Statistical Speech Synthesis for the Blizzard Challenge 2008

Pieter Scholtz, Albert Visagie, Johan du Preez

Centre for Language and Speech Technology,
Department of Electrical and Electronic Engineering,
Stellenbosch University, South Africa

pscholtz@dsp.sun.ac.za, avisagie@dsp.sun.ac.za, dupreez@dsp.sun.ac.za

Abstract

In this paper we present the details of our first entry for the Blizzard Challenge 2008 speech synthesis evaluation. Our system is a new HMM-based text-to-speech voice building system for the HTS runtime synthesis engine. We entered two UK English voices, one built from the full database provided, A, the other from the Arctic subset, B and one Mandarin voice built from the provided database. The evaluation results show that our system provides very good intelligibility, moderate naturalness and low similarity to the original speaker.

1. Introduction

In recent years the popularity of HMM-based speech synthesis has risen significantly. The HMM-based approach has numerous advantages over traditional unit selection-based techniques. Arguably the most significant of these is the relative ease with which high quality voices can be constructed in new languages. Building a voice in a new language requires a small speech database and a minimal set of linguistic resources, e.g. a lexicon and/or letter-to-sound rules.

This makes HTS a particularly attractive solution for minority languages with limited linguistic resources, as evidenced by the good quality synthesis achieved for Xhosa using a data-driven approach [1]. Similar techniques have been used to build a Zulu voice and a South African English voice using HTS.

The HMM-based approach is so efficient in modelling speech that a relatively small set of recorded speech, 30 minutes or more, and a basic set of linguistic features are required for good quality synthesis. Typically, unit selection systems require several hours of recorded speech, specialised linguistic knowledge, language-dependent prosodic models and a significant amount of fine tuning. Still, even for the best unit selection systems, it is difficult to guarantee naturalness and intelligibility of synthetic speech in the unlimited domain.

However, the best examples of unit selection, being almost indistinguishable from real speech, will always sound better than the best examples of statistical parametric synthesis (SPS) [2]. This characteristic of unit selection is very attractive, especially in limited domain applications, where synthesis contingencies can be catered for adequately during voice construction. Outside of the specified domain the synthesis quality is likely to be compromised, both in terms of naturalness and intelligibility. Furthermore, the limited domain approach is not well suited to a number of important speech applications that require stable, predictable and intelligible synthesis across an arbitrary range of utterances.

The use of TTS in E-learning and second language acquisition applications requires stable synthesis of predictable quality in the unlimited domain. A user suffering from reading difficulties or dyslexia might use a screen reader to have learning material, or an arbitrary body of text, read aloud. A user wishing to employ TTS to learn a foreign language would need assurance of the accuracy of the synthetic prosody and pronunciation. Typical flaws related to unit selection systems, like those resulting from bad joins, would be greatly detrimental to the user experiences in these scenarios.

The language independent, corpus-based architecture of SPS systems makes TTS much more accessible to minority languages, offering a clear advantage over older, reliable techniques like articulatory and formant synthesis, which require a great deal of linguistic knowledge. Building an SPS voice requires a minimal amount of knowledge about the target language, from which all pronunciation and prosodic rules can be automatically derived.

The rest of this paper will detail our statistical parametric synthesis voice building system for the HTSEngine runtime. We then present a brief overview of the evaluation results. This is followed by a discussion, highlighting the system's shortcomings and proposing solutions to improve overall quality.

2. Voice Building

This section provides a step-by-step overview of the voice building procedure. To simplify the task of voice building, our system requires a minimal set of inputs. The UK English voices were built automatically from the Unisyn lexicon, the letter-to-sound rules trained from it, the text prompts and the audio files. For the Mandarin voice we extracted a lexicon from the provided labels, and made no attempt to cater for out-of-vocabulary words.

