Words for the Untutored Mind -
Automating Discovery of Lexical Boundaries

Jeffrey Kinsey *
May 2003

Submitted to the Department of Computer Science, Yale University in partial fulfillment of the requirements for the degree of Bachelor of Science.

When listening to our native language, we perceive a sequence of clearly delineated words. However, examination of audio signals provides information contrary to this intuitive model. One aim of the project described below was to construct a natural language processing system that would model some aspects of language acquisition. The resultant project focuses on one aspect of this problem: namely, the segmentation of continuous speech into discrete words. This paper outlines existing work on speech segmentation and describes a statistical on-line algorithm and implementation attempted in the accompanying source files.

1. INTRODUCTION

Every child grappling with a new and unfamiliar language faces the problem of segmenting continuous speech into usable words. Since fluent spoken language does not contain any reliable acoustic indications of the locus of word boundaries, children presented with the task of learning language must somehow synthesize a lexicon for the target language from clues present in the structure of the speech they hear.

To fluent speakers, the sounds of their native language are perceived as sequences of discrete words. However, speech contains no known auditory indications of word boundaries which could facilitate this perception. [3]. Like children, adults encounter novel words in their linguistic interactions. Such words are easily picked out from their environment as unfamiliar strings of phonemes. One expects that children go through some similar process of extracting novel sound sequences from the context of those that are known. The extent to which this process can be extracted from the greater problem of language acquisition, and thus handled independently, is among the inquiries of this project. In the following paper, I examine a minimalist approach to word segmentation via a recognition based strategy.

2. MOTIVATION

Central to my approach to the segmentation problem is the idea that it is very unlikely that first few sounds of a sentence (or the last few) contain any word boundaries.
Implementations of the kind presented here provide a means by which theories of language acquisition may be tested. The success of such an implementation demonstrates that the assumptions made in the theory are sufficient to learn the target behavior. Aside from providing a usable engineering solution to problems of language acquisition, our model demonstrates sufficient (within the limitations of our success) conditions for the acquisition of a lexicon.

3. BACKGROUND

Much of the design of this project is based on Michael Brent’s work on speech segmentation. Brent’s INCDROP model, in particular, provided a starting point for the recognition based strategy invoked by our algorithm [1]. As outlined in his paper, Speech Segmentation and Word Discovery, there have been three main approaches to the segmentation problem.

These include

- **predictability strategy** unexpected phonemes are hypothesized as word initial
- **utterance boundary based** boundaries are placed after sequences which are commonly seen at the ends of sentences
- **word-recognition based** hypothesize boundaries before and after recognized strings.

INCDROP was the first incremental recognition based algorithm to perform reasonably well on speech segmentation. Earlier models using a recognition based strategy, i.e. explicitly storing units for future comparison, all performed batch analysis of entire dialogs. INCDROP and our algorithm both proceed incrementally, in the sense that the segmentation of a single utterance is finalized before later utterances are examined.

3.1. CORPUS AND RESOURCES

A phonemic dictionary created at Carnegie Mellon University was used in the creation of continuous phonemic transcriptions from the CHILDES database files. The CMU Pronouncing Dictionary is available via anonymous FTP from [6]. The dictionary format uses 39 distinct phonemes, with additional stress marks of none, primary, or secondary for vowels.

The CHILDES database contains transcriptions of interactions between children and adults. The collections (in American English) that I used are available from [7].

In order to construct unsegmented strings of phonemes as a source of test data, I used a small script (translate.pl, described below) to convert words in the CHILDES files into their phonemic equivalents, and write these to new files with word breaks eliminated. Although the produced phonemic sentences do represent a good approximation of the sounds actually produced in utterances, this method of translation has the shortcoming of encoding the pronunciations in isolation of the included words. Ideally, our input would contain transcriptions of the actual phonemic productions, including the transitional changes to pronunciation due to
Following this initial processing, the input to my program consists of sentences, or utterances, delimited by newlines. Each utterance contains a sequence of phonemes with no additional segmentation.

