also restricted. In English, the voiced velar nasal at the end of *hang* cannot begin a syllable, and the voiceless velar fricative at its beginning cannot end a syllable. Other languages do allow an *ng* onset and a *h* coda. The cluster that ends the word *oops* cannot appear in an onset (although some people pronounce the *p* in the trade mark 'Psion Organiser'). The cluster *sr* is inadmissible in English, and in a CCC onset, the first C is always *s* (*square, split, strong*). These structural regularities will find a use in the front end work described in chapter 5.

Not all sequences of consonants are clusters. In the word *hatrack* the consonants *t* and *r* belong to two syllables, whereas in *tray* they form a cluster. In *hatrack* they are said to *abut*. The difference between clustered and abutting consonants can often be heard. In most people's pronunciation of *tray* the *r* is a voiceless fricative, and in *hatrack* it is a voiced approximant.

The restrictions that apply to monosyllables do not apply, in English, to polysyllables. For example, *tl* is not allowed in the onset of a monosyllable[2], but it may appear as such in a polysyllable. In some people's pronunciation of *Atlantic* the *l* is fricated, indicating that it is part of the release of the previous *t* and therefore in the same syllable with it. The syllabification there is *A-tlantic*. Other people, of course, do not fricate the *l*, and they syllabify the word as *At-lan-tic*.

Some syllables do not have a vowel as a nucleus. In the pronunciation of many people there is nothing between the *t* and the *n* in *button*, or between the *t* and *l* in *little*. In the first case we have a *syllabic n* and in the second a syllabic *l*. A syllabic *m* can be found in some pronunciations of *bottom* and a syllabic *r* in *number*.

[1] Abercrombie (Abercrombie, 1967) quotes Whorf as suggesting that a CCCCC coda is possible in the last word of 'thou triumphst!', because some people insert a *p* between *m* and *f*. The cluster would be m-p-ph-s-t.

[2] For many accents of English. I have heard Yorkshire people say *tlear* instead of *clear*.

The syllabic consonants raise a difficulty in the definition of vowels and consonants. They are usually defined in phonetic terms, as I did too in chapter 2. Vowels were defined by Bloomfield (Bloomfield, 1933) as 'modifications of the voice-sound that involve no closure, friction, or contact between the tongue and lips'. Anything else is a consonant. By this definition the *y* in *yet* and the *w* in *wet* are vowels, but from the point of view of syllable structure they are consonants. Conversely, as we saw in the previous paragraph, the consonant *n* plays the role of a vowel in many pronunciations of *button.*

The usual solution is to call *y* and *w* *semi-vowels*, and to speak of syllabic consonants in words like *button.* Pike (Pike, 1943) suggested a simple way of looking at this problem, which separated the phonetic form of a segment from its phonological role. Abercrombie (Abercrombie, 1967, p 80) describes it as follows.

> Pike introduced two new terms to replace the words vowel and consonant when used with reference the phonetic form, without regard to syllabic function: *vocoid*, and non-vocoid or *contoid.* The terms are very rigorously defined. A vocoid is a segment with a stricture of open approximation, with or without velic closure, and with central passage of the air-stream[3]. All other segments are contoids. Pike then puts forward the term *syllabic* for a segment representing a V element of syllable structure, and *non-syllabic* for a segment represented a C element of syllable structure. The two sets of terms when used together give us, for any segment in a given utterance, its category in general phonetic terms and its place in [syllable] structure. Thus we have, in English, a *syllabic vocoid* in *awe*, a *non-syllabic vocoid* at the beginning of *yet*, a *syllabic contoid* in the second syllable if *people*, and a *non-syllabic contoid* at the beginning of *pet.* The traditional terms vowel and consonant, Pike suggests, can be used as synonyms of syllabic vocoid, and non-

[3]In Pike's words (Pike, 1943, p 78) 'A *vocoid* is a sound during which air leaves the mouth over the center of the tongue and without friction in the mouth.'

syllabic contoid, respectively.

The interjections *shh* and *mmm* consist of syllabic contoids.

## 4.3 Definition of the syllable

Ancient Greek rules of syllabification were sometimes just concerned with where to split the word on a line, and were based on a mixture of phonetic, phonological and grammatical considerations (Allen, 1973, p 29). More modern theories take greater care with how the units of vowel, consonant and syllable are defined. Modern theories are of different kinds. Articulatory theories look at the activity of the muscles of articulation, including the breathing muscles. Under this head respiratory and motor theories may be distinguished. Acoustic theories consider the pattern of harmonic and non-harmonic stretches in the signal. Auditory theories may count peaks of audibility and phonological theories proceed from the combinatory possibilities of the sounds of a language.

### 4.3.1 Phonological theories

In phonological theories syllables may be defined in terms of vowels and consonants, or vowels and consonants may be defined in terms of syllables. In the latter case the syllable may for example be defined as a unit of accent placement, and vowel and consonant may then be defined as its central and marginal constituents. An example of the former is that of O'Connor and Trim (1953). They begin by tabulating the numbers of common environments of the 34 different phonemes of RP[4].For example, in word-initial position, /d/ and /oo/ have the common (right) environment /r/ in the words *dream* and *aural*; in this position they found that /d/ and /oo/ have 3 common environments in all. As a further example, consider the phoneme /p/. In all the positions studied — not just word-initial — they found that /p/ has 42 and 41 environments in common with /t/ and /s/, and 10 and 16 in common with /e/ and /@/. They

[4]This is rather fewer than the 44 *std* phonemes used for RP in this thesis, because, in order not to prejudice their study, they define diphthongs like /ai/ in terms of their constituent phonemes /aa/ and /ii/.

also counted how many phonemes /p/ can be combined with. In word-initial position /p/ occurs before 14 different phonemes. Let us call a phoneme in a position a *context*; thus /@/ before /p/ and /@/ after /p/ constitute two distinct contexts. To restate the foregoing, in word-initial position /p/ occurs before 14 contexts; in all the positions studied, /p/ occurs with 46 contexts. O'Connor and Trim then compared the common contexts with the total numbers of contexts. /p/ and /t/ occur with 46 and 52 contexts respectively, and have 42 contexts in common. 42 is more than half of either 46 and 52. /p/ and /s/ occur with 46 and 59 contexts, and have 41 in common, again more than half. /p/ and /e/ occur in 42 and 59 contexts, and have 10 contexts in common; this is less than half of 42 or 59. Again, /p/ and /@/ occur with 42 and 79 contexts, and have 16 in common; this also is less than half. Comparing /e/ and /@/, they find them occurring in 59 and 79 contexts, as we have seen already, and they have 47 contexts in common; this once more is more than half. Using this simple comparison — numbers of common contexts and half the numbers of contexts altogether — they found that the phonemes fall into two groups. /p/ and /t/ fall in the same group because they share more than half the contexts with which either of them occurs separately. /e/ and /@/ also fall in the same group. /p/ and /e/, however, do not fall in the same group, and neither do /p/ and /@/, because they have fewer than half the contexts in common. O'Connor and Trim called the members of the first group *consonants* and the members of the second group *vowels.*

Having defined vowels and consonants (semi-vowels needed a more complicated argument), they define the syllable as 'a minimal pattern of phoneme combination with a vowel unit as nucleus, preceded and followed by a consonant unit or permitted consonant combination' (O'Connor & Trim, 1953, p 259). This leaves the problem of how to syllabify polysyllabic words. In some cases the point of division can be decided according to the structure of monosyllables: *anger* must be syllabified /a ng - g @/ and not /a ng g - @/ or /a - ng g @/, because there are no monosyllabic words that end or begin in /ng g/. In other cases like *aster* the example of monosyllables leave the point of

division undetermined; we may have /a - s t @/, /a s - t @/ or /a s t - @/. Here the authors suggest using the statistics of unproblematic cases as a guide. This aspect of their work has not found assent; see for example (Bell, 1970).

### 4.3.2 Acoustic theories

Acoustic theories revolve around the idea of 'sonority', which is usually conceived of as the 'audibility' of sounds, and correlates reasonably well with the level of acoustic energy. Peaks of sonority indicate syllable nuclei and troughs indicate its margins. Vowels tend to have a high sonority and consonants a low one, which agrees with the intuitive understanding of a syllable. A weakness of the idea of sonority is its inability to rank sounds of low sonority. Fricatives, for example, have a higher sonority than nasals, and a word like *station*, in which the /sh/ is a syllable margin and /n=/ is a nucleus, makes nonsense of it. The theory must deny that fricatives can ever be a syllable nucleus, a condition that is contradicted by a word like *pst!*.

### 4.3.3 Articulatory theories

Articulatory theories were popularised by de Saussure (de Saussure, 1916). He based his theory on the articulatory criterion of aperture. A syllable begins with a sequence of sounds of increasing aperture, which constitute an 'explosion', and ends with a sequence of decreasing aperture. A good example is the word *drink*, which begins with a stop and its attendant release, carries on with a continuant, then a vowel, in which the vocal tract is unobstructed, and comes to a close with a nasal and a stop, in which first the oral tract is closed off, and finally the nasal tract as well. However, the theory is straightaway contradicted by a word like *steps*, in which the aperture of /t/ is more restricted than the preceding /s/. Various ad hoc devices have been suggested to deal with such cases.

Under articulatory theories we may distinguish respiratory theories and motor theories.

### Respiratory theories

This theory describes the syllable in terms of the pulmonic air-stream mechanism, and is also known as the chest-pulse theory. According to this theory, when air is released from the lungs during speech, it is not done in a continuous movement, but in a series of puffs, which occur approximately five times per second. Each of these puffs produces a syllable, and they are called *chest-pulses*, *breath-pulses*, or *syllable-pulses*. Some of the pulses are stronger than others, and these produce stressed syllables.

In this crude form the theory is easy to disprove. For example, the English word *better*, although it consists of two syllables, can be uttered with only one chest-pulse (Abercrombie, 1967, p 36).

### Motor theories

The motor theory is a refinement of the respiratory theory, and is due to Stetson (Stetson, 1951). The respiratory theory is concerned with the abdominal chest pulse, as caused by the opposed actions of the rectus abdominis and the diaphragm muscles. As such the movement is controlled and slow, and responsible for what Stetson called the 'breath group' and the 'foot'. The 'foot' is defined as a single stressed syllable or a stressed syllable together with a group of unstressed ones. Stetson's theory is concerned with a different chest movement, as caused by the intercostal muscles. It is a 'ballistic movement', in that it is initiated by muscular action, but uncontrolled until it is arrested at the end. This rapid movement is responsible for the utterance of syllables. These small movements are superimposed on the larger breathing movement like ripples on a wave.

> The syllable is constituted by a ballistic movement of the intercostal muscles. Its delimitation is not due to a 'point of sonority', but to the conditions which define a movement as one movement. In the individuality of the syllable the sound is secondary; syllables are possible without sound. Speech is rather a set of movements made

> audible than a set of sounds produced by movements. (Stetson, 1951, p 33).

The ballistic movement of the chest pulse begins with a release and ends with an arrest. If the releasing is done by the intercostal muscles alone we have a syllable that begins with a vowel. The release may be assisted or modified by the articulators, in which case we have a syllable that begins with one or more consonants. Similarly, a syllable is open or closed according to which muscles are involved in the arrest: the intercostal muscles alone for open syllables, and these assisted by the articulators for closed syllables.

Stetson's theory gives the syllable primacy over consonants and vowels, which is observed also in Abercrombie's book.

His theory has been criticised by Ladefoged (1967), among others. Stetson's experiments were performed in the 1920s, and modern electromyographic studies have shown both that a single chest pulse can give rise to two or more syllables, and that some monosyllables, like *sport* and *stay*, can be produced by two separate bursts (Lehiste, 1970, p 109). Others, such as Fry, acknowledge that the evidence for syllable action should be sought in movement, but find his description of this movement too simple. '[T]he muscles used in speech are so numerous, the interaction of the various systems so complex that we should hardly expect to find syllabification controlled by a single muscle or even the respiratory muscles alone' (Fry, 1964, p 217).

Lehiste, after reviewing Ladefoged's criticism concedes 'However, in connected speech the bursts of intercostal activity correlate fairly well with occurrences of the principal stresses of the utterances' (Lehiste, 1970, p 109). She goes on to cite studies which suggest that there are more general neuromuscular correlates for linguistic units like syllables.

> Lenneberg (1967) makes a good case for assuming that the rhythmic structure of speech is ultimately related to the relatively constant patterns of the electrical activity of the brain, one of which has a frequency of approximately 6 cycles per second. It is surely no accident that this frequency is very close to the frequency with which

syllables are produced in speech (Lehiste, 1970, pp 155–6).

### 4.3.4 Discussion

This brief review of different attempts to define the syllable has left us no wiser. I find myself unable to choose between them, and also unwilling. The attempts described all seek to reduce the syllable to other terms, typically articulatory or acoustic ones. Perhaps we shouldn't try to. Perhaps we should regard it as a purely phonological unit, which is defined by example. This is an unpopular thing to do in modern scientific practice, but not unknown. In chemistry, for example, the term *valency* refers to the tendency of atoms to bond with other atoms. The valency of oxygen is 2 because it combines with two hydrogen atoms to form water. Valency is usually explained by reducing it to physical terms like electron hunger. This explanation is usually successful, but there are circumstances where it doesn't apply. This in no way limits the usefulness of the concept.

Another example is the concept of monetary value in economics (the example is Fodor's (1968)). Monetary value cannot be reduced to a physical phenomenon. Examples of things that *have* monetary value can be given, such as currency, cheques, bags of sugar and motor cars, but the examples themselves cannot even be characterised in a satisfactory way. Nevertheless this does not stop monetary value from having a clear use in rigorous economic discourse.

It may be similar for the concept of a syllable. Our inability to find a physical or other measurable correlate has not prevented the syllable from being a useful linguistic unit, as the large literature on the subject testifies. It may well be that such a correlate does not exsit.

## 4.4 Syllabification

The syllable is hard to define, as the variety of theories about it shows. Definition aside, there is controversy also over how words should be syllabified, particularly in English.

In some cases syllabification follows the etymology of the word. Thus we have *bee-keeper* but *beef-eater*, although the internal structure of both words is -VCV-. Again, we have *tea-tray* but *heat-ray* (both -VCCV-), and *mouse-trap* but *toe-strap* (both -VCCCV-). The syllabification shows how the words are compounded. However, this is not always the case. Some people say *war-drobe*, *teas-poon*, *ea-chother*, and even ignore word boundaries, as when they say *a-tleast* for 'at least', *a-tall* for 'at all' and *thi-safternoon* for 'this afternoon'.

The criteria for dividing English words into syllables are usually sought in the permitted initial and final phoneme sequences of words. The word *anger*, for example, must be syllabified as /a ng - g @/ because /ng/ is permitted at the end of a word and /g/ is permitted at the beginning of a word, whereas /a - ng g @/ is an incorrect syllabification, because there is no word (in RP) which begins with the sequence /ng g/. However, words are not always a guide to syllabification. The word *extra* could be syllabified in three ways, depending on the words chosen as models. Thus we could have /e k - s t r @/, /e k s - t r @/ or /e k s t - r @/ on the basis of *back stroke*, *sex trial* and *next row*.

The question becomes important in chapter 5, when we try to predict syllabic structure from phonetic clues like stop releases. The approach I have taken is to rely on my phonetic intuitions. This is not entirely satisfactory, because phonetic intuitions can be misleading. Also, different speakers vary in the way they syllabify words. We have already seen that *Atlantic*, which is normally syllabified *At-lan-tic*, may have a fricated /l/, in which case it should be be syllabified *A-tlantic*. The same applies to words like *bedroom* and *beetroot*, which, besides their 'traditional' syllabification, can also appear as *be-droom* and *bee-troot* when the /r/ is fricated.

A problem of a different kind is presented in words like *hammer*, *bidding*, *money* and *pony* (the examples are Kahn's (1976)). Is it /h a - m @/ or /h a m - @/, /b i - d i ng/ or /b i d - i ng/? Few people have strong intuitions about these, although they all agree that these are bisyllabic words. One answer is not to insist on a definite boundary between the two syllables, and class the dividing phoneme as *ambisyllabic*. Thus the /m/ in *hammer* is both syllable-final in the

first syllable /h a m/, and syllable-initial in the second syllable /m @/. This is not the solution I have adopted. I have chosen the arbitrary expedient of syllabifying *-ing* words as in /b i d - i ng/ and words like *pony* as /p ou - n ii/.

## 4.5 Is the syllable necessary?

Although the syllable is a venerable concept in linguistics, the question does arise whether it is necessary to a full understanding of the sounds that make up a language. It might be that the sound pattern of a language can be fully described without recourse to the notion of syllables at all. This would make the syllable, theoretically at least, a redundant unit.

The best known modern attempt to do without syllables, and indeed without any phonological units larger than a segment, is *The Sound Pattern of English*, by Chomsky and Halle (1968). The title is usually abbreviated SPE.

Anderson sums up their motivation as follows (Anderson, 1982, p 546).

> The 'classical' model of generative phonology (as presented in e.g Chomsky & Halle ...) recognised only one sort of structural unit in phonological and phonetic representations: the segment. There was thus no explicit provision for syllables (or other units, such as prosodic feet and the like) as significant elements contributing to the organisation of speech. This was not, as some have suggested, simple oversight or failure of imagination, but rather a matter of principle: while traditional phonetic descriptions of course frequently refer to syllable structure, and many informal statements of processes in Chomsky & Halle do so as well, the convenience of this unit for ordinary language description does not ipso facto establish its linguistic significance. If it were to turn out that all the statements we might want to make in terms of syllables were, when expressed formally, representable simply in terms of (strings of) segments, *without important loss of generality*, this would suggest that the more parsimonious and restrictive theory which only allowed

> reference to such units was in fact essentially correct, and thus to be preferred. It was the attempt to establish this programme that lay behind the exclusion of syllable structure from the formalism of early generative phonology.

However, the tide has turned against so spartan an approach. One of the first post-SPE linguists to argue against this was Kahn (Kahn, 1976). He argued that the syllable is an essential constituent for describing allophonic behaviour like flapping, glottalisation, aspiration, and /r/ insertion and deletion. Since then the position of the syllable and other units larger than the segment has become well established. The work of Liberman and Prince, for example (1977), uses units of the syllable and the foot (a unit of a stressed and an unstressed syllable) to explain the tendency in English and many other languages for stressed and unstressed syllables to alternate with each other.

## 4.6 The syllable in speech recognition

The concept of the syllable is as old as the study of language itself. Within the infant field of speech recognition the syllable also makes an early appearance, and work in this field has continued at a steady pace. The present section points out some of the milestones.

As a prehistorical preliminary, so to speak, we may consider the Vicens and Reddy system (Vicens, 1969). The front end used six acoustic parameters to find six kinds of segments; vowels, fricatives, nasals, consonants, stops and transitions. The lexicon was spelled in these terms. It was accessed one syllable at a time — a syllable contained one and only one vowel. A prematch was made against the lexicon using only fricatives and vowels. This yielded a subset of words which were then matched further. The lexicon contained 16 words.

### 4.6.1 ARPA SUR

The achievements of the ARPA SUR systems have already been described, in section 2.2. Here I will focus on their use of syllables and other higher-level

cues. Much of this material comes from (Lea & Shoup, 1979).

Syllabic information was grouped with prosodic features, and their promise was fully appreciated in the planning stages. The ARPA Final Report, which was the basic planning document, envisages phonological rules and stress and intonation rules (Newell *et al.*, 1973, p 24). Furthermore '[s]uprasegmental features such as duration, pitch, and amplitude, exhibit different characteristics if there is a word boundary between segments than if there is not (Lehiste 1970)' (p 26). However, '[t]he main difficulty with [this information] is that [it] is in generative form and their analytic counterparts appear to be much harder to formulate' (p 26). Some of the suprasegmental features made it into the systems that were built, and some did not.

The baby of the four ARPA systems, the SDC system, did achieve the detection of syllable nuclei. Mermelstein's (1975) convex hull procedure was used. This procedure could find 93% of these nuclei. Sperry Univac's algorithm for the same thing (Lea, 1976) located 91% of them. However:

> Studies at SDC and SCRL suggested that syllable *boundaries* are too difficult to reliably locate to be used in recognition schemes, and in many phonological rules the syllable boundaries could be removed or ignored without altering the effects of the rule (Hanson et al, 1976). We still await hard evidence that phonetic constraints are less across syllable boundaries than within syllables, even though such evidence, if available, would help justify syllables as units in speech analysis (Lea & Shoup, 1979, p 77).

ARPA SUR yielded over 200 phonological rules. These were applied either before lexical lookup, or after, or the rules were precompiled as in Harpy. Most of the rules included boundary information for morphemes, syllables or words, 'though studies were done to see which rules could be rewritten without boundary considerations' (Lea & Shoup, 1979, p 79).

Algorithms were available for phrase boundary detection (Lea, 1973a), stressed syllable detection (89% of them) (Lea, 1973b; Lea & Kloker, 1975), and speech rate detection (Bernstein *et al.*, 1976). Also, '[p]rosodic aids to parsing [were]

promising but largely untested additions' (1979, p 76). However, Lea and Shoup's verdict is that the impact of these methods was small:

> In general, while the acoustic data and parameter extractions were available for determining important prosodic features within each of the systems, prosodic features played only a minimal role in the final systems'. (Lea & Shoup, 1979, p 81).

## 4.7 The Convex Hull Algorithm

Mermelstein's syllable segmenter (Mermelstein, 1975), mentioned above, uses a convex hull algorithm. Sufficiently deep local minima in energy are candidates for syllable boundaries. These candidates then have to satisfy criteria of length, loudness and degree of voicing. The results are 90% accurate according to Mermelstein, but the syllables are not always the conventionally defined ones. The segmenter has been incorporated in several systems, such as (Mertens, 1987), and also by Mermelstein himself, in (Hunt *et al.*, 1983). He and his co-authors state boldly,

> The energy profile of a speech signal shows a modulation due to the syllable structure. This may make syllables the only elements of speech that can be consistently isolated independently of recognition (Hunt *et al.*, 1983, p 168).

By 'independently of recognition' the authors mean that it is possible that syllables can be found on the basis of acoustic evidence alone, without the help of lexical and syntactic processing. Even so they find that 'the acoustic form of a syllable is not independent of its context'. In their system, for example, the words *ninth October* are syllabified *ninth oct*, *nin thoct* and *ninth thoct.* They do not explain how the spurious candidates are removed without the help of higher-level processing.

The system works as follows. It processes the signal to produce mel-scale cepstrum coefficients. The syllables are defined in terms of these, and matched left to right. A beam search is used with a beam width of 300. Such a large

beam width only cost 20% more syllable matches than a beam width of 1, but reduced the error rate five-fold. Syllable matching is done under the control of a simple syntax. The syntax allows variant definitions. When a weak fricative is omitted, for example, the system obtains *nine oct* for *ninth oct.* This is solved by giving *ninth* a 'deep structure' of both *ninth* and *nine.*

## 4.8 Church's System

Church's advocacy of syllables in speech recognition has been reported already in chapter 2 (section 2.7). Instead of decrying allophonic variation in the phoneme lattice, as many speech workers did, he welcomes it, because it provides information about suprasegmental units like the syllable. Work had been going on for many years in the field of human perception, to do with the various phonetic and other cues to word and syllable boundaries (e.g (Nakatani & Dukes, 1977)), and Church gave computational expression to it.

Church's proposals are contained in his PhD thesis (Church, 1983). A large part of Church's thesis deals with theoretical issues in linguistics. He conceived of allophones as arising from underlying phonemes by a phonological process of alteration. He sought to describe these effects in terms of rules, which were devised by a human expert. This was a deliberate policy:

> [W]e will attempt to develop explicit models of allophonic processes, rather than acquiring the rules through training. This has a number of practical advantages (e.g speaker independence, reduced training, scales up with the number of allophonic distinctions) as well as the theoretic advantage of providing falsifiable models of grammar (Church, 1983, p 32).

This concern for linguistic theory runs through most of the work. It causes him, in my view, to overlook the difficulties that arise when it comes to implementation, and his detailed proposals are impractical. However, the general observation that allophones are useful and not a hindrance remains valuable, and this thesis owes its existence to it.

In outline, his system works as follows. The input consists of a string of phones, drawn from an inventory of 45. In addition to identifying the parent phoneme, the phones are marked with the following eight phonetic features (Church, 1983, p 54).

| | | | |
|---|---|---|---|
| stress | glottalisation | glide | rounding |
| aspiration | lengthening | nasalisation | unrelease |

The allophones are constructed into syllables using a chart parser with a phrase-structure grammar. Typical rules of the grammar are (p 187):

/h/ is syllable-initial

unreleased stop is syllable-final

glottal stop is syllable-final

released stop is syllable-initial

The rules produce a lattice of syllables, 'rarely more than four' deep (p 43). Syllables are hierarchical structures, of which an important intermediate level is the *sylpart*, a term which covers onset, peak and coda. The constituent allophones in the syllables are then *canonicalised* to phonemes, after which the structures are matched against a lexicon. The lexicon is organised on two levels, with sylparts forming syllables, and syllables forming words.

Church's proposals have not, to my knowledge, been implemented in a real system. Church's own software used hand transcriptions as input. There were no multiple segmentations, and no multiple candidates per segment: just a string of error-free allophones. Church recognised that this is unrealistic, and devotes a separate chapter, 'Robustness Issues' to what needs to be done if the phoneme lattice comes not from a hand transcription but from a recogniser. However, the way he goes about it is to introduce a small number of deliberate errors, which I find unconvincing.

I think Church was misled by his input into thinking that suprasegmental units could be derived purely from bottom-up allophonic cues. In fact, he went so far as to forego syntactic information altogether (p20). This thesis does not

follow that approach. It uses allophonic cues and suprasegmental units (viz syllables) too, but the syllables are not constructed bottom-up. In the front end they are looked up in a repository of predefined syllables. The details are given in the next two chapters.

Church makes the case for allophonic cues by pointing out that they can be used to derive suprasegmental constituents. These are then looked up in a lexicon spelt in these constituents. The question arises why this is a helpful thing to do. Church does not answer it, but other people have, and we turn to one of them in section 4.10.

## 4.9 SYLK

SYLK, which stands for 'statistical syllabic knowledge', aims to combine statistical matching with phonetic knowledge in a continuous speech recognition system. It was developed at the University of Sheffield. The following brief details are based on (Green *et al.*, 1990) and (Green *et al.*, 1992). SYLK uses HMM models of onset types and coda types to do the initial segmentation and labelling. Similar units are also used elsewhere: the Spanish work reported in (no *et al.*, 1989) and (Lleida *et al.*, 1991) uses demisyllables. The difference with SYLK is in their use of refinement tests of the spotted demisyllables; the tests are based on knwledge-based techniques, and incorporate arbitrary phonetic insights in a statistically admissible way.

## 4.10 Lexical studies

A different class of work looks at how syllables can reduce the search space during lexical retrieval. When vocabularies become large, the words become more confusable, and the space of words to be searched when the phoneme lattice is matched against the lexicon grows. This leads to a reduction in recognition accuracy. Waibel (1988) summarises the evidence as follows.

> In a recent study, Lee, Silverman and Dixon (1984) have studied the relationship betweeen the confusability of a vocabulary and its size.

> Their results indicate that there is a steady increase in confusability with increases in vocabulary size. Particularly interesting was the fact that confusability appears to grow more dramatically when vocabulary size exceeds the 1000 word limit. Performance figures by Smith (1977) show that the most dramatic drop in recognition performance of large vocabulary recognition systems seems to occur when vocabulary sizes increase to up to 8000 words.

Waibel himself performed various experiments with large vocabularies to discover ways of reducing the search space. I review those that involve the use of syllables.

The problem that the experiments address is that the phoneme lattice from a typical front end is full of errors, and when a string of phonemes from it is matched against the lexicon, many word candidates are obtained. If the search can be confined to a subset of the lexicon, then the number of candidates is reduced and the recognition accuracy improves. The experiments were performed on the 20,000-word Webster's dictionary, which is large by speech recognition standards. The aim is to to reduce the number of words that needs to be searched when the phonemes are matched. Suppose, for example, that in addition to the phoneme string, the number of syllables in the string is known as well. This extra information can be used to preselect from the dictionary only those words that have the right number of syllables. Such a selection is called a cohort. The size of the cohort will vary with the number of syllables; for example, in the Brown corpus (Kucera & Francis, 1967) of one million words almost 40% of the words are of two syllables, and only about 5% are of five syllables. The expected cohort size $ECS$ is the average number of words in a cohort selected in this way.

$$ECS = \sum s_i * p_s(s_i)$$

where $s_i$ ranges over the sizes of the possible cohorts, and $p_s(s_i)$ is the probability of a word's falling into a cohort of size $s_i$ (i.e the relative frequency of

words in this cohort). The expected cohort size takes into account the probability that any particular cohort size will occur. For the 20,000 word vocabulary mentioned, the $ECS$ for syllable counts is 5013, a reduction to 25%.

Another experiment looked at syllable durations. A syllable was considered to stretch from vowel nucleus to vowel nucleus. This left some word-initial consonant(s) stranded, and they were ignored. Word-final syllables, which tend to be longer, were normalised by subtracting 90 msec from their length. This can of course only be done with syllables that you know to be word-final. Syllables were now classified as long, medium and short for polysyllables, and long and short for monosyllables. The syllables of the words in the dictionary were labelled with the symbols H (high duration), M and L (low duration). 362 different patterns were found. The $ECS$ is 1249, or 6% of the dictionary.

A further study was done on the balance of voiced and unvoiced material in a syllable. A syllable that was all voiced received the label H (high). Syllables with both are labelled M (medium) or L (low) according as voicing preponderates or not. A syllable like *six*, which contains a lot of unvoiced frication, is labelled L. The results are 352 cohorts, with an $ECS$ of 909 (4.5%).

## 4.11 Summary and discussion

Despite the lack of a rigorous definition, the syllable is a widely used unit in linguistics. After a short period of obscurity in the wake of SPE, the syllable and other large units are once more respectable theoretical units.

In speech technology also, interest in the syllable continues unabated. Syllables have found two areas of application in speech recognisers: the front end and the back end. Most front end work is concerned with the direct recognition of syllables from the input signal, or of aspects of them like their nuclei or their number. In the back end we have looked at a study that tries to use this information to reduce the lexical search space.

This thesis incorporates syllables in both the front end and the back end. In both of these areas the approach taken is unconventional. In the front end syllables are not recognised from the signal, but looked up in a store which con-

tains all the syllable patterns which are expected to be spoken. It is phonemes that are recognised from the signal, and these are matched against syllables in order to remove sequences of phonemes that do not form valid syllables.

In the back end syllables are used to obtain better statistics about phoneme confusions.

# Chapter 5

# Syllable Experiments in the Front End

## 5.1 Introduction

We have seen in earlier chapters that the fundamental unit of recognition of a continuous speech recognition system is the phoneme, but that the choice of this unit is problematical because phonemes have different acoustic realisations in different settings. The problem is usually addressed by considering phonemes in their context. There are different ways of doing this, as we saw in chapter 2. One way is to use triphones. Another way is to constrain the generation of phonemes so that only those phonemes are generated which form part of valid words. These two methods are normally combined, as in the SPHINX system (Lee, 1988).

Integrated systems like SPHINX, which use high-level (i.e lexical and syntactic) constraints at the front end, pay a price for doing so: loss of flexibility. Phoneme, word and syntax models are all combined into a single multi-level Markov model, and as these systems grow in vocabulary it becomes expensive to add the new words and phrases.

The remedy proposed in this thesis is a loosely-coupled system, in which high-level constraints can be applied at the front end without sacrificing flexibility. The high-level constraints proposed here are syllables. Syllables, being

fixed in number, offer a source of constraint that is not dependent on the fluctuating contents of the lexicon. The lexicon and the grammar remain separate in the back end, so that any change in these components does not disrupt the whole system.

The use of syllables, and certainly their advocacy, is not new, as we saw in chapter 4. One proposal was marked out for particular attention, and that was the work of Church (Church, 1983). Section 4.8 described how the use of allophonic information was to help with the location of syllable boundaries. Church's was a theoretical study, because he did not have a front end that was good enough to provide him with the allophonic hypotheses he needed. One of the subsidiary questions in this chapter is whether front ends are good enough these days, and if they are, whether the expected benefits can be obtained.

The main question to be addressed is whether syllables are an effective contraint on the phonemes that are generated at the front end, in the way that words are in integrated recognition systems. This question is broken down into several subsidiary questions, as follows. In order not to keep the reader in suspense, I give abbreviated answers as well.

1. Can a modern front end using hidden Markov models provide allophones of high quality? We shall see that in some cases an enriched allophone set can give lattices of a better quality than a standard phoneme set.

2. Are the allophones effective in helping to locate syllable boundaries? The answer is, only marginally better than a standard phoneme set, although more training data may increase the difference.

3. What is the difference in performance, in a loosely-coupled system, between using syllables rather than words as a constraint on segmentation? *Given a loosely-coupled system*, syllables give better results than words.

4. What is the effect of repairing the segmentation before syllable constraints are applied? We shall find that segmentation repair, under the restricted conditions of a loosely-coupled system, has an adverse effect on performance.

| *Database* | *Phonemes* | *Syllables* | *Words* | *Phonemes per* | | *Syllables* | *Words* |
|---|---|---|---|---|---|---|---|
| | | | | *Syllable* | *Word* | *per utterance* | |
| atr | 8714 | 3335 | 2601 | 2.61 | 3.35 | 16.7 | 13.0 |
| cyt | 5226 | 2009 | 1204 | 2.60 | 4.34 | 11.8 | 7.1 |

Table 5.1: Database statistics for speaker GSW.

## 5.2 The Data

The experiments have been performed on two data sets for one male speaker, with some results also for three more male speakers. The main speaker is identified as GSW, who speaks with an accent close to RP. The other three speakers are HXB, PMS and JMR. The first two of these are from the north of England, and the third is from the south.

The first data set consists of 200 phonetically balanced sentences. They were originally recorded and labelled under contract to Advanced Telecommunications Research Institute International, of Kyoto in Japan. The data are used for academic research purposes by kind permission of the Institute. The data set will be identified as ATR, and comprises just over 11 minutes of speech. For GSW the 200 sentences were hand-segmented and labelled to phonetic level by qualified phoneticians. This data set was used for training models, and for producing closed test results. The second data set, identified as CYT, consists of 170 sentences and phrases used in a cytology laboratory, which comprises just over 9 minutes of speech. This data was transcribed to phonemic level but not segmented — that is, the phonemes that were spoken were written down, but not their start times and end times. This means certain experiments could not be performed on this data. It was used to provide the open test results for most of the experiments.

Table 5.1 provides statistics on some of the linguistic units in the two databases. Cytology has more phonemes per word because of the many long medical words like *bloodstained*, *epithelial* and *lymphocytes*.

The vocabulary size for ATR is 1242 and for CYT is 242. Although vocabulary size is commonly quoted as an indication of recognition difficulty, it is a less important indicator than confusions among similar words. Unfortunately, no

generally accepted measure exists for this.

The data for the other three speakers are less reliable. No segmentations or transcriptions are available for these speakers. The data for them was obtained as follows. A *general transcription* of the utterances was made first, based on the transcription for GSW. Table 5.2 shows the transcription of the sentence 'Some debris is present', which comes from the cytology data. The top line is the sentence as spoken by GSW, in an RP accent. A program converted this to a general transcription, which allows multiple readings, and this is shown on the lines below. The first vowel /uh/ is rendered as one of the vowels /uh, @, u/. The possibility /@/ allows for the case where vowel reduction has taken place. The possiblity /u/ is the vowel used in parts of the north of England for the /uh/ of RP. Recall that speakers HXB and PMS have traces of a northern accent. The second vowel also has two renderings in the general transcription, and these are independent of the renderings of the first vowel. The /y/ in square brackets is an optional glide between the neighbouring vowels of *debris* and *is*. The /z/ in *present* is rendered either as /z/ or as /s/.

The general transcription is then given to the segmenter, which segments the utterance accordingly, using the models trained on GSW. Where there are more than one possibility for a segment, they are all tried, and the highest-scoring one chosen. The outcome is a set of segmentations, which are declared to be an accurate representation of what was spoken. They are clearly not as good as the hand segmentations that are available for speaker GSW, for several reasons. We cannot be certain that they contain all and only the phonemes that were spoken, because idiosyncracies of pronunciation and minor slips of the tongue are not provided for. The segment boundaries — the start and stop times — are also likely to be inaccurate, because they were obtained by models trained for a different speaker. Finally, the segmentations are only at the phonemic level, whereas the hand segmentations of GSW included indications of stop releases, among other features.

The segmentations derived from the general transcriptions are used for speech encoding and model training in the normal way.

```
s uh m d e b r ii     i z p r e z @ n t
s uh m d e b r ii [y] i z p r e z @ n t
  @      @                     s
  u
```

Table 5.2: Std transcriptions of 'Some debris is present'. First line, transcription for speaker GSW. Subsequent lines, general transcription.

## 5.3 The apu sets

We saw in chapter 2 that the distributions of some allophones correlate well with their position in a syllable. For instance, released stops and light /l/ more often fall at the beginning of a syllable than at the end. To capitalise on this fact a number of apu sets were designed whose members depend in various ways on their position in the syllable. These sets are as follows.

**std** 45 apus. Standard RP phoneme set.

**stdp** 49 apus. As std, plus the four syllabic consonants /l=/, /m=/, /n=/, /r=/.

**ext02** 128 apus. Syllable-conditioned consonants. This means consonants are defined according to their position in the syllable. The /p/s in *print* and *sprint*, for example, are defined as two different allophones, as are the /s/s in *spat* and *pats*. Total 36 stop allophones, 10 /s/ allophones, 5 /z/ allophones, etc.

**ext03** 137 apus. Syllable-conditioned consonants as ext02, but stops further divided into released and unreleased. We shall say that ext03 has *syllable-conditioned acoustic stops*. Total 45 stop allophones (not $36 \times 2 = 72$ as one might expect, because some classes had to be combined to obtain enough tokens for training).

**ext04** 104 apus. Syllable-conditioned consonants as ext02, but instead of syllable-conditioned stops, stops are divided into released and unreleased only. We shall say that ext04 has *acoustic stops*. Total 12 stop allophones.

**ext05** 79 apus. Syllable-conditioned stops only, identical to those in ext02. Total 36 stop allophones. The difference from *ext02* is that the other apus are not syllable-conditioned; they are as those in stdp.

**ext06** 55 apus. Stops divided into released and unreleased (not syllable conditioned; another example of acoustic stops). Total 12 stop allophones. The difference from ext04 is that the other apus are not syllable-conditioned; they are as those in stdp.

**ext07** 65 apus. As ext05, but with some stop categories combined. 22 stop allophones.

**ext08** 72 apus. As ext07, but with the stops recombined into new classes. 29 stop allophones.

The number of apus in each case includes the silence symbol. 'Ext01' was a first attempt at an apu set whose design was embarrassingly inept, and which has been omitted from the thesis. In all the other sets the vowels and diphthongs are the standard ones used in RP. Full definitions of the apu sets can be found in appendix B.2.

