The Lessac Technologies Hybrid Concatenated System for Blizzard Challenge 2012

Reiner Wilhelms-Tricarico, Brian Mottershead, John Reichenbach, Gary Marple

Lessac Technologies, Inc., USA

{reiner.wilhelms, brian.mottershead, john.reichenbach, gary.marple} @lessactech.com

Abstract

Lessac Technologies has developed a technology for concatenated speech synthesis based on a novel approach for describing speech in which expressivity, voice quality, and speaking style are fundamental. The main aspect of our system is that instead of traditional phonetic symbols, we use a much more fine-grained and richer set of entities called Lessemes to describe speech and to label units, which allow a richer and more precise characterization of speech sounds. The front-end portion of our synthesizer translates plain input text into a sequence of these units by syntactic parsing and applying a set of rules developed from expertise. We use a Bayesian method to obtain a particular trainable mapping from linguistic and prosodic features encoded in the Lessemes to a trajectory in the acoustic parameter space. Unit selection consists of selecting the best candidate units from a data base to match them to the target trajectory, while minimizing discontinuities between them.

Index Terms: Speech Synthesis, Blizzard Challenge, Lesseme.

1. Introduction

This is our third entry to the Blizzard Challenge. For 2012 , the Blizzard Challenge voice corpus was based on a set of four Mark Twain audio book recordings from librivox.org (A Tramp Abroad; Life on the Mississippi; The Adventures of Tom Sawyer; and The Man That Corrupted Hadleyburg, and Other Stories), all read by John Greenman. As part of the Challenge, Toshiba Research Europe provided each Blizzard participant with automatically segmented and aligned data with a confidence factor for the alignment for each prompt. The original recording length in mp3 format for all four books combined runs to just over 50 hours.

For Blizzard Challenge 2011, Lessac had supplied the "Nancy" voice corpus to each participant as the basis of the challenge. For Blizzard Challenge 2012, we used similar voice building techniques to build the "Greenman" voice. Except for the approach we took to selecting which portions of the "Greenman" voice corpus to use in building our voice, we used our current standard approach to building a voice for a Lessac TTS, and we used our standard approach to synthesize the test data. The major advances between 2011 and 2012 have been further improvements in (1) pitch marking based on a neural network, and (2) our hierarchical mixtures of experts approach to predict desired acoustic parameters.

Section 2 describes the approach we took to selecting the portions of the voice corpus to use in our synthesizer. Section 3 provides a description of our text-to-speech system. Section 4 explains the Lessac process of building the "Greenman" voice. In the first half of Section 4, we explain the approach we used in developing the pre-cursor elements of building a synthesizer, such as prompts, pitch-marks, and phonetic labels. In the second half of Section 4, we outline how we used the data available in the "Greenman" voice corpus to build our complete Lessac text-to-speech synthesis system. Results from the listening test and related discussion can be found in Section 5.

2. Approach to Corpus Selection

As narrated, the four Mark Twain books took John Greenman more than 50 hours to read aloud; however, in our submitted voice, we used only 17.1 hours of speech, 5,152 prompts from "The Man that Corrupted Hadleyburg, and Other Stories", and 4,935 prompts from "A Tramp Abroad" All of these prompts were at the 100% prompt alignment level based on Toshiba's automated alignment.

Lessac has built a half dozen TTS voices from similar audio book data. We use a semi-automated tool we have developed to segment the voice recordings into sentences or prompts. The user listens to the voice recordings while simultaneously being presented with the text, on a sentence by sentence basis, and then repeatedly presses a key as he hears the end of each sentence. Since this cannot be done entirely accurately the first time through, this is followed with a second clean-up tool where the user is presented with a small amount of wave data (both graphically and audibly), and he can add, move or delete each sentence segmentation break. However, in order to get clean data, it requires listening to the equivalent of the entire recording twice. Since we did not want to invest more than 100 additional hours to get cleanly segmented data, we decided to use the Toshiba automated segmentation.

We analyzed the Toshiba segmentation by listening to a number of examples, and found that only the prompt segmentation with 100% confidence level was accurate enough to use, so we eliminated all portions of the provided voice corpus that did not have a 100% confidence level. We also eliminated prompts with foreign words.