4. METHODOLOGY

4.1. Approach and Algorithm

My approach uses an online word boundary assignment, with the influence of phoneme bigram data gleaned from the text as it is processed. Each utterance, as represented by a line of phonemes, is fully processed before moving on to subsequent utterances. Novel phoneme sequences which begin with a pair of phonemes, or bigram, which has frequently begun a line and end with a bigram which has frequently ended a line are labeled as possible segmentation points as well. A segmentation of the utterance is assigned which maximizes the total score of words and unknown phoneme sequences, normalized to the number of words in the segmentation.

This optimization is characterized by the following segmentation strategies:

- Stored words have an associated score based on frequency and recency
- Unfamiliar strings of phonemes have an associated initialized score if, and only if, they begin and end with ON-SET and ENDLINE bigrams with relatively lower frequencies of containing a boundary

As boundaries are assigned, words chosen are inserted into an initially empty lexicon, each with a score that is subsequently incremented when the word is seen again or decremented if not seen in subsequent input files. This approach builds upon the INCDROP recognition strategy. INCDROP’s method of discovering novel utterances rests on optimization of three quantities: minimize total new words, minimize sum of new word lengths, maximize the product of relative frequencies of segmented units [1].

Input files, as constructed via the Perl scripts described below, consist of newline delimited utterances, each representing a phonemic transcription of a single sentence bounded by pauses. Our algorithm is online, in the sense that a single utterance is fully processed, and assigned a segmentation, before the next utterance is considered.

Recognition of phoneme sequences provides the basis for assigning word boundaries to a given line of input. However, after a large amount of input data has been processed, the likelihood that multiple overlapping strings appear in a given sentence becomes extremely high. Thus, a method for determining which of these strings to take as the word units for the sentence is necessary. My initial methodology used only local information to determine a segmentation order. Under this paradigm, although an utterance was first fully labeled with the recognized phoneme sequences, the decision algorithm compared only adjacent word scores to determine a non-overlapping segmentation. This method proved inadequate, however.

The current implementation (with the -
b flag) uses a greedy algorithm to find a near optimization of an objective function describing the frequency of recognized strings. Novel words are stored with an initial score as specified by a command line option. This score is increased with each successive utterance segmentation which finds the given word.

Pairs of phonemes found at the start of an utterance are labeled as ONSET bigrams. Similarly, pairs found at the end of utterances are labeled as ENDING bigrams. Word segments are selected on the basis of familiarity, i.e. recognized as present in the lexicon, or as a sequence starting and ending, respectively, at bigrams with high ONSET and ENDING scores (in respect to their scores for containing a boundary).

After a segmentation is assigned to a given utterance, the scores for the units selected are increased in the lexicon, all bigrams in the utterance are updated as is appropriate by increase of quantities representing adjacency to or division by a word boundary.

Additionally, this implementation includes a representation of memory decay. As each file is completed, all word scores are decremented via a sigmoid decay function. Thus, words that have been encountered in the file will be left with an increased score, and those lexicon entries not seen will gradually decay to the point of being discarded.

syllables (repped in hashes.txt?)
phonotactics (also repped in hashes.txt?)

On each iteration of the program over a single file, the lexicon and the table of bigram scores are dumped to text files for read in by the next iteration (as controlled by the learndir.pl script).

In order to select against introduction of too many new word-units, the scoring mechanism is normalized against the number of new units introduced.

4.2. Helper Scripts

The source code for this project includes a number of additional programs to manipulate the input data and facilitate creation and organization of results.

**translate.pl** Converts files in the CHILDES *.cha format to ordinary text files with line breaks only at the end of utterances. All utterances but those of the Mother and Father and the recording sessions are eliminated.

**count.pl** Given a directory argument, produces a file ”counts.txt” containing an iterated sum of the distinct words in the input dir.

**learndir.pl** Runs the learn algorithm on given directory with the decay, cutoff and initialization values for word scores specified on the command line.