## 5.4 Measures of quality

The results that follow report the outcomes of the following kind of experiment. First the ATR data are labelled with a new apu set. This was done mainly by computer, by translating the std hand segmentation into the appropriate new symbols. Table 5.3 below shows the std and ext04 transcriptions of the sentence 'The price range is smaller than any of us expected'. The std transcription comes from the hand segmentation, and the ext04 transcription was produced by a program, which replaced one symbol by another, paying due regard to the syllables in which the original symbols appear. For example, /r2/ is the post-stop allophone of /r/ in the ext04 set, and /u-k/ is a post-vocalic unreleased /k/. /##/ is the silence symbol.

```
## dh  @ p  r  ai s  r  ei n  jh  i z  s  m  oo l  @
## dh1 @ p1 r2 ai s5 r1 ei n6 jh3 i z2 s4 m1 oo l1 @
```

the price range is smaller

```
dh  @ n  e n  i @ v  @ s  @ k  s  p  e k  t  i d
dh1 @ n2 e n1 i @ v2 @ s5 @ k1 s2 p1 e u-k t1 i d1
```

than any of us expected

Table 5.3: Std (first line) and ext04 (second line) transcriptions of 'The price range is smaller than any of us expected'. Notice how the std phoneme dh corresponds to the ext04 allophone dh1, and so on.

Next HMM models are trained on the newly labelled data. The segmenter is run with the new apu set, for both closed and open test. Next the classifier is run on the computer segmentations, with this or another apu set. There are a few variations on this theme, which are best explained when we come to them.

Three measures are used to assess the quality of the segmentation and the classification. These are as follows.

### End-point differences (epds)

The segmenter can be asked to segment according to the hand transcription. In this operation the segmenter is given a list of the apus that were spoken, in order. Its task is to match the relevant models against the speech signal and discover the start and end points of the segments. It produces exactly the same number of segments as are in the hand transcription. The only models used during the segmentation are the ones it has been given, and there is no effect due to the number of models in the set. The segments produced from such a run are compared with the hand segmentation, and the absolute differences between their end-points, measured in milliseconds, are accumulated.

This measure can be used only on the ATR data, because we have no hand segmentation for cytology.

| | |
|---|---|
| apu set 1 | group 1 |
| apu set 2 | group 1 |
| apu set 3 | group 1 |
| apu set 4 | group 2 |
| apu set 5 | group 2 |
| apu set 6 | group 2 |
| apu set 7 | group 2 |

Table 5.4: Illustration of grouped results. Rankings within a group are not statistically significant. Across groups the rankings are significant.

### Oversegmentation rate

The segmenter always produces more segments than the number that were spoken; that is, it oversegments. The oversegmentation rate is the percentage of extra segments over the hand segmentation that the segmenter produces.

### Entropy

The entropy of a classification run, using a variety of apu sets. The concept of entropy in general is discussed in appendix B, and its application to phoneme lattices is described in chapter 3.

The results of the experiments to be reported have been tested for significance using the 't' test, and are presented in tables. The lines of the tables are given in rank order and are indented to show groups whose results are different to a statistically significant degree (90% confidence level). Table 5.4 illustrates what is meant. The indentations show that the results fall in two groups. The three sets in group 1 are ranked in order, with the best set at the top. However, none of the sets in the group is significantly different from the others. Likewise the sets in group 2 are not different to a statistically significant degree among themselves. Across groups the difference *is* significant; some members of group 1 are significantly better than some members of group 2. This is not necessarily true of every member. It could be, for example, that apu set 3 and apu set 4, which are closest in rank across the groups, are not statistically significant from each other. What is true is that set 3 is closer to set 2 than to 4, and 4 is closer to 5 than to 3.

| *Set* | *epd (ms)* | *No of apus* |
|---|---|---|
| ext03 | 346 | 137 |
| ext02 | 354 | 128 |
| ext04 | 367 | 104 |
| ext05 | 386 | 79 |
| ext08 | 389 | 72 |
| ext07 | 394 | 65 |
| ext06 | 400 | 55 |
| stdp | 414 | 49 |
| std | 420 | 45 |

Table 5.5: Average end-point differences (epd) between hand segmentations, and machine segmentations given the transcription. Speaker GSW, ATR data.

## 5.5 Recognition of different apu sets

### 5.5.1 End-point differences

Table 5.5 gives the average end-point differences between the hand segmentations and the segmentations the machine produces when it is given the transcriptions. The results are from a closed test run on the ATR data. Consider the last line in the table, which is for the std apu set. This apu set contains 45 symbols, which is given in the last column. For each of the 200 utterances the segmenter is given the transcription; thus, for the first utterance it knows it must produce segments to correspond with `## dh @ p r ai s r ei n jh i z s m oo l @ dh @ n e n i @ v @ s @ k s p e k t i d` (*the price range is smaller than any of us expected*). Under this arrangement the segmenter produces exactly the right segments for each utterance, except that they begin and end at slightly different times. The interval by which each machine segment is offset from the hand segment is calculated (it is reckoned as a positive number), and the total reported for each utterance. The number in the table is the average of the 200 utterance totals, and in the case under discussion is 420 ms.

The results correlate perfectly with the number of apus. Ext03, with the largest number of apus, has the smallest average end-point difference, i.e its segments align most closely with the hand segments. The more apus there are, the more accurate is the segmentation. This is because a large set has more

| *Set* | *% oversegm* | *No of apus* |
|---|---|---|
| stdp | 3.1 | 49 |
| std | 3.5 | 45 |
| ext07 | 6.1 | 65 |
| ext06 | 6.3 | 55 |
| ext08 | 6.5 | 72 |
| ext05 | 7.1 | 79 |
| ext04 | 9.5 | 104 |
| ext02 | 9.6 | 128 |
| ext03 | 10.7 | 137 |

Table 5.6: Percent oversegmentation. Speaker GSW, ATR data.

specific models. What is called /t/ in std, for example, is called any of /t t2 t3 t4 t5 t6 t7 t8 u-t1 u-t2 u-t3 u-t4/ in ext03. This is encouraging in a small way. It means that the apu sets are not badly chosen, and that for the purpose of this exercise there is enough training data to adequately train at least 137 models. However, we could wish for a more sensitive indicator of performance. We know that the sets were chosen according to incompatible criteria. For example, ext03 has syllable-conditioned acoustic stops, and ext04 and ext06 have acoustic stops only. If these different criteria lead to different effects, they do not show up here. The most important factor is the number of models.

End-point differences for the other speakers cannot be obtained, because there are no hand segmentations to compare against.

### 5.5.2 Oversegmentation

Tables 5.6 and 5.7 show the oversegmentation rates for the different apu sets for GSW, for closed test and open test conditions respectively.

The results correlate approximately inversely with the number of apus. This is the opposite of what we obtained with end-point differences above. There larger sets gave good results, and here larger sets give worse results. The smaller sets do better because fewer apus are less competition for each other, which results in less fragmentation. However, there is an interesting exception. Std has fewer models than stdp, but performs significantly worse than stdp. The extra models in stdp are the syllabic consonants, and for this speaker at

| *Set* | *% oversegm* | *No of apus* |
|---|---|---|
| stdp | 16.8 | 49 |
| std | 17.5 | 45 |
| ext06 | 20.0 | 55 |
| ext07 | 20.4 | 65 |
| ext08 | 21.0 | 72 |
| ext05 | 22.1 | 79 |
| ext04 | 24.0 | 104 |
| ext02 | 25.6 | 128 |
| ext03 | 26.8 | 137 |

Table 5.7: Percent oversegmentation. Speaker GSW, CYT data.

least are worth it.

Figures 5.1 and 5.2 show graphs of the oversegmentation data. As can be seen, ext06 and ext04 also perform worse than the general trend suggests. The poor performance of ext06 is particularly obvious in the case of ATR: it has fewer models than ext07, its nearest competitor, yet performs worse. Ext06 and ext04 both have acoustic stops. That these should cause adverse behaviour is an interesting result, which will be explored in section 5.9 below.

The oversegmentation rates for the other speakers are not very illuminating. Recall from section 5.2 that the data for these speakers were not hand segmented, and a machine segmentation made from a general transcription was used for training models. The accuracy of the models must suffer accordingly. Another consequence is that not all the apu sets can be obtained for these speakers, since some of them rely on fine phonetic transcriptions to the level of stop releases, which cannot be reliably obtained by machine. Only four apu sets could be defined, namely std, stdp, ext02 and ext05.

Tables 5.8 and 5.9 give the closed-test and open-test oversegmentation rates for the other speakers. The trend which shows that the oversegmentation rate increases for apu sets with more models, is confirmed. For the closed test the oversegmentation rates are worse than GSW for every apu set for every speaker, which confirms that the models are of a poorer quality. In the case of open test, however, the results for speaker PMS are better than for GSW. I do not know why this is so.

We see that although stdp was a significantly better set than std for GSW,

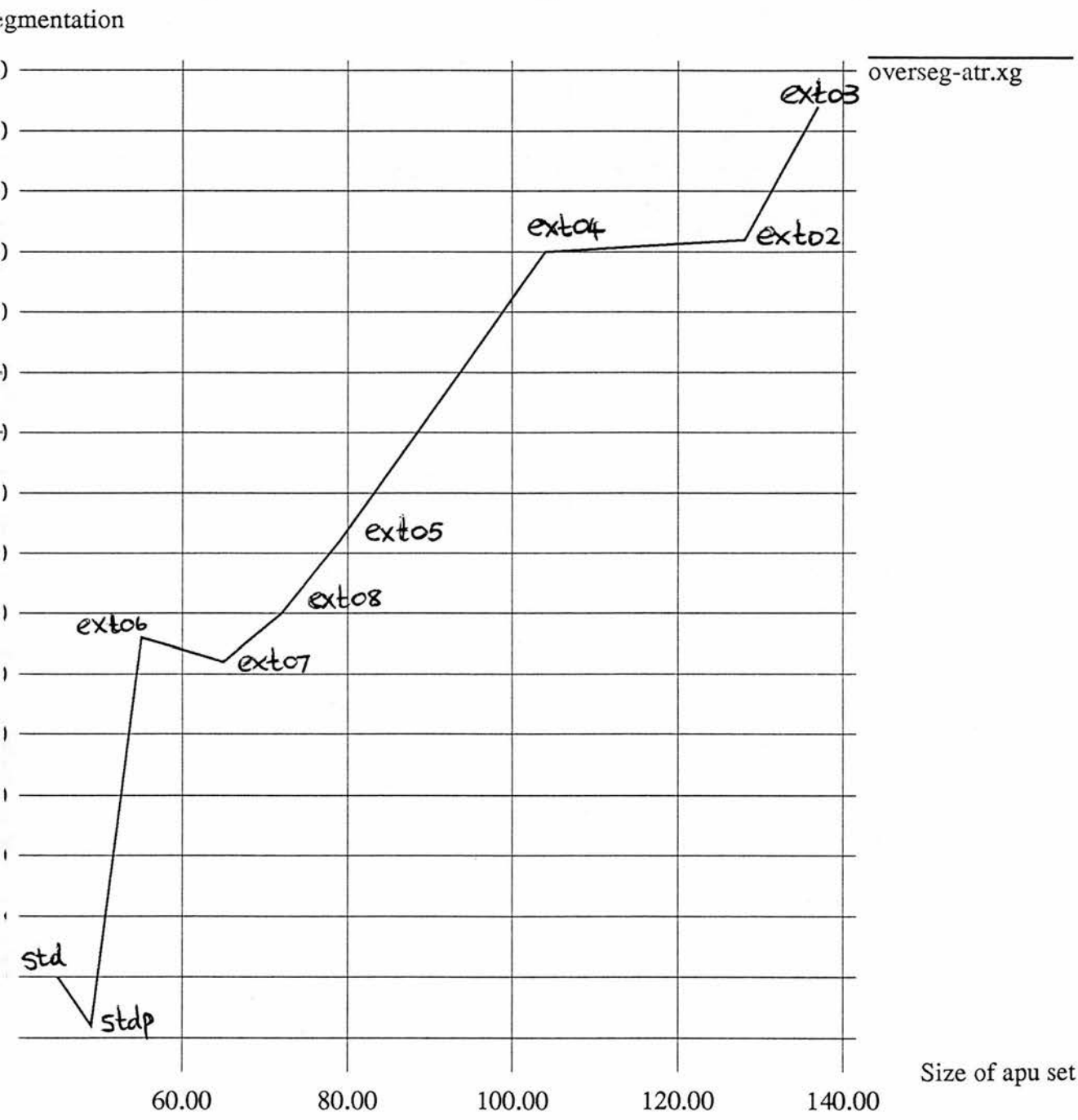

Size of apu set versus oversegmentation. ATR data
egmentation
overseg-atr.xg
ext03
ext04
ext02
ext05
ext08
ext06
ext07
std
stdp
Size of apu set
60.00
80.00
100.00
120.00
140.00

Figure 5.1:

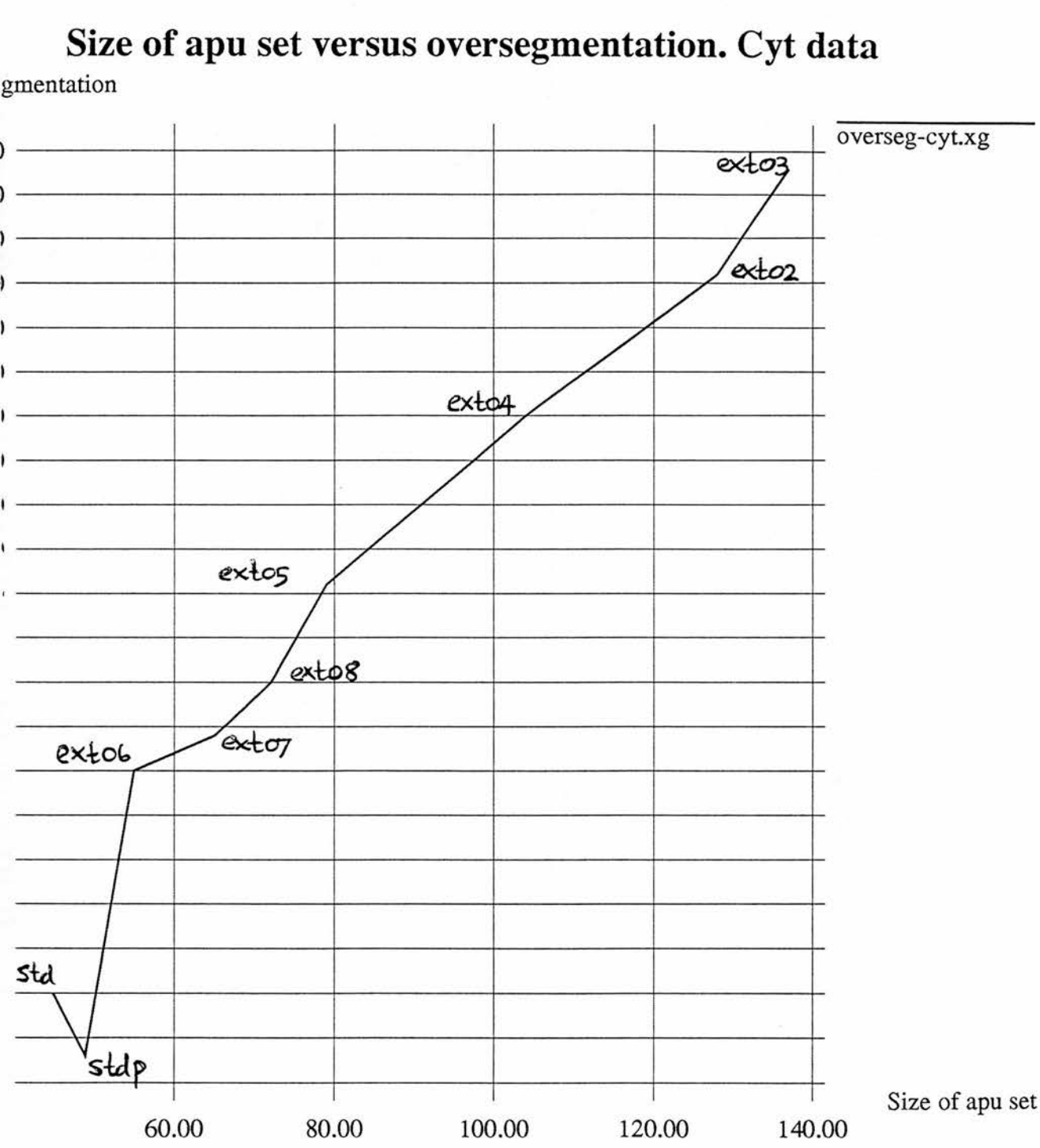

Size of apu set versus oversegmentation. Cyt data
gmentation
overseg-cyt.xg
ext03
ext02
ext04
ext05
ext08
ext07
ext06
std
stdp
Size of apu set
60.00
80.00
100.00
120.00
140.00

Figure 5.2:

| *Set* | *% oversegm* | *No of apus* |
|---|---|---|
| **hxb** | | |
| std | 6.8 | 45 |
| stdp | 6.7 | 49 |
| ext05 | 8.3 | 79 |
| ext02 | 11.7 | 128 |
| **pms** | | |
| std | 4.9 | 45 |
| stdp | 5.3 | 49 |
| ext05 | 7.7 | 79 |
| ext02 | 10.2 | 128 |
| **jmr** | | |
| std | 9.9 | 45 |
| stdp | 10.2 | 49 |
| ext05 | 12.6 | 79 |
| ext02 | 15.6 | 128 |

Table 5.8: Percent oversegmentation. Other speakers, ATR data.

this is not so for any of the other speakers. In the case of HXB this can perhaps be explained by the small number of training examples for stdp's extra apus, which are the syllabic consonants /l= m= n= r=/. It has only 14 of these consonants, whereas GSW has 68. However, the other two speakers have 80 and 92 respectively, and so in their case this is not the answer. The explanation probably lies in the poor quality of the models.

### 5.5.3 Entropies

A further indication of the quality of a segmentation might be obtained by running classifications on it. Tables 5.10 and 5.11 show the results of performing classifications on the different segmentations. To ensure a fair comparison, the same apu set, std, is used for the classifications.

Although the results for ATR and CYT look different, what can be discerned is that the apu sets which oversegment worse have a better entropy, and those which oversegment better have a worse entropy. Sets ext05, ext06, and ext07 all lower their rank when we go from segmentation quality to entropy. Sets ext02, ext03 and ext04 all increase in rank: they segment well but classify poorly

| *Set* | *% oversegm* | *No of apus* |
|---|---|---|
| **hxb** | | |
| std | 27.5 | 45 |
| stdp | 27.5 | 49 |
| ext05 | 29.7 | 79 |
| ext02 | 34.2 | 128 |
| **pms** | | |
| std | 16.5 | 45 |
| stdp | 16.6 | 49 |
| ext05 | 18.9 | 79 |
| ext02 | 23.0 | 128 |
| **jmr** | | |
| std | 21.2 | 45 |
| stdp | 21.3 | 49 |
| ext05 | 24.7 | 79 |
| ext02 | 28.2 | 128 |

Table 5.9: Percent oversegmentation. Other speakers, CYT data.

| *Set* | *Ph Entropy* | *No of apus* | *Oversegmentation ranking* |
|---|---|---|---|
| ext02 | 2.78 | 128 | stdp |
| ext04 | 2.78 | 104 | std |
| ext03 | 2.79 | 137 | ext07 |
| stdp | 2.81 | 49 | ext06 |
| std | 2.81 | 45 | ext08 |
| ext07 | 2.83 | 65 | ext05 |
| ext08 | 2.83 | 72 | ext04 |
| ext06 | 2.84 | 55 | ext02 |
| ext05 | 2.84 | 79 | ext03 |

Table 5.10: Ranked entropies of std classifications on different segmentations. The right hand column shows the ranking according to the oversegmentation rate, for comparison. Speaker GSW, ATR data.

| *Set* | *Ph Entropy* | *No of apus* | *Oversegmentation ranking* |
|---|---|---|---|
| std | 3.30 | 45 | stdp |
| stdp | 3.31 | 49 | std |
| ext04 | 3.32 | 104 | ext06 |
| ext06 | 3.34 | 55 | ext07 |
| ext03 | 3.36 | 137 | ext08 |
| ext02 | 3.36 | 128 | ext05 |
| ext08 | 3.36 | 72 | ext04 |
| ext05 | 3.37 | 79 | ext02 |
| ext07 | 3.37 | 65 | ext03 |

Table 5.11: Ranked entropies of std classifications on different segmentations. The right hand column shows the ranking according to the oversegmentation rate, for comparison. Speaker GSW, CYT data.

(ext08 stays the same for ATR and lowers its rank for CYT). We see that extra segments in the lattice boost the entropy. For ATR, std and stdp also do worse on the entropy score, but for CYT this is not so. I cannot discover a reason for this.

The reason why oversegmentations do better on the entropy scores has to do with the way the entropy is calculated. Recall from section 3.3 that lattices which need insertions tend to be more expensive (have a higher entropy) because they contribute terms to the entropy expression which are outside the duration adjustment. An oversegmented lattice needs fewer insertions, and that is why the more heavily oversegmented lattices have a better entropy. This is an unfortunate effect, because it makes entropy an inappropriate measure for comparing the quality of different segmentations.

### 5.5.4 Classification results

The previous section considered the effectiveness of the different apu sets for segmentation. We now turn to their effectiveness for classification. We investigate this by performing different classifications on a fixed segmentation. In the case of ATR both the hand segmentation and a machine segmentation using stdp were chosen for this, and for CYT, in the absence of a hand segmentation, only a stdp machine segmentation was chosen. As we saw in section 5.5.2, stdp

| *Set* | *Ph entropy* | *No of apus* |
|---|---|---|
| std | 1.86 | 45 |
| stdp | 1.88 | 49 |
| ext06 | 1.95 | 55 |
| ext07 | 2.02 | 65 |
| ext08 | 2.05 | 72 |
| ext05 | 2.09 | 79 |
| ext04 | 2.37 | 104 |
| ext02 | 2.49 | 128 |
| ext03 | 2.52 | 137 |

Table 5.12: Classifications performed on a hand segmentation. Speaker GSW, ATR data.

| *Set* | *Ph Entropy* | *No of apus* |
|---|---|---|
| ext05c | 1.84 | 79 |
| ext08c | 1.84 | 72 |
| ext07c | 1.84 | 65 |
| ext06c | 1.85 | 55 |
| ext02c | 1.85 | 128 |
| stdpc | 1.85 | 49 |
| std | 1.86 | 45 |
| ext03c | 1.86 | 137 |
| ext04c | 1.86 | 104 |

Table 5.13: Classifications performed on a hand segmentation. Lattice conflated to std phonemes. Speaker GSW, ATR data.

provides the best segmentation that can be obtained by machine.

Table 5.12 shows different classifications performed on a hand segmentation. The results correlate perfectly with the number of symbols. This is not an interesting result, because we are seeing an effect that is due to the number of apu targets. There is more to go wrong when the number of targets is large, and so the small apu sets have an unfair advantage.

We can compensate for the number of targets by conflating the phoneme lattices down to the standard set. We first classify using the full apu set. Then the lattice is collapsed down to the std set. Thus, in the case of ext03, /t1 t2 t3 t4 t5 t6 t7 t8 u-t1 u-t2 u-t3 u-t4/ in the lattice are collapsed down to just /t/, and so also for the other apus. Next the entropy is calculated. The scores in the lattice are adjusted accordingly, and the entropy calculation is done with reference to the std set.

Table 5.13 gives the results, again for a hand segmentation. Broadly speaking, the larger apu sets do better than the smaller ones. The exceptions are the three largest sets. They should be at the top of the table, but they are not. Presumably this is because there is not enough training data to train so many models. Of the three exceptions, two of them — ext03 and ext04 — actually do worse than the smallest set. They both have acoustic stops. There seem to be two effects at work here: the size of the apu set, and the type of its stop consonants. The type of stop is the lesser factor, because the small set ext06, although it also has acoustic stops, is in its right place in the table.

When we move away from a perfect segmentation, and use machine segments, the picture changes between open and closed test. Table 5.14, which is also on the closed-test data, is broadly similar to the hand segmentations. The top, middle and bottom three sets are the same, apart from the groupings. The open test results, which are given in table 5.15, are however very different. The small sets all do better than the big sets. Among the small sets the one with the acoustic stops is the best, and the same goes for the big sets.

In all cases stdp is better than std, although not always to a statistically significant extent.

The conclusion seems to be that under favourable conditions, such as those of a closed test, there is some reason for choosing an elaborated apu set. Under realistic conditions, however, large numbers of apus are a disadvantage. This disadvantage may be due to the amount of training data, as a comparison between stdp and std suggests.

## 5.6 Syllable-assisted segmentation

We will next investigate the use of syllables at the front end. Syllables will be used to constrain the segmenter, and the effect, as we will see, will be to reduce the number of segments that it produces. The mechanism that is used for this is called the sequencer, and it was described in sections 3.2.3 and 3.2.4.

The results of syllable-assisted segmentation for the different apu sets are now given.

| *Set* | *Ph Entropy* | *No of apus* |
|---|---|---|
| ext07c | 2.79 | 65 |
| ext08c | 2.79 | 72 |
| ext05c | 2.79 | 79 |
| ext02c | 2.80 | 128 |
| ext06c | 2.80 | 55 |
| stdpc | 2.80 | 49 |
| ext03c | 2.80 | 137 |
| std | 2.81 | 45 |
| ext04c | 2.81 | 104 |

Table 5.14: Classifications performed on a stdp segmentation. Lattice conflated to std phonemes. Speaker GSW, ATR data.

| *Set* | *Ph Entropy* | *No of apus* |
|---|---|---|
| ext06c | 3.30 | 55 |
| stdpc | 3.31 | 49 |
| std | 3.31 | 45 |
| ext07c | 3.31 | 65 |
| ext08c | 3.31 | 72 |
| ext05c | 3.32 | 79 |
| ext04c | 3.32 | 104 |
| ext03c | 3.33 | 137 |
| ext02c | 3.33 | 128 |

Table 5.15: Classifications performed on a stdp segmentation. Lattice conflated to std phonemes. Speaker GSW, CYT data.

| *Set* | *Oversegmentation* | | *% reduction* | *No of apus* |
|---|---|---|---|---|
| | *without* | *with syllables* | | |
| std_syll | 3.5 | 0.3 | 3.0 | 45 |
| stdp_syll | 3.1 | 0.4 | 3.0 | 49 |
| ext06_syll | 6.3 | 1.3 | 4.7 | 55 |
| ext08_syll | 6.5 | 1.9 | 4.3 | 72 |
| ext07_syll | 6.1 | 2.1 | 3.8 | 65 |
| ext05_syll | 7.1 | 2.1 | 4.6 | 79 |
| ext04_syll | 9.5 | 2.3 | 6.6 | 104 |
| ext02_syll | 9.6 | 2.6 | 6.4 | 128 |
| ext03_syll | 10.7 | 2.7 | 7.2 | 137 |

Table 5.16: Percent oversegmentation. Speaker GSW, ATR data, with and without syllables. Using syllables reduces the oversegmentation rate, to the extent shown in the third column of figures.

Tables 5.16 and 5.17 give the oversegmentation results for the case when syllable networks are used. The first table is for the ATR data, and the second is for CYT data. For convenience the oversegmentation rates without syllables (from the corresponding tables 5.6 and 5.7) are repeated in the first column. The use of syllables reduces the rate of oversegmentation, and this is indicated in the column headed *% reduction.* In the first line of the first table we see that syllable-assisted segmentation produces 3% fewer segments than unassisted segmentation. The reduction comes about because some of the spurious segments are lost in making the segments conform to a legal sequence of syllables. Since the lengths of the utterances remains the same, a small number of segments per utterance must consist of longer segments than a large number of segments. On the whole the lost segments are therefore short ones. As before the results correlate roughly inversely with the number of apus. Although the syllables reduce the oversegmentation more for the large sets than the small ones, the disadvantage that large sets have during segmentation remains after syllable processing.

Figures 5.3 and 5.4 show graphs of the oversegmentation data. The second of these, for the cytology data, is strikingly similar to the graph given earlier for ATR, 5.1. Ext06 and ext04, the sets with the acoustic stops, perform worse than expected. Stdp does unusually well. These trends are visible also in figure 5.3, but less clearly.

| *Set* | *Oversegmentation* | | *% reduction* | *No of apus* |
|---|---|---|---|---|
| | *without* | *with syllables* | | |
| stdp_syll | 16.8 | 13.9 | 17.3 | 49 |
| std_syll | 17.5 | 15.0 | 14.3 | 45 |
| ext08_syll | 21.0 | 14.7 | 30.0 | 72 |
| ext05_syll | 22.1 | 14.8 | 33.0 | 79 |
| ext07_syll | 20.4 | 15.0 | 26.5 | 65 |
| ext06_syll | 20.0 | 15.2 | 24.0 | 55 |
| ext02_syll | 25.6 | 16.7 | 34.8 | 128 |
| ext04_syll | 24.0 | 16.8 | 30.0 | 104 |
| ext03_syll | 26.8 | 17.5 | 34.7 | 137 |

Table 5.17: Percent oversegmentation. Speaker GSW, CYT data, with and without syllables. The syllables reduce the oversegmentation rate, to the extent shown in the third column of figures.

We notice from the oversegmentation tables that syllables reduce the oversegmentation rate more for CYT than for ATR. The reason for this lies in the bushiness of the syllable networks, a topic to which we must now turn.

### 5.6.1 Perplexities of Syllable Networks

The perplexity of a network measures its bushiness. Perplexity is the antilogarithm of the entropy of the phoneme sequences that make up the network; see appendix B. The perplexity of the network

```
k - a - t
```

is 1.

Table 5.18 gives the perplexities of the different syllable networks, for the ATR data. The third column gives the number of syllable types in the data. Notice that for the apu sets with acoustic stops, the number of syllable types is larger. This is because some syllables with stops appear in two forms. The syllable *pan*, for example, might appear twice, once with a released stop and once with an unreleased stop. This is because the manner of articulation — released or unreleased — of stops is not completely determined by syllable position. We saw in chapter 1 that although a majority of released stops occur in initial clusters, they do also appear in final clusters (roughly 70% versus 30% for the ATR data). Similarly unreleased stops, although mostly in final clusters

**e of apu set versus oversegmentation. Syllable-assisted, ATR**

gmentation

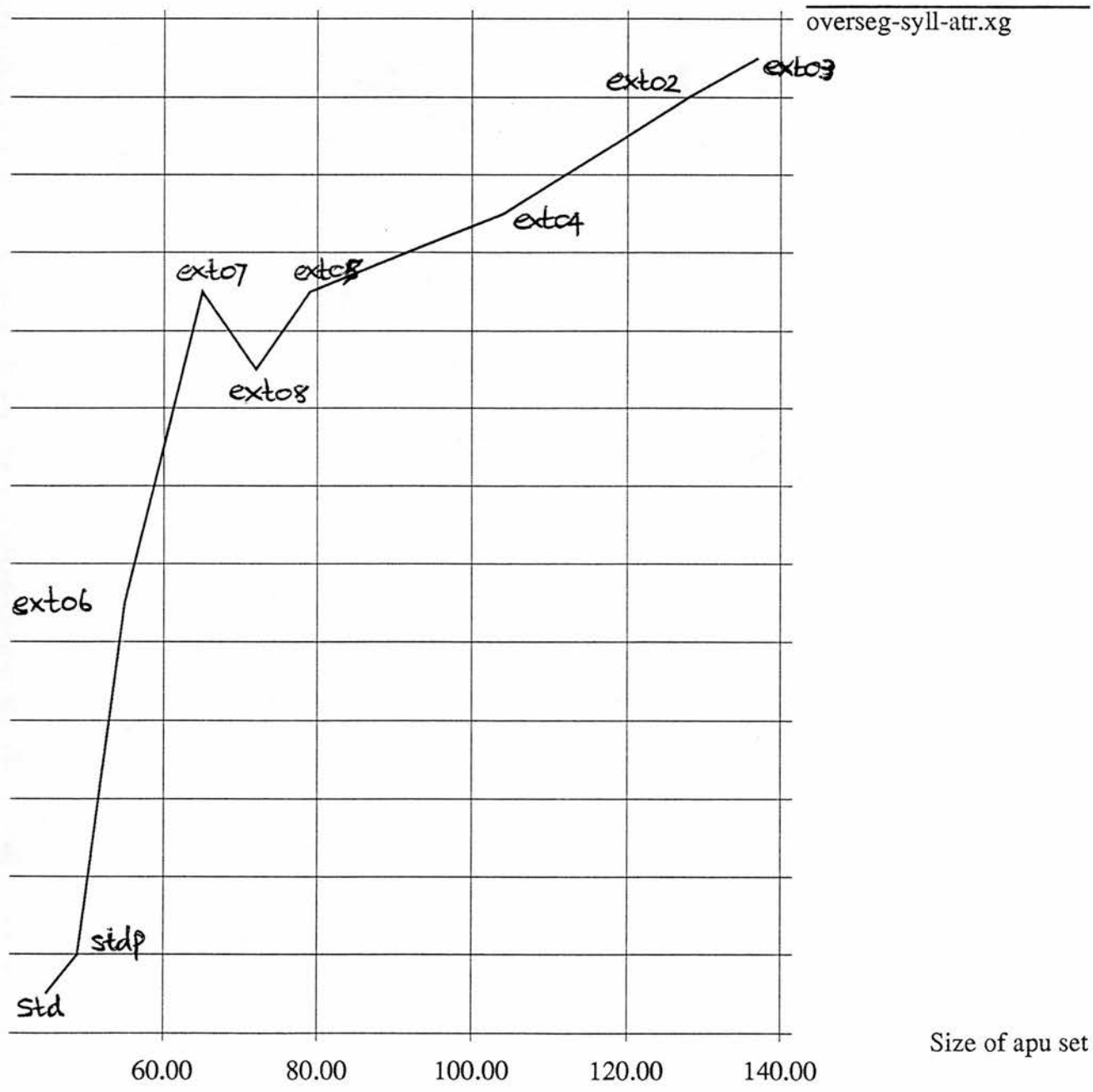



Figure 5.3:

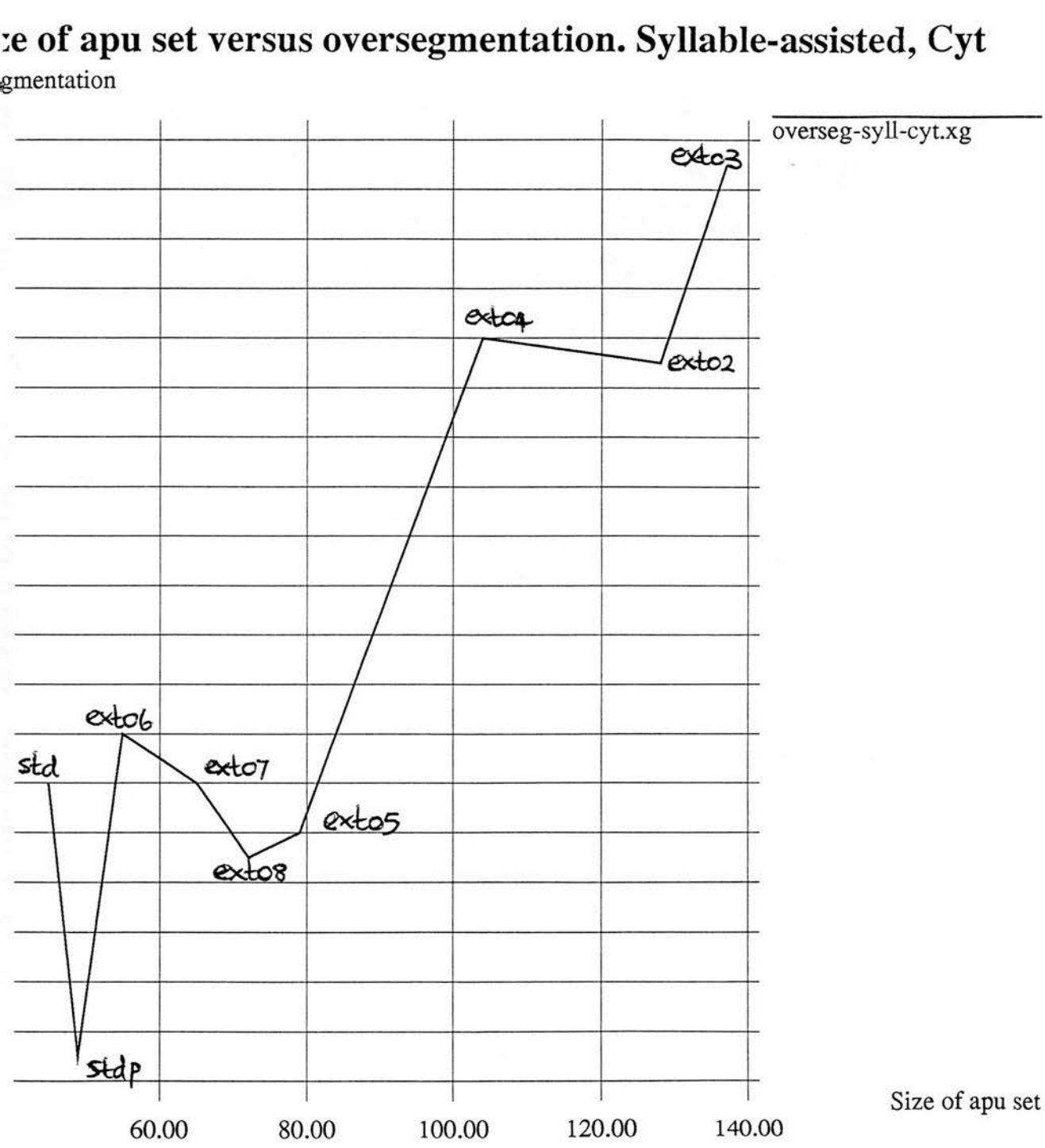

e of apu set versus oversegmentation. Syllable-assisted, Cyt
gmentation
overseg-syll-cyt.xg
ext03
ext04
ext02
ext06
std
ext07
ext05
ext08
stdp
60.00
80.00
100.00
120.00
140.00
Size of apu set

Figure 5.4:

| APU set | No of apus | No of syllables | Perplexity |
|---|---|---|---|
| ext02 | 128 | 1377 | 71.50 |
| ext03 | 139 | 1437 | 74.15 |
| ext05 | 79 | 1377 | 96.50 |
| ext08 | 72 | 1377 | 108.16 |
| ext04 | 104 | 1437 | 124.15 |
| ext07 | 65 | 1377 | 129.69 |
| ext06 | 55 | 1437 | 204.29 |
| stdp | 49 | 1377 | 222.56 |
| std | 45 | 1377 | 208.34 |

Table 5.18: Ranked perplexities of syllable models. Speaker GSW, ATR data.

(82%), also appear in initial clusters (18%). The effect of this is that a syllable like *pluck*, with initial and final stops, may appear more than once in the three sets with acoustic stops, in released and unreleased versions.

