On Building a Concatenative Speech Synthesis System from the Blizzard Challenge Speech Databases

Wael Hamza[1], Raimo Bakis[1], Zhi Wei Shuang[2], Heiga Zen[3]

[1]IBM T.J. Watson Research Center
[2] IBM China Research Lab
[3] Nagoya Institute of Technology

[1]{hamzaw, bakis}@us.ibm.com
[2]shuangzw@cn.ibm.com
[3]zen@ics.nitech.ac.jp

Abstract

In this paper, we compare two methods of building a concatenative speech synthesis system from the relatively small, "Blizzard Challenge" speech databases. In the first method we build a system directly from the Blizzard databases using the IBM Concatenetative Speech Synthesis System originally designed for very large speech databases. In the second method, a larger database is used to build the synthesis system and the output is "morphed" to match the speakers in the Blizzard databases. The second method outperformed the first while maintaining the identity of the Blizzard target speakers.

1. Introduction

To construct an utterance, a concatenative TTS system divides the utterance into segments, looks for corresponding segments in its prerecorded database, and concatenates them to form the utterance, possibly modifying the segments by signal processing. If the database is large, the search algorithm is likely to find long segments that together cover most of the utterance very well and require little or no signal processing. The resulting speech is often indistinguishable from natural utterances. With a sufficiently large and varied database, the system even has the freedom to choose segments that express a particular style or mood. In general, synthesis of realistic speech becomes easier as the size of the database increases.

Although the processing speed and memory available in a given size physical system have greatly increased in recent years, this has not translated into corresponding relaxation of constraints for marketable TTS systems. Rather, the market is now demanding more speech capability in constrained environments such as cell-phones and PDA's, as well as in automobiles. In addition, our customers want to produce systems that speak with specific, distinctive voices. The available database for such a voice may be small, and it may not be feasible to record the additional 40 or more hours of speech from the same speaker that would be needed for a conventional high-quality concatenative TTS. The need for improved synthesis from a small database is evident.

The smaller the database, the less likely the system is to find what it needs. It may, for example, have to build syllables out of individual phones or pieces of phones. Even so, the short segments will not fit together very well, and will not have the desired style, level of emphasis, intonation, or prosody. The system may be able to improve the quality by modifying the pitch, duration, formant frequencies and other parameters of the speech signal. To search for the best possible segments in the database and to modify them appropriately, the system will have to know how the target utterance would have sounded if had been in the database. Thus, it needs prior information to supplement its empirical database. To a large extent, this additional information ultimately comes from other empirical observations, not explicitly included in the small database. This may include general observations about the shapes of pitch and formant trajectories in human speech, as well as language- and dialect-specific rules about durations, pitch patterns, co-articulation, etc. This information may be obtained by human analysis of speech samples, or by mathematical algorithms applied to speech corpora other than the small database in question.

In this paper, we explored two approaches, one relying essentially only on the small database, the other synthesizing entirely from a different, large database, and treating the small one only as a source of additional information for modifying or "morphing" the synthetic speech to resemble the small-database speaker. The first method suffers from the typical problems of database inadequacy, while the second raises the difficult questions of how to measure the similarity to the target speaker, and how close is good enough?

The rest of the paper is organized as follows. After a brief overview of the IBM Concatenative Speech Synthesis system, designed primarily for large databases, we describe the application of this system to the small Blizzard databases, and the adaptations we made for that purpose. Next, we describe our other approach, in which the synthesis was done from a large database, and the results were converted by morphing to resemble the Blizzard target speakers. We present listening results both from the Blizzard challenge and additional tests we conducted for the purpose of estimating the similarity of our morphed voices to the target voices. Finally, we discuss the advantages and disadvantages of each approach, suitability for various potential applications, and work yet to be done.

2. The IBM Concatenative Speech Synthesis System Overview

A professional speaker is directed to record a large number of sentences in a friendly energetic style. This data is used to build a speaker dependent Hidden Markov Model (HMM). In these models, each phone is represented by a typical left to right 3-state HMM. The resulting models are used to align the

recorded sentences to their phonetic labels. Using the state alignments, a top-down likelihood-based acoustic tree is built for each HMM state using the standard acoustic tree growing algorithm used in speech recognition. The leaves of the resulting trees are considered to be the synthesis units and all segments aligned to any particular leaf are candidates for this leaf.

The recorded sentences are also used to grow a statistical decision tree for energy, duration, and pitch. Energy, duration, and pitch are predicted on the leaf, phone and syllable level respectively.