2.1. Vocoder Feature Extraction

For each audio file in the training set, 24th-order mel-cepstral parameters (MCP) and f_0 are extracted. We use the same mel-cepstral analysis technique provided by SPTK, as used in HTS [3]. While high quality speech can also be obtained from alternative parameterisations, e.g. line spectral pairs (LSP) [4], MCP was chosen because of the effective postfiltering provided. LSP-based voices might sound better if an appropriate postfilter is applied to the generated parameter sequence. LSP is particularly attractive for embedded solutions, requiring a lower analysis order and a lower complexity synthesis algorithm compared to the mel log spectrum approximation filter (MLSA).

Pitch contours were extracted using the script supplied by the HTS voice building environment. Due to the lack of a multi-space probability distribution HMM-based solution [5] for modelling discontinuous pitch contours, we interpolate the

pitch contour in unvoiced regions and create an additional voiced/unvoiced parameter. As the synthetic f_0 contours are quite flat using this technique, we employ a scalable root-based dynamic range compression technique, as our system is particularly sensitive to the significant range compression of the log-based procedure used by HTS.

2.2. Linguistic Feature Extraction

Linguistic features are extracted from the text prompts using our own text analyzer written in Python. This text processing frontend is particularly useful for rapid bootstrapping of new languages. It also makes experimenting with different contextual factor sets relatively easy, as appropriate question sets are automatically generated. Future investigation of the effects of certain contextual factors on synthesis quality, and the determination of an optimal set of factors, are of critical importance to SPS-based systems, as all rules governing the pronunciation and prosody of synthesis are automatically derived from the available factors. Determining an optimal subset of contextual factors is particularly important for embedded solutions in reducing overall complexity.

The modular design of the Python frontend is heavily influenced by Festival, and heterogeneous relation graphs are used to represent arbitrary linguistic information. The frontend is designed to require a minimal set of inputs, text prompts, a phoneset, a lexicon, and LTS rules. Closely following the design principles of Festival allows the text processing frontend to be interchangeable with Festival and Flite for runtime synthesis.

2.3. Training

The first phase of text processing outputs a sequence of words for each utterance and a reduced lexicon containing only the words in the dataset and all their possible pronunciations. Initial phoneme models are flatstarted to the global means and variances of the dataset. Several iterations of Baum-Welch embedded reestimation is done on the entire dataset from the word labels. The trainer is lexicon-based and creates a prototype word HMM with parallel branches to accommodate multiple pronunciations of the word in the lexicon. Skipable silence models are inserted between words and at the start and end of each utterance. This is the most time consuming of all voice building operations, taking up to 24 hours for the entire UK English dataset on a single core Pentium 4.

Once the training is complete, the prototype word models, now with fully trained phoneme models underneath, are used in a Viterbi forced alignment procedure. This procedure produces multi-level transcriptions, at the word, phoneme and senone level. Informal inspection of the alignments revealed them to be very accurate and consistent. A particularly promising aspect of this lexicon-based training procedure is the high accuracy heteronym disambiguation, a problem that is virtually impossible to solve using standard text processing techniques. It is unlikely that this feature contributes significantly to enhanced speech quality, as the HMM-based approach is quite robust to small errors in text analysis. In future, however, it might be used in conjunction with a meta-lexicon like Unisyn to provide accurate and robust dialect classification.

The silence insertion also performed quite well, although some problems were noted with plosive sounds. Words that start with a plosive sound would sometimes have a silence inserted before it, even though there is no audible or syntactic pause. While this did not prove too detrimental to synthesis quality, speech with compound plosive sounds did sound unusually rushed. Instead of using a threshold to avoid incorrect insertion of silences, we are working on a more robust solution. The idea is to train generic speech and silence models from the global means and variances. From these two generic models initial phoneme models are derived. The silence model is used for silences and the first state of plosive phonemes. The speech model is used for all the rest. While we did not employ this technique for Blizzard, we hope to integrate it into our voice building system in future.

2.4. Context Clustering

After transcribing the dataset, a second phase of text analysis is performed. The disambiguated pronunciations and inserted silences from the alignments are used directly, with the text analyzer adding a number of other linguistic features.