We see that, in general, the perplexity goes up as the number of apus decreases. This is because a more general apu belongs to more syllables. Consider an apu set in which phoneme /t/ is represented by both its post-vocalic allophone /tp/ and by its allophone as it appears in clusters /tc/. Then /tp/ appears in syllables like *pat* and *kit*, and /tc/ appears in syllables like *print* and *fast*. A smaller apu set with just the phoneme /t/, would have /t/ appearing in all the syllables mentioned: *pat*, *kit*, *print* and *fast*. The network of which /t/ is part is therefore more bushy than one in which /tp/ and /tc/ are part. A small apu set has fewer allophones, and those that it has are more general.

The perplexities of the cytology networks are given in table 5.19. The three sets with acoustic stops are separated out, because their syllables have been constructed on a different principle from the ATR case. Because the hand transcription of the cytology data was only done at the phonemic level, there is no indication of whether a stop was released or not. Thus *cat* is transcribed only as /k a t/, and we do not know whether it was spoken with released or unreleased *k*, or released or unreleased *t*. In order not to miss the version that was spoken, networks were created with the stop allophones systematically varied; thus *cat* appears four times, with released and unreleased appearing independently in the two stop positions. This gives rise to more syllables than are present in

| APU set | No of apus | No of syllables | Perplexity |
|---|---|---|---|
| ext02 | 128 | 368 | 53.25 |
| ext05 | 79 | 368 | 69.07 |
| ext08 | 72 | 368 | 76.79 |
| ext07 | 65 | 368 | 90.51 |
| stdp | 49 | 368 | 158.67 |
| std | 45 | 368 | 160.53 |
| | | | |
| ext03 | 139 | 560 | 62.31 |
| ext04 | 104 | 560 | 102.47 |
| ext06 | 55 | 560 | 152.32 |

Table 5.19: Ranked perplexities of syllable models. Speaker GSW, CYT data. The syllables in the second group do not reflect real data (see text).

the database, and explains why the acoustic sets have more syllables than the others.

The fact that the three apu sets which have acoustic stops have a less economical arrangement of syllables than the apu sets with syllabic stops, provides another reason why the acoustic sets do worse. We saw above that these sets oversegment worse than the syllabic sets. We saw above that two of them at least also oversegment worse with syllable assistance. Their larger set of syllables provides a weaker constraint during segmentation than that provided by a smaller set of syllables. This effect is additional to their poor performance during unassisted segmentation.

The matter of acoustic stops and syllabic stops is explored further in section 5.9 below.

## 5.7 Word-assisted segmentation

If, as we have seen so far, syllables reduce oversegmentation, we might wonder whether the use of words would reduce it even further. Being larger units, words impose stronger sequence restrictions than syllables, and so should winnow out more of the excess segments that the segmenter is prone to produce. Table 5.20 shows that this is not so. The table compares oversegmentation in the two cases already considered — unassisted segmentation, and syllable-assisted —

| *Database* | *Set* | *% Oversegmentation* | | |
|---|---|---|---|---|
| | | *Unassisted* | *With syllables* | *With words* |
| atr | stdp | 3.5 | 0.4 | 1.5 |
| cyt | stdp | 26.9 | 13.9 | 17.8 |
| cyt | ext02 | 35.5 | 16.8 | 18.7 |
| cyt | ext05 | 31.9 | 14.8 | 19.0 |

Table 5.20: Percent oversegmentation, for a selection of apu sets. Comparison of three segmentation methods: using no assistance, using syllable assistance, and using word assistance.

with the case where word networks are used. We see that words do reduce the oversegmentation rate, but not to the extent that syllables do. Figure 5.5 illustrates. It shows three lattices for the utterance *preliminary report.* The lattices are respectively unassisted, word-assisted and syllable-assisted segmentation, reading from bottom to top. The oversegmentation becomes less from bottom to top (the example chosen is representative of the general trend in table 5.20).

One thing about the figure is puzzling at first. It shows sequences of phonemes that do not form legal syllables or words. The middle segmentation has the sequence /r r r r/, for example. These are the best-scoring hypotheses for their respective segments. The explanation is that the sequencing is performed to produce the globally optimum path. In some cases this means not using the locally best-scoring hypothesis. The words and the syllables that were formed are not visible in the figure, and indeed are not reported by the algorithm.

The reason why word segmentation is worse than syllable segmentation is related to the size of the sequencing unit — syllables or words — and segmentation accuracy. It is clear that segmentation errors undo the work of the sequencer. Errors of deletion omit segments that the sequencer should have seen, and errors of insertion produce spurious segments that it should not have seen. Where the sequencing unit is large, this is more likely than when the sequencing unit is small.

This can be demonstrated as follows. Recall the discussion in chapter 3 about the operation of lexical access. The segmentation errors mentioned occasion lexical access to perform insertion and deletion repairs in the lattice (substitution errors only affect the identity of the apus, and not the quality of

the segmentation). Appendix C describes a study of the insertion and deletion repairs performed by lexical access. We see there that the *error-free interval* in the lattice, which we may define as the distance between successive insertions or deletions, is 4.36 phonemes for ATR and 3.29 phonemes for cytology. Recall from table 5.1 that the average syllable size is about 2.6 phonemes and the average word size is about 3.5 phonemes. Syllables are considerably smaller than either of the error-free intervals, whereas words are closer in size to them. This shows that syllable sequencing is less likely to be undone by errors in the segmentation.

Aside from the fact that, with current segmentation accuracies, words are too large a unit for safely sequencing a segmentation, they would not be a desirable unit to use in a loosely-coupled system. The reason for using syllables for sequencing segmentations is that their number is limited, which makes them immune to changes of vocabulary and grammar.

## 5.8 Segmentation repair

We have just seen that the raw segmentation that forms the input to the sequencing procedure suffers from the drawback that it contains errors. These errors of insertion and deletion mislead the sequencer, and as we saw the extent to which this happens depends on the size of the sequencing unit (syllable or word). We might ask whether it is worth repairing the errors of the raw segmentation before sequencing it. We shall find in this section that it is not.

Insertion and deletion errors in the segmentation are normally repaired in the back end, by lexical access. The lexical access module of the CSTR system was described in section 3.4, and for convenience it was referred to as 'lexax'. We could graft lexax into the front end, between the segmenter and the sequencer, in order to effect the repair. Of course we do not want to bring in the whole of lexax, complete with lexicon and grammar, because that would defeat our purpose of designing a front end that is stable in the face of changing requirements in vocabulary and syntax. Instead, we shall use a stripped-down lexax which uses syllables rather than words, and which operates without the

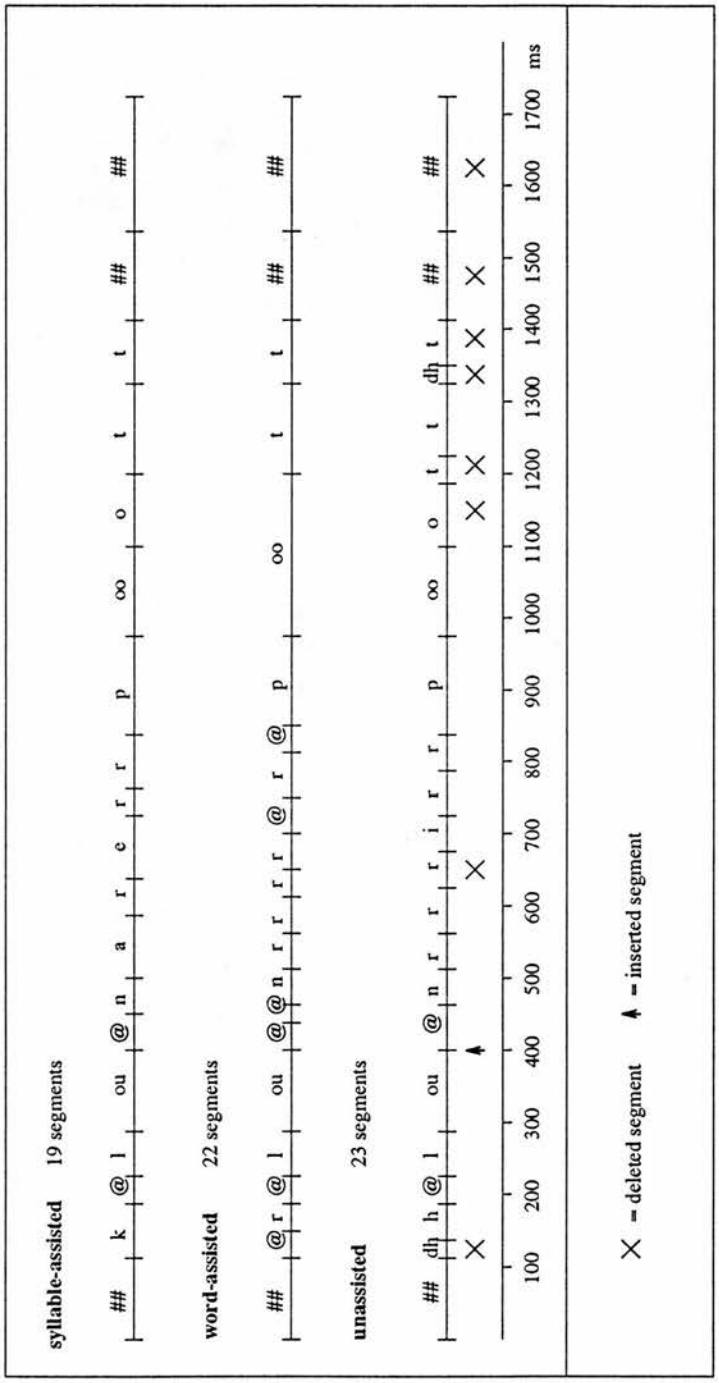



Figure 5.5: Three stdp phoneme lattices for *preliminary report*, using, from bottom, unassisted, word-assisted and syllable-assisted segmentation. All three lattices suffer from oversegmentation. The repairs that lexax had to make on the worst lattice are indicated with crosses for deletions and an arrow for the insertion.

assistance of a grammar.

The details of the implementation are as follows. The segmenter produces a regular segmentation, on which is performed a classification to produce a normal phoneme lattice. The stripped-down lexax runs, using a syllabicon and no grammar. The output file shows the sequence of phonemes that produced the output string; this sequence was obtained from the phoneme lattice, with insertions and deletions duly made. Lexax has repaired the segmentation, and has at the same time performed a syllable sequencing. The sequence has no segment boundaries, because lexax cannot know where the inserted segments begin and end. This sequence of phonemes is therefore presented to the segmenter to produce a resegmentation. This procedure is the same as that described in section 5.4 above. The resegmentation replaces the original segmentation, and the rest of the system runs as normal. The resegmentation will be called a *repaired segmentation* below.

Table 5.21 gives the oversegmentation rates of repaired segmentations. The negative numbers indicate an *under*segmentation: the repaired segmentations contain fewer segments than the hand transcription. The percentages are quite high: between a fifth and a quarter of the segments are missing. Table 5.22 shows the entropy results after std classifications have been performed on these segmentations. As we might expect, the entropies for the repaired segmentations are significantly worse than for the regular (unrepaired) ones.

We see that segmentation repair produces worse results than unassisted segmentation. Syllables are short units, which leads to a high rate of deletion. It is worth repeating that the conditions under which the segmentation repair took place were restricted: we chose syllables rather than words for lexax to use, and lookup was done without the guidance of a grammar. These restrictions are of course in line with the aim of this research, which is to investigate the behaviour of a front end which is not liable to changes of vocabulary and syntax. With these restrictions, segmentation repair is not a good idea.

| *Set* | *Dbs* | *% oversegm* |
|---|---|---|
| stdp | atr | −24.13 |
| | cyt | −19.43 |

Table 5.21: Percent oversegmentation after syllable sequencing with segmentation repair. The negative numbers indicate that fewer segments are produced than are present in the hand segmentation. Speaker GSW.

| *Set* | *Dbs* | *Ph Entropy* | | *Conf* |
|---|---|---|---|---|
| | | *Regular* | *Repaired* | |
| stdp | atr | 2.8 | 3.3 | 100% |
| | cyt | 3.3 | 3.8 | 100% |

Table 5.22: Std classification on stdp segmentations, with and without lattice repair. The confidence column indicates the probability that the repaired lattice gives worse results. Speaker GSW

## 5.9 Stop realisation and syllable position

In sections 5.5 and 5.5.4 above we noted that apu sets with acoustic stops perform worse than apu sets whose stops are defined according to their position in the syllable. These results are interesting because they corroborate findings elsewhere. The work of Mark Randolph (Randolph, 1989) is particularly relevant.

In a study of American English, he investigated the effects of phonetic and prosodic context on stop realisation. In his study a stop could be realised as *released*, *unreleased*, *flapped*, *deleted*, or *glottalised*. Table 5.23 shows his proposed predictors of these realisations. *Affix-1* is the first phoneme of an

| *Predictor* | *Values* |
|---|---|
| place | labial, alveolar, velar |
| voicing | voiced, unvoiced |
| prev context | affricate, fricative, glide, nasal, stop, vowel |
| foll context | the same |
| /s/-stop cluster | yes, no |
| syllable pos'n | onset, coda, affix-1, ambisyllabic |
| stress | high, low, rising, falling |

Table 5.23: Randolph's predictors of stop realisation.

affix. Affixes were defined in chapter 3 above. The values of the stress variable are defined according to the preceding and following vowel. If both of them are non-reduced, the stress environment is 'high'. If both of them are reduced, it is 'low'. If the preceding vowel is reduced and the following vowel is non-reduced, the stress environment is 'rising'. In the remaining case it is 'falling'.

Randolph performed a regression study between the predictors in the table and the different stop realisations. The regression was done by growing a binary tree, in which the predictor variables (the distinctive features) are correlated with a response variable (the stop realisation). The data consists of a set of vectors, of which this is an example:

(place = labial, voicing = voiced, syll posn = onset, ...,
realisation = released)

The final element of the vector is the response variable, and the earlier elements are the predictor variables. Initially all the data are placed at the root node of the tree. An attribute is then chosen for partitioning the root's sample of data. Once the choice is made the sample is split into two subsamples, and these are placed at the root's two daughter nodes. This procedure is continued until (this is the most important reason) all the samples at a node are 'pure', that is, contain acoustic realisations of only one type. The choice of attribute on which to split is made by maximising the mutual information of a subsample. Mutual information is also used to report the performance of the tree at the end. Mutual information is the a priori entropy of an acoustic realisation, reduced by a weighted sum of entropies corresponding to the individual terminal nodes.

Randolph found that the most significant influence on stop realisation is syllable position, and the next most significant was following context. Voicing and /s/-stop cluster have almost no influence. The oversegmentation results for ext06 and ext04 support this finding, because the other apu sets, which have syllable-conditioned stops, segment better. Ext03 also has acoustic stops, just like ext06 and ext04, but does not show up clearly as performing worse. This could be because its released and unreleased stops are themselves syllable-conditioned.

## 5.10 Summary and Conclusions

To retain the benefit of constrained generation of phonemes, I propose the use of syllables.

We have investigated the possibility of using enriched apu sets in the front end of a speech recogniser. Different apu sets were tried for both segmentation and classification. The apu sets were chosen according to their positions in syllables, and syllable networks were used to assist during segmentation. The results were measured in three ways, using end-point differences, oversegmentation rate, and entropies.

The results are variable. Under favourable conditions (known transcription during segmentation, and a closed test set during classification), the large sets do better than the small ones. Under realistic conditions, the large sets do worse, although this is not an absolute rule. Where stdp is different from std, it does better. An important factor in this is the quality of the segmentation for training. Where a high-quality hand segmentation is available, as it was for speaker GSW, stdp indeed does better than std. Where the training data consists of a machine segmentation, as it did for the other speakers, the advantage that stdp enjoys over std disappears. This advantage, in the case of GSW, suggests that some of the larger sets might not do so poorly if more training data were available.

The use of syllables reduces oversegmentation, as one would expect. Even with syllable assistance the large sets are at a disadvantage compared to the small ones: they still oversegment worse. The use of words rather than syllables to assist segmentation was shown to be ineffective. So was the attempt to repair the segmentation before sequencing.

Sets with acoustic (released and unreleased) stops are shown to be worse than sets with syllable-conditioned stops: they oversegment worse than their size would suggest. This applies to both unassisted and syllable-assisted segmentation. Sets with acoustic stops also have less efficient syllable networks: compared with the other sets they need more syllables to describe the same data.

At this point we may recall Church's recommendation, which was that the identification of allophones (enriched apu sets) would help recognition accuracy. With the amount of training data available at CSTR, and with the syllable mechanism used in this thesis, his recommendation is not borne out by experience.

The lessons learnt for the builders of loosely-coupled systems are as follows. The constraint provided by using syllables to sequence the segmentation can be enhanced by a judicious choice of apu set. The stdp set is a better choice than the std set, for example. It is possible that with more training data some of the other apu sets will come into their own. Sets with acoustic stops are to be avoided.

# Chapter 6

# Syllable Experiments in the Back End

## 6.1 Introduction

We saw in the previous chapter that syllables improve the performance of the front end and we now investigate the effect they have in the back end.

In chapter 3 the CSTR recognition system was described. It is a modular system, in which the front end and the back end are separate stages. As we saw, this enables variations of the system to be tested easily, and their effect measured independently of other factors. This approach is not free of difficulties, as we shall see in this chapter, but it is convenient for experimentation.

The aim in this chapter is to use syllables somehow to improve the performance of the back end. Given that syllables lie halfway between phonemes and words, one possibility is to use them as an intermediate data structure: that is, to produce syllables out of phonemes, and then look up the syllable strings in a special lexicon which spells words in terms of syllables. However, I could not see any obvious gain from this approach, and did not use it. Another possibility is to use syllables to get better statistics for phoneme confusions, and this is the approach of this chapter. Recall from chapter 3 that lexical access matches apu strings against words by performing substitutions, insertions and deletions. This is done on the basis of a confusion matrix which has been trained in ad-

vance. In the unmodified lexical access there is a single confusion matrix. In the approach taken in this chapter, multiple confusion matrices will be used, to to reflect the different behaviour of phonemes in different positions of a syllable.

This approach can be justified as follows. We know that it is common for speakers to pronounce phonemes indistinctly or to omit them altogether. Usually this phenomenon is modelled by considering only the left and right neighbours of a phoneme. However, the position of the phoneme in the syllable is relevant as well. For example, phoneme deletion is more common at the end of a syllable than at the beginning. Some systems (e.g (Lee, 1988)) take account of this fact, but only at the ends of words. A small experiment indicates that this is worth doing at word-internal syllable-boundaries as well.

The 200 ATR sentences were run through the CSTR recogniser and the number and places where the phoneme /d/ was deleted were counted. We expect the deletions to occur particularly at the ends of words and syllables, either because speakers don't say them or say them so indistinctly that the recogniser misses them. The experiment bears this out. A count of /d/ deletions shows that of 115 deletions, only 12 were not in the syllable coda. Of the 103 syllable-final deletions, nearly a quarter (25) were word-internal.

Multiple confusion matrices allow us to take account of this fact. With a single matrix, phoneme deletions are corrected in a way that depends only on the identity of the phoneme, and not on its position in the word or sentence. By introducing multiple matrices, one for each appropriate position in a syllable, the correction of phoneme deletions can be done in accordance with their frequency of occurrence in the different parts of the syllable.

Multiple confusion matrices are the main concern of this chapter, but before we get there a preliminary question is addressed, one which is raised by the previous chapter.

- Syllable-assisted segmentation is better than an unassisted segmentation, as we saw in the last chapter. Does this improvement make itself felt also in the back end? We shall see that the answer is no, for reasons that are probably related to redundancy in the lattice.

The main question of this chapter is

- Can we improve back-end performance by making lexical access aware of the positions in a syllable that the phonemes in the lattice occupy? We shall see that the answer is yes, but the performance can go down as well as up.

## 6.2 Measurement of Word String Quality

We measure the quality of the word strings produced by the back end by calculating a weighted error which shows how many substitutions, insertions and deletions it takes to turn the top scoring word string into the correct answer.

*weighted error = no of substitutions + ( no of indels / 2)*

An *indel* is an insertion or a deletion. Note that in calculating this measure the substitutions, insertions and deletions are made on *words*, unlike the operation of the back end itself, where these operations are performed on phonemes. The substitutions and indels are done under the control of a dynamic programming algorithm, and are the cheapest that can be obtained.

Here are some examples.

| | |
|---|---|
| Example: | Cytology sentence number 150 |
| Correct sentence: | *Pleural aspirates.* |
| Back end produces: | `needle aspirates` |
| Weighted error: | 1 (one substitution, *pleural* for `needle`) |

| | |
|---|---|
| Example: | Cytology sentence number 85 |
| Correct sentence: | *The specimen contains superficial squames.* |
| Back end produces: | `specimen contains superficial squames` |
| Weighted error: | 0.5 (one insertion, *the*) |

| | |
|---|---|
| Example: | Cytology sentence number 14 |
| Correct sentence: | *Microscopy shows scanty material with occasional* |

| | |
|---|---|
| | *groups of epithelial cells.* |
| Back end produces: | `microscopy shows scanty material with occasional groups of epithelial cells with a clot` |
| Weighted error: | 1.5 (three deletions, `with a clot`) |

As is obvious from the formula, the error function is such that an indel scores 0.5 and a substitution scores 1.0. The deletion of an incorrect word followed by the insertion of the correct one would therefore score the same as the straightforward substitution of the incorrect word by the correct one. For a perfect answer the weighted error is zero.

Occasionally the weighted error gives scores that are intuitively wrong.

| | |
|---|---|
| Example: | Cytology sentence number 69 |
| Correct sentence: | *Microscopy shows very scanty epithelial cells.* |
| Back end version 1: | `microscopy shows scanty epithelial cells` |
| Weighted error: | 0.5 (one insertion, *very*) |
| Back end version 2: | `microscopy shows a scanty epithelial cells` |
| Weighted error: | 1 (one substitution, *very* for `a`) |

The second version has a worse score than the first one, but it is arguable that it is a better recognition. The two versions differ on only one word, *a*, which is missing in the first version. The word is in fact wrong (it should be *very*), but at least the second version has given us the right number of words. Despite this occasional misbehaviour, the weighted error is widely used. The DARPA word error rate described in section 3.5 is an example.

Once the back end has run, the weighted error is calculated for every utterance. To compare the results of two runs, a $t$ test is performed on the weighted errors of the two runs. For ATR this means comparing 200 weighted errors from one run with 200 weighted errors from another run.

| *Database* | *Segm* | *Seg count* | *Conf* | *Ph entropy* | *Conf* |
|---|---|---|---|---|---|
| atr | reg | 9020 | | 2.88 | |
| | syll | 8752 | 100% | 2.75 | 100% |
| cyt | reg | 6622 | | 3.37 | |
| | syll | 5947 | 100% | 3.26 | 100% |

Table 6.1: Segment counts and phoneme entropies for the front end lattices of speaker GSW, using stdp segmentation and classification. The *Segm* column gives the type of segmentation: regular and syllable-assisted. The second line of a pair gives confidence levels that it is significantly different from the previous line.

## 6.3 The Data

Table 6.1 gives some statistics about the quality of the front end lattices that are fed to the back end. It gives in summary form the sort of thing that was presented in detail in chapter 5. The stdp apu set was used for both segmentation and classification. Two kinds of segmentation are shown: a regular segmentation without the use of syllables, and a syllable-assisted segmentation. The table gives the number of segments and the average phoneme entropy of the lattices. The lines should be read in pairs. The second line of the pair gives the confidence level that it is different from the first line (the confidence levels have been rounded to the nearest whole number). Recall from the previous chapter that because the segmenter oversegments, low segment counts are better than high ones, and that for entropies also low is better than high. We see from the table that for both ATR and CYT the use of syllables makes a significant improvement both to the segment counts and to the entropies of stdp classifications. Back end experiments will be performed on both kinds of lattices.

The back end is run with one of three grammar options: zero grammar, a bigram grammar, and a full grammar. We recall from section 3.4.4 that grammars help to restrict the number of candidates that lexax needs to consider when it is trying to make words out of the phoneme lattice. Zero grammar means that the number of words is not restricted, and at every possible word boundary in the phoneme lattice lexax needs to try to construct every word in the lexicon. A bigram grammar consists of a list of word pairs. After the first

| *Grammar* | *Database* | *Perplexity* |
|---|---|---|
| zero | atr | 1240 |
| | cyt | 240 |
| full | atr | 1.6 |
| | cyt | 3.5 |

Table 6.2: Perplexities for two kinds of grammar.

word of an utterance has been constructed, the candidates for the second word are drawn only from the pairs that start with the first word. Candidates for the third word depend on the second word, and so on. A further restriction is provided by a full grammar, in which only the sentences in the database can be constructed.

The extent to which lexax's choice of words is restricted is measured by *perplexity*, a term which we used already in chapter 5. The perplexity gives the average number of words that lexax needs to consider at any one point. Table 6.2 gives the perplexities of two grammars, for the ATR and CYT data.

With the zero grammar the perplexities are simply the sizes of the ATR and CYT lexicons. The perplexity of the full grammar for CYT is higher than for ATR because, even though ATR has more words, it also has longer sentences (as was shown in table 5.1). Long sentences have a larger number of predictable words than short ones. The first few words are usually enough to identify a sentence, and the remaining words can then be predicted with certainty. For each of these remaining words, therefore, lexax has only one choice. These fixed choices ensure a low perplexity, and since longer sentences have more of them, the average perplexity for ATR is lower than for CYT.

## 6.4 Syllable-conditioned Phoneme Lattices

Before we look for improvements to make in the back end, we would like to know how the lattices we produced with the help of syllables in the front end fare in the existing back end. As we saw above the syllable-conditioned lattices have a better entropy than the regular ones, and we expect them to do well in the back end also.

To answer the question two sets of runs were done with the baseline (unmodified) back end: one set used regular phoneme lattices as input and the other used syllable-conditioned ones. To give a fair comparison, the back end was trained separately on the two kinds of input. For the regular runs, the back end's confusion matrix was trained on the 200 ATR sentences, using lattices produced without syllable assistance. Lexax was then run on these files again to produce closed-test results, and the run was repeated on the cytology data to give open test results. For the other set of runs, the confusion matrix was trained on the ATR sentences, using lattices produced with syllable-conditioned segmentation. Closed and open test results were then produced for this option as well. All the runs were performed three times over, for each of the three grammar options.

The results are summarised in table 6.3. The lines in the table should be read in pairs. The first line of the pair is for results produced from a phoneme lattice prepared without the aid of syllables. The second line is for results produced from a syllable-conditioned phoneme lattice. The results consist of the word and sentence recognition rates achieved, together with the weighted error. The weighted error was described in section 6.2, and is the average for all the sentences in the set. The weighted error in the top line of the table is 5.72: this means it takes about 6 word substitutions or about 12 indels to turn a word string produced by the back end into the right answer. The weighted error for cytology is on average less than that for ATR because the cytology sentences are shorter (7.1 vs 13.0 words per sentence, as we noted in section 5.2). The confidence level in the second line of each pair indicates whether or not the weighted error is significantly different from the one in the previous line. The confidence level derives from a $t$ test which is performed on the weighted errors of the runs — 200 weighted errors for an ATR run and 170 errors for a cytology run. From the first two lines of the table, we see that we are 60% sure that syllable conditioned lattices perform worse than regular ones. I take a confidence of 90% as significant.

We see that syllable-conditioned phoneme lattices generally produce worse

| *Grammar* | *Database* | *Segm* | *% Words* | *% Sents* | *Error* | *Conf* |
|---|---|---|---|---|---|---|
| zero | atr | reg | 52.37 | 0.00 | 5.72 | |
| | | syll | 52.14 | 0.00 | 5.75 | 60% |
| | cyt | reg | 72.44 | 15.29 | 1.68 | |
| | | syll | 70.55 | 13.53 | 1.79 | 95% |
| bigram | atr | reg | 96.92 | 85.50 | 0.37 | |
| | | syll | 95.84 | 79.50 | 0.50 | 91% |
| | cyt | reg | 97.73 | 88.24 | 0.14 | |
| | | syll | 97.87 | 90.00 | 0.13 | 62% |
| full | atr | reg | 100 | 100 | 0.00 | |
| | | syll | 99.63 | 99.50 | 0.05 | 84% |
| | cyt | reg | 100 | 100 | 0.00 | |
| | | syll | 99.71 | 99.41 | 0.02 | 84% |

Table 6.3: Results for speaker GSW produced by the baseline back end, reading stdp lattices. The *Segm* column gives the type of segmentation: regular or syllable-assisted. The second line of a pair gives the confidence level that it is significantly different from the previous line.

results than the regular ones. In only one case does the syllable-conditioned lattice produce better results (bigram syntax on the cytology data), but this case is not significant, the confidence level being only 62%.

Given that the regular lattices themselves are worse than the syllable-conditioned ones as measured by entropy (table 6.1), the fact that they can give rise to better back-end results requires explanation. Figures 6.1 and 6.3 show two ways it can arise. Both figures show two fragments of phoneme lattices, one from a regular lattice, and one from a syllable-assisted one. The lattice is shown as a sequence of line segments, each labelled with an apu. Each segment in fact has multiple candidate phonemes, but these are shown only in a few cases, where it matters for the purpose of illustration. The second, third, etc candidates are shown stacked vertically over the first. They are given in score order, with the best candidate at the bottom. Below each lattice the phonemes assigned to the segments by the back end are shown in bold. These are called the *lexical phonemes*, and they are obtained from the phonemes in the lattice (called in this context the *surface phonemes*) by means of substitutions or insertions.

The first figure shows the lexical phonemes of the phrase *deposit of epithelial cells*. All of them were obtained from the surface phonemes by means of substi-

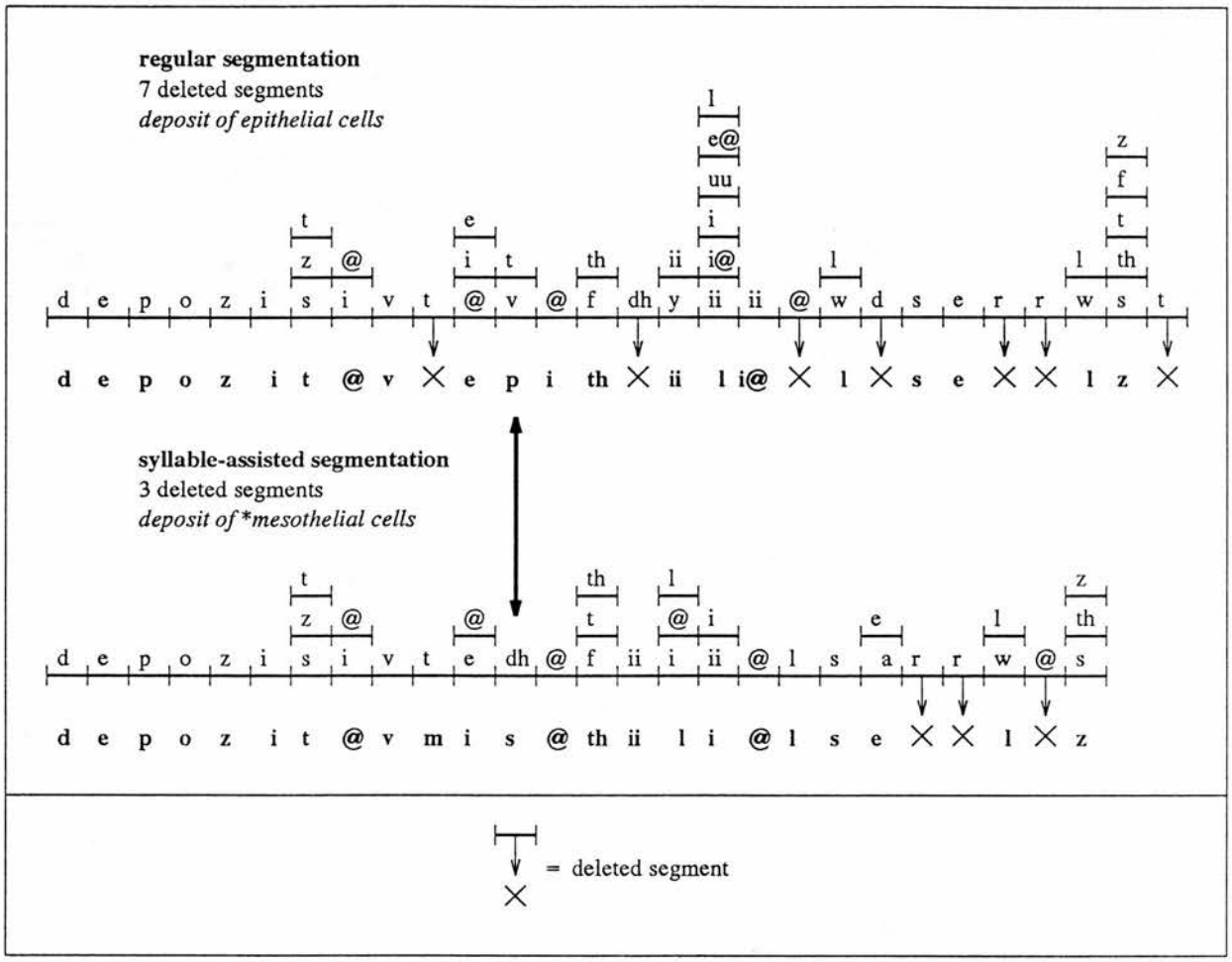



Figure 6.1: Back-end action on two phoneme lattices, using a bigram grammar. 'deposit of epithelial cells' and 'deposit of *mesothelial cells'.

tution. In the regular segmentation they are all identity substitutions: lexical /d/ was obtained from surface /d/, lexical /e/ from surface /e/, and so on. The lexical /t/ was obtained from the third-best scoring surface phoneme, also a /t/; the two better-scoring phonemes, /s/ and /z/, were not used. Multiple candidates are only shown in the figures in order to show the surface phoneme that lexax used to form the string. Segments that were skipped entirely (*deleted segments*), are shown with a down-pointing arrow and a ×. Inserted segments, which lexax had to provide in order to form a word, are shown by up-pointing arrows. The first figure has no inserted segments.

Consider now figure 6.1 in more detail. The illustrations are fragments

from larger lattices[1].The full lattice of the regular segmentation is worse than that of the syllable-assisted segmentation: the utterance entropies are 219 and 206 respectively[2].We can see the difference in quality even from the fragments shown: the top lattice has needed seven deletions to form a phrase, while the bottom lattice has needed only three. Nevertheless, the top lattice produces the right answer, *deposit of epithelial cells*, while the bottom lattice produces the incorrect phrase *deposit of *mesothelial cells* (the star indicates the incorrect word). The cause of the error is the arrowed segment. Its immediate environment is shown in more detail in figure 6.2. The segments are shown with their scores. The scores are identical where the segment boundaries are the same. One boundary is different, however, which is shown by the dashed arrow. The segment labelled /@, i, e/ in the top lattice is somewhat shorter than its equivalent in the bottom lattice, and the next segment, labelled /v, t, dh/, is somewhat longer. The difference gives rise to different scores on the affected segments, and to the lexical:surface substitution **p**:t at the top and the substitution **s**:dh at the bottom. From this follows *epithelial* at the top and **mesothelial* at the bottom.

A somewhat different case is illustrated in figure 6.3. These are fragments from a lattice for the sentence *microscopy shows scanty material with occasional groups of epithelial cells* (utterance number 14). The fragments show the tail end of the lattice in the two cases. Here also the regular lattice is worse than the syllable-assisted one (utterance entropies 206 and 177), although this is not evident from the fragments shown. The top and bottom fragments are shown aligned. The gaps that have been left in the top lattice are purely to make the alignment graphically possible, and have no other significance (i.e there are no breaks in the top lattice). Up to the end of the utterance, both lattices produce the same words. At the end however, the top lattice ends with the word *cells*, while the bottom lattice ends with *cells *with *a *clot.* This is despite the fact

[1]The full lattice is *The specimen contains a scanty deposit of epithelial cells and inflammatory cells*, utterance number 4.

[2]These utterance entropies are the sums of the entropies of the individual phonemes, normalised for utterance length.

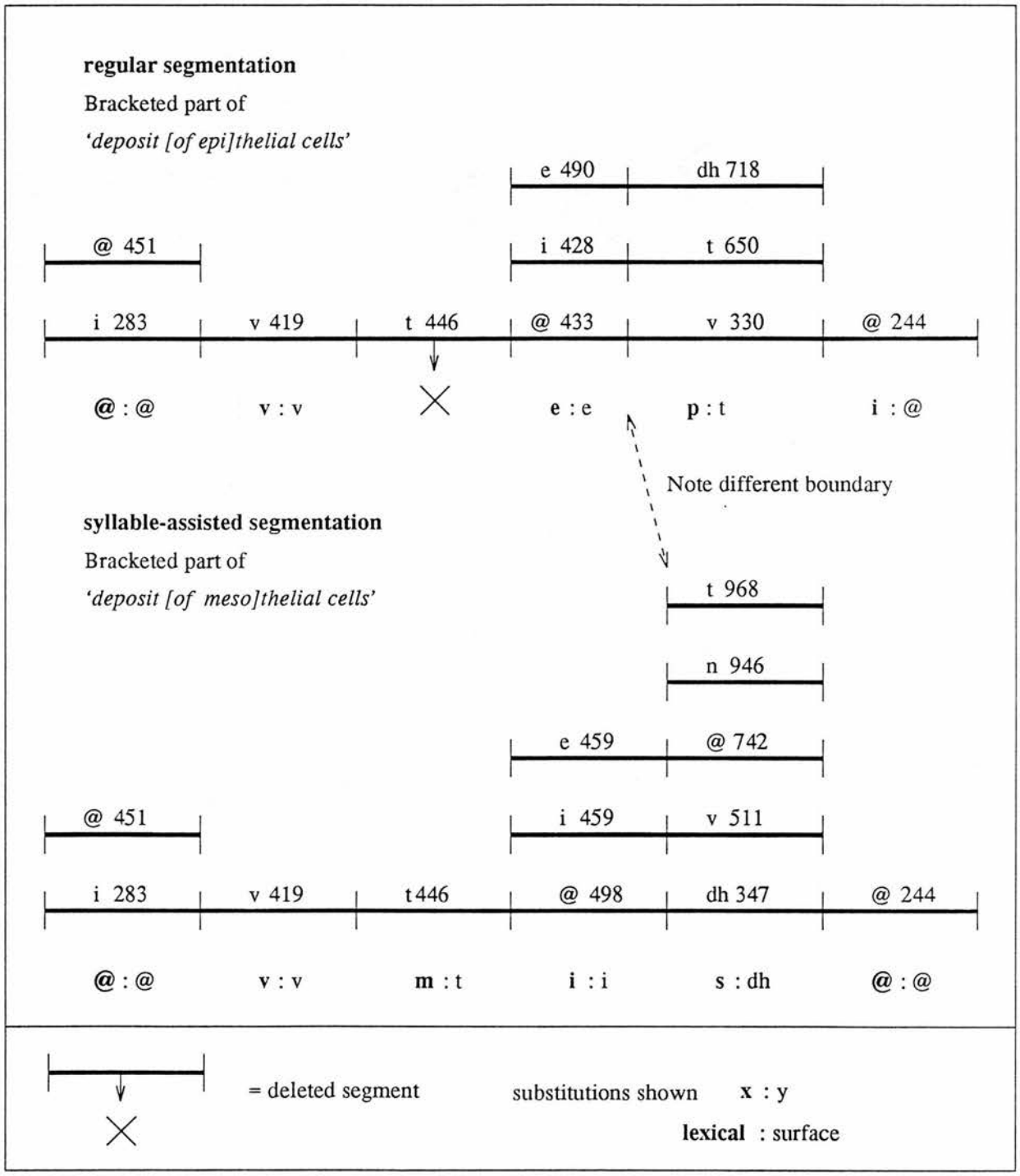



Figure 6.2: Back-end action on two phoneme lattices, using a bigram grammar. Excerpts from 'deposit of epithelial cells' and 'deposit of *mesothelial cells.