In addition to that, all words that occurred in the recorded sentences along with an index to the corresponding speech segments are kept in a dictionary [1] henceforth called the "alternates' dictionary." The alternates' dictionary is used during the segment search to allow segments that belong to a certain word to be considered during the search even if the front end predicts different phonetic pronunciation for this word.

During synthesis, the input is processed by a typical front end. The acoustic decision trees use the resulting phonetic sequence to generate one of leaves. Energy, duration, and pitch trees are used to generate the target prosody for the given sentence. Then, for the given leaf sequence, a standard segment selection takes place through a lattice of segment candidates that corresponds to the given leaf sequence.

The output speech is constructed from the search-generated segment sequence as follows. To avoid large pitch modification to the original signal, a piecewise linear connection of the observed end pitch of each selected segment is constructed. This contour is smoothed [2] to eliminate any rapid fluctuations which sound as if the speaker were shaking or distressed. Speech is then generated using the contiguous bypass algorithm described in [1], which aims to minimize distortion introduced by signal processing by bypassing this processing in cases in which the engine is using a sequence ("chunk") of segments which were contiguous in the original recordings. Using a signal processing algorithm similar to Frequency Domain Pitch Synchronous Overlap Add [3], segments not belonging to contiguous chunks are then modified in pitch to match the smoothed pitch contour and, optionally, in duration to match the target duration. Contiguous chunks are broken into two parts, "internal" segments and "boundary" segments. The internal segments are copied sample by sample to the output buffer, while boundary segments on either side provide a smooth transition between the sample-by-sample copy and our regular signal processing. This algorithm eliminates signal processing distortion in large contiguous segment chunks and provides a seamless transition to regular signal processing. The contiguous bypass algorithm is applied only to contiguous chunks that exceed a specified number of segments. This constraint prevents distortion resulting from switching too frequently between regular signal processing and sample-by-sample copying. The overall cost function is tuned to bias toward selecting long contiguous chunks; doing so makes the contiguous bypass algorithm more effective. Detailed description of the IBM system can be found in [2].

3. Direct System Building Method

In the Blizzard challenge, the CMU ARCTIC databases consist of 2 male (BDL and RMS) and 2 female (SLT and CLB) speakers were provided. For each speaker, 1132 phonetically balanced sentences greedily extracted from the Gutenberg novels were recorded at 16 kHz sampling.

We used the IBM regular building process to build the speakers' databases directly from the Blizzard databases. No manual corrections were made to the resulting phonetic alignments or pitch prediction. The acoustic decision trees were made a little smaller than the regular ones to compensate for the small size of the speech database.

Instead of using the Blizzard data to build duration and pitch prediction trees, we used ones built from a very large speech databases. Those databases are the same ones used by the voice morphing method in the next section.

After building whole systems, system setting and cost weights were tweaked by hand using sentences not included in the test materials. The alternates' dictionary was not used because it was found to degrade the output speech quality. This may be due to the small number of words in the speech database. Building time of individual systems was about 6 hours and their footprints were around 120 MB.

4. Voice Morphing Building Method

The second approach did not use the small databases for synthesis at all. Instead the required utterances were synthesized by the IBM Concatenative Speech Synthesis System using speaker databases already in our repertory, each one containing in excess of 8000 sentences. The texts of the utterances were supplied to the system with no markup other than normal punctuation. The speakers were, however, selected to be appropriate for the voices which were to be simulated.

Four types of modification were applied to the synthesized utterance in order to mimic the target speaker:

- Rate of speech adjustment (words per minute)
- Static and dynamic pitch contour modification.
- Vocal tract length adjustment
- Spectral envelope equalization.

The rate of speech factor was applied uniformly to all sounds, and the appropriate factor was determined by listening.

Pitch contour adjustment consisted of two parts: static pitch mapping, and modification of dynamic characteristics. Static mapping consisted of a linear transform applied to the F0 value in the log domain. Thus, if F_{0s} is the source pitch and F_{0t} the target pitch, then: $\log F_{0t} = a + b \log F_{0s}$, where a and b are chosen to transform the average log pitch of the source speaker to the average log pitch of the target speaker, and to transform the log pitch variance of the source speaker to that of the target speaker.

Dynamic pitch adjustment consisted of a linear filter applied to the log pitch function. Specifically, in this example, it was found necessary to apply a smoothing filter to the pitch contours of some speakers, because the synthetic speech had more rapid pitch fluctuations than the target speech. A filter with zero phase-shift was used, to prevent the pitch contour from becoming desynchronized with the text.

