CS 224N Final Project: Speech Summarization for Rapid Playback

Charles DuHadway

June 7, 2008

1 Introduction

Searching audio streams is currently a tedious task. There exist no tools that allow a user to quickly scan through a long video lecture or audio book using only audio cues.

In this paper I propose and build a simple system that alleviates the burden of searching through a large amount of audio data. This system use concepts from other text and speech summarization work, but focuses on the problem of search.

2 Motivation

Digital speech recordings in the form of lectures, books, news, etc... are increasingly common. Many universities, including Stanford, provide video recordings of courses. These videos can be used to shift learning in time and space, which is incredibly useful. However, the value of these video sources could be further extended by increasing their accessibility to search.

At Stanford a typical exam may cover material from twelve 75 minute lectures. To listen to each lecture once requires 15 hours. Searching for a specific topic in those twelve 75 minute lectures could, in the worst case, require 15 hours.

Of course, one could increase the playback speed of the lectures while applying pitch correction. At best this could decrease the amount of time required by a factor of 2: only 7.5 hours required to search through the lectures.

Imagine if the same task could be performed with a speedup factor of 10, 20 or 50. This would reduce search times to 1.5 hours, 45 minutes, or even 18 minutes. I propose a simple system that achieves arbitrarily high levels of playback while minimizing the information loss. This system is designed for quickly locating areas of interest in a long audio stream.

One common solution to this problem is simply to replace the audio data with text. This immediately gives all the advantages of text, including efficient

search. However, this also immediately removes the advantages of an audio stream, such as no dependency on a proper screen.

3 System Description

The goal of the proposed system is to reduce playback time of an audio stream while minimizing information loss. We'll accomplish this by selectively throwing away those parts of the audio stream that contain the least amount of information.

As the user specifies higher playback speeds we'll throw away more and more pieces of the audio stream. While doing this we'll also need to ensure that the remaining pieces are, to a first approximation, evenly spread in time. This prevents large contiguous sections of the original audio stream from being completely removed which would prevent their content from being searched.

We accomplish this in two steps. First, the original audio stream is partitioned into segments of roughly equal size, as measured in seconds. Consider the following phrase taken from our test data set. It has been partitioned into four segments as indicated by the "|" symbols:

> The Massachusetts Senate gets to work on a three hundred fifty million dollar | deficit reduction package today, brth with republican leaders | preparing to unveil a series of amendments brth they say will | chop an additional two hundred fifty million dollars off the deficit.

Given a specific partition the next step is to eliminate those parts of each segment, or window, with minimal information content. This might result in the following progressive cropping:

> Massachusetts Senate work on three hundred fifty million dollar | deficit reduction package republican leaders | unveil series of amendments they say will | chop an additional two hundred fifty million dollars off deficit
>
> Massachusetts Senate work | deficit reduction package republican leaders | unveil amendments will | chop additional dollars deficit
>
> Massachusetts Senate | deficit reduction | unveil amendments | chop deficit
>
> Senate | deficit | amendments | chop

Each level of cropping represents a different trade-off between audio length and content.

We need a method of estimating the information density of pieces of the audio stream. This same problem has been solved many times in the text

and speech summarization literature. We'll explore several possibilities in the implementation section below.

4 Implementation

Dividing the original input into windows guarantees that the output audio sequence will contain utterances drawn from each section of the original. The window size can range from the entire stream to the size of the average utterance. Larger windows provide more flexibility in maximizing the information density of chosen utterances, but provide only weak guarantees about timing uniformity of the chosen utterances.

The maximum duration of chosen utterances within a window is a function of window count, $|w|$, and the maximum duration of the output stream, d_s:

$$d_w \leq \frac{d_s}{|w|}$$

For each window we can score each utterance as a function of its frequency within the window, $f_{u,w}$, its frequency within the total corpus, f_u and its duration, d_u:

$$s_u = \frac{f_{u,w}}{f_u} \frac{1}{d_u}$$

We can then sort the utterances based on their score, s_u and greedily choose from the top of the list until adding another utterance would exceed d_w. We then sort the chosen utterances by time, so that their original ordering is preserved.