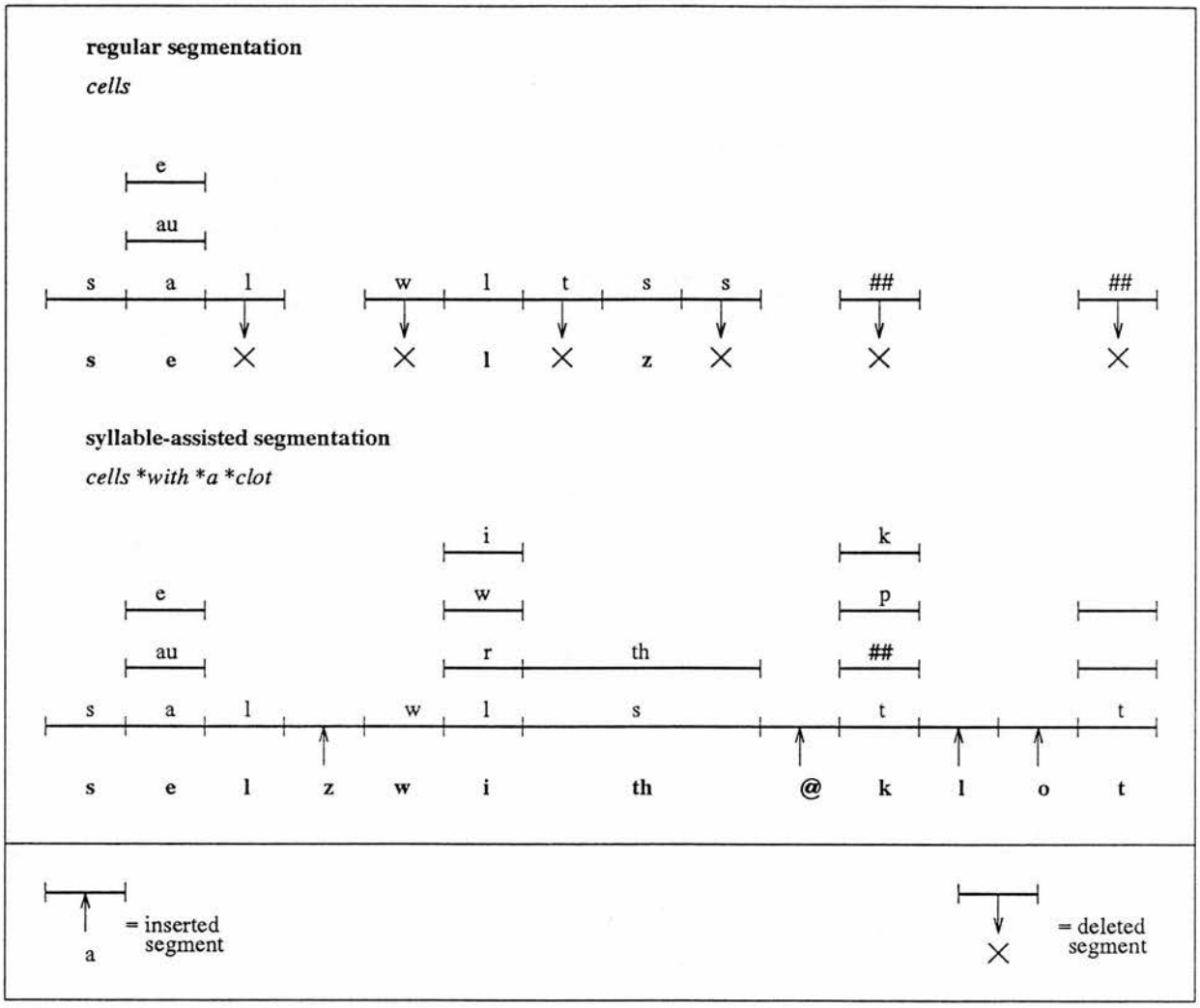



Figure 6.3: Back-end action on two phoneme lattices, using a bigram grammar. '...cells' and '...cells with a clot'.

that the top fragment has ten segments and the bottom one only eight. The difference is caused by a gross difference in the length of one segment. The segment labelled /s, th/ in the bottom lattice corresponds to three segments in the top (except for this one case, the segments are not drawn to scale). Three of the eight phonemes in *with a clot* had to be inserted to construct these words.

In the first example ('deposit of epithelial cells') different word strings were produced as a result of slightly different scores in the phoneme lattices. In the second example ('cells with a clot') a shorter sequence of segments gave rise to extra words because of a large difference in segment lengths. In both examples

the entropy tells us that it should be easier to produce the right answer from the syllable-assisted segmentation than from the regular one, but in both cases the back end prefers the regular one. Of course we can tell nothing from isolated examples like these. Isolated examples can always be found that contradict a general trend. What is more important is that table 6.3 shows that the back end almost always prefers the regular lattices, which have worse entropies.

One possible reason which may occur to the reader must be eliminated from consideration. We know that the front end sequences its lattices against syllables without guidance from a grammar. It is likely that in some cases it will choose the wrong syllables, that is, syllables which are not in the correct transcription. This will adversely affect the lattice. However, this is not enough to lower the quality of the lattice below that of an unsequenced lattice, as we can tell from the entropy scores. The entropy scores tell us that sequenced lattices are of a higher quality than unsequenced ones, even though the sequencing is prone to error.

The likely reason why the sequenced (syllable-conditioned) lattices do less well at the back end lies with oversegmentation. We have seen (tables 5.16 and 5.17) that syllable-conditioned lattices are smaller by 3% in the case of ATR and 17% in the case of cytology. The back end prefers the regular, more heavily oversegmented lattices. The front end evaluation program, which calculates the entropies, penalises such profligacy. It does this in the light of the transcriptions, which are not available to the back end. The back end needs to choose words from the entire lexicon, or from such subsets as are suggested by the grammar. It has a more difficult task than the front evaluation program. It may view the extra segments not as misleading, but as a source of redundancy. The extra segments seem to help the back end, rather than hinder it.

Aside from the fact that the back end prefers verbose lattices, we note two further points concerning the difference between front end and back end performance. The first point is specific to the CSTR system. The evaluation program in the front end, which computes entropies, has different information available to it from that available to the back end. The front end computes entropies

according to a confusion matrix similar to the one used in the back end, but the two confusion matrices are not the same. Recall from section 3.3 that the front end calculates a confusion matrix according to a *test set* of utterances (namely, the even-numbered utterances) which are not the same as the *closed set* of utterances (namely, the ATR utterances) from which the back end calculates its confusion matrix. We may therefore expect the back end to behave slightly differently from what the front end suggests.

The second point concerning front end evaluation and back end performance is more general. Because entropies are calculated in the light of the transcriptions, which are not available to the back end, they reflect performance that is closer to ideal than the back end can achieve. We may expect the back end to fall short of what the front end predicts. The back end is a complicated program, whose behaviour cannot always be foreseen. The CSTR system is not alone in yielding surprises at the back end. It was mentioned in section 2.6 that in SPHINX, adding duration modelling for words leads to a higher word accuracy, but not when a grammar is used.

The moral is that performance of the back end is not directly predictable from that of the front end. In particular, table 6.3 shows that the back end prefers verbose lattices, even though the front end considers them to be worse.

## 6.5 Multiple Confusion Matrices

We are now ready to run lexical access with multiple confusion matrices. Three confusion matrices are used: one for the apus in the syllable onset, one for the nucleus, and one for the coda. We shall call this set of confusion matrices the O-N-C matrices.

Runs were performed on the two kinds of input — regular phoneme lattices and syllable-assisted ones — and the results are given in two tables, 6.4 and 6.5. Each table shows the word and sentence recognition rates for two back end runs: one with a single (global) confusion matrix, and one with O-N-C matrices. As before each run was done with three grammar options, and as before the lines should be read in pairs, with the confidence level on the second line giving

| *Grammar* | *Database* | *Matrix* | *% Words* | *% Sents* | *Error* | *Conf* |
|---|---|---|---|---|---|---|
| zero | atr | global | 52.37 | 0.00 | 5.72 | |
| | | O-N-C | 50.73 | 0.00 | 5.92 | 100% |
| | cyt | global | 72.44 | 15.29 | 1.68 | |
| | | O-N-C | 72.49 | 17.65 | 1.67 | 53% |
| bigram | atr | global | 96.92 | 85.50 | 0.37 | |
| | | O-N-C | 97.19 | 85.00 | 0.34 | 67% |
| | cyt | global | 97.73 | 88.24 | 0.14 | |
| | | O-N-C | 98.45 | 91.76 | 0.09 | 96% |
| full | atr | global | 100 | 100 | 0.00 | |
| | | O-N-C | 98.17 | 97.50 | 0.22 | 99% |
| | cyt | global | 100 | 100 | 0.00 | |
| | | O-N-C | 96.13 | 93.53 | 0.24 | 99% |

Table 6.4: Word and sentence recognition rates, using regular phoneme lattices. Comparison of global and syllable-sensitive confusion matrices. O-N-C stands for onset-nucleus-coda

the probability that the two lines are different. For example, the first two lines of the first table show that the global matrix has a word recognition rate of approximately 52% and the O-N-C matrices have a rate of approximately 51%. The confidence level shows that the difference is not due to chance; we are 100% certain that the global recognition rate is better than the O-N-C one.

The tables show that the O-N-C matrices by and large do worse than a global matrix. Of the twelve results, the O-N-C matrices do worse in seven cases, and better in five. Seven of the twelve results are significant at the 90% level, and of these the O-N-C matrices do better in only two cases. However, an argument can be made for excluding the full grammar runs. The full grammar contains all and only the correct sentences, and can only be used in applications where these are known in advance. In realistic applications this is not so, and the other grammar options are the only ones that can be used. There are eight results for these non-full grammar options. Of these, the O-N-C matrices are better in five cases, and worse in three. If only significant results are considered, the comparison is better in two cases and worse in two cases. O-N-C matrices are therefore worth considering, but more experiments are necessary to discover exactly under what circumstances they are better.

In the runs under consideration O-N-C is at a disadvantage because its train-

| *Grammar* | *Database* | *Matrix* | *% Words* | *% Sents* | *Error* | *Conf* |
|---|---|---|---|---|---|---|
| zero | atr | global | 52.14 | 0.00 | 5.75 | |
| | | O-N-C | 51.21 | 0.00 | 5.86 | 96% |
| | cyt | global | 70.55 | 13.53 | 1.79 | |
| | | O-N-C | 71.57 | 15.29 | 1.73 | 91% |
| bigram | atr | global | 95.84 | 79.50 | 0.50 | |
| | | O-N-C | 95.50 | 81.00 | 0.54 | 70% |
| | cyt | global | 97.87 | 90.00 | 0.13 | |
| | | O-N-C | 98.50 | 91.18 | 0.09 | 89% |
| full | atr | global | 99.63 | 99.50 | 0.05 | |
| | | O-N-C | 99.38 | 99.00 | 0.08 | 66% |
| | cyt | global | 99.71 | 99.41 | 0.02 | |
| | | O-N-C | 98.16 | 96.47 | 0.11 | 96% |

Table 6.5: Word and sentence recognition rates, using syllable-conditioned phoneme lattices. Comparison of global and syllable-sensitive confusion matrices. O-N-C stands for onset-nucleus-coda

| | | |
|---|---|---|
| No of phonemes in | onset | 3934 |
| | nucleus | 3318 |
| | coda | 2544 |
| Average, onset and coda | | 3239 |

Table 6.6: Phoneme statistics for ATR data.

ing data is spread across three confusion matrices, compared to one for the global case. We can compensate for this by training the global matrix on a reduced set of utterances. The vowels get the same amount of training, but the training data for the consonants needs to be cut down to be approximately the same as the average of the number of onset consonants and coda consonants. Table 6.6 shows that this average is approximately the same as the number of vowels, and so the training regime is easy: use 200 sentences for training vowels, and 100 sentences for training consonants.

Table 6.7 shows the results when the single matrix is given reduced training in this sense. The first thing to note is that in two cases the reduced matrices do better than the full matrices. If we compare the results for the global matrix in table 6.4, where full training was used, we see that with reduced training the word recognition rates are 0.04% better for ATR and 0.09% better for cytology,

| *Grammar* | *Database* | *Matrix* | *% Words* | *% Sents* | *Error* | *Conf* |
|---|---|---|---|---|---|---|
| zero | atr | reduced | 51.71 | 0.00 | 5.80 | |
| | | O-N-C | 50.73 | 0.00 | 5.92 | 95% |
| | cyt | reduced | 72.44 | 14.71 | 1.68 | |
| | | O-N-C | 72.49 | 17.65 | 1.67 | 53% |
| bigram | atr | reduced | 96.96 | 84.50 | 0.37 | |
| | | O-N-C | 97.19 | 85.00 | 0.34 | 65% |
| | cyt | reduced | 97.82 | 88.24 | 0.13 | |
| | | O-N-C | 98.45 | 91.76 | 0.09 | 95% |
| full | atr | reduced | 100 | 100 | 0.00 | |
| | | O-N-C | 98.17 | 97.50 | 0.22 | 99% |
| | cyt | reduced | 100 | 100 | 0.00 | |
| | | O-N-C | 96.13 | 93.53 | 0.24 | 99% |

Table 6.7: Word and sentence recognition rates, using regular phoneme lattices. Comparison of a global confusion matrix with reduced training and syllable-sensitive confusion matrices. O-N-C stands for onset-nucleus-coda.

for the bigram grammar option (96.96% reduced versus 96.02% full for ATR and 97.82% reduced versus 97.73% full for cytology). We expect matrices from reduced training to be less robust than the fully trained ones. Perhaps this is true in general, and the two cited cases are exceptions.

A comparison between table 6.7 and table 6.4 shows that the pattern of results are the same. In each case, the six runs show the O-N-C matrices to be worse, better, better, better, worse and worse respectively than the single matrix. In each case the O-N-C matrices are significantly worse in three cases and significantly better in one case. If the full grammar option results are ignored, the O-N-C matrices are significantly better once and significantly worse once, again identically for full and reduced training. The conclusion is that any disadvantages that the O-N-C matrices have is not due to their having less training data than the global matrix.

## 6.6 Conclusions

The results in this chapter show promise but call for further investigation before firm conclusions are drawn. We have seen that the unmodified back end does better on regular phoneme lattices than on syllable-conditioned ones — that is, it does better on lattices that have a worse entropy. The reasons for this are

not clear, but they could be related to redundancy in the lattice. A less heavily oversegmented lattice like a syllable-conditioned one gets a better entropy score at the front end, but the back end prefers fuller lattices. Fuller lattices seem to be more tolerant of error than lean ones.

We performed further experiments, in which, following the hint that /d/ deletions occur more frequently at the end of a syllable than elsewhere, phoneme confusions were made to depend on syllable position. This was achieved with multiple confusion matrices, which were called O-N-C matrices. In some cases the multiple confusion matrices did better than the customary single confusion matrix, and in some cases they did worse. There is some indication that the multiple matrices are better in runs without a full grammar, but further experiments are needed to confirm this.

# Chapter 7

# Conclusions

## 7.1 Introduction

A recent trend in continuous speech recognition systems has been to move away from exclusive reliance on phonemes, words and syntax, towards a more linguistically informed approach. An example of this trend is SPHINX, which, with its triphone models takes account of the fact that the acoustic form of a phoneme is affected by its neighbours.

The work described in this thesis is part of the trend towards linguistic sophistication. It draws the environment of a phoneme wider than SPHINX, to include the syllable. This was done in order to capture variations in the phoneme that can be conveniently ascribed to its position in a syllable: a phoneme's neighbours are not the only influence on a phoneme.

## 7.2 Summary of results

The effect of using syllables in a continuous speech recognition system were investigated. They were exploited in three ways: in two ways in the front end of the recogniser, and in one way in the back end. The first way in the front end was as a reference against which to define the apus to be recognised. The form of some apus depends on their position in the syllable: an example is stops: syllable-final stops are more likely to be released than stops in other

syllable positions. Various sets of apus were defined; the smallest such set had 45 members and the largest had 137 members. The experiments were conducted as a result of a suggestion of Church's. He (Church, 1983) had proposed syllable-defined apus as a way of improving recognition accuracy. The finding here is that although syllable-defined apus are of some help, they are not as useful as Church had hoped.

The second way syllables were exploited was as a top-down constraint on the front end of a modular or loosely-coupled system. In speech recognisers where the front end and the back end are closely coupled, segmentation and classification normally take place under the control of lexical and syntactic information. Such top-down information makes a big difference to recognition accuracy. Modular systems like the one in use at CSTR suffer the disadvantage that this information is not available to the segmenter and classifier. As a partial remedy, this thesis offered syllable information to the segmenter and classifier. The use of syllables leads to a better segmentation than what is possible without them. In fact, the syllables give a better segmentation than words do, if a grammar is not used. The use of syllables during segmentation is only a partial remedy, because words plus grammar, as used in closely coupled systems, give better results than the use of syllables (or words) without a grammar.

One discovery in the front end has been that it is better to define stop allophones by their syllable position, than defining them as released and unreleased. It is true that released stops more often fall at the beginning of a syllable than at the end, but there seems to be more to it than that. Defining stops by syllable position leads to better phoneme lattices than defining them as released and unreleased. This is true whether or not the lattices are prepared with syllable-assisted segmentation.

The third way syllables have been exploited is in the back end. They were used to specialise the confusion statistics which lexical access uses when it matches the phoneme lattice against the lexicon. Usually the confusion statistics are contained in a single, global matrix. On the strength of our knowledge about phoneme deletion, which is that some phonemes are more often deleted at

the end of a syllable than at the beginning, the confusion matrix was split into three, with one matrix containing the confusion statistics for phonemes in the syllable onset, another matrix for the phonemes in the nucleus, and the third matrix for the syllable coda. The expectation is that these syllable-sensitive confusion matrices will produce better results than a single confusion matrix. The results however were equivocal. Syllable-sensitive confusion matrices perform worse when a full grammar is used, and better with a bigram or zero grammar.

## 7.3 Limitations of the use of syllables

This thesis has concentrated on the effect that syllable position has on the realisation of a phoneme. Syllables are not of course the only influence on phoneme realisation. Some of the influences come from units smaller than the syllable, and some from units that are larger. The former are readily seen in assimilation, coarticulation and reduction effects, which are due to the phoneme's neighbours. These effects also occur when the neighbours fall across syllable and word boundaries. Mention has already been made of the assimilation /d i jh uu/ for *did you*, which goes across a word boundary.

Effects from units larger than the syllable include those due to words and phrases. For example, the position of a phoneme in a word or phrase has an effect on its duration: it tends to be longer at the end of a word, and there is a further lengthening at the end of a phrase.

A large influence on phoneme realisation is stress. Stress often determines whether a word is reduced or not. Stressed vowels are generally acoustically more distinct than unstressed ones (Hieronymus *et al.*, 1992).

None of these effects can be captured by means of syllables, but the mechanisms by which they could be captured are not incompatible with syllables. It would be possible to combine syllables with smaller or larger units, as the case may be.

## 7.4 Future work

As always in work which depends on statistical training, the first recommendation is that the results be repeated on a larger set of training data. The training data used here, the 200 ATR sentences, is smaller than I would wish, particularly for the results on some of the larger apu sets. The front end results indicated that the larger *stdp* set (49 apus) does better than the *std* set (45 apus). This result shows that *stdp*'s extra apus, the syllabic consonants /l=/, /m=/, /n=/, and /r=/ are robust units. However, none of the sets that were larger still (*ext06* with 55 apus, for example) could exceed the performance of *stdp*. It is possible that a larger training set could change this ranking, and reveal further robust units.

The following is a list of specific things to be done.

In the front end:

**Multiple segmentations** The segmentations produced at the front end have all been single segmentations, that is, the utterance is spanned by a single chain of segments. In general, multiple segmentations improve the quality of the lattice. It would be interesting to see what effect this has on syllable-asssisted segmentation, and on segmentations using the other apu sets.

**Probabilistic arcs** The syllable networks used for the segmentation have no probabilities on them. The use of probabilities would presumably lead to a greater improvement than the one that has been obtained already.

In the back end:

**Wider range of confusion matrices** The multiple confusion matrices introduced at the back end separated the confusion statistics for segments from the syllable-initial cluster, from the nucleus, and from the syllable-final cluster. Further investigation should elaborate this three-way split with matrices for different syllable types like CVC, VCC, etc. Such an investigation would need a large amount of training data.

**Breakdown of benefits** The results as presented in chapter 6 do not indicate where the benefit of using multiple confusion matrices is coming from. Does it make sense to separate the statistics for consonant clusters between the initial and final parts of the syllable? Are some confusions more prevalent in the initial part than the final part?

**Vowels** The three-way split of the confusion matrix affected only the consonants of a syllable. The statistics for the nucleus were not affected by the split. It is likely that the nucleus does have different characteristics in different syllable types, particularly as between open and closed syllables. This matter needs to be investigated as well.

**Stress** (Hieronymus *et al.*, 1992) have found that stressed syllables are easier to recognise than unstressed ones. Their investigation was focussed on vowels: stressed vowels are more distinct than unstressed ones. Does this effect hold also for the consonants of stressed syllables? If so, does it depend on the position of the consonant in the syllable?

## 7.5 Final Word

It is well known that some verbal behaviour can be characterised by reference to syllables. Our knowledge of this behaviour can find a use also in speech recognition systems. Our understanding of how best to apply this knowledge is limited. This thesis has described a couple of ways of doing so. The recommended work will improve our understanding and lead to better recognition systems in the future. It is possible that on the way to this goal some general linguistic principles will be illuminated as well.

# References


Abercrombie, David. 1967. *Elements of General Phonetics.* Edinburgh: Edinburgh University Press.

Allen, W Sidney. 1973. *Accent and Rhythm: Prosodic Features of Latin and Greek: a Study in Theory and Reconstruction.* London: Cambridge University Press.

Allerhand, Michael. 1987. *Knowledge-based Speech Pattern Recognition.* London: Kogan Page.

Anderson, S. 1982. The Analysis of French Schwa. *Language*, **50**(3), 534 – 573.

Averbuch, A, Bahl, L, Bakis, R, Brown, P, Daggett, G, Das, S, Davies, K, Gennaro, S De, de Souza, P, Epstein, E, Fraleigh, D, Jelinek, F, Lewis, B, Mercer, R, Moorhead, J, Nadas, A, Nahamoo, D, Picjeny, M, Shichman, G, Spinelli, P, van Campernolle, D, & Wilkens, H. 1987. Experiments with the Tangora 20,000 word speech recogniser. *Pages 701 – 704 of: IEEE ICASSP.*

Bahl, L R, Bakis, R, Cohen, P S, Cole, A G, Jelinek, F, Lewis, B L, & Mercer, R L. 1980. Further results on the recognition of a continuously read natural corpus. *Pages 000 – 000 of: International Conference on Acoustics, Speech and Signal Processing.*

Bahl, L R, Jelinek, F, & Mercer, R L. 1983. Maximum likelihood approach to speech recognition. *Pattern Analysis and Machine Intelligence*, **PAMI-5**(2), 179 – 190.

Bahl, L R, Brown, P F, Souza, P V De, & Mercer, R L. 1988. A new algorithm of the estimation of hidden Markov model parameters. *Pages 000 – 000 of: International Conference on Acoustics, Speech and Signal Processing.*

Bell, A E. 1970. *A state-process approach to syllabicity and syllable structure.* Ph.D. thesis, Stanford, Palo Alto, Ca.

Bellman, R E. 1957. *Dynamic Programming.* Princeton, NJ: Princeton University Press.

Bernstein, M, *et al.* 1976. *Interactive systems research: Final report to the director, ARPA.* Tech. rept. TM-5246/006/00. System Development Corporation, Santa Monica, Ca.

Bloomfield, L. 1933. *Language.* New York: Holt.

Bridle, J S, & Brown, M D. 1979. Connected-Word Recognition using Whole-Word Templates. *In: British Institute of Acoustics Autumn Conference.*

Campbell, W N. 1988. Foot-level shortening in the Spoken English Corpus. *Pages 489 - 494 of: Seventh FASE Symposium.*

Cherry, Colin. 1978. *On Human communication: a review, a survey, and a criticism, 3rd edition.* Cambridge, Mass: MIT Press.

Chomsky, N, & Halle, M. 1968. *The Sound Pattern of English.* New York: Harper and Row.

Christie, W M. 1974. Some Cues for Syllable Juncture in English. *Journal of the Acoustical Society of America*, **55**(4), 819 - 821.

Church, K W. 1983. *Phrase-structure Parsing: A Method for Taking Advantage of Allophonic Constraints.* Ph.D. thesis, MIT, Cambridge, Mass.

Cox, S J. 1988. Hidden Markov Models for Automatic Speech Recognition: Theory and Application. *British Telecom Technology Journal*, **6**, 105 - 115.

Crowe, A S. forthcoming. *Title.* Ph.D. thesis, University of Edinburgh, Edinburgh.

de Saussure, Ferdinand. 1916. *Cours de linguistique générale.* Paris: Payot.

Fodor, J A. 1968. *Psychological explanation: an introduction to the philosophy of psychology.* New York: Random House.

Forney, G D. 1973. The Viterbi algorithm. *Pages 268 - 78 of: Proceedings IEEE 61.*

Fry, D B. 1964. The function of the syllable. *Zphon*, **17**, 215 - 000.

Gimson, A C. 1980. *An Introduction to the Pronunciation of English, 3d ed.* London: Edward Arnold.

Green, P D, Simons, A J H, & Roach, P J. 1990. The SYLK Project: Foundations and Overview. *Pages 000-000 of: British Institute of Acoustics Autumn Conference.*

Green, P D, Boucher, L A, Kew, N R, & Simons, A J H. 1992. *The SYLK Project — Final Report.* Tech. rept. CS-92-18. University of Sheffield Department of Computer Science, Sheffield, United Kingdom.

Gupta, V N, Lennig, M, & Mermelstein, P. 1987. Integration of Acoustic Information in a large vocabulary word recogniser. *Pages 697 - 700 of: International Conference on Acoustics, Speech and Signal Processing.*

Hieronymus, J L, McKelvie, D, & McInnes, F R. 1992. Use of Acoustic sentence level and lexical stress in hsmm speech recognition. *Pages 225 - 227 of: International Conference on Acoustics, Speech and Signal Processing.*

Hunt, Melvyn J, Lennig, Matthew, & Mermelstein, Paul. 1983. *Use of Dynamic Programming in a Syllable-based Continuous Speech Recognition System.* Reading, Mass: Addison-Wesley. Pages 163 - 187.

Kahn, D. 1976. *Syllable-based Generalisations in English Phonology.* Ph.D. thesis, MIT, Cambridge, Mass.

Klatt, Dennis H. 1977. Review of the ARPA Speech Understanding Project. *Journal of the Acoustical Society of America*, **62**, 1345 - 1366.

Klatt, Dennis H. 1979. *Scriber and Lafs: Two New Approaches to Speech Analysis.* Englewood Cliffs, New Jersey: Prentice-Hall. Pages 000 - 000.

Kruskal, Joseph B. 1983. *The Symmetric Time-Warping Problem: From Continuous to Discrete.* Reading, Mass: Addison-Wesley. Chap. 1.

Kruskal, Joseph B, & Liberman, Mark. 1983. *The Symmetric Time-Warping Problem: From Continuous to Discrete.* Reading, Mass: Addison-Wesley. Pages 125 - 161.

Kucera, H, & Francis, W N. 1967. *Computational Analysis of Present-Day American English.* Providence, RI: Brown University Press.

Ladefoged, Peter. 1967. *Three Areas of Experimental Phonetics.* London: Oxford University Press.

Lamel, L F, Rabiner, L R, Rosenberg, A E, & Wilpon, J G. 1981. An Improved Endpoint Detector for Isolated Word Recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, **ASSP-29**, 777 - 785.

Lea, Wayne A. 1973a. An Algorithm for Locating Stressed Syllables in Continuous Speech. *Journal of the Acoustical Society of America*, **55**, 411 (A).

Lea, Wayne A. 1973b. An Approach to Syntactic Recognition without Phonemics. *IEEE Trans on Audio and Electroacoustics*, **AU-21**, 249 - 358.

Lea, Wayne A. 1976. *Prosodic aids to speech recognition iX: Acoustic-prosodic patterns in delected English phrase structures.* Tech. rept. PX11963. Sperry Univac DSD, St Paul, Minn.

Lea, Wayne A, & Kloker, D R. 1975. *Prosodic aids to speech recognition: VI. Timing Cues to Linguistic Structures.* Tech. rept. PX11534. Sperry Univac DSD, St Paul, Minn.

Lea, Wayne A, & Shoup, June E. 1979. *Review of the ARPA SUR Project and Survey of Current Technology in Speech Understanding.* Tech. rept. N00014-77-C-0570. Office of Naval Research, Arlington, Virginia.

Lee, K-F. 1988. *Large-Vocabulary Speaker-Independent Continuous Speech Recognition: The SPHINX system.* Ph.D. thesis, Carnegie-Mellon University, Pittsburgh, Pa.

Lee, Kai-Fu. 1989. Hidden Markov models: past, present, and future. *Pages 148 – 155 of: European Conference on Speech Communication and Technology.*

Lee, Y, Silverman, H F, & Dixon, N R. 1984. Preliminary Results for an Operational Definition and Methodology for Predicting Large Vocabulary DUR Confusability from Phonetic Transcriptions. *Pages 26.2.1 – 26.2.4 of: International Conference on Acoustics, Speech and Signal Processing.*

Lehiste, Ilse. 1960. An Acoustic-Phonetic Study of Internal Open Juncture. *Phonetica*, **Suppl 5.**

Lehiste, Ilse. 1970. *Suprasegmentals.* Cambridge, Mass: MIT Press.

Lenneberg, Eric H. 1967. *Biological Foundations of Language.* New York: John Wiley and Sons, Inc.

Liberman, M Y, & Prince, A. 1977. On Stress and Linguistic Rhythm. *Linguistic Enquiry*, **8**(2), 249 – 336.

Lleida, E, Marino, J B, Nadeu, C, & Salavedra, J. 1991. Demisyllable-based HMM spotting for continuous speech recognition. *Pages 709–712 of: International Conference on Acoustics, Speech and Signal Processing.*

Lowerre, B T. 1976. *The Harpy Recognition System.* Ph.D. thesis, Carnegie-Mellon University, Pittsburgh, Pa.

Lowerre, Bruce, & Reddy, Raj. 1980. *The HARPY Speech Understanding System.* Englewood Cliffs, NJ: Prentice-Hall. Chap. 15.

Marino, J B, Nadeu, C, Moreno, A, Lleida, E, & Monte, E. 1989. Recognition of Numbers and Strings of Numbers by Using Demisyllables: One Speaker Experiment. *Pages 102 – 105 of: European Conference on Speech Communication and Technology.*

Markel, J D, & Gray, A H. 1986. *Linear Prediction of Speech.* blah: Springer-Verlag.

McInnes, F R. 1988. *Adaptation of Reference Patterns in Word-Based Speech Recognition.* Ph.D. thesis, University of Edinburgh, Edinburgh, Scotland.

McInnes, F R. 1993. *Entropy Evaluation.* Draft of an unsubmitted manuscript.

McInnes, F R. September 1992. *Polyglot final report.* Tech. rept. Centre for Speech Technology Research, Edinburgh.

McInnes, F R, & Jack, M A. 1988. *Automatic Speech Recognition using Word Reference Patterns.* Edinburgh: Edinburgh University Press. Chap. 1.

McInnes, F R, McKelvie, D, & Hiller, S M. 1990. The Structure, Strategy and Performance of a Modular Continuous Speech Recognition System. *In: British Institute of Acoustics Autumn Conference.*

McInnes, F R, McKelvie, D, & Hiller, S M. 1991. *The* CSTR *Recognition System.* Draft of an unsubmitted manuscript.

Mermelstein, Paul. 1975. Automatic Segmentation of Speech into Syllabic Units. *Journal of the Acoustical Society of America*, **58**(4).

Mertens, P. 1987. Automatic Segmentation of Speech into Syllables. *Page ?? of: European Conference on Speech Communication and Technology.*

Nakagawa, Sciichi, & Jilan, Mohammed M. 1986. Syllable-Based Connected Spoken Word Recognition by Two Pass O(n) DP Matching and Hidden Markov Models. *Pages 1117 - 1120 of: International Conference on Acoustics, Speech and Signal Processing.*

Nakatani, L H, & Dukes, K. 1977. Locus of Segmental Cues for Word Juncture. *Journal of the Acoustical Society of America*, **62**(3), 714 - 719.

Newell, A, Barbett, J, Forgie, J W, Green, C, Klatt, D, Licklider, J C R, Munson, J, Reddy, D R, & Woods, W A. 1973. *Speech Understanding Systems: Final Report of a Study Group.* Amsterdam: North Holland/American Elsevier.

O'Connor, J D, & Trim, J L M. 1953. Vowel, consonant and syllable — a phonological definition. *Word*, **9**, 103 - 000. Reprinted in (Jones & Laver, 1973). Page references are to this edition.

Papadimitriou, Christos H, & Steiglitz, Kenneth. 1982. *Combinatorial Optimisation: Algorithms and Complexity.* Englewood Cliffs, NJ: Prentice-Hall.

Paul, D B, & Martin, E A. 1988. Speaker Stress-Resistant Continuous Speech Recognition. *Pages 000–000 of: International Conference on Acoustics, Speech and Signal Processing.*

Pierre, Donald A. 1969, 1986. *Optimisation Theory with Applications.* New York: Dover Publications, Inc.

Pike, K L. 1943. *Phonetics.* Ann Arbor, Mich: University of Michigan Press.

Rabiner, L R, & Juang, B H. 1986. An Introduction to Hidden Markov Models. *IEEE ASSP Magazine*, 4 - 16.

Rakerd, Brad, Sennett, William, & Fowler, Carol A. 1987. Domain-Final Lengthening and Foot-Level Shortening in Spoken English. *Phonetica*, **44**, 147 - 155.

Randolph, Mark A. 1989. *Syllable-based Constraints on Properties of English Sounds.* Ph.D. thesis, MIT, Cambridge, Mass.

Reddy, D Raj (ed). 1975. *Speech Recognition.* New York: Academic Press.

Russell, M J, Ponting, K M, Peeling, S M, Browning, S R, Bridle, J S, Moore, R K, Galiano, I, & Howell, P. 1990a. The ARM Continuous Speech Recognition System. *Pages 69 - 72 of: International Conference on Acoustics, Speech and Signal Processing.*

Russell, M J, Ponting, K M, & Peeling, S M. 1990b. The Armada Speech Recognition System. *In: Voice Systems Worldwide.*

Saito, S, & Itakura, F. 1966. *The Theoretical Consideration of Statistically Optimum Methods for Speech Spectral Density.* Tech. rept. 3107. Electrical Communication Laboratory, NTT, Tokyo.

Sankoff, David, & Kruskal, Joseph B. 1983. *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison.* Reading, Mass: Addison-Wesley.

Shannon, Claude E. 1951. Prediction and Entropy of Printed English. *Bell System Technical Journal*, **30**, 50 - 64.

Shannon, Claude E, & Weaver, Warren. 1949. *The Mathematical Theory of Communication.* Urbana, Illinois: University of Illinois Press.

Smith, A R. 1977. *Word Hypothesisation for Large-Vocabulary Speech Understanding Systems.* Ph.D. thesis, Carnegie-Mellon University, Pittsburgh, Pa.

Stetson, R H. 1951. *Motor Phonetics.* Amsterdam: North Holland.

Thompson, H S. 1984. Speech transcription: an incremental, interactive approach. *Pages 697 - 704 of: Sixth European Conference on Artificial Intelligence.*

Thompson, Henry S, & Ritchie, Graeme. 1984. Implementing Natural Language Parsers. *Chap. 9 of:* O'Shea, Tim, & Eisenstadt, Marc (eds), *Artificial Intelligence and Applications.* Harper and Row.

Vaissiere, J. 1989. On automatic extraction of prosodic information for automatic speech recognition system. *Pages 202–205 of: European Conference on Speech Communication and Technology.*

Vicens, P. 1969. *Aspects of Speech Recognition by Computer.* Ph.D. thesis, Computer Science Dept, Stanford University.

Waibel, Alex. 1988. *Prosody and Speech Recognition.* London / San Mateo, Calif: Pitman / Morgan Kaufmann.

Wood, L C, & Pearce, D J B. 1990. Sub-Word HMM Recognition: An Investigation of Phone Context Modelling and Improved Discrimination. *Pages 181–188 of: British Institute of Acoustics Autumn Conference.*

# Appendix A

# The machine-readable phonemic alphabet

Table A.1 defines the machine-readable phonemic alphabet in use at CSTR.

Table A.2 gives the relative frequencies of *std* phonemes, based on Gordon Watson's pronunciation of 200 ATR sentences.

| mrpa | IPA | mrpa | IPA | mrpa | IPA | mrpa | IPA |
|---|---|---|---|---|---|---|---|
| p | p | zh | ʒ | y | j | oo | ɔ |
| t | t | f | f | w | w | o | ɒ |
| k | k | v | v | ii | i | aa | ɑ |
| b | b | th | θ | i | ɪ | ei | eɪ |
| d | d | dh | ð | e | ɛ | ai | aɪ |
| g | g | h | h | a | a | oi | ɔɪ |
| ch | ʧ | m | m | @ | ə | ou | oʊ |
| jh | ʤ | n | n | @@ | ɜ | au | aʊ |
| s | s | ng | ŋ | uh | ʌ | i@ | iə |
| z | z | l | l | uu | u | e@ | ɛə |
| sh | ʃ | r | r | u | ʊ | u@ | ʊə |

Table A.1: RP English phonemes, expressed in mrpa (machine readable phonemic alphabet) and IPA symbols.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ## | 0.022952 | | 0.110971 | | 0.006312 | a | 0.014459 |
| aa | 0.012050 | ai | 0.018706 | au | 0.006771 | b | 0.024329 |
| ch | 0.007804 | d | 0.033165 | dh | 0.038100 | e | 0.021804 |
| e@ | 0.004820 | ei | 0.014574 | f | 0.021001 | g | 0.011361 |
| h | 0.009525 | i | 0.059559 | i@ | 0.004361 | ii | 0.030640 |
| jh | 0.008836 | k | 0.032017 | l | 0.042460 | m | 0.028230 |
| n | 0.059100 | ng | 0.008951 | o | 0.012623 | oi | 0.003098 |
| oo | 0.012050 | ou | 0.012164 | p | 0.022493 | r | 0.032821 |
| s | 0.053133 | sh | 0.008722 | t | 0.069084 | th | 0.007000 |
| u | 0.004246 | u@ | 0.001148 | uh | 0.012394 | uu | 0.011820 |
| v | 0.017787 | w | 0.024214 | y | 0.008263 | z | 0.031673 |
| zh | 0.002410 | | | | | | |

Table A.2: Phoneme frequencies. Speaker GSW, ATR data.