Vocal tract length adjustment consisted of a linear stretching or compression of the frequency scale, as needed. A preliminary estimate was computed by minimizing the distance

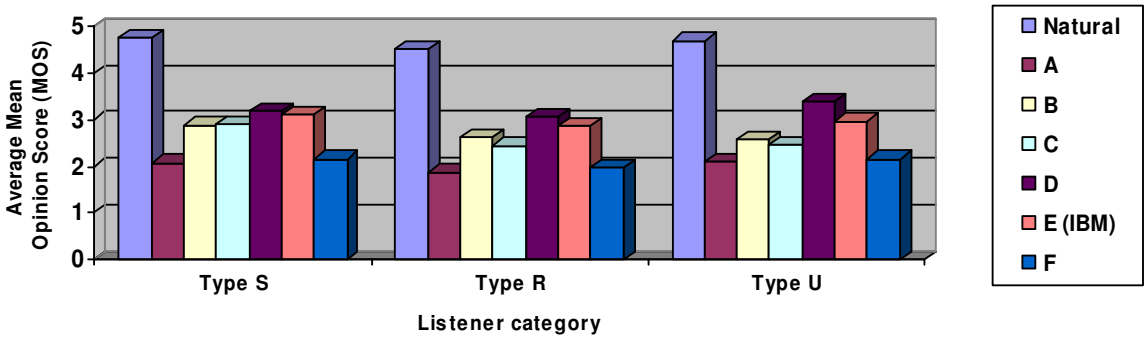

Figure 1: Average Mean Opinion Scores of the natural and synthetic speech evaluated in listening test conducted in the Blizzard Challenge 2005.

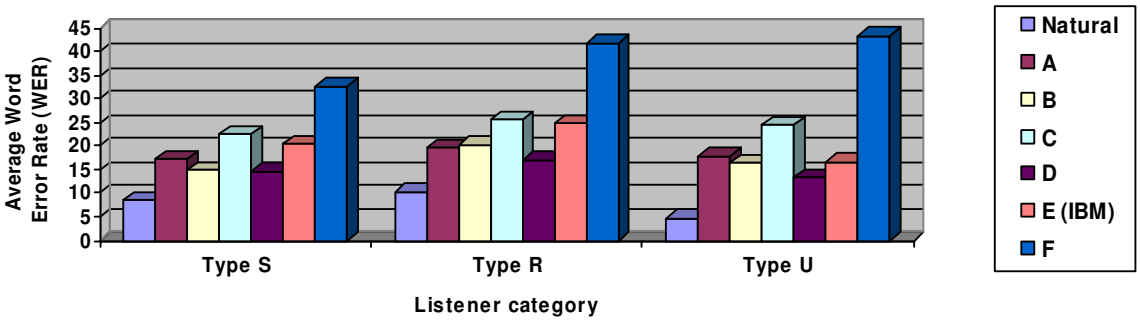

Figure 2: Average Word Error Rates of the natural and synthetic speech evaluated in the listening test conducted in the Blizzard Challenge 2005.

between the average target spectrum and the average warped source spectrum. This estimate was then refined by ear.

Spectral envelope equalization was calculated from the average power spectra of the source and target speakers after vocal tract length normalization.

The STRAIGHT [4] algorithm was used to re-synthesize the modified signal.

5. Results

Test sentences provided by the organizer consist of five different domains:

- Gutenberg novels
- Standard news text
- Conversational/dialog sentences
- DRT/MRT phonetically confusable words within sentences
- Semantically unpredictable sentences

They were independently generated and selected to minimize text processing, and out of vocabulary word pronunciation. The listening tests done in the Blizzard challenge consisted of MOS (Mean Opinion Scores), where listeners heard sentences from different systems in random order and gave a score of 1-5, or, in the case of phonetically confusable words and semantically unpredictable sentences, typed in the words they heard, and the word error rate (WER) was measured. In this evaluation, 3 types of listeners were used: US English native undergraduate students (Type U, around fifteen individuals), speech synthesis experts (Type S, around fifty individuals), and also open evaluation by anyone on the web (Type R, around seventy individuals). Figures 1 and 2 show the average MOS and WER of the natural and synthetic speech, respectively. Our direct system was the second in the MOS tests and fourth or fifth in the WER. The results were not as good as we thought. It may be because our direct system is optimized for larger databases. Details about the Blizzard listening test done by the organizers can be found in [5].

