A RECURSIVE ALGORITHM FOR THE FORCED ALIGNMENT OF VERY LONG AUDIO SEGMENTS

Pedro J. Moreno, Chris Joerg, Jean-Manuel Van Thong and Oren Glickman

Cambridge Research Laboratory,
Compaq Computer Corporation,
One Kendall Square, Building 700,
Cambridge, Massachusetts 02139, U.S.A.

ABSTRACT

In this paper we address the problem of aligning very long (often more than one hour) audio files to their corresponding textual transcripts in an effective manner. We present an efficient recursive technique to solve this problem that works well even on noisy speech signals. The key idea of this algorithm is to turn the forced alignment problem into a recursive speech recognition problem with a gradually restricting dictionary and language model. The algorithm is tolerant to acoustic noise and errors or gaps in the text transcript or audio tracks.

We report experimental results on a 3 hour audio file containing TV and radio broadcasts. We will show accurate alignments on speech under a variety of real acoustic conditions such as speech over music and speech over telephone lines. We also report results when the same audio stream has been corrupted with white additive noise or compressed using a popular web encoding format such as RealAudio.

This algorithm has been used in our internal multimedia indexing project. It has processed more than 200 hours of audio from varied sources, such as WGBH NOVA documentaries and NPR web audio files. The system aligns speech media content in about one to five times realtime, depending on the acoustic conditions of the audio signal.

1. INTRODUCTION

As web search engines evolve from simply indexing text to indexing audio and video, the need grows for tools and algorithms that properly align and index speech to a textual transcription of that speech. In this paper we present an effective recursive technique to force-align very long, and possibly noisy, speech signals to their associated transcript.

The recent popularity and increase of multimedia content on the web has created a problem for the current web indexing technology, which is text based. Since indexing technologies rely on text, multimedia files have to be text transcribed. The obvious solution for this problem is speech recognition, and even though a lot of progress has been achieved in this area (see for example [5] or [4]), the results are still far from perfect.

Very often, however, audio content is available with its associated transcriptions (see *http://www.npr.org* for example). In that case, the problem is not to recognize words but to align the given text with the audio track. When used for indexing purpose, the precision required in the alignment is not critical and an error in alignment of up to 2 seconds is acceptable.

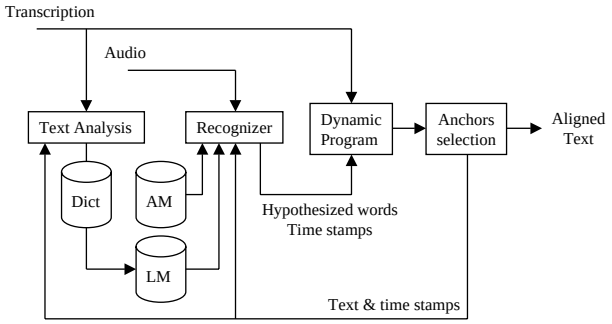

Figure 1: Block diagram of the aligner program.

While aligning speech to its corresponding text might seem a solved problem, in practice it can be very difficult. Two main causes make this a difficult problem, first the length of the audio streams, and secondly the varied acoustic recording conditions. When the speech segment is very long several difficulties arise. For example, the Viterbi search algorithm fails to scale well as the length of the speech segment grows. Its memory requirements increase dramatically, as does the time required to perform the alignment. Furthermore, if the Viterbi algorithm loses alignment at some point in time it might not be able to recover leading to aligned documents which are totally wrong. Simple solutions to these problems such as increasing the beam search width work only on relatively short and noise free speech segments.

The outline of the paper is as follows. In the next section we describe the algorithm in detail. In section 3 we describe our experimental results of running our algorithm on a three hour audio file consisting of TV and radio broadcasts with varied acoustic conditions. We conclude this paper with an analysis of our results.

2. ALGORITHM DESCRIPTION

The novel idea of this algorithm is to turn forced alignment into a recursive speech recognition problem with a gradually restricting

dictionary and language model. Also, a key component of this algorithm is the fact that the alignment is done in several steps, looking at the data several times. This feature helps to prevent it from making mistakes at an early stage from which the algorithm will not be able to recover.

Figure 1 gives a block diagram of the alignment algorithm.

We assume that the algorithm starts with a feature representation of the audio file. In our case we use a 13 dimensional mel-cepstrum feature vector augmented by its first and second order derivatives resulting in a 39 dimensional vector. The text analysis module processes the text file and creates a dictionary. We use the CMU public domain dictionary (*http://www.speech.cs.cmu.edu/cgi-bin/cmudict*) to translate each word into a phonetic sequence. For those words not present in the CMU dictionary we use an automatic module build by Kevin Lenzo at CMU and publicly available which is a modification of an algorithm introduced by Daelemons [2].