# Appendix B

# Entropy and Perplexity

Most recognition systems report their performance as a hit rate of some kind. Lee (Lee, 1989), for example, quotes a version of the SPHINX system as having a 96.2% 'word accuracy', where word accuracy is defined as the percentage of words in the sentence correct, less the percentage of (correct) words that had to be inserted in the sentence by the parser. Lee's is an example of a word hit rate. Hit rates are an appealing measure of performance, and they will probably be always with us. However, they can be misleading as well, because they leave unsaid the number of hypotheses required to achieve the hit rate. If a spoken sentence contains 20 words, the recogniser will typically generate many more than just 20 words in an attempt to recognise it. To be quite sure of getting it right, a 1000-word recogniser could generate all 1000 words at every word position. Such a 'recogniser' achieves a 100% hit rate, but it in effect does nothing for us, because it leaves us with the problem of identifying and eliminating the enormous number of false alarms.

To cope with this and other problems the *cstr* recogniser uses *entropy* as a measure of recognition accuracy. Specifically, it is used to measure the quality of the phoneme lattice. The following section introduces the concept in general, and the one after that explains how it is applied to the phoneme lattice.

## B.1 Entropy

'Entropy' was first used in dynamical systems (physical systems that are in motion) to express the degree of disorganisation in them. The mathematical expression for entropy is very similar to that derived in information theory to measure the amount of information in a signal. Consequently 'entropy' has come to mean 'amount of information' as well. In this section the expression for entropy

$$H = -\sum_i p_i \log p_i \qquad (B.1)$$

is derived. The discussion draws mainly on (Cherry, 1978) and (Shannon & Weaver, 1949).

Information theory, which deals with the measurement of information, was developed in the first half of this century, and was brought to its fruition by Claude Shannon. See (Cherry, 1978) for a historical discussion. The theory arose out of the problems of transmitting messages over telegraph wires.

Suppose two friends have an arrangement whereby one of them visits the other most weeks, on the day that is convenient to the other. This person, the sender, informs the receiver of which day he should come, by sending him a message naming the day of the week. If the visit would be inconvenient, he sends the phrase *Don't come.* The receiver therefore expects one of eight possible messages, namely, *Sunday, Monday, ..., Saturday, Don't come.* Suppose he receives the message *Saturday.* How do we measure the information in this message? One factor that seems relevant is the number of possible messages. Suppose the receiver knows he is to come at the weekend, and needs only to be told which day. In this case he is expecting one of two possible messages, namely *Saturday, Sunday.* Suppose he receives the message *Saturday.* We intuitively feel that there is less information when this message is part of this new case than the original one. In the new case he can expect one of two answers, instead of one of eight. If the answers are equally likely, then the probability of the message *Saturday* in the new case is $\frac{1}{2}$, and in the original case is $\frac{1}{8}$. The message in the new case resolves less uncertainty. This is a key observation.

| | |
|---|---|
| 000 | Sunday |
| 001 | Monday |
| 010 | Tuesday |
| 011 | Wednesday |
| 100 | Thursday |
| 101 | Friday |
| 110 | Saturday |
| 111 | Don't come |

Table B.1: Eight equally probable messages and the binary encoding of their selection

To carry the argument further let us now look at the signal instead of the messages themselves. The signal *encodes* the messages. Suppose the signal is transmitted over a wire as a sequence of binary digits, 0 and 1. One way of encoding the messages is to associate *Sunday* with 1, *Monday* with 2, and so on. *Don't come* will be associated with 8. The message *Saturday* is associated with 7, which would be transmitted in binary form as 111. This is a very efficient way of encoding the messages, as the next paragraph explains.

In the original example the receiver knows that the message will be one of eight possibilities. A fruitful way of looking at the signal is as instructions to select from this repertoire. The most efficient way to do this is by the method of *binary subdivisions*. The sender first instructs the receiver to find the message in either the first half or the second half of the repertoire. We can represent the selection with a 0 for the first half — *Sunday* to *Wednesday* — and a 1 for the second half — *Thursday* to *Don't Come*. The number of possibilities has now been reduced to four. The next instruction asks the receiver to select either the first half or the second half of this new set. As before, we represent the selection with a 0 or a 1. The number of possibilities is now two, and a final instruction tells the receiver to select the first one or the second one. The receiver has made three selections to arrive at the intended message, which can be written as a sequence of three bits, such as 101, which selects *Friday*. Table B.1 shows the binary encodings of the eight messages.

We shall say that the information in this repertoire of 8 messages is 3 bits.

This is because $2^3 = 8$, or $\log 8 = 3$. (As elsewhere in the thesis, log means logarithm to the base 2). Each bit halves the range of uncertainty of the receiver. At first its range of uncertainty is 8. After the first bit its range of uncertainty is 4, after the second bit it is 2, and after the third bit it is 1. 3 bits is also the information for each sign. We have assumed that each sign is equally probable, and each needs three binary selections to specify it. The information per sign is usually given the symbol $H_i$. $i$ ranges between 1 and 8 in this case, and $H_1 = H_2 = \cdots = H_8 = 3$. Each sign has a probability of $\frac{1}{8}$ of occurring. We can express the information per sign also as

$$H_i = -\log \frac{1}{8} = -(-3) = 3$$

bits per sign.

The choice of the logarithm of the number of messages as the information measure can be justified further when we consider what happens when we install a second channel (a second trasmission wire). We are to send messages down the two channels simultaneously. The number of messages that can now be sent is the square of eight, because while we are sending *Sunday* on the first channel, we can be sending any of the other eight possibilities down the second channel, and so also for *Monday*, *Tuesday*, etc. Our intuition tells us that adding the extra channel doubles the information that can be sent at any time, and this is reflected nicely in the addition rule for logarithms. With a repertoire of $8 \times 8 = 64$ messages, we have an information measure of $\log 64 = 6$, which is 3+3. Similarly, if we add a third channel, the number of messages is $8 \times 8 \times 8 = 512$, while the information is $3 + 3 + 3 = 9 = \log 512$.

The addition rule also applies when we consider messages that consist of more than one sign. So far the whole message sent by the sender has consisted of a single word, or sign, namely *Sunday*, *Monday*, etc. A more realistic message contains a sequence of words or signs. We might think that the information associated with a two-word message, a three-word message, etc, is just double, treble, etc, the information of a one-word message, by the same argument we used when we added extra channels alongside the original one. This is indeed so

| | |
|---|---|
| $\frac{1}{2}$ | Sunday |
| $\frac{1}{4}$ | Monday |
| $\frac{1}{16}$ | Tuesday |
| $\frac{1}{16}$ | Wednesday |
| $\frac{1}{32}$ | Thursday |
| $\frac{1}{32}$ | Friday |
| $\frac{1}{32}$ | Saturday |
| $\frac{1}{32}$ | Don't come |

Table B.2: Messages with unequal probabilities

for the case where the different words don't depend on each other. The number of possibilities for a two-word message is again $8 \times 8 = 64$, and the information associated with such a repertoire is $3 + 3 = 6$.

In all these cases

$$H_i = -\log p_i$$

So far the signs have been assumed to be equally probable. The signs all had a probability of $\frac{1}{8}$ of occurring. In any natural language some words occur more frequently than others. Let us alter the example and give the messages the probabilities shown in Table B.2.

Call these probabilities $p_i$. Thus $p_1 = \frac{1}{2}$, $p_8 = \frac{1}{32}$, and of course they add up to 1:

$$\sum_i p_i = 1$$

The probabilities have been carefully chosen for the argument to follow, but we shall remedy this shortly. Instead of dividing the repertoire up into equal groups as before, we now divide it into groups of equal probability. There are four groups of different sizes. The first group has only one member, namely *Sunday*, because that is the only message with probability $\frac{1}{2}$. The second group also has one member, namely *Monday*, the third group has two members, namely *Tuesday* and *Wednesday*, and the fourth group has the rest. With this arrangement the bits of the signal once more halves the range of uncertainty each time. If the first bit is 0 it selects the first group, whose probability is $\frac{1}{2}$. If it is 1 it selects the other groups, the sum of whose probabilities is also $\frac{1}{2}$. A signal of

| | |
|---|---|
| 0 | Sunday |
| 10 | Monday |
| 1100 | Tuesday |
| 1101 | Wednesday |
| 11100 | Thursday |
| 11101 | Friday |
| 11110 | Saturday |
| 11111 | Don't come |

Table B.3: Eight messages of different and the binary encoding of their selection

10 selects the second group, whose probability is $\frac{1}{4}$, and a signal of 11 selects groups 3 and 4, the sum of whose probabilities is also $\frac{1}{4}$. Table B.3 shows the encoding of all the messages.

Notice that the more probable a message, the fewer selections are needed to identify it, i.e the lower is its information content. This fact can be exploited in the encoding. We obtain a more efficient encoding by using codes of different lengths, and assigning short codes to the more frequent words. This is the basis of Morse code, which uses a two-state code — dots and dashes — to encode the letters of the Latin alphabet. The frequent letters $e$ and $t$ have the codes $\cdot$ and $-$ respectively; rare letters like $q$ and $z$ are $- - \cdot -$ and $- - \cdot\cdot$.

From the table we see that $H_1 = 1$ while $H_5 = H_6 = H_7 = H_8 = 5$. In general

$$H_i = -\log p_i$$

as before. The average information $H$ per sign is

$$\begin{aligned} H &= \frac{1}{2\times 1} + \frac{1}{4\times 2} + \frac{1}{16\times 4} + \frac{1}{16\times 4} + \frac{1}{32\times 5} + \frac{1}{32\times 5} + \frac{1}{32\times 5} + \frac{1}{32\times 5} \\ &= 2.125 \end{aligned}$$

The units are bits per sign. The general form of this is

$$H = -\sum_i p_i \log p_i$$

bits per sign. This is the same as equation B.1.

In the above argument the alphabet of signs conveniently divided into equally likely subgroups. A more general case is now considered. Up to now the examples have been chosen so that we could see the information content of individual signs. In this more general case we will need to deal with averages of signs. Suppose the alphabet consists of signs $a_1, a_2, \ldots, a_n$. We assume once more that the signs are independent. We need to estimate the probabilities $p_1$, $p_2, \ldots, p_n$, with which they occur. We do this by observing a sample of signs in a message. The larger the sample we observe, the better our estimates will be. This is actually only true if the signs are *ergodic* or *statistically stationary.* An ergodic sequence is one which is regular in the following sense: its parameters (the probabilities $p_i$ in this case) can be estimated by choosing the sample in any way. If one sample consists of every even-numbered sign, for example, and another of every odd-numbered sign, then the estimates of the probabilities will, in the long run, still agree if the sequence is ergodic.

Consider now an ensemble of all the samples of length $S$ that can be taken, where $S$ is a large number. The samples will differ from each other only in the order of the signs within them. Call the number of samples $N$. They have nearly equal probabilities, since the source is ergodic. Call this probability $p(S)$. $N = 1/p(S)$. Since the samples are equiprobable, the information content $H(S)$ for each sample is

$$H(S) = \log N = \log \frac{1}{p(S)} = -\log p(S)$$

Now $p(S)$ is equal to the product of the probabilities of the signs making up the sequence, since the signs are independent. The sequence consists of the signs $a_1, a_2, \ldots, a_n$, each occurring a number of times. Since the probability of $a_1$ is $p_1$, $a_1$ will occur about $Sp_1$ times, and so also for the other signs. That is,

$$p(S) = p_1^{Sp_1} \times p_2^{Sp_2} \times \cdots \times p_n^{Sp_n}$$

We now have

$$\begin{aligned} H(S) &= -\log p(S) \\ &= -\log(p_1^{Sp_1} \times p_2^{Sp_2} \times \cdots \times p_n^{Sp_n}) \\ &= -S \sum_i p_i \log p_i, \end{aligned}$$

by the addition rule for logarithms. The average information per sign is

$$\begin{aligned} H &= \frac{H(S)}{S} \\ &= -\sum_i p_i \log p_i \end{aligned}$$

This is the same expression as equation B.1.

We have obtained a more efficient encoding by taking into account the probabilities with which words (or in the case of Morse code, letters) occur in the English language. We have assumed that they occur independently of each other, i.e we have considered only their *a priori* probabilities. This is not a warranted assumption for English. We have assumed for example that the word *of* or the word *house* occur with their a priori probabilities in any position of a sentence. That is a false assumption. If a sentence starts with *The*, then the next word is more likely to be *house* than *of*. The frequency with which *house* follows *the* determines its *conditional probability*. This dependency extends over more than just the immediately preceding word. Consider the word *reason*. It may be followed by, among other words, *that* and *why*. The probabilities of these words are not much different. After the phrase *the reason*, these probabilities are not much changed. However, after the phrase *for the reason*, *that* is much more likely than *why*, because *for the reason why* is unidiomatic.

A process that produces a sequence of words, letters, or any other signs, in which their probabilities depend on foregoing signs, is called a *stochastic process*. The derivation of the formula for the information associated with a general stochastic process is beyond the scope of this appendix. The formula is the same as equation B.1.

## B.2 Perplexity

As used in the front end, the syllable networks operate to restrict the segments that can appear in the lattice. During segmentation the phonemes are at first hypothesised independently of each other, and can therefore appear in any order. When the segmenter operates without syllable networks, the string of top-scoring phonemes determine the segmentation. When the segmenter operates with syllable networks, the networks rule out certain phoneme sequences in the top-scoring string as illegal. Different phonemes, which are not in the top-scoring string, are then brought forward to replace them. Only phoneme sequences that form legal syllables appear in the final string. In this way the syllable networks allow some phoneme sequences, while disallowing the rest. We would like to measure the extent to which the networks do this.

The measure adopted in this thesis is *perplexity*, a concept first proposed by Jelinek (Bahl *et al.*, 1983). Jelinek used perplexity in connection with a grammar of the English language. A grammar restricts the order in which signs can appear in a string. It makes the signs more predictable than they would be in a random selection, and so reduces their entropy. Consider for example the 'grammar' imposed on letters and the interword space by written English words. This grammar stipulates, for example, that the strings 'anecdote' and 'noted ace' are words, but 'acdeenot' is not. Without this grammar the signs (26 letters plus space) are equiprobable, and their average entropy is $-\log \frac{1}{27} = 4.76$ bits per letter. With the grammar, according to Shannon, the entropy is approximately one bit per letter (Shannon, 1951)[1].

A good idea of the organising power of a grammar can be therefore be got from the entropy $H$ per symbol of the strings that it allows. It follows from equation B.1 above that strings with entropy $H$ can be thought of as being constructed from signs that are chosen equiprobably from an alphabet of size

[1]The fact that English text is about 80% redundant does not mean that four fifths of the letters can be removed to leave an intelligible text. It means rather that other encodings of the text can be found that are up to 80% more efficient.

$PP$, where

$$PP = 2^H$$

$PP$ is called the *perplexity* of the grammar.

$H$ can be expressed for syllable networks as follows.

$$H = \frac{\sum^{nodes} p_{node} \log b_{node}}{(\sum^{nodes} p_{node}) - 1}$$

where $p_{node}$ is the probability of arriving at the node, understood as the product of the transition probabilities up to this point. $p_0$, the probability of arriving at the first node, is 1. $b_{node}$ is the number of branches leaving this node.

# Appendix C

# Repair interval in Phoneme Lattices

In section 5.7 the effect of sequencing the phoneme lattice with the help of syllables and words is discussed. It is shown there that word sequencing does not reduce the oversegmentation as much as syllable sequencing. The explanation lies with the size of the sequencing unit — syllables or words — and how often the segments that fall within the unit are likely to be wrong. This latter measure is expressed as a *repair interval*, and this appendix calculates its size.

The repair interval is the distance between successive insertions or deletions (indels), as performed later by lexical access when it is constructing words out of the lattice. We recall from chapter 3 that lexical access needs to make three kinds of repairs to the phoneme lattice: substitutions, insertions and deletions. We may ignore substitutions, because they do not affect the sequencer; the sequencer has all the phoneme identities available to it for each segment (this fact is explained in section 3.2.2). On the other hand, insertions and deletions affect the work of the sequencer directly: the sequencer will not have not seen the inserted segments, and it should not have seen the deleted segments. The more such repairs prove necessary, i.e the smaller the repair interval, the greater is the extent to which the sequencer worked with incorrect segments.

Table C.1 is based on table 6.3 and shows the results of lexax runs using three grammar options on two kinds of phoneme lattice, for each of ATR and

| | Grammar | Database | Segm | % Words | Repair interval |
|---|---|---|---|---|---|
| 1 | zero | atr | reg | 47.54 | 4.37 |
| 2 | | | syll | 49.77 | 5.10 |
| 3 | | cyt | reg | 66.54 | 3.77 |
| 4 | | | syll | 67.41 | 4.03 |
| 5 | bigram | atr | reg | 93.77 | 4.46 |
| 6 | | | syll | 92.25 | 5.04 |
| 7 | | cyt | reg | 97.15 | 3.61 |
| 8 | | | syll | 96.76 | 3.98 |
| 9 | full | atr | reg | 99.77 | 3.45 |
| 10 | | | syll | 99.23 | 3.76 |
| 11 | | cyt | reg | 99.52 | 2.12 |
| 12 | | | syll | 100 | 2.22 |

Table C.1: Words correct and repair interval for speaker GSW produced by the baseline back end, reading regular and syllable-sequenced lattices.

CYT. The repair interval was calculated by counting the number of segments between successive indels, and dividing by the number of indels. The repair interval is consistently worse (smaller) for the cytology database, reflecting the fact that it is the 'open test' case. Note, incidentally, that there seems to be little correlation between words correct and the repair interval: the number of words correct in line 5 is nearly twice as high as in line 1, but its repair interval is only slightly lower.

The average repair interval for ATR is 4.36 and for CYT is 3.29.

# Appendix D

# Operation of Hidden Markov models

In continuous speech recognisers, hidden Markov models are used in the stage immediately before lexical lookup. It is a pattern matching stage that turns the encoded speech signal into phonemes. This stage is a sequence comparison, of an input string against stored strings, with the strings consisting of acoustic codes. In the CSTR system these codes are VQ indices (see section 3.2.1). The stored strings are called *models*. There is in principle one model for every phoneme, but in practice there are more than one, in order to model more accurately phonetic and acoustic variants of a phoneme. We shall not be concerned with such refinements in this appendix.

Consider a phoneme as a sequence of VQ codes. In the CSTR system a relatively short phoneme of, say, 30 milliseconds, will consist of six codes, since the frame length is 5 ms. The same phoneme spoken at a different time (i.e a different *token* of the phoneme) will consist of a similar but not identical sequence of codes, and will possibly be different in length. This is because different tokens of the same phoneme are seldom identical. To match such variants successfully against a model of the phoneme, we need to take into account their statistical distribution.

The codes as they appear in the sequence are not statistically independent, because the articulators (tongue, lips, and so on) move more slowly than the

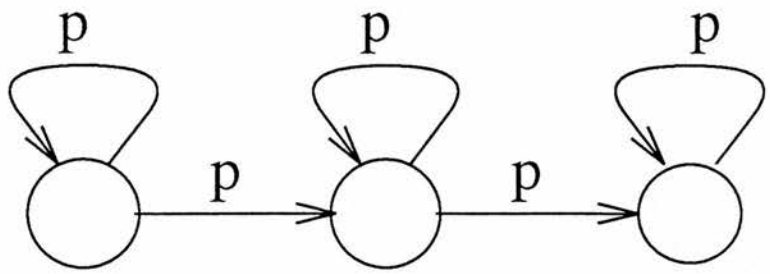



Figure D.1: A three-state Markov model with self-transitions. The *p*s are transition probabilities

encoding rate. This leads to the idea of a stochastic sequence, in which the probability of a VQ code depends not only on the code itself, but also on earlier codes. A sequence like this is produced by a stochastic *process*. A stochastic process in which the dependence is only on the immediately foregoing code, and not on the ones before that, is called a *Markov* process.

A Markov model consists of a sequence of *states*, normally three or five. In the CSTR system each model has three states. The states are linked left to right by means of *transitions*, and the states are usually also linked to themselves by means of *self-transitions*. Figure D.1 illustrates. Each state *emits* several codes, with different probabilities. Thus state 1 can emit several codes, each with its own probability. State two can emit several codes as well, which may be the same or different from the ones emitted by state 1. Each of the codes emitted at stage two will also have its own *emission probability*, as it is called. Similarly, stage 3 has its own set of codes and emission probabilities. Each transition or self-transition also has a different probability associated with it, called a *transition probability*.

An input phoneme, which consists of a string of codes, is recognised by matching it against a model. The model is traversed from left to right in the process, and a match score is calculated by multiplying the emission and transition probabilities. For concreteness, consider a stop phoneme, which consists of a period of silence, a burst release and a period of aspiration. The silence might be represented by the first three VQ codes, the burst by another code, and the aspiration by two codes, making six codes altogether. The states of the

model each have a probability distribution for all the legal VQ codes. The three states of the model describe, respectively, the beginning, middle section, and the end of the phoneme; we may assume here that the first state describes the portion of silence, the second state the burst release, and the third state the aspiration portion of the stop. The probabilities for silence codes in the first state will therefore be high, and the probabilities for all the other codes will be low.

Recognition begins with the first code of the phoneme, and at the first state of the model. The probability of the first silence code is found, and becomes the first factor in the score. Now there are two possibilities. The transition to the second state can be taken, or the self-transition back to the first state can be taken. In fact both possibilities will be tried, and the path that leads to the highest probability will be chosen. This is likely to be the self-transition, since the second code of the phoneme is also from the silence portion. The transition probability is multiplied into the score, followed by the emission probability of the second code. The third code of the phoneme is again from the silence portion, and a further self-transition is taken, with its probability multiplied into the score. After the emission probability of the third code has been mulitplied in, the fourth code is considered. This code describes the burst release. The transition to the second state is now taken, and the transition probability multiplied in. The emission probability of the burst code is found there and multiplied in. For the fifth code a transition is taken to state 3. Processing continues until all the codes have been looked up. The final score consists of the product of successive emission and transition probabilities.

It follows from the explanation above that a particular model can assign different scores to the same input. Naturally we will want to use the best of these scores when we come to compare it against the scores for the other models. However, to try out all the different ways that the input can be matched, and then pick out the highest-scoring one, would be prohibitively expensive. Luckily this exhaustive procedure is not necessary. The *Viterbi algorithm*, which is based on the principle of *dynamic programming*, will find the highest score

without considering all the possibilities. Dynamic programming is described in appendix D.1.

There remains to be discussed the way the models are obtained in the first place. This is done in a special training session, which is performed once, in advance of the recognition. Several examples of a phoneme are presented to the training program. These examples will not be identical, but will show slight variations. The training program needs to find the best way of assigning the emission and transition probabilities in the face of these variations. The algorithm that does this is called the *forward-backward* or *Baum-Welch* algorithm.

Further information about hidden Markov models can be found in (Rabiner & Juang, 1986) and (Cox, 1988).

## D.1 Summary

Speech is a progression of different sounds, which we can approximate as a sequence of separate acoustic events. We can use standard pattern-matching techniques to recognise unknown speech. But the events for a particular word are not constant over different speakers or different performances of the same speaker. This variability leads to partial matches, and hence goodness of fit. The language of stochastic processes provides us with a mathematical description of a sequence, and an optimisation programme for training.

# Appendix E

# Dynamic Programming

Dynamic programming is an algorithm for finding the best sequence of operations from all the sequences that can be formed, without actually trying all of them. This statement will become clear in the light of the example below. In this thesis dynamic programming (DP) is used in both the front end and the back end, as mentioned in chapter 3. The example below is from the back end, and concerns matching the input phoneme string against words in the lexicon. A particular string can be matched against a particular word in many different ways, and the dynamic programming algorithm is needed to find the best way of doing so. The DP algorithm is used several times like this, once for each word that is tried, and at the end the word that gave the best match emerges as the winner. The words in the lexicon are stored as strings of phonemes, and in this application are often called *templates*. When dynamic programming is used in template matching like this, it is called *dynamic time warping* (DTW).

In figure E.1 the input phonemes and the template phonemes have been written against rectangular axes. The inpout phonemes are /w h @@/, and the template phonemes are /h aa d/, for the word *hard*. Suppose we have started with the first input phoneme. There are three possibilities. We may match it against the first template phoneme, either directly or via a substitution. This is shown by the diagonal line. Secondly, we may skip the input phoneme and pay a deletion penalty; this is the horizontal line. Thirdly, we may decide to hold the input phoneme over until the next template phoneme. In this case we

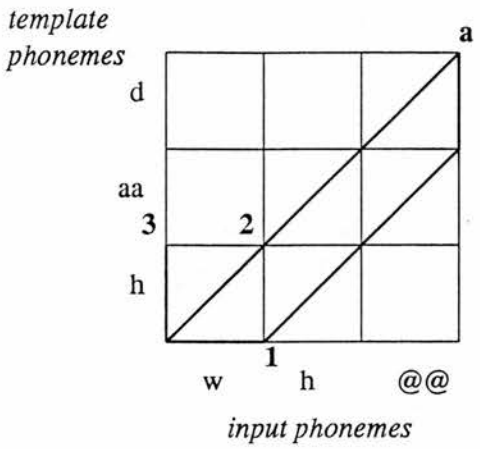



Figure E.1: Representation of lexax operations along two axes. A horizontal line denotes a deletion, a vertical line denotes an insertion, and a diagonal line denotes a substitution. Different sequences of operations are shown by different paths.

treat the current template phoneme as an insertion, and pay the appropriate penalty. This is the vertical line. We arrive at one of the points **1**, **2** or **3**. We now continue from any of these points, variously matching input and template phonemes, skipping an input phoneme because it needs to be deleted, or accepting a template phoneme as an insertion in the input. Two such paths are shown in the figure, both terminating at point **a**. (The reader may care to refer back to chaoter 3. The path that passes through point **1** corresponds to the operations in figure 3.5, and the path that passes through point **2** corresponds to the operations in figure 3.6.). These are just two ways of reaching **a**; there are many others, which are not shown in the diagram. How do we find the shortest (lowest-scoring) path? One way would be to trace out all the paths, compare their scores at the end, and pick the shortest. This would be be a computationally intensive way of doing it. The DTW algorithm allows us to do it more efficiently.

Consider another example, shown in figure E.2. Let point **b** lie along a path to the end which is still under construction. We are trying to find the shortest such path. The figure shows that point **b** has been reached in two ways. It was reached via an insertion from **2** or via a match from **3**. We need to retain only the shorter of the two. If our shortest path to the end does lie along **b**,

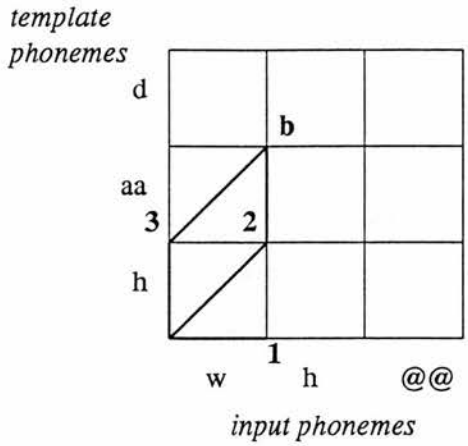



Figure E.2: Dynamic time warping algorithm. In tracing a forward path through the diagram, only the shortest path at each point needs to be kept.

it cannot include the longer piece, because then we would replace it with the shorter one, and obtain an even shorter path. This is the principle of dynamic programming, which was mentioned in chapter 1. It states that if a sequence of operations is optimal, then any subsequence of it is also optimal[1]. As we advance through the diagram, we need to keep only the shortest path for any point that we reach. The longer routes we may discard; at the end the shortest path cannot lie along them.

The computational complexity of this algorithm, that is, the minimum possible computation time that is needed, is proportional to $n^2$ (Sankoff & Kruskal, 1983, p 29), where $n$ is the number of phonemes in the input.

Sequence comparison using dynamic programming is used in many fields besides speech recognition. (Kruskal, 1983, pp 23–4) lists nine independent discoverers of this method, in the fields of molecular biology, speech processing and computer science. A good general description of its application to speech can be found in Kruskal and Liberman (1983). A survey of the different refinements can be found in (McInnes & Jack, 1988). Dynamic programming in the wider context of combinatorial optimisation is treated in (Pierre, 1969, 1986)

[1] Bellman (1957) expresses it thus: 'An optimal policy has the property that, whatever the initial state and the initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision' (for 'policy' read 'sequence of decisions').

and (Papadimitriou & Steiglitz, 1982).

# Appendix F

# Definitions of the extended apu sets

The nine APU sets were informally described in chapter 5. The *std* and *stdp* sets were completely characterised there. The others were sketched briefly. The following gives complete definitions.

The first column gives the serial number of the apu, by which it is known internally in the recognition system. The name in bold gives the apu name. The identity of the parent phoneme is readily apparent from this name. The first apu of the ext02 set is **i-b1**, for example. This identifies it as an allophone of /b/. The body of the table shows which clusters this allophone appears in. (x y — ) means a syllable-initial cluster and ( — x y) means a syllable-final cluster. Look at apu **i-b3**. The clusters show that this is an allophone of /b/ that appears syllable-initially before an /l/ or a /y/. In the ATR training data there were 11 examples where it appears before /l/ and 1 example where it appears before /y/. This gives a total of 12 tokens for this apu.

## F.1 Ext02

128 apus, including silence. Syllable-conditioned consonants. Total 36 stop allophones, 10 /s/ allophones, 5 /z/ allophones, etc.

The naming convention for this set is / i-xn / for x in syllable-initial cluster, and / f-xn / for x in syllable-final cluster. This convention was dropped in later sets, when I had to combine some of the syllable-initial and syllable-final definitions.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | **i-b1** | Total | 161 | | | | | |
| | | (b — ) | 161 | | | | | |
| 2 | **i-b2** | Total | 11 | | | | | |
| | | (b r — ) | 11 | | | | | |
| 3 | **i-b3** | Total | 12 | | | | | |
| | | (b l — ) | 11 | (b y — ) | 1 | | | |
| 49 | **f-b1** | Total | 18 | | | | | |
| | | ( — b) | 18 | | | | | |
| 50 | **f-b2** | Total | 10 | | | | | |
| | | ( — b d) | 5 | ( — b z) | 3 | ( — l b) | 1 | |
| | | ( — l b d) | 1 | | | | | |
| 4 | **i-d1** | Total | 105 | | | | | |
| | | (d — ) | 105 | | | | | |
| 5 | **i-d2** | Total | 10 | | | | | |
| | | (d r — ) | 8 | (d w — ) | 1 | (d y — ) | 1 | |
| 51 | **f-d1** | Total | 109 | | | | | |
| | | ( — d) | 101 | ( — l d) | 8 | | | |
| 52 | **f-d2** | Total | 24 | | | | | |
| | | ( — n d) | 21 | ( — m d) | 1 | ( — ng d) | 1 | |
| | | ( — l m d) | 1 | | | | | |
| 53 | **f-d3** | Total | 16 | | | | | |
| | | ( — v d) | 8 | ( — z d) | 4 | ( — dh d) | 2 | |
| | | ( — n z d) | 1 | ( — l v d) | 1 | | | |
| 54 | **f-d4** | Total | 14 | | | | | |
| | | ( — b d) | 5 | ( — g d) | 2 | ( — l b d) | 1 | |
| | | ( — jh d) | 4 | ( — l jh d) | 1 | ( — n jh d) | 1 | |
| 55 | **f-d5** | Total | 11 | | | | | |
| | | ( — d z) | 7 | ( — l d z) | 2 | ( — n d z) | 1 | |
| | | ( — d s t) | 1 | | | | | |
| 6 | **i-g1** | Total | 53 | | | | | |
| | | (g — ) | 53 | | | | | |

7 **i-g2** Total 21
(g r — ) 15 (g l — ) 4 (g w — ) 1
(g y — ) 1

56 **f-g1** Total 25
( — g) 19 ( — g z) 3 ( — g d) 2
( — ng g th s) 1

8 **i-p1** Total 86
(p — ) 86

9 **i-p2** Total 26
(p r — ) 26

10 **i-p3** Total 13
(p l — ) 13

11 **i-p4** Total 15
(s p — ) 9 (s p y — ) 1 (p y — ) 2
(s p r — ) 2 (s p l — ) 1

57 **f-p1** Total 35
( — p) 35

58 **f-p2** Total 23
( — m p) 5 ( — m p t) 1 ( — p s) 4
( — m p s) 1 ( — m p f s) 1 ( — m p s t) 1
( — m p t s) 1 ( — l p s) 2 ( — l p) 1
( — s p t) 1 ( — s p s) 2 ( — l p t) 1
( — p s t) 1 ( — p th) 1

12 **i-t1** Total 256
(t — ) 256

13 **i-t2** Total 31
(t r — ) 26 (t w — ) 4 (t y — ) 1

14 **i-t3** Total 36
(s t — ) 28 (s t r — ) 6 (s t y — )

259 **f-t1** Total 142
( — t) 142

60 **f-t2** Total 27
( — n t) 25 ( — m t) 2

61 **f-t3** Total 50
( — s t) 24 ( — sh t) 3 ( — ch t) 2
( — n ch t) 2 ( — l ch t) 1 ( — f t) 4
( — p s t) 1 ( — t s t) 1 ( — l t s t) 1
( — k s t) 1 ( — d s t) 1 ( — l s t) 2
( — n s t) 2 ( — ng s t) 1 ( — m p t s) 1
( — m p s t) 1 ( — l f t) 1 ( — th t) 1

62 **f-t4** Total 13
( — k t) 7 ( — l p t) 1 ( — l k t) 1
( — s p t) 1 ( — s k t) 1 ( — m p t) 1
( — ng k t) 1

63 **f-t5** Total 47
( — t s) 27 ( — t sh) 1 ( — t s t) 1
( — m t s) 1 ( — l t) 7 ( — l t s) 1
( — l t s t) 1 ( — f t s) 1 ( — s t s) 1
( — n t s) 3 ( — t th) 1 ( — n t th s) 1
( — t th s) 1

15 **i-k1** Total 125
(k — ) 125

16 **i-k2** Total 15
(k r — ) 15

17 **i-k3** Total 18
(k l — ) 10 (k w — ) 5 (k y — ) 3

18 **i-k4** Total 14
(s k — ) 10 (s k r — ) 2 (s k w — ) 2

64 **f-k1** Total 64
( — k) 64

65 **f-k2** Total 32
( — k s) 18 ( — l k) 2 ( — l k s) 1
( — s k s) 1 ( — k s t) 1 ( — ng k) 5
( — ng k s) 3 ( — ng k th) 1

66 **f-k3** Total 11
( — k t) 7 ( — l k t) 1 ( — s k) 1
( — s k t) 1 ( — ng k t) 1

19 **i-z1** Total 32
(z — ) 32

67 **f-z1** Total 189
( — z) 189

68 **f-z2** Total 22
( — n z) 15 ( — m z) 5 ( — ng z) 1
( — l m z) 1

69 **f-z3** Total 16
( — b z) 3 ( — d z) 7 ( — g z) 3
( — l d z) 2 ( — n d z) 1

70 **f-z4** Total 17
( — l z) 5 ( — v z) 4 ( — l v z) 2
( — n z d) 1 ( — z d) 4 ( — dh z) 1

20 **i-zh1** Total 12
(zh — ) 12

71 **f-zh1** Total 9
( — zh) 7 ( — n zh) 2

21 **i-jh1** Total 42
(jh — ) 40 (jh y — ) 2

72 **f-jh1** Total 24
( — jh) 24

73 **f-jh2** Total 11
( — n jh) 4 ( — jh d) 4 ( — l jh d) 1
( — n jh d) 1 ( — l jh) 1

22 **i-s1** Total 140
(s — ) 139 (s f — ) 1

23 **i-s2** Total 48
(s p — ) 9 (s p y — ) 1 (s t — ) 28
(s k — ) 10

24 **i-s3** Total 15
(s p r — ) 2 (s t r — ) 6 (s k r — ) 2
(s k w — ) 2 (s t y — ) 2 (s p l — ) 1

25 **i-s4** Total 17
(s w — ) 6 (s l — ) 4 (s y — ) 1
(s m — ) 5 (s n — ) 1

74 **f-s1** Total 113
( — s) 113

75 **f-s2** Total 50
( — p s) 4 ( — m p s) 1 ( — t s) 27
( — k s) 18

76 **f-s3** Total 42
( — s t) 24 ( — s k) 1 ( — s k t) 1
( — s p t) 1 ( — t s t) 1 ( — m p s t) 1
( — n s t) 2 ( — ng s t) 1 ( — l s t) 2
( — d s t) 1 ( — k s t) 1 ( — p s t) 1
( — l t s t) 1 ( — s p s) 2 ( — s k s) 1
( — s t s) 1

77 **f-s4** Total 11
( — n s) 9 ( — m s) 1 ( — l s) 1

78 **f-s5** Total 17
( — l p s) 2 ( — l t s) 1 ( — l k s) 1
( — ng k s) 3 ( — n t s) 3 ( — m t s) 1
( — f t s) 1 ( — m p t s) 1 ( — s p s) 2
( — s t s) 1 ( — s k s) 1

79 **f-s6** Total 12
( — th s) 3 ( — f s) 2 ( — n th s) 1
( — l th s) 1 ( — t th s) 1 ( — ng g th s) 1
( — n t th s) 1 ( — m p f s) 1 ( — l f s) 1