The set of linguistic features used differs significantly from that used by HTS. A minimal set of counting features are used. Counting features consist of counting up and down from the current position, and also the total number of elements. The counting features used are: the position and number of phonemes in the syllable, the position and number of syllables in the word, the position and number of words in the phrase, and the position and number of phrases in the utterance. The other factors used include syllable type and structure, parts-of-speech and word emphasis, phrase type and utterance style.

Due to the large number of unique permutations of the contextual factors, training HMMs from the full context labels is problematic. It is not feasible to create an HMM for each context and load them all into memory at the same time. Instead we train each utterance separately. An HMM is created for each unique label in the utterance by cloning the appropriate monophone model. A few iterations are performed on the utterance after which the sufficient statistics are dumped to a file. When all the utterances have been trained the statistics are loaded and merged for CART building.

We build the CART trees directly from the sufficient statistics, as opposed to mean and diagonal covariance vectors. This greatly reduces the computational complexity, by replacing multiplications with additions. The sum and squared-sum statistics and the occupation counts are simply accumulated for slices of the dataset. Determining the likelihood of a slice of the dataset is then simply a matter of calculating a covariance vector from the accumulated statistics in the slice, which, together with the occupation, the minimum description length gain is determined [6].

Due to memory constraints, it is necessary to train separate trees for the mel-cepstral parameters for larger databases like the full UK and Mandarin voices. The statistics are split according to HMM state position prior to training. After all state trees have been trained, their roots are merged. The f_0 and duration trees require considerably less memory, and are built all at once, using questions about state positions. For voice A seven trees were built, 5 spectral trees for each HMM state, an f_0 tree and a duration tree, all using the same question set. All trees were built sequentially on a single core Pentium 4 in less than 7 hours. The trees for voice B were built in less than 2 hours. The Mandarin voice also took about 6 hours. Memory usage never exceeded 1 GB.

Another factor possibly contributing to this performance improvement over HTS tree building is the pre-parsing of all full context labels prior to clustering. All separator tokens are stripped and the fields are arranged in a large matrix of contextual factors. The columns of the matrix correspond to specific

contextual factors, whereas each row represents a unique label. The questions operate directly on field positions. As literally thousands of questions are asked prior to each split, avoiding costly string comparisons and pattern matching can make a significant difference.

One of the major weaknesses of the current system is that f_0 trees are built from statistics derived from the alignments, which are produced using spectral parameters. Thus, HMM-modelling of f_0 is never performed. After the f_0 tree is built, the clusters' corresponding voiced/unvoiced parameters are gathered to determine whether each leaf is voiced or not. This procedure is much simpler than HTS's MSD modelling, but clearly quality is compromised. This is especially true for intonation and tonal languages, where not only naturalness and similarity suffer, but also intelligibility. The results of the Mandarin evaluation reflect this. If any single component of the current system can be highlighted for improvement, it is this one.

The duration tree is also trained directly from the alignments. The HMM-state occupancy counts are used as single dimensional statistics. Therefore, durations are modelled and predicted on a state level. This approach to duration modelling seems to produce very good and stable results.

To summarize, we outlined a fully automated voice building procedure requiring minimal inputs. Large corpus voices, like voice A and the Mandarin voice, are built in less than 35 hours on a single CPU, whereas voice B was built in less than 10 hours. By far the most time consuming part of the system is the embedded reestimation from flatstarted models. After the dataset is transcribed, the rest of the training is completed relatively quickly. The most significant improvement in overall speech quality can be obtained by implementing an explicit f_0 modelling routine, such as MSD for HTS [5]. Spectral fidelity can also be improved by untying initial spectral clusters, retraining full context models cloned from these clusters, then clustering again, and finally retraining the clustered models. This technique is used by HTS to good effect [7].

3. Synthesis

The accessibility of text-to-speech synthesis is largely dependent on the portability of the runtime synthesis engine. We have designed our system to build voices specifically for the HTSEngine platform, because it is small, fast and portable.