This module also creates a language model for the transcript. Rather than building a complete language model based on N-grams we build a simple word pair and triple model. Computing such simple models gives us significant speed ups, while causing no observed loss in accuracy.

The next step consists of running the speech recognition system using the previously generated language model and dictionary. We use the speaker independent SPHINX II speech recognition system [3] as our speech recognition engine. In fact, any large vocabulary speech recognition system could have been used. No specific training, adaptation, or robust speech processing of the acoustic models is performed.

Once a complete hypothesis text string is produced for the whole audio stream we align it with the correct transcript. To perform this alignment we use dynamic programming techniques to find the globally best alignment. Next we choose *anchors* which are the parts of the alignment of the hypothesized text and transcript that we are most confident are correct. These anchors are used to partition the text segment and the audio segment into unaligned and aligned segments. Then we iterate, repeating the algorithm on each unaligned subsection. At each iteration the language model and dictionary are rebuilt to limit the list of active words and words sequences to those found in the transcript of this subsection. This has the double effect of speeding up the recognition and ensuring that only those words and their word pairs and triples that we know are available in the segment are actually searched for.

These steps are repeated on the unaligned segments until a termination condition is reached. The termination condition on a segment can be fully aligned text, or when a particular unaligned segment has a duration less that a predetermined threshold. The recursion also terminates if the recognizer is unable to find any additional words in the audio segment.

Figure 2 shows how the algorithm progresses in the alignment of a long segment. Each block represents a word and the block lengths represent the word duration. Each iteration of the algorithm is represented with two lines, labeled *a* and *b*. Lines labeled

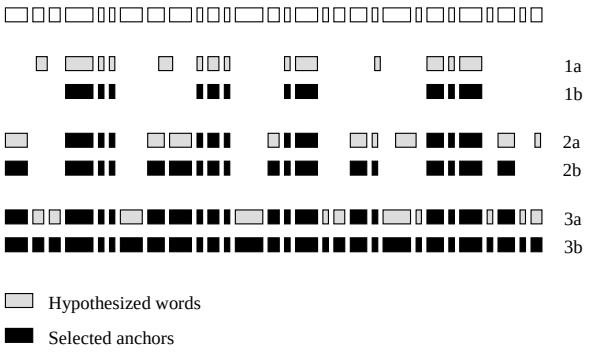

Figure 2: Example of the aligner run. Each set of two lines, *a* and *b*, represents one iteration of the algorithm.

with *a* show hypothesized words which the dynamic programming module aligned with the transcript. Lines labeled *b* show which of these hypothesized words have been chosen as anchors. In addition, each line includes all the words previously selected as anchors.

2.1. Anchor selection

The goal of choosing the anchors or *islands of confidence* is to choose sections that are highly likely to be correctly aligned. Once chosen, the anchors will be assumed to be correct and only the segments between anchors will be selected for another pass of the recognition system.

To choose anchors, we make use of the observation that after performing the dynamic alignment, any long sequence of consecutive words that are aligned between the transcript and the hypothesized words are usually correctly aligned. So we chose a simple heuristic to select anchors, namely, choosing sequences of N or more consecutive words that are aligned.

The confidence in the anchors is related to the number of words N in the islands. A large value of N forces the algorithm to trust only long islands, resulting in fewer but longer unaligned segments for subsequent processing. On the other hand, a small value of N generates more but shorter unaligned segments. A large value of N decreases the probability of error at the cost of increasing the number of iterations and thereby the runtime. The value of N is dynamically decreased as we keep iterating and processing shorter and shorter audio segments.

There are many other techniques one could use for choosing anchors. For example, we could give greater weight to long words when choosing anchors. Alternatively, acoustic confidence metrics (*e.g.* [1]) could be used in combination with the current criterion. Our simple metric for choosing anchors has worked well enough that we have not tried any of these alternate techniques.

2.2. Benefits of recursion

There are a number of benefits to using this recursive technique. Firstly, the algorithm begins by making decisions (ie. choosing anchors) only about the portions of the audio where it is most confident. Decisions that are more likely to be wrong are delayed

to the later iterations, where an incorrect decision will have less impact.

Secondly, at the lower iteration levels, we deal with fairly small pieces which have a very restricted language model. This helps the alignment work well in the face of noisy inputs.

Thirdly, this technique works well even when there are gaps or errors in the transcript text. The recursive process will typically align the correctly transcribed parts first, leaving the area around the errors for later. This puts the errors in a smaller region, thus limiting the effect of the transcript errors.