We built a voice from the nearly 30 remaining hours of the voice corpus that Toshiba had segmented at a 100% confidence level using our standard automated voice-building approach, and submitted this voice as our Blizzard demonstration voice in 2011. However, we were concerned that this demonstration voice had a number of undesirable characteristics.

Upon further exploration, we discovered that "Life on the Mississippi" was recorded with something that sounds like a fan noise in the background. We used signal processing to eliminate this background noise. While we could perceptually eliminate the fan noise, concatenated joins between units drawn from this book joined to units drawn from other books were too often jarring. We also eliminated Tom Sawyer because of the extensive use of dialect, which was not very well handled by our dictionary; we did not want to invest the time to expand our dictionary to encompass all of the eccentric dialectical spellings used in Tom Sawyer.

3. Lessac Technologies Text-to-Speech System

Similar to other systems, Lessac Technologies text-to-speech system consists of two main components: the front-end, which takes plain text as input and outputs a sequence of graphic symbols, and the back-end, which takes the graphic symbols as input to produce synthesized speech as output. In what follows, we briefly discuss the properties that distinguish our system from others and, we believe, play an important role in producing expressive synthesized speech.

3.1 Use of Lessemes

Successful production of natural sounding synthesized speech requires developing a sufficiently accurate symbolic set of sound representations that can be derived from the input text, and that relate the input text to be pronounced with the corresponding synthesized speech utterances that are heard by the listener. Rather than adopting traditional symbolic representations, such as IPA, SAMPA, or ARPAbet, Lessac Technologies has derived an extended set of symbolic representations called Lessemes from the phonosensory symbol set for expressive speech as conceived by Arthur Lessac [1]. The Lesseme system for annotating text explicitly captures the musicality of speech, and from the start avoids the artificial separation of prosodic and linguistic features of speech.

In their basic form and meaning, Lessemes are symbolic representations that carry in their base form segmental information just like traditional symbolic representations. To be able to describe speech more accurately and to include in the symbol set information that is not carried by a typical phonetic symbol, each base Lesseme can be sub-typed into several more specific symbols which then represent phonetic information found in traditional phonetic symbols plus descriptors for co-articulation and suprasegmental information. Acoustic data demonstrate different properties of a set of Lessemes which are normally collapsed under one phonetic label in other systems [2].

For General American English, with the present Lesseme specification, there can be as many as 1,500 different Lessemes. Compared to other sets of representations which usually contain about 50 symbols, Lessemes allow more fine-grained distinction of sounds. Units of the same type share closely similar acoustic properties. By having supra-segmental information directly encoded in Lessemes, we believe our system can target available units for concatenation better than a system with a relatively impoverished intonation annotation scheme. This should be useful especially when trying to produce expressive speech from a very large database.

3.2. Front-end with extensive linguistic knowledge

The front-end which derives Lessemes from plain text input is a rules-based system. The rules are based on expert linguistic knowledge from a wide variety of fields including phonetics, phonology, morphology, syntax, light semantics, and discourse. Simplistically, the Lessac front-end labels text, building from, at the lowest level, letters, spaces and punctuation marks. These letters, spaces and punctuation marks are interpreted by the front-end, and assembled as syllables, words, phrases, sentences, and paragraphs to be spoken, along with context-aware labeling for appropriate co-articulations, intonation, inflection, and prosodic breaks.

First, the input text is processed by a syntactic parser which generates the most likely syntactic tree for each sentence, and tags words with part-of-speech (POS) information. In the next step, words are transcribed by use of a pronunciation dictionary into base Lessemes accompanied by lexical stress. Homograph disambiguation based on POS tags takes place at this step. Subsequent processing steps modify the base Lessemes by making successive decisions based on the overall phrase and sentence structure. In particular, prosodic breaks are inserted in meaningful places by taking into consideration factors such as punctuation, phrase length, syntactic constituency, and balance. In most phrases, an operative word is marked which carries the highest pitch prominence within the phrase. In addition, Lessemes are assigned inflection profiles and one of two degrees of emphasis. Context-based co-articulations across word boundaries are also captured. The result is a full Lesseme for each sound which encodes expressive intonational content in addition to the segmental information found in traditional phonetic symbols.