Speech parameters are generated from single mixture Gaussian distributions using maximum likelihood parameter generation [8]. Simple pulse-noise excitation signals are generated from f_0 values. The generated mel-cepstral parameters are postfiltered and used in the mel log spectral approximation filter to generate speech waveforms [3].

As HTSEngine does not perform text analysis, we used our own internal Python text processor. The Python frontend is ideal for rapid prototyping of text analyzers for new languages. For deployment on speech servers or mobile devices, a more efficient solution is required. Work is under way to develop a Flite module integrating HTSEngine. We hope that this work will bring HTS voices to a wider audience on multiple platforms. Initial experiments have shown that an unmodified HTSEngine, with a Flite frontend, can produce speech faster than realtime on high-end mobile devices. This requires the analysis order and sampling rates to be reduced, and the lower complexity LSP vocoder to be used.

The difficulty in achieving faster than realtime synthesis on mobile devices is caused by the lack of floating point calculation units. It is likely that better quality and performance can be achieved on mobile devices with a true fixed point conversion. Either way, HTS voices are already accessible in a wide array of languages, on multiple platforms.

4. Results

For the Blizzard Challenge we submitted all three voices, the full UK English voice, A, the Arctic subset voice, B, and the full Mandarin voice.

The results of the evaluation have, in general, shown our system to be of average intelligibility with a WER of 29%, the same as the overall WER. Our system scored below average in naturalness and similarity to original speaker. We are satisfied with this result, recognizing the importance of predictability and stability of synthetic speech. However, to be truly competitive with the very high quality unit selection systems, significant work needs to be done.

While our system generally produces speech with fairly natural prosody, which is relatively easy to understand, listeners did judge it rather harshly in the naturalness and similarity categories. This might be due to how these criteria were defined. Both naturalness and similarity tests were done with explicit reference to real speech samples. Given this formulation the listener would be correct to prefer synthetic utterances concatenated from unmodified waveform segments with relatively unnatural, discontinuous prosody, over smooth, stable speech exhibiting vocoding artifacts. Surely the naturalness criteria can be defined so that jarring jumps in prosody are judged inferior to vocoding artifacts. In the case of the similarity to original speaker spectral brightness and fidelity should play a more central role. Speech experts have a particularly low opinion of synthetic speech lacking the crispness expected of state-of-the-art unit selection synthesis.

Interestingly, the intelligibility of our Mandarin voice was much lower compared to the English voices. We attribute this to the relatively poor intonation modelling.

5. Discussion

In order to understand where significant improvements to our system can be made, it is important to clearly identify the ways in which original speaker characteristics are distorted. Three distinct types of distortion have been noted:

- *Buzz* - Originates unambiguously from the HTS vocoder generating excitation signals as pulse train for voiced and noise for unvoiced segments.
- *Drone* - Caused by a flattening of synthetic f_0 contours, reducing pitch variability resulting in speech with an increased monotonous quality.
- *Muffle* - Statistical averaging of vocoder features causes smearing of the frequency response resulting in muffled speech.

The synergistic effect of these distortions results in decidedly dehumanised speech, which may be perfectly intelligible, but fails to meet quality standards set by traditional unit selection.

We propose the following solutions which should, based on previous research, alleviate much of the quality issues of our current system:

- Buzziness can be effectively alleviated using an efficient mixed excitation scheme, such as those used in [4, 9, 10, 11]

- More varied and accurate f_0 contours can be synthesised if modelled explicitly using HMM, similar to MSD-HMM used by HTS [5].
- Spectral clarity can be improved using an LSP vocoder with an effective postfilter [4]. We could also perform more iterations of embedded reestimation after the first phase of context clustering [7].

We expect that these solutions would greatly improve the subjective opinion of our system, and may also improve intelligibility, especially for tonal and intonation languages.

We are hesitant to rely on newer algorithms like the parameter generation considering global variance [12] for quality improvement as this algorithm is too computationally complex for embedded devices. In this context, an effective and efficient postfiltering technique is preferred.