Note that I haven't yet defined an utterance. In this context an utterance could exist of a unigram, bigram or trigram.

5 Training and Testing

The Boston University Radio Speech Corpus, as provided by the LDC, was used for training the system. The data set contains over seven hours of speech and over 120,000 separate utterances. Word-aligned transcription and POS tagging are included with the data set.

Word-aligned transcription was the main reason for using this specific data set. This allowed me to easily segment and recombine the audio streams based on word level utterances.

I was unable to find any human summaries of the type this system is designed to produce. To gain some automatic measurements of the system I created a small set of human summaries. The human summaries were made by first presenting a subject with a paragraph level excerpt of a radio broadcast. This excerpt was then segmented into windows, as described above, and for each segment the subject was asked to pick the first and second most descriptive

words in the segment. By posing the same task to the automatic system I could then compare the results.

Two different human subjects each provided summaries for two separate excerpts. The target playback speed was set for a factor of 10, and the widows were sized to allow approximately one utterance per window. The level of agreement between each summary was measured as the proportion of windows in which the first choice of one summarizer matched either the first or second choice of the second.

Note that this results in an asymmetric measure, so both results are reported in Figure 1.

	System	Human 1	Human 2
System	1.0	0.43	0.19
Human 1	0.20	1.0	0.48
Human 2	0.12	0.53	1.0

Figure 1: Agreement between system and human summaries.

Observing the differences between the human and automatic summaries led me to make two additions to the automatic system. First, the system was more likely to choose very short words such as "me", "to" or "two" than humans which also tended to be difficult to understand when playing back the output audio stream. Second, the rate of disfluencies, such as "uh", is so low in this particular data set that the system was giving them high enough scores to be included in some summaries.

I added penalty terms to very short words and to disfluencies as defined by POS tagging. This resulted in an increase in agreement with the human summarizers as shown in Figure 2.

It also appeared that the human summarizers, particularly Human 2, preferred using nouns to verbs in their summaries. However, no simple encoding of this fact in the scoring mechanisms led to improved results and so it was not accounted for in the final system.

	Automatic System	Human 1	Human 2
Automatic System	1.0	0.52	0.24
Human 1	0.37	1.0	0.48
Human 2	0.17	0.53	1.0

Figure 2: Agreement between system and human summaries.

The collected data also seemed to show that the human summarizers were not taking into account the length of the utterance. By removing this term from the utterance scoring equation in the previous section agreement between the automatic and human systems rose from 0.52 and 0.24 to 0.57 and 0.29

respectively. However, removing this term appeared to decrease the quality of the actual output audio streams as fewer utterances were included.

6 Conclusion and Future Work

Using word frequency as the main criteria for estimating significance was first introduced in [2]. Other features could also be used. With enough labeled training data it would be possible to train simple classifiers that could choose the most informative features. One interesting possibility would be to use only prosodic features, eliminating the need for transcription or POS tagging. [1] and [3] have shown that this approach can be effective in the context of speech summarization.

The main limitation of the above work was an inability to accurately measure the performance of the system. Ideally, one would want to do task oriented studies to measure the effect such a system has on accomplishing specific tasks, such as searching audio streams for specified topics. Collecting a larger amount of human summaries would also be helpful for testing potential improvements to the current system.

A second limitation of this work was a function of the data set used. Radio news segments have specific characteristics that are not typical of many audio streams. For example, they contain very low rates of disfluencies and have relatively high information density.

For this work only single words were considered when estimating information content. It would be interesting to extend the system to consider bigram and trigram utterances as well. This would likely require a larger data set for more accurate language modeling.

References

[1] Konstantinos Koumpis and Steve Renals. Automatic summarization of voicemail messages using lexical and prosodic features. *ACM Trans. Speech Lang. Process.*, 2(1):1, 2005.

[2] H. P. Luhn. The automatic creation of literature abstracts. *IBM Journal of Research Development*, 2(2):159–165, 1958.

[3] Sameer Maskey and Julia Hirschberg. Summarizing speech without text using hidden markov models. In *HLT-NAACL*, 2006.