The front-end process is able to develop a complete Lesseme label stream with plain normally punctuated text as the sole input. This Lesseme stream is delivered to the signal processing back-end.

3.3. Voice database construction

In addition to the machine readable form used as the input to the signal processing back-end, Lessemes are also used in creating new voices, namely to automatically generate a human readable graphic output stream which can be thought of as annotated text plus a musical score, as illustrated in figure 1.

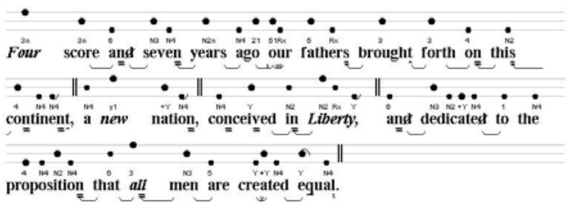
Four score and seven years ago our fathers brought forth on this
continent, a new nation, conceived in Liberty, and dedicated to the
proposition that all men are created equal.

Figure 1: *Lessac Technologies annotated text*

In the annotation, vowel orthographic forms are designated with Arthur Lessac's phonosensory symbols. Consonant orthographic forms are marked with information indicating whether the consonant is sustainable (double underlined) or percussive, i.e. pronounced with a brief contact within the mouth (single underlined), as well as how the consonant is linked to the next sound in connected speech. The musical score on top of the orthographic forms depicts notes which represent the intonation pattern that a person with sufficient voice training can follow. Each syllable corresponds to a note. Higher notes are pronounced with higher pitch. Large notes define stressed syllables while small notes refer to unstressed syllables. Some notes are further specified with an inflection, which reflects a particular shape of pitch movement within the syllable.

During the voice database construction, the text to-be-recorded is first processed by the front-end, yielding the stream of Lessemes. In building a full Lessac voice, the resulting stream is then transformed into a human readable form, as seen in figure 1, which we use as the combined script and score for the trained voice talent during the recordings. The way the voice talent records the prompts is controlled by the annotated text and musical score. The recordings of the prompts are then segmented and labeled with the same Lessemes that underlie the script and score that the voice talent followed. The fact that the same Lessemes are output for the voice talent script as well as the labeling of the database creates a direct link between each speech snippet and its Lesseme label, thus a high degree of correspondence between the symbols and the sounds as actually recorded by the voice talent.

However, for Lessac TTS voices constructed from audio book data, such as the "Greenman" voice, such a high degree of symbol-to-sound correspondence is not guaranteed. Using our current voice-building techniques, unless the voice building process includes a labor intensive manual notation process, the symbol-to-sound correspondence reflects only the expert knowledge contained in our front-end. Our front-end prosody represents an idealized "reportorial" prosody, which although relatively accurate for most speakers, is only one of many speaking styles that a voice actor could use to read the text.

We make use of this correspondence in the unit selection process by evaluating units in the data base according to the context dependent linguistic and prosodic features, in order to preselect a subset of unit candidates, which are then evaluated by the model described in the following.

3.4. Hierarchical Mixture of Experts for mapping linguistic features to acoustic parameters

To enhance methods for target cost calculation and unit selection, we apply the Hierarchical Mixture of Experts (HME) model [3] [4] to learn the parameters of a statistical model of the relationship between the Lesseme representation of the input text and the ideal acoustic observables in the recordings.

A functional diagram of the HME model is shown in figure 2.