As SPS is still a relatively new field of research in TTS we plan to use our system for furthering research. A technique possibly worth considering is to lump the zeroth cepstral coefficient, or the log energy in case of LSP, together with f_0 and a maximum voiced frequency parameter [4]. It is possible that the loudness, pitch and degree of voicing are more closely correlated than loudness and spectrum. This technique could therefore provide a much better separation of intonation and articulation specific parameters. This should also reduce the dominance of loudness over spectral parameters during training and clustering.

6. Conclusion

Our first participation in the Blizzard Challenge has proved an invaluable learning experience. It has helped us to clearly identify those components which require the most urgent attention. We are very pleased that our system scored highly in the intelligibility tests, as we feel this is the strongest component of our system. However, it remains clear that to be truly competitive in the TTS field today, voice quality cannot be compromised. Listeners have a clear preference for clean, crisp sounding speech, and tend to forgive certain prosodic quirks much more readily than speech consistently degraded by vocoding artifacts.

7. Acknowledgments

We hereby express our gratitude towards the National Research Foundation for the sponsorship of this project (GUN 2074784) in terms of the Japan-South African Intergovernmental Science and Technology Cooperation programme.

We would also like to thank the HTS working group for their invaluable work on HMM-based speech synthesis, and for releasing their software under a BSD-style license. Without their work, this project would not have been possible.

8. References

[1] J. C. Roux and A. S. Visagie, "Data-driven approach to rapid prototyping Xhosa speech synthesis," *SSW6*, pp. 143–147, 2007.

[2] A.W. Black, H. Zen, and K. Tokuda, "Statistical Parametric Speech Synthesis," *Proc. of ICASSP*, pp. pp.1229–1232, 2007.

[3] K. Tokuda, H. Matsumura, T. Kobayashi, and S. Imai, "Speech coding based on adaptive mel-cepstral analysis," *Proc. ICASSP*, pp. pp. 197–200, 1994.

[4] S.-J. Kim and M. Hahn, "Two-Band Excitation for HMM-Based Speech Synthesis," *IEICE - Trans. Inf. Syst.*, vol. E90-D, no. 1, pp. 378–381, 2007.

[5] K. Tokuda, T. Masuko, N. Miyazaki, and T. Kobayashi, "Hidden Markov Models Based on Multi-Space Probability Distribution for Pitch Pattern Modeling," *Proc. ICASSP*, vol. vol. 1, pp. pp. 229–232, 1999.

[6] H. Zen, K. Tokuda, and T. Kitamura, "Decision Tree-based Simultaneous Clustering of Phonetic Contexts, Dimensions, and State Positions for Acoustic Modeling," *Proc. of Eurospeech*, pp. pp. 3189–3192, 2003.

[7] H. Zen, T. Nose, J. Yamagishi, S. Sako, T. Masuko, A.W. Black, and K. Tokuda, "The HMM-based Speech Synthesis System Version 2.0," *Proc. of ISCA SSW6*, pp. 294–299, 2007.

[8] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura, "Speech parameter generation algorithms for HMM-based speech synthesis," *Proc. of ICASSP*, vol. vol. 3, pp. pp. 1315–1318, 2000.

[9] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura, "Mixed Excitation for HMM-based Speech Synthesis," *Proc. of Eurospeech*, vol. vol. 3, pp. pp. 2263–2266, 2001.

[10] C. Hemptinne, "Master Thesis: Integration of the Harmonic plus Noise Model (HNM) into the Hidden Markov Model-Based Speech Synthesis System (HTS)," IDIAP, IDIAP-RR 69, 2006.

[11] R. Maia, T. Toda, H. Zen, Y. Nankaku, and K. Tokuda, "An excitation model for HMM-based speech synthesis based on residual modeling," *Proc. ISCA SSW6*, pp. 131–136, 2007.

[12] T. Toda and K. Tokuda, "Speech parameter generation algorithm considering global variance for HMM-based speech synthesis," *Proc. Interspeech*, pp. 2801–2804, 2005.