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 26 | **i-sh1** | Total | 53 | | | | | | | | |
| | | (sh — ) | 52 | (sh r — ) | 1 | | | | | | |
| 80 | **f-sh1** | Total | 23 | | | | | | | | |
| | | ( — sh) | 18 | ( — sh t) | 3 | ( — t sh) | 1 | | | | |
| | | ( — l sh) | 1 | | | | | | | | |
| 27 | **i-ch1** | Total | 36 | | | | | | | | |
| | | (ch — ) | 35 | (ch y — ) | 1 | | | | | | |
| 81 | **f-ch1** | Total | 32 | | | | | | | | |
| | | ( — ch) | 24 | ( — ch t) | 2 | ( — n ch t) | 2 | | | | |
| | | ( — n ch) | 2 | ( — l ch t) | 1 | ( — l ch) | 1 | | | | |
| 28 | **i-v1** | Total | 60 | | | | | | | | |
| | | (v — ) | 55 | (v y — ) | 3 | (v r — ) | 2 | | | | |
| 82 | **f-v1** | Total | 79 | | | | | | | | |
| | | ( — v) | 79 | | | | | | | | |
| 83 | **f-v2** | Total | 16 | | | | | | | | |
| | | ( — v d) | 8 | ( — v z) | 4 | ( — l v z) | 2 | | | | |
| | | ( — l v) | 1 | ( — l v d) | 1 | | | | | | |
| 29 | **i-dh1** | Total | 301 | | | | | | | | |
| | | (dh — ) | 301 | | | | | | | | |
| 84 | **f-dh1** | Total | 31 | | | | | | | | |
| | | ( — dh) | 28 | ( — dh d) | 2 | ( — dh z) | 1 | | | | |
| 30 | **i-f1** | Total | 107 | | | | | | | | |
| | | (f — ) | 107 | | | | | | | | |
| 31 | **i-f2** | Total | 28 | | | | | | | | |
| | | (f r — ) | 19 | (f l — ) | 5 | (f y — ) | 3 | | | | |
| | | (s f — ) | 1 | | | | | | | | |
| 85 | **f-f1** | Total | 31 | | | | | | | | |
| | | ( — f) | 31 | | | | | | | | |
| 86 | **f-f2** | Total | 17 | | | | | | | | |
| | | ( — f t) | 4 | ( — f t s) | 1 | ( — l f t) | 1 | | | | |
| | | ( — m f) | 3 | ( — l f) | 3 | ( — f s) | 2 | | | | |
| | | ( — l f s) | 1 | ( — f th) | 1 | ( — m p f s) | 1 | | | | |
| 32 | **i-th1** | Total | 18 | | | | | | | | |
| | | (th — ) | 18 | | | | | | | | |
| 33 | **i-th2** | Total | 9 | | | | | | | | |
| | | (th r — ) | 7 | (th y — ) | 1 | (th w — ) | 1 | | | | |
| 87 | **f-th1** | Total | 17 | | | | | | | | |
| | | ( — th) | 17 | | | | | | | | |

88 **f-th2** Total 17

( — th s) 3 ( — t th s) 1 ( — p th) 1
( — t th) 1 ( — ng g th s) 1 ( — ng k th) 1
( — n t th s) 1 ( — n th s) 1 ( — l th s) 1
( — l th) 2 ( — f th) 1 ( — th t) 1
( — n th) 1 ( — m th) 1

34 **i-h1** Total 83

(h — ) 82 (h y — ) 1

35 **i-l1** Total 136

(l — ) 135 (l y — ) 1

36 **i-l2** Total 48

(p l — ) 13 (k l — ) 10 (b l — ) 11
(g l — ) 4 (s p l — ) 1 (f l — ) 5
(s l — ) 4

89 **f-l1** Total 98

( — l) 98

90 **f-l2** Total 29

( — l p) 1 ( — l p t) 1 ( — l t) 7
( — l t s t) 1 ( — l k) 2 ( — l k t) 1
( — l b) 1 ( — l b d) 1 ( — l d) 8
( — l p s) 2 ( — l t s) 1 ( — l k s) 1
( — l d z) 2

91 **f-l3** Total 18

( — l z) 5 ( — l v z) 2 ( — l v) 1
( — l v d) 1 ( — l ch t) 1 ( — l ch) 1
( — l jh d) 1 ( — l jh) 1 ( — l m) 2
( — l m d) 1 ( — l n) 1 ( — l m z) 1

92 **f-l4** Total 12

( — l f) 3 ( — l f t) 1 ( — l th) 2
( — l sh) 1 ( — l s) 1 ( — l s t) 2
( — l f s) 1 ( — l th s) 1

04 **l=** Total 29

(l= ) 29

37 **i-r1** Total 118

(r — ) 118

38 **i-r2** Total 101

(p r — ) 26 (t r — ) 26 (k r — ) 15
(b r — ) 11 (d r — ) 8 (g r — ) 15

39 **i-r3** Total 29

(f r — ) 19 (th r — ) 7 (sh r — ) 1
(v r — ) 2

40 **i-r4** Total 10

(s p r — ) 2 (s t r — ) 6 (s k r — ) 2

93 **f-r1** Total 26
( — r) 26

05 **r=** Total 2
(r= ) 2

41 **i-m1** Total 134
(m — ) 127 (s m — ) 5 (m y — ) 2

94 **f-m1** Total 79
( — m) 79

95 **f-m2** Total 15
( — m p) 5 ( — m p t) 1 ( — m p t s) 1
( — m p s t) 1 ( — m t) 2 ( — m d) 1
( — m p f s) 1 ( — m p s) 1 ( — m t s) 1
( — l m d) 1

96 **f-m3** Total 13
( — m z) 5 ( — l m z) 1 ( — m f) 3
( — m th) 1 ( — m s) 1 ( — l m) 2

06 **m=** Total 6
(m= ) 6

42 **i-n1** Total 111
(n — ) 106 (n y — ) 4 (s n — ) 1

97 **f-n1** Total 278
( — n) 278

98 **f-n2** Total 51
( — n t) 25 ( — n d) 21 ( — n t s) 3
( — n t th s) 1 ( — n d z) 1

99 **f-n3** Total 15
( — n z) 15

100 **f-n4** Total 11
( — n s) 9 ( — n s t) 2

101 **f-n5** Total 15
( — n jh) 4 ( — n jh d) 1 ( — n zh) 2
( — n z d) 1 ( — n ch t) 2 ( — n ch) 2
( — n th s) 1 ( — n th) 1 ( — l n) 1

107 **n=** Total 34
(n= ) 34

102 **f-ng1** Total 64
( — ng) 64

103 **f-ng2** Total 14
( — ng k) 5 ( — ng d) 1 ( — ng k s) 3
( — ng k t) 1 ( — ng s t) 1 ( — ng g th s) 1
( — ng k th) 1 ( — ng z) 1

43 **i-y1** Total 42
(y — ) 42

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 44 | i-y2 | Total | 12 | | | | | |
| | | (p y — ) | 2 | (s p y — ) | 1 | (t y — ) | 1 | |
| | | (k y — ) | 3 | (b y — ) | 1 | (d y — ) | 1 | |
| | | (g y — ) | 1 | (s t y — ) | 2 | | | |
| 45 | i-y3 | Total | 10 | | | | | |
| | | (n y — ) | 4 | (m y — ) | 2 | (v y — ) | 3 | |
| | | (h y — ) | 1 | | | | | |
| 46 | i-y4 | Total | 9 | | | | | |
| | | (f y — ) | 3 | (th y — ) | 1 | (s y — ) | 1 | |
| | | (jh y — ) | 2 | (ch y — ) | 1 | (l y — ) | 1 | |
| 47 | i-w1 | Total | 191 | | | | | |
| | | (w — ) | 191 | | | | | |
| 48 | i-w2 | Total | 20 | | | | | |
| | | (t w — ) | 4 | (k w — ) | 5 | (d w — ) | 1 | |
| | | (g w — ) | 1 | (s w — ) | 6 | (s k w — ) | 2 | |
| | | (th w — ) | 1 | | | | | |
| 108 | ii | Total | 267 | | | | | |
| | | (ii) | 267 | | | | | |
| 109 | e | Total | 190 | | | | | |
| | | (e) | 190 | | | | | |
| 110 | a | Total | 126 | | | | | |
| | | (a) | 126 | | | | | |
| 111 | uu | Total | 103 | | | | | |
| | | (uu) | 103 | | | | | |
| 112 | u | Total | 37 | | | | | |
| | | (u) | 37 | | | | | |
| 113 | oo | Total | 105 | | | | | |
| | | (oo) | 105 | | | | | |
| 114 | o | Total | 110 | | | | | |
| | | (o) | 110 | | | | | |
| 115 | aa | Total | 105 | | | | | |
| | | (aa) | 105 | | | | | |
| 116 | i | Total | 519 | | | | | |
| | | (i) | 519 | | | | | |
| 117 | @@ | Total | 55 | | | | | |
| | | (@@) | 55 | | | | | |
| 118 | uh | Total | 108 | | | | | |
| | | (uh) | 108 | | | | | |
| 119 | ei | Total | 127 | | | | | |
| | | (ei) | 127 | | | | | |
| 120 | ou | Total | 106 | | | | | |
| | | (ou) | 106 | | | | | |

| | | | |
|---|---|---|---|
| 121 | au | Total | 59 |
| | | (au) | 59 |
| 122 | ai | Total | 163 |
| | | (ai) | 163 |
| 123 | oi | Total | 27 |
| | | (oi) | 27 |
| 124 | i@ | Total | 38 |
| | | (i@) | 38 |
| 125 | e@ | Total | 42 |
| | | (e@) | 42 |
| 126 | u@ | Total | 10 |
| | | (u@) | 10 |
| 127 | @ | Total | 967 |
| | | (@) | 967 |

## F.2 Ext03

137 apus, including silence. Syllable-conditioned consonants as ext02, but stops further divided into released and unreleased. Released /d/ called /d d2 d3/ etc. Unreleased /d/ called /u-d u-d1 u-d2/ etc. Total 45 stop allophones (not $36 \times 2 = 72$ as one would expect, because some classes had to be combined to make up the numbers).

```
1   b    Total     152
         (b — )    152
2   b2   Total      10
         (b r — )  10
3   b3   Total      12
         (b l — )  11      (b y — )  1
4   b4   Total      22
         ( — b)    17      ( — b d)  2
         ( — b z)  3
5   u-b  Total      15
         (u-b — )  9       ( — u-b d)  3      ( — l u-b)  1
         ( — u-b)  1       ( — l u-b d)  1
6   d    Total     85
         (d — )    85
7   d2   Total      8
         (d r — )  6       (d w — )  1        (d y — )  1
8   d3   Total     64
         ( — d)    59      ( — l d)  5
9   d4   Total      18
         ( — n d)  16      ( — m d)  1        ( — ng d)  1
10  d5   Total      10
         ( — v d)  6       ( — z d)  3        ( — n z d)  1
11  d6   Total       8
         ( — b d)  2       ( — u-b d)  3      ( — g d)  1
         ( — u-g d)  1     ( — l u-b d)  1
12  d7   Total       8
         ( — d z)  5       ( — l d z)  2      ( — n d z)  1
13  u-d  Total      22
         (u-d — )  20      (u-d r — )  2
14  u-d2         Total     51
                 ( — u-d)  42      ( — n u-d)  5      ( — l m u-d)  1
    ( — l u-d)   3
```

15 u-d3 Total 15
( — v u-d) 2 ( — dh u-d) 2 ( — z u-d) 1
( — l v u-d) 1 ( — jh u-d) 4 ( — l jh u-d) 1
( — n jh u-d) 1 ( — u-d z) 2 ( — u-d s t) 1

16 g Total 51
(g — ) 51

17 g2 Total 21
(g r — ) 15 (g l — ) 4 (g w — ) 1
(g y — ) 1

18 g3 Total 18
( — g) 13 ( — g z) 3 ( — g d) 1
( — ng g th s) 1

19 u-g Total 9
( — u-g) 6 (u-g — ) 2 ( — u-g d) 1

20 p Total 86
(p — ) 86

21 p2 Total 26
(p r — ) 26

22 p3 Total 13
(p l — ) 13

23 p4 Total 14
(s p — ) 9 (p y — ) 2 (s p r — ) 2
(s p l — ) 1

24 p5 Total 30
( — p) 30

25 p6 Total 18
( — m p) 5 ( — p s) 4 ( — m p s t) 1
( — l p s) 2 ( — l p) 1 ( — s p s) 2
( — l p t) 1 ( — p s t) 1 ( — p th) 1

26 u-p Total 9
( — u-p) 5 ( — m u-p u-t s) 1 ( — s u-p u-t) 1
( — m u-p f s) 1 ( — m u-p t) 1

27 t Total 248
(t — ) 248

28 t2 Total 30
(t r — ) 25 (t w — ) 4 (t y — ) 1

29 t3 Total 31
(s t — ) 26 (s t r — ) 4 (s t y — ) 1

30 t4 Total 60
( — t) 60

31 t5 Total 21
( — n t) 19 ( — m t) 2

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 32 | t6 | Total | 28 | | | | | | | | |
| | | ( — s t) | 14 | ( — sh t) | 2 | ( — ch t) | 2 | | | | |
| | | ( — l ch t) | 1 | ( — f t) | 3 | ( — p s t) | 1 | | | | |
| | | ( — l s t) | 1 | ( — u-d s t) | 1 | ( — ng s t) | 1 | | | | |
| | | ( — m p s t) | 1 | ( — l f t) | 1 | | | | | | |
| 33 | t7 | Total | 10 | | | | | | | | |
| | | ( — k t) | 3 | ( — u-k t) | 2 | ( — l p t) | 1 | | | | |
| | | ( — l k t) | 1 | ( — s u-k t) | 1 | ( — m u-p t) | 1 | | | | |
| | | ( — ng k t) | 1 | | | | | | | | |
| 34 | t8 | Total | 29 | | | | | | | | |
| | | ( — t s) | 19 | ( — t sh) | 1 | ( — t s u-t) | 1 | | | | |
| | | ( — m t s) | 1 | ( — l t) | 3 | ( — l t s) | 1 | | | | |
| | | ( — f t s) | 1 | ( — n t s) | 1 | ( — t th) | 1 | | | | |
| 35 | u-t | Total | 14 | | | | | | | | |
| | | (u-t — ) | 8 | (s u-t — ) | 2 | (s u-t r — ) | 2 | | | | |
| | | (u-t r — ) | 1 | (s u-t y — ) | 1 | | | | | | |
| 36 | u-t2 | Total | 85 | | | | | | | | |
| | | ( — u-t) | 82 | ( — k u-t) | 1 | ( — u-k u-t) | 1 | | | | |
| | | ( — s u-p u-t) | 1 | | | | | | | | |
| 37 | u-t3 | Total | 24 | | | | | | | | |
| | | ( — n u-t) | 6 | ( — n u-t s) | 2 | ( — l u-t s u-t) | 1 | | | | |
| | | ( — u-t s) | 8 | ( — u-t th s) | 1 | ( — l u-t) | 4 | | | | |
| | | ( — l u-t s u-t) | 1 | ( — n u-t th s) | 1 | | | | | | |
| 38 | u-t4 | Total | 22 | | | | | | | | |
| | | ( — s u-t) | 10 | ( — n s u-t) | 2 | ( — l s u-t) | 1 | | | | |
| | | ( — sh u-t) | 1 | ( — t s u-t) | 1 | ( — s u-t s) | 1 | | | | |
| | | ( — m u-p u-t s) | 1 | ( — f u-t) | 1 | ( — n ch u-t) | 2 | | | | |
| | | ( — k s u-t) | 1 | ( — th u-t) | 1 | | | | | | |
| 39 | k | Total | 123 | | | | | | | | |
| | | (k — ) | 123 | | | | | | | | |
| 40 | k2 | Total | 15 | | | | | | | | |
| | | (k r — ) | 15 | | | | | | | | |
| 41 | k3 | Total | 18 | | | | | | | | |
| | | (k l — ) | 10 | (k w — ) | 5 | (k y — ) | 3 | | | | |
| 42 | k4 | Total | 14 | | | | | | | | |
| | | (s k — ) | 10 | (s k r — ) | 2 | (s k w — ) | 2 | | | | |
| 43 | k5 | Total | 54 | | | | | | | | |
| | | ( — k) | 54 | | | | | | | | |
| 44 | k6 | Total | 38 | | | | | | | | |
| | | ( — k s) | 18 | ( — l k) | 2 | ( — k t) | 3 | | | | |
| | | ( — k u-t) | 1 | ( — l k t) | 1 | ( — l k s) | 1 | | | | |
| | | ( — s k) | 1 | ( — s k s) | 1 | ( — k s u-t) | 1 | | | | |
| | | ( — ng k) | 4 | ( — ng k t) | 1 | ( — ng k s) | 3 | | | | |
| | | ( — ng k th) | 1 | | | | | | | | |

45 u-k Total 17
( — u-k) 10 ( — u-k t) 2 (u-k — ) 2
( — s u-k t) 1 ( — ng u-k) 1 ( — u-k u-t) 1

46 z Total 32
(z — ) 32

47 z2 Total 189
( — z) 189

48 z3 Total 22
( — n z) 15 ( — m z) 5 ( — ng z) 1
( — l m z) 1

49 z4 Total 16
( — b z) 3 ( — d z) 5 ( — u-d z) 2
( — g z) 3 ( — l d z) 2 ( — n d z) 1

50 z5 Total 17
( — l z) 5 ( — v z) 4 ( — l v z) 2
( — n z d) 1 ( — z d) 3 ( — z u-d) 1
( — dh z) 1

51 zh Total 12
(zh — ) 12

52 zh2 Total 9
( — zh) 7 ( — n zh) 2

53 jh Total 42
(jh — ) 40 (jh y — ) 2

54 jh2 Total 24
( — jh) 24

55 jh3 Total 11
( — n jh) 4 ( — jh u-d) 4 ( — l jh) 1
( — l jh u-d) 1 ( — n jh u-d) 1

56 s Total 140
(s — ) 139 (s f — ) 1

57 s2 Total 47
(s p — ) 9 (s t — ) 26 (s u-t — ) 2
(s k — ) 10

58 s3 Total 15
(s p r — ) 2 (s t r — ) 4 (s u-t r — ) 2
(s k r — ) 2 (s k w — ) 2 (s t y — ) 1
(s u-t y — ) 1 (s p l — ) 1

59 s4 Total 17
(s w — ) 6 (s l — ) 4 (s y — ) 1
(s m — ) 5 (s n — ) 1

60 s5 Total 113
( — s) 113

61 s6 Total 49
( — p s) 4 ( — t s) 19 ( — u-t s) 8
( — k s) 18

62 s7 Total 42
( — s t) 14 ( — s u-t) 10 ( — s k) 1
( — s u-k t) 1 ( — s u-p u-t) 1 ( — t s u-t) 1
( — m p s t) 1 ( — n s u-t) 2 ( — ng s t) 1
( — l s t) 1 ( — l s u-t) 1 ( — u-d s t) 1
( — k s u-t) 1 ( — p s t) 1 ( — l u-t s u-t) 1
( — s p s) 2 ( — s u-t s) 1 ( — s k s) 1

63 s8 Total 11
( — n s) 9 ( — m s) 1 ( — l s) 1

64 s9 Total 17
( — l p s) 2 ( — l t s) 1 ( — l k s) 1
( — ng k s) 3 ( — n t s) 1 ( — n u-t s) 2
( — m t s) 1 ( — f t s) 1 ( — m u-p u-t s) 1
( — s p s) 2 ( — s u-t s) 1 ( — s k s) 1

65 s10 Total 12
( — th s) 3 ( — f s) 2 ( — n th s) 1
( — l th s) 1 ( — u-t th s) 1 ( — ng g th s) 1
( — n u-t th s) 1 ( — m u-p f s) 1 ( — l f s) 1

66 sh Total 53
(sh — ) 52 (sh r — ) 1

67 sh2 Total 23
( — sh) 18 ( — sh t) 2 ( — sh u-t) 1
( — t sh) 1 ( — l sh) 1

68 ch Total 36
(ch — ) 35 (ch y — ) 1

69 ch2 Total 32
( — ch) 24 ( — ch t) 2 ( — n ch u-t) 2
( — n ch) 2 ( — l ch t) 1 ( — l ch) 1

70 v Total 60
(v — ) 55 (v y — ) 3 (v r — ) 2

71 v2 Total 79
( — v) 79

72 v3 Total 16
( — v d) 6 ( — v u-d) 2 ( — v z) 4
( — l v z) 2 ( — l v) 1 ( — l v u-d) 1

73 dh Total 301
(dh — ) 301

74 dh2 Total 31
( — dh) 28 ( — dh u-d) 2 ( — dh z) 1

75 f Total 107
(f — ) 107

76 f2 Total 28
(f r — ) 19 (f l — ) 5 (f y — ) 3
(s f — ) 1

77 f3 Total 31
( — f) 31

78 f4 Total 17
( — f t) 3 ( — f u-t) 1 ( — f t s) 1
( — l f t) 1 ( — m f) 3 ( — l f) 3
( — f s) 2 ( — l f s) 1 ( — f th) 1
( — m u-p f s) 1

79 th Total 18
(th — ) 18

80 th2 Total 9
(th r — ) 7 (th y — ) 1 (th w — ) 1

81 th3 Total 17
( — th) 17

82 th4 Total 17
( — th s) 3 ( — u-t th s) 1 ( — p th) 1
( — t th) 1 ( — ng g th s) 1 ( — ng k th) 1
( — n u-t th s) 1 ( — n th s) 1 ( — l th s) 1
( — l th) 2 ( — f th) 1 ( — th u-t) 1
( — n th) 1 ( — m th) 1

83 h Total 83
(h — ) 82 (h y — ) 1

84 l Total 136
(l — ) 135 (l y — ) 1

85 l2 Total 48
(p l — ) 13 (k l — ) 10 (b l — ) 11
(g l — ) 4 (s p l — ) 1 (f l — ) 5
(s l — ) 4

86 l3 Total 98
( — l) 98

87 l4 Total 29
( — l p) 1 ( — l p t) 1 ( — l t) 3
( — l u-t) 4 ( — l u-t s u-t) 1 ( — l k) 2
( — l k t) 1 ( — l u-b) 1 ( — l u-b d) 1
( — l d) 5 ( — l u-d) 3 ( — l p s) 2
( — l t s) 1 ( — l k s) 1 ( — l d z) 2

88 l5 Total 18
( — l z) 5 ( — l v z) 2 ( — l v) 1
( — l v u-d) 1 ( — l ch t) 1 ( — l ch) 1
( — l jh u-d) 1 ( — l jh) 1 ( — l m) 2
( — l m u-d) 1 ( — l n) 1 ( — l m z) 1

89 l6 Total 12
( — l f) 3 ( — l f t) 1 ( — l th) 2
( — l sh) 1 ( — l s) 1 ( — l s t) 1
( — l s u-t) 1 ( — l f s) 1 ( — l th s) 1

90 l= Total 29
(l= ) 29

91 r Total 118
(r — ) 118

92 r2 Total 101
(p r — ) 26 (t r — ) 25 (u-t r — ) 1
(k r — ) 15 (b r — ) 11 (d r — ) 6
(u-d r — ) 2 (g r — ) 15

93 r3 Total 29
(f r — ) 19 (th r — ) 7 (sh r — ) 1
(v r — ) 2

94 r4 Total 10
(s p r — ) 2 (s t r — ) 4 (s u-t r — ) 2
(s k r — ) 2

95 r5 Total 26
( — r) 26

96 r= Total 2
(r= ) 2

97 m Total 134
(m — ) 127 (s m — ) 5 (m y — ) 2

98 m2 Total 79
( — m) 79

99 m3 Total 14
( — m p) 5 ( — m u-p t) 1 ( — m u-p u-t s) 1
( — m p s t) 1 ( — m t) 2 ( — m d) 1
( — m u-p f s) 1 ( — m t s) 1 ( — l m u-d) 1

100 m4 Total 13
( — m z) 5 ( — l m z) 1 ( — m f) 3
( — m th) 1 ( — m s) 1 ( — l m) 2

101 m= Total 6
(m= ) 6

102 n Total 111
(n — ) 106 (n y — ) 4 (s n — ) 1

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 103 | n2 | Total | 278 | | | | | | | | |
| | | ( — n) | 278 | | | | | | | | |
| 104 | n3 | Total | 51 | | | | | | | | |
| | | ( — n t) | 19 | ( — n u-t) | 6 | ( — n d) | 16 | | | | |
| | | ( — n u-d) | 5 | ( — n t s) | 1 | ( — n u-t s) | 2 | | | | |
| | | ( — n u-t th s) | 1 | ( — n d z) | 1 | | | | | | |
| 105 | n4 | Total | 15 | | | | | | | | |
| | | ( — n z) | 15 | | | | | | | | |
| 106 | n5 | Total | 11 | | | | | | | | |
| | | ( — n s) | 9 | ( — n s u-t) | 2 | | | | | | |
| 107 | n6 | Total | 15 | | | | | | | | |
| | | ( — n jh) | 4 | ( — n jh u-d) | 1 | ( — n zh) | 2 | | | | |
| | | ( — n z d) | 1 | ( — n ch u-t) | 2 | ( — n ch) | 2 | | | | |
| | | ( — n th s) | 1 | ( — n th) | 1 | ( — l n) | 1 | | | | |
| 108 | n= | Total | 34 | | | | | | | | |
| | | (n= ) | 34 | | | | | | | | |
| 109 | ng | Total | 64 | | | | | | | | |
| | | ( — ng) | 64 | | | | | | | | |
| 110 | ng2 | Total | 14 | | | | | | | | |
| | | ( — ng k) | 4 | ( — ng u-k) | 1 | ( — ng d) | 1 | | | | |
| | | ( — ng k s) | 3 | ( — ng k t) | 1 | ( — ng s t) | 1 | | | | |
| | | ( — ng g th s) | 1 | ( — ng k th) | 1 | ( — ng z) | 1 | | | | |
| 111 | y | Total | 42 | | | | | | | | |
| | | (y — ) | 42 | | | | | | | | |
| 112 | y2 | Total | 11 | | | | | | | | |
| | | (p y — ) | 2 | (t y — ) | 1 | (k y — ) | 3 | | | | |
| | | (b y — ) | 1 | (d y — ) | 1 | (g y — ) | 1 | | | | |
| | | (s t y — ) | 1 | (s u-t y — ) | 1 | | | | | | |
| 113 | y3 | Total | 10 | | | | | | | | |
| | | (n y — ) | 4 | (m y — ) | 2 | (v y — ) | 3 | | | | |
| | | (h y — ) | 1 | | | | | | | | |
| 114 | y4 | Total | 9 | | | | | | | | |
| | | (f y — ) | 3 | (th y — ) | 1 | (s y — ) | 1 | | | | |
| | | (jh y — ) | 2 | (ch y — ) | 1 | (l y — ) | 1 | | | | |
| 115 | w | Total | 191 | | | | | | | | |
| | | (w — ) | 191 | | | | | | | | |
| 116 | w2 | Total | 20 | | | | | | | | |
| | | (t w — ) | 4 | (k w — ) | 5 | (d w — ) | 1 | | | | |
| | | (g w — ) | 1 | (s w — ) | 6 | (s k w — ) | 2 | | | | |
| | | (th w — ) | 1 | | | | | | | | |
| 117 | ii | Total | 267 | | | | | | | | |
| | | (ii) | 267 | | | | | | | | |

| | | | |
|---|---|---|---|
| 118 | e | Total | 190 |
| | | (e) | 190 |
| 119 | a | Total | 126 |
| | | (a) | 126 |
| 120 | uu | Total | 103 |
| | | (uu) | 103 |
| 121 | u | Total | 37 |
| | | (u) | 37 |
| 122 | oo | Total | 105 |
| | | (oo) | 105 |
| 123 | o | Total | 110 |
| | | (o) | 110 |
| 124 | aa | Total | 105 |
| | | (aa) | 105 |
| 125 | i | Total | 519 |
| | | (i) | 519 |
| 126 | @@ | Total | 55 |
| | | (@@) | 55 |
| 127 | uh | Total | 108 |
| | | (uh) | 108 |
| 128 | ei | Total | 127 |
| | | (ei) | 127 |
| 129 | ou | Total | 106 |
| | | (ou) | 106 |
| 130 | au | Total | 59 |
| | | (au) | 59 |
| 131 | ai | Total | 163 |
| | | (ai) | 163 |
| 132 | oi | Total | 27 |
| | | (oi) | 27 |
| 133 | i@ | Total | 38 |
| | | (i@) | 38 |
| 134 | e@ | Total | 42 |
| | | (e@) | 42 |
| 135 | u@ | Total | 10 |
| | | (u@) | 10 |
| 136 | @ | Total | 967 |
| | | (@) | 967 |

## F.3 Ext04

104 apus, including silence. Syllable-conditioned consonants as ext02, but instead of syllable-conditioned stops, stops are divided into released and unreleased only. Unreleased /b/ is called /u-b/, etc. Total 12 stop allophones.

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | b | Total | 196 | | | | | | | | |
| | | (b — ) | 152 | (b r — ) | 10 | (b l — ) | 11 | | | | |
| | | (b y — ) | 1 | ( — b) | 17 | ( — b d) | 2 | | | | |
| | | ( — b z) | 3 | | | | | | | | |
| 2 | u-b | Total | 15 | | | | | | | | |
| | | (u-b — ) | 9 | ( — u-b d) | 3 | ( — l u-b) | 1 | | | | |
| | | ( — u-b) | 1 | ( — l u-b d) | 1 | | | | | | |
| 3 | d | Total | 201 | | | | | | | | |
| | | (d — ) | 85 | (d r — ) | 6 | (d w — ) | 1 | | | | |
| | | (d y — ) | 1 | ( — d) | 59 | ( — l d) | 5 | | | | |
| | | ( — n d) | 16 | ( — m d) | 1 | ( — ng d) | 1 | | | | |
| | | ( — v d) | 6 | ( — z d) | 3 | ( — n z d) | 1 | | | | |
| | | ( — b d) | 2 | ( — u-b d) | 3 | ( — g d) | 1 | | | | |
| | | ( — u-g d) | 1 | ( — l u-b d) | 1 | ( — d z) | 5 | | | | |
| | | ( — l d z) | 2 | ( — n d z) | 1 | | | | | | |
| 4 | u-d | Total | 88 | | | | | | | | |
| | | (u-d — ) | 20 | (u-d r — ) | 2 | ( — u-d) | 42 | | | | |
| | | ( — n u-d) | 5 | ( — l m u-d) | 1 | ( — l u-d) | 3 | | | | |
| | | ( — v u-d) | 2 | ( — dh u-d) | 2 | ( — z u-d) | 1 | | | | |
| | | ( — l v u-d) | 1 | ( — jh u-d) | 4 | ( — l jh u-d) | 1 | | | | |
| | | ( — n jh u-d) | 1 | ( — u-d z) | 2 | ( — u-d s t) | 1 | | | | |
| 5 | g | Total | 90 | | | | | | | | |
| | | (g — ) | 51 | (g r — ) | 15 | (g l — ) | 4 | | | | |
| | | (g w — ) | 1 | (g y — ) | 1 | ( — g) | 13 | | | | |
| | | ( — g z) | 3 | ( — g d) | 1 | ( — ng g th s) | 1 | | | | |
| 6 | u-g | Total | 9 | | | | | | | | |
| | | ( — u-g) | 6 | (u-g — ) | 2 | ( — u-g d) | 1 | | | | |
| 7 | p | Total | 187 | | | | | | | | |
| | | (p — ) | 86 | (p r — ) | 26 | (p l — ) | 13 | | | | |
| | | (s p — ) | 9 | (p y — ) | 2 | (s p r — ) | 2 | | | | |
| | | (s p l — ) | 1 | ( — m p) | 5 | ( — p) | 30 | | | | |
| | | ( — p s) | 4 | ( — m p s t) | 1 | ( — l p s) | 2 | | | | |
| | | ( — l p) | 1 | ( — s p s) | 2 | ( — l p t) | 1 | | | | |
| | | ( — p s t) | 1 | ( — p th) | 1 | | | | | | |
| 8 | u-p | Total | 9 | | | | | | | | |
| | | ( — u-p) | 5 | ( — m u-p u-t s) | 1 | ( — s u-p u-t) | 1 | | | | |
| | | ( — m u-p f s) | 1 | ( — m u-p t) | 1 | | | | | | |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 9 | t | Total | 457 | | | | | |
| | | (t — ) | 248 | (t r — ) | 25 | (t w — ) | 4 | |
| | | (t y — ) | 1 | (s t — ) | 26 | (s t r — ) | 4 | |
| | | (s t y — ) | 1 | ( — t) | 60 | ( — n t) | 19 | |
| | | ( — m t) | 2 | ( — s t) | 14 | ( — sh t) | 2 | |
| | | ( — ch t) | 2 | ( — l ch t) | 1 | ( — f t) | 3 | |
| | | ( — p s t) | 1 | ( — l s t) | 1 | ( — u-d s t) | 1 | |
| | | ( — ng s t) | 1 | ( — m p s t) | 1 | ( — l f t) | 1 | |
| | | ( —k t ) | 3 | ( — u-k t) | 2 | ( — l p t) | 1 | |
| | | ( — l k t) | 1 | ( — s u-k t) | 1 | ( — m u-p t) | 1 | |
| | | ( — ng k t) | 1 | ( — t s) | 19 | ( — t sh) | 1 | |
| | | ( — t s u-t) | 1 | ( — m t s) | 1 | ( — l t) | 3 | |
| | | ( — l t s) | 1 | ( — f t s) | 1 | ( — n t s) | 1 | |
| | | ( — t th) | 1 | | | | | |
| 10 | u-t | Total | 145 | | | | | |
| | | (u-t — ) | 8 | (s u-t — ) | 2 | (s u-t r — ) | 2 | |
| | | (u-t r — ) | 1 | (s u-t y — ) | 1 | ( — u-t) | 82 | |
| | | ( — k u-t) | 1 | ( — u-k u-t) | 1 | ( — s u-p u-t) | 1 | |
| | | ( — n u-t) | 6 | ( — n u-t s) | 2 | ( — l u-t s u-t) | 1 | |
| | | ( — u-t s) | 8 | ( — u-t th s) | 1 | ( — l u-t) | 4 | |
| | | ( — l u-t s u-t) | 1 | ( — n u-t th s) | 1 | ( — s u-t) | 10 | |
| | | ( — n s u-t) | 2 | ( — l s u-t) | 1 | ( — sh u-t) | 1 | |
| | | ( — t s u-t) | 1 | ( — s u-t s) | 1 | ( — m u-p u-t s) | 1 | |
| | | ( — f u-t) | 1 | ( — n ch u-t) | 2 | ( — k s u-t) | 1 | |
| | | ( — th u-t) | 1 | | | | | |
| 11 | k | Total | 262 | | | | | |
| | | (k — ) | 123 | (k r — ) | 15 | (k l — ) | 10 | |
| | | (k w — ) | 5 | (k y — ) | 3 | (s k — ) | 10 | |
| | | (s k r — ) | 2 | (s k w — ) | 2 | ( — k) | 54 | |
| | | ( — k s) | 18 | ( — l k) | 2 | ( — k t) | 3 | |
| | | ( — k u-t) | 1 | ( — l k t) | 1 | ( — l k s) | 1 | |
| | | ( — s k) | 1 | ( — s k s) | 1 | ( — k s u-t) | 1 | |
| | | ( — ng k) | 4 | ( — ng k t) | 1 | ( — ng k s) | 3 | |
| | | ( — ng k th) | 1 | | | | | |
| 12 | u-k | Total | 17 | | | | | |
| | | ( — u-k) | 10 | ( — u-k t) | 2 | (u-k — ) | 2 | |
| | | ( — s u-k t) | 1 | ( — ng u-k) | 1 | ( — u-k u-t) | 1 | |
| 13 | z | Total | 32 | | | | | |
| | | (z — ) | 32 | | | | | |
| 14 | z2 | Total | 189 | | | | | |
| | | ( — z) | 189 | | | | | |
| 15 | z3 | Total | 22 | | | | | |
| | | ( — n z) | 15 | ( — m z) | 5 | ( — ng z) | 1 | |
| | | ( — l m z) | 1 | | | | | |

16 z4 Total 16
( — b z) 3 ( — d z) 5 ( — u-d z) 2
( — g z) 3 ( — l d z) 2 ( — n d z) 1

17 z5 Total 17
( — l z) 5 ( — v z) 4 ( — l v z) 2
( — n z d) 1 ( — z d) 3 ( — z u-d) 1
( — dh z) 1

18 zh Total 12
(zh — ) 12

19 zh2 Total 9
( — zh) 7 ( — n zh) 2

20 jh Total 42
(jh — ) 40 (jh y — ) 2

21 jh2 Total 24
( — jh) 24

22 jh3 Total 11
( — n jh) 4 ( — jh u-d) 4 ( — l jh) 1
( — l jh u-d) 1 ( — n jh u-d) 1

23 s Total 140
(s — ) 139 (s f — ) 1

24 s2 Total 47
(s p — ) 9 (s t — ) 26 (s u-t — ) 2
(s k — ) 10

25 s3 Total 15
(s p r — ) 2 (s t r — ) 4 (s u-t r — ) 2
(s k r — ) 2 (s k w — ) 2 (s t y — ) 1
(s u-t y — ) 1 (s p l — ) 1

26 s4 Total 17
(s w — ) 6 (s l — ) 4 (s y — ) 1
(s m — ) 5 (s n — ) 1

27 s5 Total 113
( — s) 113

28 s6 Total 49
( — p s) 4 ( — t s) 19 ( — u-t s) 8
( — k s) 18

29 s7 Total 42
( — s t) 14 ( — s u-t) 10 ( — s k) 1
( — s u-k t) 1 ( — s u-p u-t) 1 ( — t s u-t) 1
( — m p s t) 1 ( — n s u-t) 2 ( — ng s t) 1
( — l s t) 1 ( — l s u-t) 1 ( — u-d s t) 1
( — k s u-t) 1 ( — p s t) 1 ( — l u-t s u-t) 1
( — s p s) 2 ( — s u-t s) 1 ( — s k s) 1

30 s8 Total 11
( — n s) 9 ( — m s) 1 ( — l s) 1

31 s9 Total 17
( — l p s) 2 ( — l t s) 1 ( — l k s) 1
( — ng k s) 3 ( — n t s) 1 ( — n u-t s) 2
( — m t s) 1 ( — f t s) 1 ( — m u-p u-t s) 1
( — s p s) 2 ( — s u-t s) 1 ( — s k s) 1

32 s10 Total 12
( — th s) 3 ( — f s) 2 ( — n th s) 1
( — l th s) 1 ( — u-t th s) 1 ( — ng g th s) 1
( — n u-t th s) 1 ( — m u-p f s) 1 ( — l f s) 1

33 sh Total 53
(sh — ) 52 (sh r — ) 1

34 sh2 Total 23
( — sh) 18 ( — sh t) 2 ( — sh u-t) 1
( — t sh) 1 ( — l sh) 1

35 ch Total 36
(ch — ) 35 (ch y — ) 1

36 ch2 Total 32
( — ch) 24 ( — ch t) 2 ( — n ch u-t) 2
( — n ch) 2 ( — l ch t) 1 ( — l ch) 1

37 v Total 60
(v — ) 55 (v y — ) 3 (v r — ) 2

38 v2 Total 79
( — v) 79

39 v3 Total 16
( — v d) 6 ( — v u-d) 2 ( — v z) 4
( — l v z) 2 ( — l v) 1 ( — l v u-d) 1

40 dh Total 301
(dh — ) 301

41 dh2 Total 31
( — dh) 28 ( — dh u-d) 2 ( — dh z) 1

42 f Total 107
(f — ) 107

43 f2 Total 28
(f r — ) 19 (f l — ) 5 (f y — ) 3
(s f — ) 1

44 f3 Total 31
( — f) 31

45 f4 Total 17
( — f t) 3 ( — f u-t) 1 ( — f t s) 1
( — l f t) 1 ( — m f) 3 ( — l f) 3
( — f s) 2 ( — l f s) 1 ( — f th) 1
( — m u-p f s) 1