2.3. Related Work

We are aware of two previous research efforts that deal with the alignment of large audio files. Placeway [6] introduces the concept of using transcripts to improve recognition performance. He also mentions the problem of aligning a long speech sequence with its corresponding text and suggests the use of a global minimum string edit distance alignment between the decoded string and the correct one. Our use of dynamic programming to align the hypothesized and correct text strings is similar in spirit to Placeway's work. However, we go beyond by introducing the concept of anchor segments or islands of confidence.

Robert [7] presents a system which is similar in spirit to the one presented here. There are however some important differences between this work and Robert's. First, we use a large vocabulary speech recognition system rather than a simple phoneme recognizer. Secondly, we introduce the concept of iterative alignments, where we perform recursions which continually narrow the active dictionary and language models and improve the alignment qualities. In contrast, Robert's system searched for specific utterances over the whole audio segment one by one. There is no concept of narrowing the scope of the language model and dictionary.

3. EXPERIMENTAL RESULTS

To evaluate the performance of the system we present experimental results on the 1997 hub4 evaluation audio file [5]. This file is about three hours long and is composed of several audio broadcasts recorded in varied acoustic conditions. Portions of the audio stream are quite clean speech while others contains speech over music, telephone interviews, interviews in the field with difficult background noises, etc.

Since sentence time marks are also provided with this audio, we can easily run a normal Viterbi aligning algorithm on each sentence to produce "ground truth" time alignments. These true alignments can then be used to compare with the alignments produced by running our algorithm on the entire three hour audio stream.

Figure 3 shows a histogram of the difference in time between the ground truth time alignments and the alignments produced by our algorithm. 98.5% of the words are off by less than 0.5 seconds from the true alignments and 99.75% are within 2 seconds.

Since the testing audio stream is labeled acording to acoustic conditions it is easy to break down the histogram. For example, in the

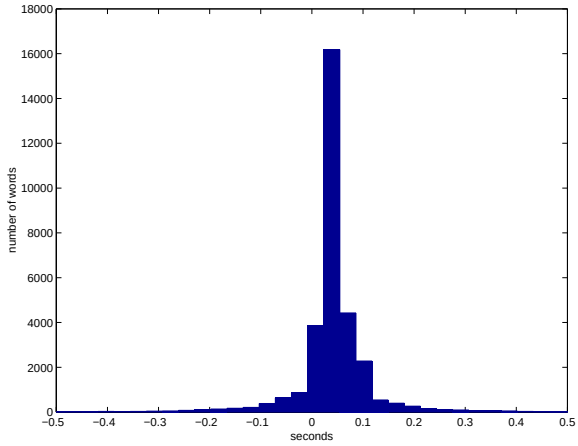

Figure 3: Histogram of the time difference between the ground truth and the alignments produced by our algorithm for the whole 1997 hub4 evaluation data.

speech over music acoustic condition, which is the one which performs the worst, 99.52% of the words are within a 2 second of the true alignment.

Since the percentage of clean and easy to anchor segments is quite large (44% in our dataset), it can be argued that the results on the noisy speech segments are overly optimistic. The clean segments provide lots of good anchors making the alignment of noisy speech much easier. In addition, this hub4 dataset might not be representative of more realistic acoustic conditions.

To test our algorithm under conditions which are noisier and conditions which are similar to those encountered on audio streams distributed over the web, we performed two additional experiments. In the first one we added artificially generated white noise at a signal to noise ratio of 15 dB. In the second experiment we encoded our dataset using the popular RealAudio signal encoder and then decoded the signal. We used the most widely used format available on the internet, the 8.5 Kbps voice audio codec. Then we reran the alignment algorithm on this data and compared the results to the ground truth.

Figure 4 shows that even when the audio signal is contaminated with white noise at a global SNR of 15 dB, our algorithm still performs well. The mean of the time error is 2.4 seconds with a standard deviation of 19.4 seconds. 94.3% of the words exhibit an error of less than 2 seconds, compared to 99.7% for the original audio signal.

When the signal is RealAudio encoded the performance degrades slightly but still provides results that are good enough for indexing purposes. The histogram in Figure 5 shows that the difference between the ground truth and the estimated alignments has a mean value of 0.062 seconds and a standard deviation of 0.59 seconds. 99.02% of the words exhibit an error of less than 2 seconds, compared to 99.7% for the original audio signal.