**supervise.pl** Runs the learning algorithm as a supervised algorithm with the same syntax as learndir.pl. That is, selects correct lexical entries after one pass and reinforces these for a second iteration.

**check.pl** Computes the precision and recall for a given iteration of the algo-
5. OPTIMIZATION

As referenced in the source code documentation, the -o option requests that an optimal summation of the potential word segment scores be computed. Insofar as this is an exceedingly time and memory intensive process (with a possible memory leak), we have included a non-optimal sum-maximization procedure as the default (lacking -o) option.

The optimal summation algorithm performs an exhaustive bit-wise summation of non-overlapping word segment scores in order to select the boundary set with the highest overall score (normalized for the number of words included).

The non-optimal solution produces a best score via a small number of iterations through the line making bitwise switches of inclusion/exclusion of boundaries based on effect on the best score.

6. RESULTS

6.1. Method of Evaluation

Two standard methods of quantifying word segmentation schemas were used to evaluate the performance of this system: the ratio of words discovered to words actually present (true positives / (true positives + false negatives)), or recall, and the ratio of words correctly added to the lexicon to total words added to the lexicon (true positives / (true positives + false positives)), or precision. These two ratios provide a balanced means of evaluating the performance of a given algorithm insofar as they undergo an inverse relationship with changes in the segmentation strategy. If we raise the tendency to add new words to the lexicon, then recall will increase, to the detriment of precision. On the other hand, if we raise the tendency to discard words not immediately reinforced, than fewer false positives will remain in the lexicon, thus increasing precision, but possibly to the detriment of recall.

6.2. Variations

This project is structured in such a way as to facilitate a wide variety of segmentation modes. The rate of memory decay, a threshold below which words are discarded, and an initialization value are each variable quantities specified at runtime. Additional modes include the usage or negligence of bigram information in the segmentation strategy, rejection or allowance of single phoneme word units, and the option of preference for words with highest stress on the first vowel.

6.3. Performance

As determined by a full iteration of compare.pl, one of the best set of parameters (a number of variations produce similar results) is decay of 1%, threshold of .40, and initialization of .50. (Note that the syntax for learndir.pl requires that this be entered as learndir.pl 99 41 50). As run on the bates/free20 directory of [7], this produces 68% recall and 51% precision. In other words, 68% of the words contained in the files (29 in total) of this database where discovered, while 51% of the resul-
tant lexicon correspond to actual words from the CMU dictionary. Equivalently, we have a run on the same database with 1% decay, threshold of .40, initialization of .50 with resultant precision of 48% and recall of 70%.

6.4. Limitations of the Methodology
Unfortunately, there appears to be an implicit limitation on the size of the lexicon that will be created with any given set of running parameters which imply memory trace decay. This is due, in part, to the nature of our memory trace decay system. If this decay is, instead, modeled by simple application of a threshold upon completion of processing, then the limitation is lifted, but to the detriment of precision.

6.5. Known bugs
The perl scripts included in the source archive were written for release v5.6.1. It is known that a bug with the peril function "split" in release v5.8.0 causes problems with these scripts.

The non-optimal segmentation selection algorithm is a severe shortcoming of this implementation. In the processing of short lists of files, the behavior is nearly indistinguishable from the optimal solution. However, as the bigram table grows, the behavior of this algorithm declines rapidly. That we are still able to achieve 68% recall and 51% precision with such a decision process, is in fact, quite surprising.

7. FUTURE WORK
One strong contribution of this project is the incorporation of two distinct prior methods of speech segmentation: a recognition based strategy with the addition of a feature which learns tactic constraints. However, it has become clear that the restriction of lexical entries to those sequences which satisfy our bigram constraints is not beneficial. Rather, the bi-gram information is useful only in determining previously unknown word units.

REFERENCES
6. CMU pronouncing dictionary (v0.3).
ftp.cs.cmu.edu/project/speech/dict

http://childes.psy.cmu.edu/data/english-usa