In order to compare with the second method, two internal listening tests were performed. These two tests were applied to one male (BDL) and one female (SLT) speakers from the Blizzard speakers. In the first test, we selected 30 sentences (15 for each speaker) from the Blizzard listening sentences and generated them with both direct and voice morphing methods. The resulting 60 sentences were presented to a group of 30 native speakers of US English and they were asked to give a score of 1-5. Results are shown in Figure 3.

The results were not surprising. The IBM morphed output was expected to be significantly better than the direct one since the voice database is larger and the IBM system was originally designed to work with very large databases. In addition to that, the quality degradation due to morphing seemed to be overlooked because of the poor synthesis quality of the direct method. The question that remains after this test is, how similar does the morphed speech sound to the target Blizzard speaker?

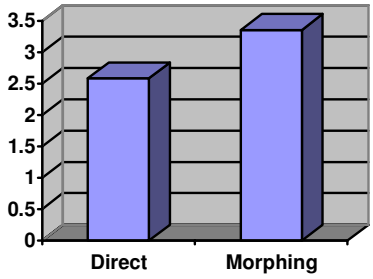

Figure 3: Mean Opinion Score results from the internal listening test comparing the two approaches

In order to answer this question, a similarity test was conducted. 10 natural sentences from the target speaker (Blizzard Natural), 10 natural sentences from the source speaker (IBM Natural), 10 synthetic sentences from the direct method (Direct), and 10 synthetic sentences from the voice morphing method (IBM Morphed) were presented to a group of 30 listeners of native speakers of US English. The listeners were asked whether the sentence sounds like the source speaker or the target speaker using a binary choice. The listeners have the chance to listen to the source and the target speaker speech each time they here a sentence from the listening stimuli. Results from this test are shown in Figure 4.

It is very clear from these results that the morphed speech sounds like the Blizzard speaker. This is due to the proper choice of the source speaker and the manual tuning of the morphing parameters.

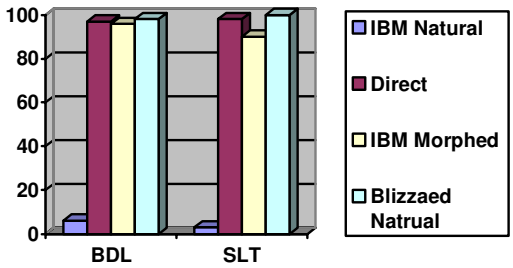

Figure 4: Similarity test results. The Y axis represents the percentage of the listeners voting that the sentence they hear sounds like the Blizzard speaker. Results are presented for BDL (male) and SLT (female) speakers

6. Discussion and Future Work

Because any small-database TTS system will use additional information from other sources, at least implicitly, we decided to explore the benefits of explicit introduction of such data by using an alternate, large database for the actual synthesis. We found that the four types of signal modification described in section 4 go a long way toward making the synthetic speech similar to the target speaker. These types of modification are relatively easy to do with little danger of serious signal distortion. They can not, however, handle serious differences in dialect. Codebook-based methods are somewhat more flexible, but are more difficult to apply.

In addition to the formal listening test we conducted, an additional criterion might be the acceptability of the morphed voice when interspersed with recorded phrases from the original voice. For example, the morphed voice might be used to synthesize variables in carrier phrases recorded by the original voice.

7. Acknowledgement

The authors would like to thank Allen Delmar for his help in conducting the listening tests for comparing the two methods and for the speaker identity comparison test. We also would like to thank Ellen Eide of IBM Research for the very interesting discussion on designing the similarity test.

8. References

[1] Hamza, W., Eide, E., Bakis, R., "Reconciling Pronunciation Differences between the Front-End and the Back-End in the IBM Speech Synthesis System," *Proc. ICSLP 2004*, Korea.

[2] Eide, E., *et al.*, "Recent Improvements to the IBM Trainable Speech Synthesis System." *Proc. ICASSP 2003*, Hong Kong, Vol. 1, pp. 708-711.

[3] Moulines, E. and F. Charpentier, "Pitch-Synchronous Waveform Processing Techniques for Text-to-Speech Synthesis Using Diphones," *Speech Communication*, Vol. 9, No. 5-6, 1990, pp. 453–467.

[4] Kawahara H., "Restructuring speech representations using a pitch-adaptive time frequency smoothing and a instantaneous-frequency-based F0 extraction: Possible role of a repetitive structure in sound", *Speech Communication* 27 (1999), pp. 187-207.

[5] Bennett, T., Syrdal, A., "Large Scale Evaluation of Corpus-Based Synthesizers: Results and Lessons from the 2005 Blizzard Challenge", Submitted to Interspeech 2005.