In terms of computational requirements, running on a 400 MHz alpha processor with 256 MB of memory, the system process the

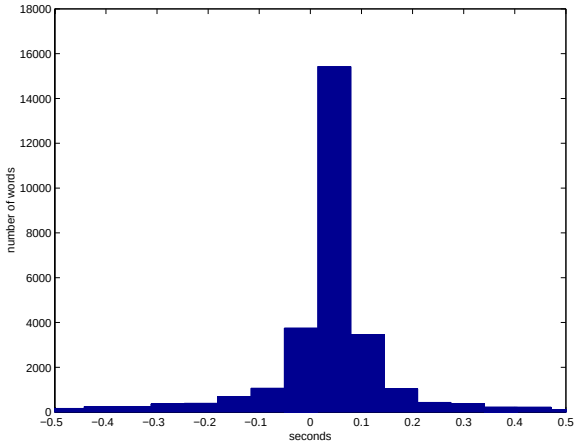

Figure 4: Histogram of the time difference between the ground truth and the alignments produced by our algorithm for the hub4 evaluation audio data with white noise added at a global SNR of 15 dB.

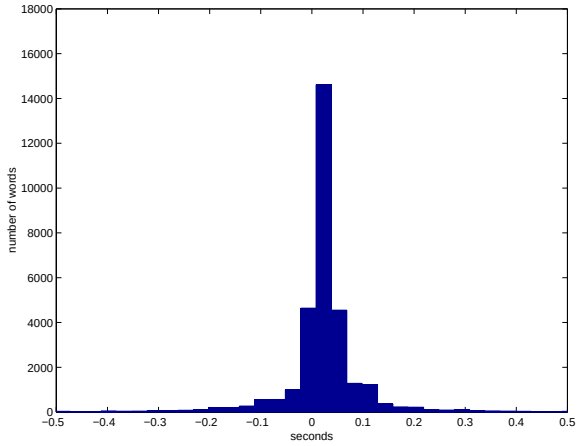

Figure 5: Histogram of the time difference between the ground truth and the alignments produced by our algorithm for the RealAudio encoded 1997 hub4 evaluation data.

audio stream in 1.7 times real time for the original audio, 2.4 times real time for the 15 dB artificially corrupted audio, and 1.9 times real time for the RealAudio processed signal.

The system has run on more than 200 hours of speech audio streams, ranging from WGBH video documentaries, to internal Digital Equipment Corporation archives and on National Public Radio (NPR) RealAudio encoded programs. Although we do not have ground truth alignments to compare to, the quality of the alignments appears to be similar to that presented earlier.

4. CONCLUSIONS

We have presented a new approach to force align very long audio streams (one hour or more) to their text transcriptions. Experiments show that the accuracy of the system is good enough to efficiently index audio content. We also showed that the algorithm is effective in noisy conditions because of the way it proceeds, by selecting anchors first, and iterating recursively on the portion of audio in between these islands of confidence. If the islands of confidence are misaligned, the error only propagates around these words. Future work will focus on improving the confidence on islands in order to increase robustness of the algorithm regarding noise and audio or text insertions/deletions. The algorithm has been successfully integrated in our multimedia indexing system which provides Web access to hundreds of hours of television documentaries, corporate archives, and radio programs.

5. ACKNOWLEDGMENTS

We would like to thank all the members of the Speech and Multimedia Indexing groups at the Compaq Cambridge Research Lab, in particular Brian Eberman and Mike Sokolov for their help in integrating this system to the MediaVista multimedia indexing engine. We also thank Dave Goddeau for his help in the language modeling aspects of the algorithm. We would also like to thank Dave Kovalcin from the Compaq Unix Group for his valuable feedback and remarks while testing the algorithm.

6. REFERENCES

1. L. Chase. Word and Acoustic Confidence Annotation for Large Vocabulary Speech Recognition. In *Proceedings Eurospeech*, 1997.
2. W. Daelons and Bosch V. D. TabTalk : Reusability in data-oriented grapheme-to-phoneme conversion . In *Proceedings Eurospeech*, pages 1459–1462, 1993.
3. X. Huang, F. Alleva, H. Hon, M. Y. Hwuang, K. F. Lee, and R. Rosenfeld. The SPHINX II Speech Recognition System: An Overview. *Computer Speech and Language*, 2(7):137–148, 1993.
4. D. S. Pallet. Overview of the 1997 DARPA Speech Recogntion Workshop . In *Proceedings of the DARPA Speech Recognition Workshop*, 1997.
5. D. S. Pallet, J. G. Fiscus, A. Martin, and M. A. Przybocki. 1997 Broadcast News Benchmark Test Results: English and Non-English. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, 1998.
6. P. Placeway and J. Lafferty. Cheating with Imperfect Transcripts. In *Proceedings ICSLP*, 1996.
7. J. Robert-Ribes and R. G Mukhtar. Automatic Generation of Hyperlinks between Audio and Transcripts. In *Proceedings Eurospeech*, 1997.