46 th Total 18
(th — ) 18

47 th2 Total 9
(th r — ) 7 (th y — ) 1 (th w — ) 1

48 th3 Total 17
( — th) 17

49 th4 Total 17
( — th s) 3 ( — u-t th s) 1 ( — p th) 1
( — t th) 1 ( — ng g th s) 1 ( — ng k th) 1
( — n u-t th s) 1 ( — n th s) 1 ( — l th s) 1
( — l th) 2 ( — f th) 1 ( — th u-t) 1
( — n th) 1 ( — m th) 1

50 h Total 83
(h — ) 82 (h y — ) 1

51 l Total 136
(l — ) 135 (l y — ) 1

52 l2 Total 48
(p l — ) 13 (k l — ) 10 (b l — ) 11
(g l — ) 4 (s p l — ) 1 (f l — ) 5
(s l — ) 4

53 l3 Total 98
( — l) 98

54 l4 Total 29
( — l p) 1 ( — l p t) 1 ( — l t) 3
( — l u-t) 4 ( — l u-t s u-t) 1 ( — l k) 2
( — l k t) 1 ( — l u-b) 1 ( — l u-b d) 1
( — l d) 5 ( — l u-d) 3 ( — l p s) 2
( — l t s) 1 ( — l k s) 1 ( — l d z) 2

55 l5 Total 18
( — l z) 5 ( — l v z) 2 ( — l v) 1
( — l v u-d) 1 ( — l ch t) 1 ( — l ch) 1
( — l jh u-d) 1 ( — l jh) 1 ( — l m) 2
( — l m u-d) 1 ( — l n) 1 ( — l m z) 1

56 l6 Total 12
( — l f) 3 ( — l f t) 1 ( — l th) 2
( — l sh) 1 ( — l s) 1 ( — l s t) 1
( — l s u-t) 1 ( — l f s) 1 ( — l th s) 1

57 l= Total 29
(l= ) 29

58 r Total 118
(r — ) 118

59 r2 Total 101
(p r — ) 26 (t r — ) 25 (u-t r — ) 1
(k r — ) 15 (b r — ) 11 (d r — ) 6
(u-d r — ) 2 (g r — ) 15

60 r3 Total 29
(f r — ) 19 (th r — ) 7 (sh r — ) 1
(v r — ) 2

61 r4 Total 10
(s p r — ) 2 (s t r — ) 4 (s u-t r — ) 2
(s k r — ) 2

62 r5 Total 26
( — r) 26

63 r= Total 2
(r= ) 2

64 m Total 134
(m — ) 127 (s m — ) 5 (m y — ) 2

65 m2 Total 79
( — m) 79

66 m3 Total 14
( — m p) 5 ( — m u-p t) 1 ( — m u-p u-t s) 1
( — m p s t) 1 ( — m t) 2 ( — m d) 1
( — m u-p f s) 1 ( — m t s) 1 ( — l m u-d) 1

67 m4 Total 13
( — m z) 5 ( — l m z) 1 ( — m f) 3
( — m th) 1 ( — m s) 1 ( — l m) 2

68 m= Total 6
(m= ) 6

69 n Total 111
(n — ) 106 (n y — ) 4 (s n — ) 1

70 n2 Total 278
( — n) 278

71 n3 Total 51
( — n t) 19 ( — n u-t) 6 ( — n d) 16
( — n u-d) 5 ( — n t s) 1 ( — n u-t s) 2
( — n u-t th s) 1 ( — n d z) 1

72 n4 Total 15
( — n z) 15

73 n5 Total 11
( — n s) 9 ( — n s u-t) 2

74 n6 Total 15
( — n jh) 4 ( — n jh u-d) 1 ( — n zh) 2
( — n z d) 1 ( — n ch u-t) 2 ( — n ch) 2
( — n th s) 1 ( — n th) 1 ( — l n) 1

75 n= Total 34
(n= ) 34

76 ng Total 64
( — ng) 64

77 ng2 Total 14
( — ng k) 4 ( — ng u-k) 1 ( — ng d) 1
( — ng k s) 3 ( — ng k t) 1 ( — ng s t) 1
( — ng g th s) 1 ( — ng k th) 1 ( — ng z) 1

78 y Total 42
(y — ) 42

79 y2 Total 11
(p y — ) 2 (t y — ) 1 (k y — ) 3
(b y — ) 1 (d y — ) 1 (g y — ) 1
(s t y — ) 1 (s u-t y — ) 1

80 y3 Total 10
(n y — ) 4 (m y — ) 2 (v y — ) 3
(h y — ) 1

81 y4 Total 9
(f y — ) 3 (th y — ) 1 (s y — ) 1
(jh y — ) 2 (ch y — ) 1 (l y — ) 1

82 w Total 191
(w — ) 191

83 w2 Total 20
(t w — ) 4 (k w — ) 5 (d w — ) 1
(g w — ) 1 (s w — ) 6 (s k w — ) 2
(th w — ) 1

84 ii Total 267
(ii) 267

85 e Total 190
(e) 190

86 a Total 126
(a) 126

87 uu Total 103
(uu) 103

88 u Total 37
(u) 37

89 oo Total 105
(oo) 105

90 o Total 110
(o) 110
91 aa Total 105
(aa) 105
92 i Total 519
(i) 519
93 @@ Total 55
(@@) 55
94 uh Total 108
(uh) 108
95 ei Total 127
(ei) 127
96 ou Total 106
(ou) 106
97 au Total 59
(au) 59
98 ai Total 163
(ai) 163
99 oi Total 27
(oi) 27
100 i@ Total 38
(i@) 38
101 e@ Total 42
(e@) 42
102 u@ Total 10
(u@) 10
103 @ Total 967
(@) 967

## F.4 Ext05

79 apus, including silence. Syllable-conditioned stops only, identical to those in ext02. Total 36 stop allophones. Non-stop apus are as those in stdp.

```
1   b    Total     161
         (b — )    161
2   b2   Total      11
         (b r — )  11
3   b3   Total      12
         (b l — )  11      (b y — )  1
4   b4   Total      18
         ( — b)    18
5   b5   Total      10
         ( — b d)   5      ( — b z)  3      ( — l b)  1
         ( — l b d) 1
6   d    Total     105
         (d — )    105
7   d2   Total      10
         (d r — )   8      (d w — )  1      (d y — )  1
8   d3   Total     109
         ( — d)    101     ( — l d)  8
9   d4   Total      24
         ( — n d)  21      ( — m d)  1      ( — ng d)  1
         ( — l m d) 1
10  d5   Total      16
         ( — v d)   8      ( — z d)  4      ( — dh d)  2
         ( — n z d) 1      ( — l v d) 1
11  d6   Total      14
         ( — b d)   5      ( — g d)  2      ( — l b d)  1
         ( — jh d)  4      ( — l jh d) 1    ( — n jh d)  1
12  d7   Total      11
         ( — d z)   7      ( — l d z) 2     ( — n d z)  1
         ( — d s t) 1
13  g    Total     53
         (g — )    53
14  g2   Total      21
         (g r — )  15      (g l — )  4      (g w — )  1
         (g y — )  1
15  g3   Total      25
         ( — g)    19      ( — g z)  3      ( — g d)  2
         ( — ng g th s) 1
16  p    Total     86
         (p — )    86
```

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 17 | p2 | Total | 26 | | | | | | | | |
| | | (p r — ) | 26 | | | | | | | | |
| 18 | p3 | Total | 13 | | | | | | | | |
| | | (p l — ) | 13 | | | | | | | | |
| 19 | p4 | Total | 14 | | | | | | | | |
| | | (s p — ) | 9 | (p y — ) | 2 | (s p r — ) | 2 | | | | |
| | | (s p l — ) | 1 | | | | | | | | |
| 20 | p5 | Total | 35 | | | | | | | | |
| | | ( — p) | 35 | | | | | | | | |
| 21 | p6 | Total | 22 | | | | | | | | |
| | | ( — m p) | 5 | ( — m p t) | 1 | ( — p s) | 4 | | | | |
| | | ( — m p f s) | 1 | ( — m p s t) | 1 | ( — m p t s) | 1 | | | | |
| | | ( — l p s) | 2 | ( — l p) | 1 | ( — s p t) | 1 | | | | |
| | | ( — s p s) | 2 | ( — l p t) | 1 | ( — p s t) | 1 | | | | |
| | | ( — p th) | 1 | | | | | | | | |
| 22 | t | Total | 256 | | | | | | | | |
| | | (t — ) | 256 | | | | | | | | |
| 23 | t2 | Total | 31 | | | | | | | | |
| | | (t r — ) | 26 | (t w — ) | 4 | (t y — ) | 1 | | | | |
| 24 | t3 | Total | 36 | | | | | | | | |
| | | (s t — ) | 28 | (s t r — ) | 6 | (s t y — ) | 2 | | | | |
| 25 | t4 | Total | 142 | | | | | | | | |
| | | ( — t) | 142 | | | | | | | | |
| 26 | t5 | Total | 27 | | | | | | | | |
| | | ( — n t) | 25 | ( — m t) | 2 | | | | | | |
| 27 | t6 | Total | 50 | | | | | | | | |
| | | ( — s t) | 24 | ( — sh t) | 3 | ( — ch t) | 2 | | | | |
| | | ( — n ch t) | 2 | ( — l ch t) | 1 | ( — f t) | 4 | | | | |
| | | ( — p s t) | 1 | ( — t s t) | 1 | ( — l t s t) | 1 | | | | |
| | | ( — k s t) | 1 | ( — d s t) | 1 | ( — l s t) | 2 | | | | |
| | | ( — n s t) | 2 | ( — ng s t) | 1 | ( — m p t s) | 1 | | | | |
| | | ( — m p s t) | 1 | ( — l f t) | 1 | ( — th t) | 1 | | | | |
| 28 | t7 | Total | 13 | | | | | | | | |
| | | ( — k t) | 7 | ( — l p t) | 1 | ( — l k t) | 1 | | | | |
| | | ( — s p t) | 1 | ( — s k t) | 1 | ( — m p t) | 1 | | | | |
| | | ( — ng k t) | 1 | | | | | | | | |
| 29 | t8 | Total | 47 | | | | | | | | |
| | | ( — t s) | 27 | ( — t sh) | 1 | ( — t s t) | 1 | | | | |
| | | ( — m t s) | 1 | ( — l t) | 7 | ( — l t s) | 1 | | | | |
| | | ( — l t s t) | 1 | ( — f t s) | 1 | ( — s t s) | 1 | | | | |
| | | ( — n t s) | 3 | ( — t th) | 1 | ( — n t th s) | 1 | | | | |
| | | ( — t th s) | 1 | | | | | | | | |

30 k Total 125
(k — ) 125

31 k2 Total 15
(k r — ) 15

32 k3 Total 18
(k l — ) 10 (k w — ) 5 (k y — ) 3

33 k4 Total 14
(s k — ) 10 (s k r — ) 2 (s k w — ) 2

34 k5 Total 64
( — k) 64

35 k6 Total 32
( — k s) 18 ( — l k) 2 ( — l k s) 1
( — s k s) 1 ( — k s t) 1 ( — ng k) 5
( — ng k s) 3 ( — ng k th) 1

36 k7 Total 11
( — k t) 7 ( — l k t) 1 ( — s k) 1
( — s k t) 1 ( — ng k t) 1

37 z Total 276
(z — ) 32 ( — z) 189 ( — n z) 15
( — m z) 5 ( — ng z) 1 ( — l m z) 1
( — b z) 3 ( — d z) 7 ( — g z) 3
( — l d z) 2 ( — n d z) 1 ( — l z) 5
( — v z) 4 ( — l v z) 2 ( — n z d) 1
( — z d) 4 ( — dh z) 1

38 zh Total 21
(zh — ) 12 ( — zh) 7 ( — n zh) 2

39 jh Total 77
(jh — ) 40 (jh y — ) 2 ( — jh) 24
( — n jh) 4 ( — jh d) 4 ( — l jh d) 1
( — n jh d) 1 ( — l jh) 1

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 40 | s | Total | 463 | | | | | |
| | | (s — ) | 139 | (s f — ) | 1 | (s p — ) | 9 | |
| | | (s t — ) | 28 | (s k — ) | 10 | (s p r — ) | 2 | |
| | | (s t r — ) | 6 | (s k r — ) | 2 | (s k w — ) | 2 | |
| | | (s t y — ) | 2 | (s p l — ) | 1 | (s w — ) | 6 | |
| | | (s l — ) | 4 | (s y — ) | 1 | (s m — ) | 5 | |
| | | (s n — ) | 1 | ( — p s) | 4 | ( — t s) | 27 | |
| | | ( — k s) | 18 | ( — s) | 113 | ( — s t) | 24 | |
| | | ( — s k) | 1 | ( — s k t) | 1 | ( — s p t) | 1 | |
| | | ( — t s t) | 1 | ( — m p s t) | 1 | ( — n s t) | 2 | |
| | | ( — ng s t) | 1 | ( — l s t) | 2 | ( — d s t) | 1 | |
| | | ( — k s t) | 1 | ( — p s t) | 1 | ( — l t s t) | 1 | |
| | | ( — s p s) | 2 | ( — s k s) | 1 | ( — s t s) | 1 | |
| | | ( — n s) | 9 | ( — m s) | 1 | ( — l s) | 1 | |
| | | ( — l p s) | 2 | ( — l t s) | 1 | ( — l k s) | 1 | |
| | | ( — ng k s) | 3 | ( — n t s) | 3 | ( — m t s) | 1 | |
| | | ( — f t s) | 1 | ( — m p t s) | 1 | ( — s p s) | 2 | |
| | | ( — s t s) | 1 | ( — s k s) | 1 | ( — th s) | 3 | |
| | | ( — f s) | 2 | ( — n th s) | 1 | ( — l th s) | 1 | |
| | | ( — t th s) | 1 | ( — ng g th s) | 1 | ( — n t th s) | 1 | |
| | | ( — m p f s) | 1 | ( — l f s) | 1 | | | |
| 41 | sh | Total | 76 | | | | | |
| | | (sh — ) | 52 | (sh r — ) | 1 | ( — sh) | 18 | |
| | | ( — sh t) | 3 | ( — t sh) | 1 | ( — l sh) | 1 | |
| 42 | ch | Total | 68 | | | | | |
| | | (ch — ) | 35 | (ch y — ) | 1 | ( — ch) | 24 | |
| | | ( — ch t) | 2 | ( — n ch t) | 2 | ( — n ch) | 2 | |
| | | ( — l ch t) | 1 | ( — l ch) | 1 | | | |
| 43 | v | Total | 155 | | | | | |
| | | (v — ) | 55 | (v y — ) | 3 | (v r — ) | 2 | |
| | | ( — v) | 79 | ( — v d) | 8 | ( — v z) | 4 | |
| | | ( — l v z) | 2 | ( — l v) | 1 | ( — l v d) | 1 | |
| 44 | dh | Total | 332 | | | | | |
| | | (dh — ) | 301 | ( — dh) | 28 | ( — dh d) | 2 | |
| | | ( — dh z) | 1 | | | | | |
| 45 | f | Total | 183 | | | | | |
| | | (f — ) | 107 | (f r — ) | 19 | (f l — ) | 5 | |
| | | (f y — ) | 3 | (s f — ) | 1 | ( — f) | 31 | |
| | | ( — f t) | 4 | ( — f t s) | 1 | ( — l f t) | 1 | |
| | | ( — m f) | 3 | ( — l f) | 3 | ( — f s) | 2 | |
| | | ( — l f s) | 1 | ( — f th) | 1 | ( — m p f s) | 1 | |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 46 | th | Total | 61 | | | | | | |
| | | (th — ) | 18 | (th r — ) | 7 | (th y — ) | 1 | | |
| | | (th w — ) | 1 | ( — th) | 17 | ( — th s) | 3 | | |
| | | ( — t th s) | 1 | ( — p th) | 1 | ( — t th) | 1 | | |
| | | ( — ng g th s) | 1 | ( — ng k th) | 1 | ( — n t th s) | 1 | | |
| | | ( — n th s) | 1 | ( — l th s) | 1 | ( — l th) | 2 | | |
| | | ( — f th) | 1 | ( — th t) | 1 | ( — n th) | 1 | | |
| | | ( — m th) | 1 | | | | | | |
| 47 | h | Total | 83 | | | | | | |
| | | (h — ) | 82 | (h y — ) | 1 | | | | |
| 48 | l | Total | 341 | | | | | | |
| | | (l — ) | 135 | (l y — ) | 1 | (p l — ) | 13 | | |
| | | (k l — ) | 10 | (b l — ) | 11 | (g l — ) | 4 | | |
| | | (s p l — ) | 1 | (f l — ) | 5 | (s l — ) | 4 | | |
| | | ( — l) | 98 | ( — l p) | 1 | ( — l p t) | 1 | | |
| | | ( — l t) | 7 | ( — l t s t) | 1 | ( — l k) | 2 | | |
| | | ( — l k t) | 1 | ( — l b) | 1 | ( — l b d) | 1 | | |
| | | ( — l d) | 8 | ( — l p s) | 2 | ( — l t s) | 1 | | |
| | | ( — l k s) | 1 | ( — l d z) | 2 | ( — l z) | 5 | | |
| | | ( — l v z) | 2 | ( — l v) | 1 | ( — l v d) | 1 | | |
| | | ( — l ch t) | 1 | ( — l ch) | 1 | ( — l jh d) | 1 | | |
| | | ( — l jh) | 1 | ( — l m) | 2 | ( — l m d) | 1 | | |
| | | ( — l n) | 1 | ( — l m z) | 1 | ( — l f) | 3 | | |
| | | ( — l f t) | 1 | ( — l th) | 2 | ( — l sh) | 1 | | |
| | | ( — l s) | 1 | ( — l s t) | 2 | ( — l f s) | 1 | | |
| | | ( — l th s) | 1 | | | | | | |
| 49 | l= | Total | 29 | | | | | | |
| | | (l= ) | 29 | | | | | | |
| 50 | r | Total | 284 | | | | | | |
| | | (r — ) | 118 | (p r — ) | 26 | (t r — ) | 26 | | |
| | | (k r — ) | 15 | (b r — ) | 11 | (d r — ) | 8 | | |
| | | (g r — ) | 15 | (f r — ) | 19 | (th r — ) | 7 | | |
| | | (sh r — ) | 1 | (v r — ) | 2 | (s p r — ) | 2 | | |
| | | (s t r — ) | 6 | (s k r — ) | 2 | ( — r) | 26 | | |
| 51 | r= | Total | 2 | | | | | | |
| | | (r= ) | 2 | | | | | | |
| 52 | m | Total | 240 | | | | | | |
| | | (m — ) | 127 | (s m — ) | 5 | (m y — ) | 2 | | |
| | | ( — m) | 79 | ( — m p) | 5 | ( — m p t) | 1 | | |
| | | ( — m p t s) | 1 | ( — m p s t) | 1 | ( — m t) | 2 | | |
| | | ( — m d) | 1 | ( — m p f s) | 1 | ( — m t s) | 1 | | |
| | | ( — l m d) | 1 | ( — m z) | 5 | ( — l m z) | 1 | | |
| | | ( — m f) | 3 | ( — m th) | 1 | ( — m s) | 1 | | |
| | | ( — l m) | 2 | | | | | | |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 53 | m= | Total | 6 | | | | | | |
| | | (m= ) | 6 | | | | | | |
| 54 | n | Total | 481 | | | | | | |
| | | (n — ) | 106 | (n y — ) | 4 | (s n — ) | 1 | | |
| | | ( — n) | 278 | ( — n t) | 25 | ( — n d) | 21 | | |
| | | ( — n t s) | 3 | ( — n t th s) | 1 | ( — n d z) | 1 | | |
| | | ( — n z) | 15 | ( — n s) | 9 | ( — n s t) | 2 | | |
| | | ( — n jh) | 4 | ( — n jh d) | 1 | ( — n zh) | 2 | | |
| | | ( — n z d) | 1 | ( — n ch t) | 2 | ( — n ch) | 2 | | |
| | | ( — n th s) | 1 | ( — n th) | 1 | ( — l n) | 1 | | |
| 55 | n= | Total | 34 | | | | | | |
| | | (n= ) | 34 | | | | | | |
| 56 | ng | Total | 78 | | | | | | |
| | | ( — ng) | 64 | ( — ng k) | 5 | ( — ng d) | 1 | | |
| | | ( — ng k s) | 3 | ( — ng k t) | 1 | ( — ng s t) | 1 | | |
| | | ( — ng g th s) | 1 | ( — ng k th) | 1 | ( — ng z) | 1 | | |
| 57 | y | Total | 72 | | | | | | |
| | | (y — ) | 42 | (p y — ) | 2 | (t y — ) | 1 | | |
| | | (k y — ) | 3 | (b y — ) | 1 | (d y — ) | 1 | | |
| | | (g y — ) | 1 | (s t y — ) | 2 | (n y — ) | 4 | | |
| | | (m y — ) | 2 | (v y — ) | 3 | (h y — ) | 1 | | |
| | | (f y — ) | 3 | (th y — ) | 1 | (s y — ) | 1 | | |
| | | (jh y — ) | 2 | (ch y — ) | 1 | (l y — ) | 1 | | |
| 58 | w | Total | 211 | | | | | | |
| | | (w — ) | 191 | (t w — ) | 4 | (k w — ) | 5 | | |
| | | (d w — ) | 1 | (g w — ) | 1 | (s w — ) | 6 | | |
| | | (s k w — ) | 2 | (th w — ) | 1 | | | | |
| 59 | ii | Total | 267 | | | | | | |
| | | (ii) | 267 | | | | | | |
| 60 | e | Total | 190 | | | | | | |
| | | (e) | 190 | | | | | | |
| 61 | a | Total | 126 | | | | | | |
| | | (a) | 126 | | | | | | |
| 62 | uu | Total | 103 | | | | | | |
| | | (uu) | 103 | | | | | | |
| 63 | u | Total | 37 | | | | | | |
| | | (u) | 37 | | | | | | |
| 64 | oo | Total | 105 | | | | | | |
| | | (oo) | 105 | | | | | | |
| 65 | o | Total | 110 | | | | | | |
| | | (o) | 110 | | | | | | |
| 66 | aa | Total | 105 | | | | | | |
| | | (aa) | 105 | | | | | | |

| | | | |
|---|---|---|---|
| 67 | i | Total | 519 |
| | | (i) | 519 |
| 68 | @@ | Total | 55 |
| | | (@@) | 55 |
| 69 | uh | Total | 108 |
| | | (uh) | 108 |
| 70 | ei | Total | 127 |
| | | (ei) | 127 |
| 71 | ou | Total | 106 |
| | | (ou) | 106 |
| 72 | au | Total | 59 |
| | | (au) | 59 |
| 73 | ai | Total | 163 |
| | | (ai) | 163 |
| 74 | oi | Total | 27 |
| | | (oi) | 27 |
| 75 | i@ | Total | 38 |
| | | (i@) | 38 |
| 76 | e@ | Total | 42 |
| | | (e@) | 42 |
| 77 | u@ | Total | 10 |
| | | (u@) | 10 |
| 78 | @ | Total | 967 |
| | | (@) | 967 |

## F.5 Ext06

55 apus, including silence. Stops divided into released and unreleased (not syllable-conditioned). Total 12 stop allophones. Unreleased /b/ called /u-b/, etc. Other apus as those in stdp.

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | b | Total | 196 | | | | | | | | |
| | | (b — ) | 152 | (b r — ) | 10 | (b l — ) | 11 | | | | |
| | | (b y — ) | 1 | ( — b) | 17 | ( — b d) | 2 | | | | |
| | | ( — b z) | 3 | | | | | | | | |
| 2 | u-b | Total | 15 | | | | | | | | |
| | | (u-b — ) | 9 | ( — u-b d) | 3 | ( — l u-b) | 1 | | | | |
| | | ( — u-b) | 1 | ( — l u-b d) | 1 | | | | | | |
| 3 | d | Total | 201 | | | | | | | | |
| | | (d — ) | 85 | (d r — ) | 6 | (d w — ) | 1 | | | | |
| | | (d y — ) | 1 | ( — d) | 59 | ( — l d) | 5 | | | | |
| | | ( — n d) | 16 | ( — m d) | 1 | ( — ng d) | 1 | | | | |
| | | ( — v d) | 6 | ( — z d) | 3 | ( — n z d) | 1 | | | | |
| | | ( — b d) | 2 | ( — u-b d) | 3 | ( — g d) | 1 | | | | |
| | | ( — u-g d) | 1 | ( — l u-b d) | 1 | ( — d z) | 5 | | | | |
| | | ( — l d z) | 2 | ( — n d z) | 1 | | | | | | |
| 4 | u-d | Total | 88 | | | | | | | | |
| | | (u-d — ) | 20 | (u-d r — ) | 2 | ( — u-d) | 42 | | | | |
| | | ( — n u-d) | 5 | ( — l m u-d) | 1 | ( — l u-d) | 3 | | | | |
| | | ( — v u-d) | 2 | ( — dh u-d) | 2 | ( — z u-d) | 1 | | | | |
| | | ( — l v u-d) | 1 | ( — jh u-d) | 4 | ( — l jh u-d) | 1 | | | | |
| | | ( — n jh u-d) | 1 | ( — u-d z) | 2 | ( — u-d s t) | 1 | | | | |
| 5 | g | Total | 90 | | | | | | | | |
| | | (g — ) | 51 | (g r — ) | 15 | (g l — ) | 4 | | | | |
| | | (g w — ) | 1 | (g y — ) | 1 | ( — g) | 13 | | | | |
| | | ( — g z) | 3 | ( — g d) | 1 | ( — ng g th s) | 1 | | | | |
| 6 | u-g | Total | 9 | | | | | | | | |
| | | ( — u-g) | 6 | (u-g — ) | 2 | ( — u-g d) | 1 | | | | |
| 7 | p | Total | 187 | | | | | | | | |
| | | (p — ) | 86 | (p r — ) | 26 | (p l — ) | 13 | | | | |
| | | (s p — ) | 9 | (p y — ) | 2 | (s p r — ) | 2 | | | | |
| | | (s p l — ) | 1 | ( — m p) | 5 | ( — p) | 30 | | | | |
| | | ( — p s) | 4 | ( — m p s t) | 1 | ( — l p s) | 2 | | | | |
| | | ( — l p) | 1 | ( — s p s) | 2 | ( — l p t) | 1 | | | | |
| | | ( — p s t) | 1 | ( — p th) | 1 | | | | | | |
| 8 | u-p | Total | 9 | | | | | | | | |
| | | ( — u-p) | 5 | ( — m u-p u-t s) | 1 | ( — s u-p u-t) | 1 | | | | |
| | | ( — m u-p f s) | 1 | ( — m u-p t) | 1 | | | | | | |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 9 | t | Total | 457 | | | | | |
| | | (t — ) | 248 | (t r — ) | 25 | (t w — ) | 4 | |
| | | (t y — ) | 1 | (s t — ) | 26 | (s t r — ) | 4 | |
| | | (s t y — ) | 1 | ( — t) | 60 | ( — n t) | 19 | |
| | | ( — m t) | 2 | ( — s t) | 14 | ( — sh t) | 2 | |
| | | ( — ch t) | 2 | ( — l ch t) | 1 | ( — f t) | 3 | |
| | | ( — p s t) | 1 | ( — l s t) | 1 | ( — u-d s t) | 1 | |
| | | ( — ng s t) | 1 | ( — m p s t) | 1 | ( — l f t) | 1 | |
| | | ( — k t) | 3 | ( — u-k t) | 2 | ( — l p t) | 1 | |
| | | ( — l k t) | 1 | ( — s u-k t) | 1 | ( — m u-p t) | 1 | |
| | | ( — ng k t) | 1 | ( — t s) | 19 | ( — t sh) | 1 | |
| | | ( — t s u-t) | 1 | ( — m t s) | 1 | ( — l t) | 3 | |
| | | ( — l t s) | 1 | ( — f t s) | 1 | ( — n t s) | 1 | |
| | | ( — t th) | 1 | | | | | |
| 10 | u-t | Total | 145 | | | | | |
| | | (u-t — ) | 8 | (s u-t — ) | 2 | (s u-t r — ) | 2 | |
| | | (u-t r — ) | 1 | (s u-t y — ) | 1 | ( — u-t) | 82 | |
| | | ( — k u-t) | 1 | ( — u-k u-t) | 1 | ( — s u-p u-t) | 1 | |
| | | ( — n u-t) | 6 | ( — n u-t s) | 2 | ( — l u-t s u-t) | 1 | |
| | | ( — u-t s) | 8 | ( — u-t th s) | 1 | ( — l u-t) | 4 | |
| | | ( — l u-t s u-t) | 1 | ( — n u-t th s) | 1 | ( — s u-t) | 10 | |
| | | ( — n s u-t) | 2 | ( — l s u-t) | 1 | ( — sh u-t) | 1 | |
| | | ( — t s u-t) | 1 | ( — s u-t s) | 1 | ( — m u-p u-t s) | 1 | |
| | | ( — f u-t) | 1 | ( — n ch u-t) | 2 | ( — k s u-t) | 1 | |
| | | ( — th u-t) | 1 | | | | | |
| 11 | k | Total | 262 | | | | | |
| | | (k — ) | 123 | (k r — ) | 15 | (k l — ) | 10 | |
| | | (k w — ) | 5 | (k y — ) | 3 | (s k — ) | 10 | |
| | | (s k r — ) | 2 | (s k w — ) | 2 | ( — k) | 54 | |
| | | ( — k s) | 18 | ( — l k) | 2 | ( — k t) | 3 | |
| | | ( — k u-t) | 1 | ( — l k t) | 1 | ( — l k s) | 1 | |
| | | ( — s k) | 1 | ( — s k s) | 1 | ( — k s u-t) | 1 | |
| | | ( — ng k) | 4 | ( — ng k t) | 1 | ( — ng k s) | 3 | |
| | | ( — ng k th) | 1 | | | | | |
| 12 | u-k | Total | 17 | | | | | |
| | | ( — u-k) | 10 | ( — u-k t) | 2 | (u-k — ) | 2 | |
| | | ( — s u-k t) | 1 | ( — ng u-k) | 1 | ( — u-k u-t) | 1 | |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 13 | z | Total | 276 | | | | | | | | |
| | | (z — ) | 32 | ( — z) | 189 | ( — n z) | 15 | | | | |
| | | ( — m z) | 5 | ( — ng z) | 1 | ( — l m z) | 1 | | | | |
| | | ( — b z) | 3 | ( — d z) | 5 | ( — u-d z) | 2 | | | | |
| | | ( — g z) | 3 | ( — l d z) | 2 | ( — n d z) | 1 | | | | |
| | | ( — l z) | 5 | ( — v z) | 4 | ( — l v z) | 2 | | | | |
| | | ( — n z d) | 1 | ( — z d) | 3 | ( — z u-d) | 1 | | | | |
| | | ( — dh z) | 1 | | | | | | | | |
| 14 | zh | Total | 21 | | | | | | | | |
| | | (zh — ) | 12 | ( — zh) | 7 | ( — n zh) | 2 | | | | |
| 15 | jh | Total | 77 | | | | | | | | |
| | | (jh — ) | 40 | (jh y — ) | 2 | ( — jh) | 24 | | | | |
| | | ( — n jh) | 4 | ( — jh u-d) | 4 | ( — l jh) | 1 | | | | |
| | | ( — l jh u-d) | 1 | ( — n jh u-d) | 1 | | | | | | |
| 16 | s | Total | 463 | | | | | | | | |
| | | (s — ) | 139 | (s f — ) | 1 | (s p — ) | 9 | | | | |
| | | (s t — ) | 26 | (s u-t — ) | 2 | (s k — ) | 10 | | | | |
| | | (s p r — ) | 2 | (s t r — ) | 4 | (s u-t r — ) | 2 | | | | |
| | | (s k r — ) | 2 | (s k w — ) | 2 | (s t y — ) | 1 | | | | |
| | | (s u-t y — ) | 1 | (s p l — ) | 1 | (s w — ) | 6 | | | | |
| | | (s l — ) | 4 | (s y — ) | 1 | (s m — ) | 5 | | | | |
| | | (s n — ) | 1 | ( — p s) | 4 | ( — t s) | 19 | | | | |
| | | ( — u-t s) | 8 | ( — k s) | 18 | ( — s t) | 14 | | | | |
| | | ( — s u-t) | 10 | ( — s k) | 1 | ( — s u-k t) | 1 | | | | |
| | | ( — s u-p u-t) | 1 | ( — t s u-t) | 1 | ( — m p s t) | 1 | | | | |
| | | ( — n s u-t) | 2 | ( — ng s t) | 1 | ( — l s t) | 1 | | | | |
| | | ( — l s u-t) | 1 | ( — u-d s t) | 1 | ( — k s u-t) | 1 | | | | |
| | | ( — p s t) | 1 | ( — l u-t s u-t) | 1 | ( — s p s) | 2 | | | | |
| | | ( — s u-t s) | 1 | ( — s k s) | 1 | ( — s) | 113 | | | | |
| | | ( — n s) | 9 | ( — m s) | 1 | ( — l s) | 1 | | | | |
| | | ( — l p s) | 2 | ( — l t s) | 1 | ( — l k s) | 1 | | | | |
| | | ( — m t s) | 1 | ( — f t s) | 1 | ( — m u-p u-t s) | 1 | | | | |
| | | ( — s p s) | 2 | ( — s u-t s) | 1 | ( — s k s) | 1 | | | | |
| | | ( — th s) | 3 | ( — f s) | 2 | ( — n th s) | 1 | | | | |
| | | ( — l th s) | 1 | ( — u-t th s) | 1 | ( — ng g th s) | 1 | | | | |
| | | ( — n u-t th s) | 1 | ( — m u-p f s) | 1 | ( — l f s) | 1 | | | | |
| 17 | sh | Total | 76 | | | | | | | | |
| | | (sh — ) | 52 | (sh r — ) | 1 | ( — sh) | 18 | | | | |
| | | ( — sh t) | 2 | ( — sh u-t) | 1 | ( — t sh) | 1 | | | | |
| | | ( — l sh) | 1 | | | | | | | | |
| 18 | ch | Total | 68 | | | | | | | | |
| | | (ch — ) | 35 | (ch y — ) | 1 | ( — ch) | 24 | | | | |
| | | ( — ch t) | 2 | ( — n ch u-t) | 2 | ( — n ch) | 2 | | | | |
| | | ( — l ch t) | 1 | ( — l ch) | 1 | | | | | | |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 19 | v | Total | 155 | | | | | | | | |
| | | (v — ) | 55 | (v y — ) | 3 | (v r — ) | 2 | | | | |
| | | ( — v) | 79 | ( — v d) | 6 | ( — v u-d) | 2 | | | | |
| | | ( — v z) | 4 | ( — l v z) | 2 | ( — l v) | 1 | | | | |
| | | ( — l v u-d) | 1 | | | | | | | | |
| 20 | dh | Total | 332 | | | | | | | | |
| | | (dh — ) | 301 | ( — dh) | 28 | ( — dh u-d) | 2 | | | | |
| | | ( — dh z) | 1 | | | | | | | | |
| 21 | f | Total | 183 | | | | | | | | |
| | | (f — ) | 107 | (f r — ) | 19 | (f l — ) | 5 | | | | |
| | | (f y — ) | 3 | (s f — ) | 1 | ( — f) | 31 | | | | |
| | | ( — f t) | 3 | ( — f u-t) | 1 | ( — f t s) | 1 | | | | |
| | | ( — l f t) | 1 | ( — m f) | 3 | ( — l f) | 3 | | | | |
| | | ( — f s) | 2 | ( — l f s) | 1 | ( — f th) | 1 | | | | |
| | | ( — m u-p f s) | 1 | | | | | | | | |
| 22 | th | Total | 61 | | | | | | | | |
| | | (th — ) | 18 | (th r — ) | 7 | (th y — ) | 1 | | | | |
| | | (th w — ) | 1 | ( — th) | 17 | ( — th s) | 3 | | | | |
| | | ( — u-t th s) | 1 | ( — p th) | 1 | ( — t th) | 1 | | | | |
| | | ( — ng g th s) | 1 | ( — ng k th) | 1 | ( — n u-t th s) | 1 | | | | |
| | | ( — n th s) | 1 | ( — l th s) | 1 | ( — l th) | 2 | | | | |
| | | ( — f th) | 1 | ( — th u-t) | 1 | ( — n th) | 1 | | | | |
| | | ( — m th) | 1 | | | | | | | | |
| 23 | h | Total | 83 | | | | | | | | |
| | | (h — ) | 82 | (h y — ) | 1 | | | | | | |
| 24 | l | Total | 341 | | | | | | | | |
| | | (l — ) | 135 | (l y — ) | 1 | (p l — ) | 13 | | | | |
| | | (k l — ) | 10 | (b l — ) | 11 | (g l — ) | 4 | | | | |
| | | (s p l — ) | 1 | (f l — ) | 5 | (s l — ) | 4 | | | | |
| | | ( — l) | 98 | ( — l p) | 1 | ( — l p t) | 1 | | | | |
| | | ( — l t) | 3 | ( — l u-t) | 4 | ( — l u-t s u-t) | 1 | | | | |
| | | ( — l k) | 2 | ( — l k t) | 1 | ( — l u-b) | 1 | | | | |
| | | ( — l u-b d) | 1 | ( — l d) | 5 | ( — l u-d) | 3 | | | | |
| | | ( — l p s) | 2 | ( — l t s) | 1 | ( — l k s) | 1 | | | | |
| | | ( — l d z) | 2 | ( — l z) | 5 | ( — l v z) | 2 | | | | |
| | | ( — l v) | 1 | ( — l v u-d) | 1 | ( — l ch t) | 1 | | | | |
| | | ( — l ch) | 1 | ( — l jh u-d) | 1 | ( — l jh) | 1 | | | | |
| | | ( — l m) | 2 | ( — l m u-d) | 1 | ( — l n) | 1 | | | | |
| | | ( — l m z) | 1 | ( — l f) | 3 | ( — l f t) | 1 | | | | |
| | | ( — l th) | 2 | ( — l sh) | 1 | ( — l s) | 1 | | | | |
| | | ( — l s t) | 1 | ( — l s u-t) | 1 | ( — l f s) | 1 | | | | |
| | | ( — l th s) | 1 | | | | | | | | |
| 25 | l= | Total | 29 | | | | | | | | |
| | | (l= ) | 29 | | | | | | | | |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 26 | r | Total | 284 | | | | | | | | |
| | | (r — ) | 118 | (p r — ) | 26 | (t r — ) | 25 | | | | |
| | | (u-t r — ) | 1 | (k r — ) | 15 | (b r — ) | 11 | | | | |
| | | (d r — ) | 6 | (u-d r — ) | 2 | (g r — ) | 15 | | | | |
| | | (f r — ) | 19 | (th r — ) | 7 | (sh r — ) | 1 | | | | |
| | | (v r — ) | 2 | (s p r — ) | 2 | (s t r — ) | 4 | | | | |
| | | (s u-t r — ) | 2 | (s k r — ) | 2 | ( — r) | 26 | | | | |
| 27 | r= | Total | 2 | | | | | | | | |
| | | (r= ) | 2 | | | | | | | | |
| 28 | m | Total | 240 | | | | | | | | |
| | | (m — ) | 127 | (s m — ) | 5 | (m y — ) | 2 | | | | |
| | | ( — m) | 79 | ( — m p) | 5 | ( — m u-p t) | 1 | | | | |
| | | ( — m u-p u-t s) | 1 | ( — m p s t) | 1 | ( — m t) | 2 | | | | |
| | | ( — m d) | 1 | ( — m u-p f s) | 1 | ( — m t s) | 1 | | | | |
| | | ( — l m u-d) | 1 | ( — m z) | 5 | ( — l m z) | 1 | | | | |
| | | ( — m f) | 3 | ( — m th) | 1 | ( — m s) | 1 | | | | |
| | | ( — l m) | 2 | | | | | | | | |
| 29 | m= | Total | 6 | | | | | | | | |
| | | (m= ) | 6 | | | | | | | | |
| 30 | n | Total | 481 | | | | | | | | |
| | | (n — ) | 106 | (n y — ) | 4 | (s n — ) | 1 | | | | |
| | | ( — n) | 278 | ( — n t) | 19 | ( — n u-t) | 6 | | | | |
| | | ( — n d) | 16 | ( — n u-d) | 5 | ( — n t s) | 1 | | | | |
| | | ( — n u-t s) | 2 | ( — n u-t th s) | 1 | ( — n d z) | 1 | | | | |
| | | ( — n z) | 15 | ( — n s) | 9 | ( — n s u-t) | 2 | | | | |
| | | ( — n jh) | 4 | ( — n jh u-d) | 1 | ( — n zh) | 2 | | | | |
| | | ( — n z d) | 1 | ( — n ch u-t) | 2 | ( — n ch) | 2 | | | | |
| | | ( — n th s) | 1 | ( — n th) | 1 | ( — l n) | 1 | | | | |
| 31 | n= | Total | 34 | | | | | | | | |
| | | (n= ) | 34 | | | | | | | | |
| 32 | ng | Total | 78 | | | | | | | | |
| | | ( — ng) | 64 | ( — ng k) | 4 | ( — ng u-k) | 1 | | | | |
| | | ( — ng d) | 1 | ( — ng k s) | 3 | ( — ng k t) | 1 | | | | |
| | | ( — ng s t) | 1 | ( — ng g th s) | 1 | ( — ng k th) | 1 | | | | |
| | | ( — ng z) | 1 | | | | | | | | |
| 33 | y | Total | 72 | | | | | | | | |
| | | (y — ) | 42 | (p y — ) | 2 | (t y — ) | 1 | | | | |
| | | (k y — ) | 3 | (b y — ) | 1 | (d y — ) | 1 | | | | |
| | | (g y — ) | 1 | (s t y — ) | 1 | (s u-t y — ) | 1 | | | | |
| | | (n y — ) | 4 | (m y — ) | 2 | (v y — ) | 3 | | | | |
| | | (h y — ) | 1 | (f y — ) | 3 | (th y — ) | 1 | | | | |
| | | (s y — ) | 1 | (jh y — ) | 2 | (ch y — ) | 1 | | | | |
| | | (l y — ) | 1 | | | | | | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 34 | w | Total | 211 | | | | |
| | | (w — ) | 191 | (t w — ) | 4 | (k w — ) | 5 |
| | | (d w — ) | 1 | (g w — ) | 1 | (s w — ) | 6 |
| | | (s k w — ) | 2 | (th w — ) | 1 | | |
| 35 | ii | Total | 267 | | | | |
| | | (ii) | 267 | | | | |
| 36 | e | Total | 190 | | | | |
| | | (e) | 190 | | | | |
| 37 | a | Total | 126 | | | | |
| | | (a) | 126 | | | | |
| 38 | uu | Total | 103 | | | | |
| | | (uu) | 103 | | | | |
| 39 | u | Total | 37 | | | | |
| | | (u) | 37 | | | | |
| 40 | oo | Total | 105 | | | | |
| | | (oo) | 105 | | | | |
| 41 | o | Total | 110 | | | | |
| | | (o) | 110 | | | | |
| 42 | aa | Total | 105 | | | | |
| | | (aa) | 105 | | | | |
| 43 | i | Total | 519 | | | | |
| | | (i) | 519 | | | | |
| 44 | @@ | Total | 55 | | | | |
| | | (@@) | 55 | | | | |
| 45 | uh | Total | 108 | | | | |
| | | (uh) | 108 | | | | |
| 46 | ei | Total | 127 | | | | |
| | | (ei) | 127 | | | | |
| 47 | ou | Total | 106 | | | | |
| | | (ou) | 106 | | | | |
| 48 | au | Total | 59 | | | | |
| | | (au) | 59 | | | | |
| 49 | ai | Total | 163 | | | | |
| | | (ai) | 163 | | | | |
| 50 | oi | Total | 27 | | | | |
| | | (oi) | 27 | | | | |
| 51 | i@ | Total | 38 | | | | |
| | | (i@) | 38 | | | | |
| 52 | e@ | Total | 42 | | | | |
| | | (e@) | 42 | | | | |
| 53 | u@ | Total | 10 | | | | |
| | | (u@) | 10 | | | | |

| | | | |
|---|---|---|---|
| 54 | @ | Total | 967 |
| | | (@) | 967 |

## F.6 Ext07

65 apus, including silence. Syllable-conditioned stops only, derived from those in ext05. Ext05's 36 stop allophones were combined to give 22 stop allopones. Non-stop apus are as those in stdp.

```
1   b    Total      161
         (b — )     161
2   b2   Total       23      (b r — )    11
         (b l — )    11      (b y — )     1
3   b3   Total       28
         ( — b)      18      ( — b d)     5      ( — b z)    3
         ( — l b)     1      ( — l b d)   1
4   d    Total      115
         (d — )     105      (d r — )     8      (d w — )    1
         (d y — )     1
5   d2   Total      109
         ( — d)     101      ( — l d)     8
6   d3   Total       24
         ( — n d)    21      ( — m d)     1      ( — ng d)   1
         ( — l m d)   1
7   d4   Total       30
         ( — v d)     8      ( — z d)     4      ( — dh d)   2
         ( — n z d)   1      ( — l v d)   1      ( — b d)    5
         ( — g d)     2      ( — l b d)   1      ( — jh d)   4
         ( — l jh d)  1      ( — n jh d)  1
8   d5   Total       11
         ( — d z)     7      ( — l d z)   2      ( — n d z)  1
         ( — d s t)   1
9   g    Total       53
         (g — )      53
10  g2   Total       21
         (g r — )    15      (g l — )     4      (g w — )    1
         (g y — )     1
11  g3   Total            25
         ( — g)            19      ( — g z)    3      ( — g d)    2
         ( — ng g th s)     1
12  p    Total       95
         (p — )      86      (s p — )     9
13  p2   Total       44
         (p r — )    26      (p l — )    13      (p y — )    2
         (s p r — )   2      (s p l — )   1
```

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 14 | p3 | Total | 57 | | | | | |
| | | ( — p) | 35 | ( — m p) | 5 | ( — m p t) | 1 | |
| | | ( — p s) | 4 | ( — m p f s) | 1 | ( — m p s t) | 1 | |
| | | ( — m p t s) | 1 | ( — l p s) | 2 | ( — l p) | 1 | |
| | | ( — s p t) | 1 | ( — s p s) | 2 | ( — l p t) | 1 | |
| | | ( — p s t) | 1 | ( — p th) | 1 | | | |
| 15 | t | Total | 284 | | | | | |
| | | (t — ) | 256 | (s t — ) | 28 | | | |
| 16 | t2 | Total | 39 | | | | | |
| | | (t r — ) | 26 | (s t r — ) | 6 | (t w — ) | 4 | |
| | | (s t y — ) | 2 | (t y — ) | 1 | | | |
| 17 | t3 | Total | 279 | | | | | |
| | | ( — t) | 142 | ( — t s) | 27 | ( — n t) | 25 | |
| | | ( — s t) | 24 | ( — k t) | 7 | ( — sh t) | 3 | |
| | | ( — m t) | 2 | ( — ch t) | 2 | ( — n ch t) | 2 | |
| | | ( — l ch t) | 1 | ( — f t) | 4 | ( — p s t) | 1 | |
| | | ( — t s t) | 1 | ( — l t s t) | 1 | ( — k s t) | 1 | |
| | | ( — d s t) | 1 | ( — l s t) | 2 | ( — n s t) | 2 | |
| | | ( — ng s t) | 1 | ( — m p t s) | 1 | ( — m p s t) | 1 | |
| | | ( — l f t) | 1 | ( — th t) | 1 | ( — l p t) | 1 | |
| | | ( — l k t) | 1 | ( — s p t) | 1 | ( — s k t) | 1 | |
| | | ( — m p t) | 1 | ( — ng k t) | 1 | ( — t sh) | 1 | |
| | | ( — t s t) | 1 | ( — m t s) | 1 | ( — l t) | 7 | |
| | | ( — l t s) | 1 | ( — l t s t) | 1 | ( — f t s) | 1 | |
| | | ( — s t s) | 1 | ( — n t s) | 3 | ( — t th) | 1 | |
| | | ( — n t th s) | 1 | ( — t th s) | 1 | | | |
| 18 | k | Total | 125 | | | | | |
| | | (k — ) | 125 | | | | | |
| 19 | k2 | Total | 15 | | | | | |
| | | (k r — ) | 15 | | | | | |
| 20 | k3 | Total | 32 | | | | | |
| | | (k l — ) | 10 | (k w — ) | 5 | (k y — ) | 3 | |
| | | (s k — ) | 10 | (s k r — ) | 2 | (s k w — ) | 2 | |
| 21 | k4 | Total | 64 | | | | | |
| | | ( — k) | 64 | | | | | |
| 22 | k5 | Total | 43 | | | | | |
| | | ( — k s) | 18 | ( — l k) | 2 | ( — l k s) | 1 | |
| | | ( — s k s) | 1 | ( — k s t) | 1 | ( — ng k) | 5 | |
| | | ( — ng k s) | 3 | ( — ng k th) | 1 | ( — k t) | 7 | |
| | | ( — l k t) | 1 | ( — s k) | 1 | ( — s k t) | 1 | |
| | | ( — ng k t) | 1 | | | | | |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 23 | z | Total | 276 | | | | | | |
| | | (z — ) | 32 | ( — z) | 189 | ( — n z) | 15 | | |
| | | ( — m z) | 5 | ( — ng z) | 1 | ( — l m z) | 1 | | |
| | | ( — b z) | 3 | ( — d z) | 7 | ( — g z) | 3 | | |
| | | ( — l d z) | 2 | ( — n d z) | 1 | ( — l z) | 5 | | |
| | | ( — v z) | 4 | ( — l v z) | 2 | ( — n z d) | 1 | | |
| | | ( — z d) | 4 | ( — dh z) | 1 | | | | |
| 24 | zh | Total | 21 | | | | | | |
| | | (zh — ) | 12 | ( — zh) | 7 | ( — n zh) | 2 | | |
| 25 | jh | Total | 77 | | | | | | |
| | | (jh — ) | 40 | (jh y — ) | 2 | ( — jh) | 24 | | |
| | | ( — n jh) | 4 | ( — jh d) | 4 | ( — l jh d) | 1 | | |
| | | ( — n jh d) | 1 | ( — l jh) | 1 | | | | |
| 26 | s | Total | 463 | | | | | | |
| | | (s — ) | 139 | (s f — ) | 1 | (s p — ) | 9 | | |
| | | (s t — ) | 28 | (s k — ) | 10 | (s p r — ) | 2 | | |
| | | (s t r — ) | 6 | (s k r — ) | 2 | (s k w — ) | 2 | | |
| | | (s t y — ) | 2 | (s p l — ) | 1 | (s w — ) | 6 | | |
| | | (s l — ) | 4 | (s y — ) | 1 | (s m — ) | 5 | | |
| | | (s n — ) | 1 | ( — p s) | 4 | ( — t s) | 27 | | |
| | | ( — k s) | 18 | ( — s) | 113 | ( — s t) | 24 | | |
| | | ( — s k) | 1 | ( — s k t) | 1 | ( — s p t) | 1 | | |
| | | ( — t s t) | 1 | ( — m p s t) | 1 | ( — n s t) | 2 | | |
| | | ( — ng s t) | 1 | ( — l s t) | 2 | ( — d s t) | 1 | | |
| | | ( — k s t) | 1 | ( — p s t) | 1 | ( — l t s t) | 1 | | |
| | | ( — s p s) | 2 | ( — s k s) | 1 | ( — s t s) | 1 | | |
| | | ( — n s) | 9 | ( — m s) | 1 | ( — l s) | 1 | | |
| | | ( — l p s) | 2 | ( — l t s) | 1 | ( — l k s) | 1 | | |
| | | ( — ng k s) | 3 | ( — n t s) | 3 | ( — m t s) | 1 | | |
| | | ( — f t s) | 1 | ( — m p t s) | 1 | ( — s p s) | 2 | | |
| | | ( — s t s) | 1 | ( — s k s) | 1 | ( — th s) | 3 | | |
| | | ( — f s) | 2 | ( — n th s) | 1 | ( — l th s) | 1 | | |
| | | ( — t th s) | 1 | ( — ng g th s) | 1 | ( — n t th s) | 1 | | |
| | | ( — m p f s) | 1 | ( — l f s) | 1 | | | | |
| 27 | sh | Total | 76 | | | | | | |
| | | (sh — ) | 52 | (sh r — ) | 1 | ( — sh) | 18 | | |
| | | ( — sh t) | 3 | ( — t sh) | 1 | ( — l sh) | 1 | | |
| 28 | ch | Total | 68 | | | | | | |
| | | (ch — ) | 35 | (ch y — ) | 1 | ( — ch) | 24 | | |
| | | ( — ch t) | 2 | ( — n ch t) | 2 | ( — n ch) | 2 | | |
| | | ( — l ch t) | 1 | ( — l ch) | 1 | | | | |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 29 | v | Total | 155 | | | | | | | | |
| | | (v — ) | 55 | (v y — ) | 3 | (v r — ) | 2 | | | | |
| | | ( — v) | 79 | ( — v d) | 8 | ( — v z) | 4 | | | | |
| | | ( — l v z) | 2 | ( — l v) | 1 | ( — l v d) | 1 | | | | |
| 30 | dh | Total | 332 | | | | | | | | |
| | | (dh — ) | 301 | ( — dh) | 28 | ( — dh d) | 2 | | | | |
| | | ( — dh z) | 1 | | | | | | | | |
| 31 | f | Total | 183 | | | | | | | | |
| | | (f — ) | 107 | (f r — ) | 19 | (f l — ) | 5 | | | | |
| | | (f y — ) | 3 | (s f — ) | 1 | ( — f) | 31 | | | | |
| | | ( — f t) | 4 | ( — f t s) | 1 | ( — l f t) | 1 | | | | |
| | | ( — m f) | 3 | ( — l f) | 3 | ( — f s) | 2 | | | | |
| | | ( — l f s) | 1 | ( — f th) | 1 | ( — m p f s) | 1 | | | | |
| 32 | th | Total | 61 | | | | | | | | |
| | | (th — ) | 18 | (th r — ) | 7 | (th y — ) | 1 | | | | |
| | | (th w — ) | 1 | ( — th) | 17 | ( — th s) | 3 | | | | |
| | | ( — t th s) | 1 | ( — p th) | 1 | ( — t th) | 1 | | | | |
| | | ( — ng g th s) | 1 | ( — ng k th) | 1 | ( — n t th s) | 1 | | | | |
| | | ( — n th s) | 1 | ( — l th s) | 1 | ( — l th) | 2 | | | | |
| | | ( — f th) | 1 | ( — th t) | 1 | ( — n th) | 1 | | | | |
| | | ( — m th) | 1 | | | | | | | | |
| 33 | h | Total | 83 | | | | | | | | |
| | | (h — ) | 82 | (h y — ) | 1 | | | | | | |
| 34 | l | Total | 341 | | | | | | | | |
| | | (l — ) | 135 | (l y — ) | 1 | (p l — ) | 13 | | | | |
| | | (k l — ) | 10 | (b l — ) | 11 | (g l — ) | 4 | | | | |
| | | (s p l — ) | 1 | (f l — ) | 5 | (s l — ) | 4 | | | | |
| | | ( — l) | 98 | ( — l p) | 1 | ( — l p t) | 1 | | | | |
| | | ( — l t) | 7 | ( — l t s t) | 1 | ( — l k) | 2 | | | | |
| | | ( — l k t) | 1 | ( — l b) | 1 | ( — l b d) | 1 | | | | |
| | | ( — l d) | 8 | ( — l p s) | 2 | ( — l t s) | 1 | | | | |
| | | ( — l k s) | 1 | ( — l d z) | 2 | ( — l z) | 5 | | | | |
| | | ( — l v z) | 2 | ( — l v) | 1 | ( — l v d) | 1 | | | | |
| | | ( — l ch t) | 1 | ( — l ch) | 1 | ( — l jh d) | 1 | | | | |
| | | ( — l jh) | 1 | ( — l m) | 2 | ( — l m d) | 1 | | | | |
| | | ( — l n) | 1 | ( — l m z) | 1 | ( — l f) | 3 | | | | |
| | | ( — l f t) | 1 | ( — l th) | 2 | ( — l sh) | 1 | | | | |
| | | ( — l s) | 1 | ( — l s t) | 2 | ( — l f s) | 1 | | | | |
| | | ( — l th s) | 1 | | | | | | | | |
| 35 | l= | Total | 29 | | | | | | | | |
| | | (l= ) | 29 | | | | | | | | |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 36 | r | Total | 284 | | | | | | | | |
| | | (r — ) | 118 | (p r — ) | 26 | (t r — ) | 26 | | | | |
| | | (k r — ) | 15 | (b r — ) | 11 | (d r — ) | 8 | | | | |
| | | (g r — ) | 15 | (f r — ) | 19 | (th r — ) | 7 | | | | |
| | | (sh r — ) | 1 | (v r — ) | 2 | (s p r — ) | 2 | | | | |
| | | (s t r — ) | 6 | (s k r — ) | 2 | ( — r) | 26 | | | | |
| 37 | r= | Total | 2 | | | | | | | | |
| | | (r= ) | 2 | | | | | | | | |
| 38 | m | Total | 240 | | | | | | | | |
| | | (m — ) | 127 | (s m — ) | 5 | (m y — ) | 2 | | | | |
| | | ( — m) | 79 | ( — m p) | 5 | ( — m p t) | 1 | | | | |
| | | ( — m p t s) | 1 | ( — m p s t) | 1 | ( — m t) | 2 | | | | |
| | | ( — m d) | 1 | ( — m p f s) | 1 | ( — m t s) | 1 | | | | |
| | | ( — l m d) | 1 | ( — m z) | 5 | ( — l m z) | 1 | | | | |
| | | ( — m f) | 3 | ( — m th) | 1 | ( — m s) | 1 | | | | |
| | | ( — l m) | 2 | | | | | | | | |
| 39 | m= | Total | 6 | | | | | | | | |
| | | (m= ) | 6 | | | | | | | | |
| 40 | n | Total | 481 | | | | | | | | |
| | | (n — ) | 106 | (n y — ) | 4 | (s n — ) | 1 | | | | |
| | | ( — n) | 278 | ( — n t) | 25 | ( — n d) | 21 | | | | |
| | | ( — n t s) | 3 | ( — n t th s) | 1 | ( — n d z) | 1 | | | | |
| | | ( — n z) | 15 | ( — n s) | 9 | ( — n s t) | 2 | | | | |
| | | ( — n jh) | 4 | ( — n jh d) | 1 | ( — n zh) | 2 | | | | |
| | | ( — n z d) | 1 | ( — n ch t) | 2 | ( — n ch) | 2 | | | | |
| | | ( — n th s) | 1 | ( — n th) | 1 | ( — l n) | 1 | | | | |
| 41 | n= | Total | 34 | | | | | | | | |
| | | (n= ) | 34 | | | | | | | | |
| 42 | ng | Total | 78 | | | | | | | | |
| | | ( — ng) | 64 | ( — ng k) | 5 | ( — ng d) | 1 | | | | |
| | | ( — ng k s) | 3 | ( — ng k t) | 1 | ( — ng s t) | 1 | | | | |
| | | ( — ng g th s) | 1 | ( — ng k th) | 1 | ( — ng z) | 1 | | | | |
| 43 | y | Total | 72 | | | | | | | | |
| | | (y — ) | 42 | (p y — ) | 2 | (t y — ) | 1 | | | | |
| | | (k y — ) | 3 | (b y — ) | 1 | (d y — ) | 1 | | | | |
| | | (g y — ) | 1 | (s t y — ) | 2 | (n y — ) | 4 | | | | |
| | | (m y — ) | 2 | (v y — ) | 3 | (h y — ) | 1 | | | | |
| | | (f y — ) | 3 | (th y — ) | 1 | (s y — ) | 1 | | | | |
| | | (jh y — ) | 2 | (ch y — ) | 1 | (l y — ) | 1 | | | | |
| 44 | w | Total | 211 | | | | | | | | |
| | | (w — ) | 191 | (t w — ) | 4 | (k w — ) | 5 | | | | |
| | | (d w — ) | 1 | (g w — ) | 1 | (s w — ) | 6 | | | | |
| | | (s k w — ) | 2 | (th w — ) | 1 | | | | | | |

45 ii Total 267
(ii) 267
46 e Total 190
(e) 190
47 a Total 126
(a) 126
48 uu Total 103
(uu) 103
49 u Total 37
(u) 37
50 oo Total 105
(oo) 105
51 o Total 110
(o) 110
52 aa Total 105
(aa) 105
53 i Total 519
(i) 519
54 @@ Total 55
(@@) 55
55 uh Total 108
(uh) 108
56 ei Total 127
(ei) 127
57 ou Total 106
(ou) 106
58 au Total 59
(au) 59
59 ai Total 163
(ai) 163
60 oi Total 27
(oi) 27
61 i@ Total 38
(i@) 38
62 e@ Total 42
(e@) 42
63 u@ Total 10
(u@) 10
64 @ Total 967
(@) 967

## F.7 Ext08

72 apus, including silence. Syllable-conditioned stops only, derived from those in ext05. Ext05's 36 stop allophones were combined to give 29 stop allopones. Non-stop apus are as those in stdp.

```
1   b    Total      173
         (b — )     161      (b l — )    11      (b y — )    1
2   b2   Total       11
         (b r — )    11
3   b3   Total       28
         ( — b)      18      ( — b d)     5      ( — b z)    3
         ( — l b)     1      ( — l b d)   1
4   d    Total      105
         (d — )     105
5   d2   Total       10
         (d r — )     8      (d w — )     1      (d y — )    1
6   d3   Total      149
         ( — d)      101      ( — l d)     8      ( — n d)    21
         ( — m d)      1      ( — ng d)    1      ( — l m d)   1
         ( — v d)      8      ( — z d)     4      ( — dh d)    2
         ( — n z d)    1      ( — l v d)   1
7   d4   Total       14
         ( — b d)     5      ( — g d)     2      ( — l b d)   1
         ( — jh d)    4      ( — l jh d)  1      ( — n jh d)  1
8   d5   Total       11
         ( — d z)     7      ( — l d z)   2      ( — n d z)   1
         ( — d s t)   1
9   g    Total       59
         (g — )      53      (g l — )     4      (g w — )    1
         (g y — )     1
10  g2   Total       15
         (g r — )    15
11  g3   Total            25
         ( — g)             19      ( — g z)    3      ( — g d)    2
         ( — ng g th s)      1
12  p    Total       86
         (p — )      86
13  p2   Total       26
         (p r — )    26
14  p3   Total       15
         (p l — )    13      (p y — )     2
```

15 p4 Total 12
(s p — ) 9 (s p r — ) 2 (s p l — ) 1

16 p5 Total 35
( — p) 35

17 p6 Total 22
( — m p) 5 ( — m p t) 1 ( — p s) 4
( — m p f s) 1 ( — m p s t) 1 ( — m p t s) 1
( — l p s) 2 ( — l p) 1 ( — s p t) 1
( — s p s) 2 ( — l p t) 1 ( — p s t) 1
( — p th) 1

18 t Total 284
(t — ) 256 (s t — ) 28

19 t2 Total 39
(t r — ) 26 (s t r — ) 6 (t w — ) 4
(t y — ) 1 (s t y — ) 2

20 t3 Total 142
( — t) 142

21 t4 Total 77
( — n t) 25 ( — m t) 2 ( — s t) 24
( — sh t) 3 ( — ch t) 2 ( — n ch t) 2
( — l ch t) 1 ( — f t) 4 ( — p s t) 1
( — t s t) 1 ( — l t s t) 1 ( — k s t) 1
( — d s t) 1 ( — l s t) 2 ( — n s t) 2
( — ng s t) 1 ( — m p t s) 1 ( — m p s t) 1
( — l f t) 1 ( — th t) 1

22 t5 Total 13
( — k t) 7 ( — l p t) 1 ( — l k t) 1
( — s p t) 1 ( — s k t) 1 ( — m p t) 1
( — ng k t) 1

23 t6 Total 47
( — t s) 27 ( — t sh) 1 ( — t s t) 1
( — m t s) 1 ( — l t) 7 ( — l t s) 1
( — l t s t) 1 ( — f t s) 1 ( — s t s) 1
( — n t s) 3 ( — t th) 1 ( — n t th s) 1
( — t th s) 1

24 k Total 135
(k — ) 125 (s k — ) 10

25 k2 Total 17
(k r — ) 15 (s k r — ) 2

26 k3 Total 20
(k l — ) 10 (k w — ) 5 (s k w — ) 2
(k y — ) 3

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | k4 | Total | 72 | | | | | | | | |
| | | ( — k) | 64 | ( — l k) | 2 | ( — ng k) | 5 | | | | |
| | | ( — s k) | 1 | | | | | | | | |
| 28 | k5 | Total | 25 | | | | | | | | |
| | | ( — k s) | 18 | ( — l k s) | 1 | ( — s k s) | 1 | | | | |
| | | ( — k s t) | 1 | ( — ng k s) | 3 | ( — ng k th) | 1 | | | | |
| 29 | k6 | Total | 10 | | | | | | | | |
| | | ( — k t) | 7 | ( — l k t) | 1 | ( — s k t) | 1 | | | | |
| | | ( — ng k t) | 1 | | | | | | | | |
| 30 | z | Total | 276 | | | | | | | | |
| | | (z — ) | 32 | ( — z) | 189 | ( — n z) | 15 | | | | |
| | | ( — m z) | 5 | ( — ng z) | 1 | ( — l m z) | 1 | | | | |
| | | ( — b z) | 3 | ( — d z) | 7 | ( — g z) | 3 | | | | |
| | | ( — l d z) | 2 | ( — n d z) | 1 | ( — l z) | 5 | | | | |
| | | ( — v z) | 4 | ( — l v z) | 2 | ( — n z d) | 1 | | | | |
| | | ( — z d) | 4 | ( — dh z) | 1 | | | | | | |
| 31 | zh | Total | 21 | | | | | | | | |
| | | (zh — ) | 12 | ( — zh) | 7 | ( — n zh) | 2 | | | | |
| 32 | jh | Total | 77 | | | | | | | | |
| | | (jh — ) | 40 | (jh y — ) | 2 | ( — jh) | 24 | | | | |
| | | ( — n jh) | 4 | ( — jh d) | 4 | ( — l jh d) | 1 | | | | |
| | | ( — n jh d) | 1 | ( — l jh) | 1 | | | | | | |
| 33 | s | Total | 463 | | | | | | | | |
| | | (s — ) | 139 | (s f — ) | 1 | (s p — ) | 9 | | | | |
| | | (s t — ) | 28 | (s k — ) | 10 | (s p r — ) | 2 | | | | |
| | | (s t r — ) | 6 | (s k r — ) | 2 | (s k w — ) | 2 | | | | |
| | | (s t y — ) | 2 | (s p l — ) | 1 | (s w — ) | 6 | | | | |
| | | (s l — ) | 4 | (s y — ) | 1 | (s m — ) | 5 | | | | |
| | | (s n — ) | 1 | ( — p s) | 4 | ( — t s) | 27 | | | | |
| | | ( — k s) | 18 | ( — s) | 113 | ( — s t) | 24 | | | | |
| | | ( — s k) | 1 | ( — s k t) | 1 | ( — s p t) | 1 | | | | |
| | | ( — t s t) | 1 | ( — m p s t) | 1 | ( — n s t) | 2 | | | | |
| | | ( — ng s t) | 1 | ( — l s t) | 2 | ( — d s t) | 1 | | | | |
| | | ( — k s t) | 1 | ( — p s t) | 1 | ( — l t s t) | 1 | | | | |
| | | ( — s p s) | 2 | ( — s k s) | 1 | ( — s t s) | 1 | | | | |
| | | ( — n s) | 9 | ( — m s) | 1 | ( — l s) | 1 | | | | |
| | | ( — l p s) | 2 | ( — l t s) | 1 | ( — l k s) | 1 | | | | |
| | | ( — ng k s) | 3 | ( — n t s) | 3 | ( — m t s) | 1 | | | | |
| | | ( — f t s) | 1 | ( — m p t s) | 1 | ( — s p s) | 2 | | | | |
| | | ( — s t s) | 1 | ( — s k s) | 1 | ( — th s) | 3 | | | | |
| | | ( — f s) | 2 | ( — n th s) | 1 | ( — l th s) | 1 | | | | |
| | | ( — t th s) | 1 | ( — ng g th s) | 1 | ( — n t th s) | 1 | | | | |
| | | ( — m p f s) | 1 | ( — l f s) | 1 | | | | | | |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 34 | sh | Total | 76 | | | | | | | | |
| | | (sh — ) | 52 | (sh r — ) | 1 | ( — sh) | 18 | | | | |
| | | ( — sh t) | 3 | ( — t sh) | 1 | ( — l sh) | 1 | | | | |
| 35 | ch | Total | 68 | | | | | | | | |
| | | (ch — ) | 35 | (ch y — ) | 1 | ( — ch) | 24 | | | | |
| | | ( — ch t) | 2 | ( — n ch t) | 2 | ( — n ch) | 2 | | | | |
| | | ( — l ch t) | 1 | ( — l ch) | 1 | | | | | | |
| 36 | v | Total | 155 | | | | | | | | |
| | | (v — ) | 55 | (v y — ) | 3 | (v r — ) | 2 | | | | |
| | | ( — v) | 79 | ( — v d) | 8 | ( — v z) | 4 | | | | |
| | | ( — l v z) | 2 | ( — l v) | 1 | ( — l v d) | 1 | | | | |
| 37 | dh | Total | 332 | | | | | | | | |
| | | (dh — ) | 301 | ( — dh) | 28 | ( — dh d) | 2 | | | | |
| | | ( — dh z) | 1 | | | | | | | | |
| 38 | f | Total | 183 | | | | | | | | |
| | | (f — ) | 107 | (f r — ) | 19 | (f l — ) | 5 | | | | |
| | | (f y — ) | 3 | (s f — ) | 1 | ( — f) | 31 | | | | |
| | | ( — f t) | 4 | ( — f t s) | 1 | ( — l f t) | 1 | | | | |
| | | ( — m f) | 3 | ( — l f) | 3 | ( — f s) | 2 | | | | |
| | | ( — l f s) | 1 | ( — f th) | 1 | ( — m p f s) | 1 | | | | |
| 39 | th | Total | 61 | | | | | | | | |
| | | (th — ) | 18 | (th r — ) | 7 | (th y — ) | 1 | | | | |
| | | (th w — ) | 1 | ( — th) | 17 | ( — th s) | 3 | | | | |
| | | ( — t th s) | 1 | ( — p th) | 1 | ( — t th) | 1 | | | | |
| | | ( — ng g th s) | 1 | ( — ng k th) | 1 | ( — n t th s) | 1 | | | | |
| | | ( — n th s) | 1 | ( — l th s) | 1 | ( — l th) | 2 | | | | |
| | | ( — f th) | 1 | ( — th t) | 1 | ( — n th) | 1 | | | | |
| | | ( — m th) | 1 | | | | | | | | |
| 40 | h | Total | 83 | | | | | | | | |
| | | (h — ) | 82 | (h y — ) | 1 | | | | | | |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 41 | l | Total | 341 | | | | | |
| | | (l — ) | 135 | (l y — ) | 1 | (p l — ) | 13 | |
| | | (k l — ) | 10 | (b l — ) | 11 | (g l — ) | 4 | |
| | | (s p l — ) | 1 | (f l — ) | 5 | (s l — ) | 4 | |
| | | ( — l) | 98 | ( — l p) | 1 | ( — l p t) | 1 | |
| | | ( — l t) | 7 | ( — l t s t) | 1 | ( — l k) | 2 | |
| | | ( — l k t) | 1 | ( — l b) | 1 | ( — l b d) | 1 | |
| | | ( — l d) | 8 | ( — l p s) | 2 | ( — l t s) | 1 | |
| | | ( — l k s) | 1 | ( — l d z) | 2 | ( — l z) | 5 | |
| | | ( — l v z) | 2 | ( — l v) | 1 | ( — l v d) | 1 | |
| | | ( — l ch t) | 1 | ( — l ch) | 1 | ( — l jh d) | 1 | |
| | | ( — l jh) | 1 | ( — l m) | 2 | ( — l m d) | 1 | |
| | | ( — l n) | 1 | ( — l m z) | 1 | ( — l f) | 3 | |
| | | ( — l f t) | 1 | ( — l th) | 2 | ( — l sh) | 1 | |
| | | ( — l s) | 1 | ( — l s t) | 2 | ( — l f s) | 1 | |
| | | ( — l th s) | 1 | | | | | |
| 42 | l= | Total | 29 | | | | | |
| | | (l= ) | 29 | | | | | |
| 43 | r | Total | 284 | | | | | |
| | | (r — ) | 118 | (p r — ) | 26 | (t r — ) | 26 | |
| | | (k r — ) | 15 | (b r — ) | 11 | (d r — ) | 8 | |
| | | (g r — ) | 15 | (f r — ) | 19 | (th r — ) | 7 | |
| | | (sh r — ) | 1 | (v r — ) | 2 | (s p r — ) | 2 | |
| | | (s t r — ) | 6 | (s k r — ) | 2 | ( — r) | 26 | |
| 44 | r= | Total | 2 | (r= ) | 2 | | | |
| 45 | m | Total | 240 | | | | | |
| | | (m — ) | 127 | (s m — ) | 5 | (m y — ) | 2 | |
| | | ( — m) | 79 | ( — m p) | 5 | ( — m p t) | 1 | |
| | | ( — m p t s) | 1 | ( — m p s t) | 1 | ( — m t) | 2 | |
| | | ( — m d) | 1 | ( — m p f s) | 1 | ( — m t s) | 1 | |
| | | ( — l m d) | 1 | ( — m z) | 5 | ( — l m z) | 1 | |
| | | ( — m f) | 3 | ( — m th) | 1 | ( — m s) | 1 | |
| | | ( — l m) | 2 | | | | | |
| 46 | m= | Total | 6 | | | | | |
| | | (m= ) | 6 | | | | | |
| 47 | n | Total | 481 | | | | | |
| | | (n — ) | 106 | (n y — ) | 4 | (s n — ) | 1 | |
| | | ( — n) | 278 | ( — n t) | 25 | ( — n d) | 21 | |
| | | ( — n t s) | 3 | ( — n t th s) | 1 | ( — n d z) | 1 | |
| | | ( — n z) | 15 | ( — n s) | 9 | ( — n s t) | 2 | |
| | | ( — n jh) | 4 | ( — n jh d) | 1 | ( — n zh) | 2 | |
| | | ( — n z d) | 1 | ( — n ch t) | 2 | ( — n ch) | 2 | |
| | | ( — n th s) | 1 | ( — n th) | 1 | ( — l n) | 1 | |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 48 | n= | Total | 34 | | | | | | |
| | | (n= ) | 34 | | | | | | |
| 49 | ng | Total | 78 | | | | | | |
| | | ( — ng) | 64 | ( — ng k) | 5 | ( — ng d) | 1 | | |
| | | ( — ng k s) | 3 | ( — ng k t) | 1 | ( — ng s t) | 1 | | |
| | | ( — ng g th s) | 1 | ( — ng k th) | 1 | ( — ng z) | 1 | | |
| 50 | y | Total | 72 | | | | | | |
| | | (y — ) | 42 | (p y — ) | 2 | (t y — ) | 1 | | |
| | | (k y — ) | 3 | (b y — ) | 1 | (d y — ) | 1 | | |
| | | (g y — ) | 1 | (s t y — ) | 2 | (n y — ) | 4 | | |
| | | (m y — ) | 2 | (v y — ) | 3 | (h y — ) | 1 | | |
| | | (f y — ) | 3 | (th y — ) | 1 | (s y — ) | 1 | | |
| | | (jh y — ) | 2 | (ch y — ) | 1 | (l y — ) | 1 | | |
| 51 | w | Total | 211 | | | | | | |
| | | (w — ) | 191 | (t w — ) | 4 | (k w — ) | 5 | | |
| | | (d w — ) | 1 | (g w — ) | 1 | (s w — ) | 6 | | |
| | | (s k w — ) | 2 | (th w — ) | 1 | | | | |
| 52 | ii | Total | 267 | | | | | | |
| | | (ii) | 267 | | | | | | |
| 53 | e | Total | 190 | | | | | | |
| | | (e) | 190 | | | | | | |
| 54 | a | Total | 126 | | | | | | |
| | | (a) | 126 | | | | | | |
| 55 | uu | Total | 103 | | | | | | |
| | | (uu) | 103 | | | | | | |
| 56 | u | Total | 37 | | | | | | |
| | | (u) | 37 | | | | | | |
| 57 | oo | Total | 105 | | | | | | |
| | | (oo) | 105 | | | | | | |
| 58 | o | Total | 110 | | | | | | |
| | | (o) | 110 | | | | | | |
| 59 | aa | Total | 105 | | | | | | |
| | | (aa) | 105 | | | | | | |
| 60 | i | Total | 519 | | | | | | |
| | | (i) | 519 | | | | | | |
| 61 | @@ | Total | 55 | | | | | | |
| | | (@@) | 55 | | | | | | |
| 62 | uh | Total | 108 | | | | | | |
| | | (uh) | 108 | | | | | | |
| 63 | ei | Total | 127 | | | | | | |
| | | (ei) | 127 | | | | | | |
| 64 | ou | Total | 106 | | | | | | |
| | | (ou) | 106 | | | | | | |

| | | | |
|---|---|---|---|
| 65 | au | Total | 59 |
| | | (au) | 59 |
| 66 | ai | Total | 163 |
| | | (ai) | 163 |
| 67 | oi | Total | 27 |
| | | (oi) | 27 |
| 68 | i@ | Total | 38 |
| | | (i@) | 38 |
| 69 | e@ | Total | 42 |
| | | (e@) | 42 |
| 70 | u@ | Total | 10 |
| | | (u@) | 10 |
| 71 | @ | Total | 967 |
| | | (@) | 967 |