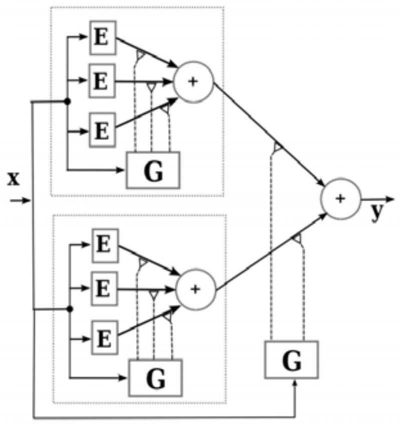

Figure 2: *Hierarchical Mixture of Experts model. (E: experts, G: gates, x: input, y: output)*

The HME model applied to the problem of mapping prosodic features to acoustic observables makes use of the interpretation of the model as a parameterized mixture of Gaussians. Each expert in the model represents one multi-dimensional normal distribution with a variable expectation vector that depends on the input x. The parameters for each expert also include a full covariance matrix that is estimated and updated during the training. Each block of experts in a group or clique (Figure 2 shows 3 experts in each of 2 cliques) together with a gating network represent one mixture of Gaussians whereby the mixture coefficients are computed in the gates as a function of the input. Multiple groups of experts can be combined by another gate in a similar way. The complete network represents a mixture of Gaussians whose parameters are trained from pairs of known input and output. During the learning process, the parameters in the experts and gates are adjusted so that, for a given known input x, the probability of obtaining the desired known output y is maximized over all available data.

In our application of the HME model, the input x includes the linguistic and prosodic features and the output y are acoustic observables, which include MFCC's, F0, duration, and intensity, mostly the same type of parameters used in database segmentation, see 4.3 below. The model is applied and trained as a recurrent system, which means that the predictions of acoustic observables, y[n], for one sound at time index n are included in the input x[n+1] for the prediction of the next y[n+1].

We use supervised learning with the HME model to map linguistic feature sequences to a trajectory in the acoustic parameter space, which is represented by via points and for some of the parameters their velocity or rate of change. The structure of the model is shown in figure 3. The system steps through a sequence of Lessemes and predicts for each Lesseme the vector of acoustic parameters that specify the unit, whereby the input to the model consists of the feature information of the previous, the current and the next two Lessemes. Further, by feeding back the previously predicted acoustic parameter vectors as input to the model, the model becomes partially auto-regressive. This facilitates the learning task because the model only has to learn to predict the current acoustic vector conditioned on the last two acoustic vectors and the input linguistic features. Learning proceeds in two phases. Initially, the looped-back input to the model consists of the actual acoustic vectors until the model begins to converge. Then, training is continued by having the predictions for the last two time slots become inputs for the prediction of the current time slot. Learning then proceeds by repeatedly processing a large number of sentences in the database, until the error variance cannot be lowered further.

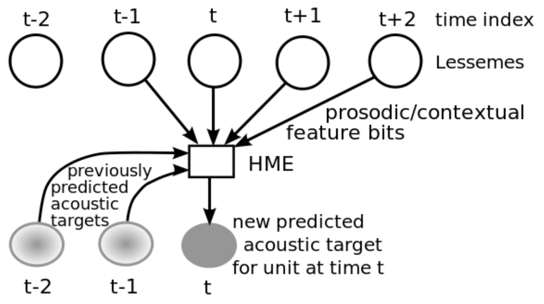
t-2
t-1
t
t+1
t+2
time index
Lessemes
prosodic/contextual feature bits
HME
previously predicted acoustic targets
new predicted acoustic target for unit at time t
t-2
t-1
t

Figure 3: *Recurrent and partially auto-regressive prediction of intonation contour and other acoustic targets by HME*

During the target cost calculation process, we compute the cost as the distance of the acoustic parameters of a candidate unit from the ideal trajectory, which is in turn directly predicted from the linguistic feature variables. This distance measure makes use of the predicted mixture covariance matrix which is obtained by combining the experts' covariances according to the gating weights, (see Figure 2). To reduce processing time, we reduce the number of candidates first by applying a rapid search with binary patterns generated from some of the features, and then compute the exact target cost for a smaller subset of close candidates. Since the HME provides the parameters of a probability density in the acoustic parameter space, we compute for the remaining candidates their probability under this distribution and use as target cost a penalty that is proportional to the negative logarithm of the candidates' probability.

Using the Lesseme representation of speech sounds, the output of the front-end results in a large number of features, which is augmented further by bundling neighboring features as shown in figure 3. The HME model overcomes the sparsity problem in the acoustic database by mapping the Lesseme features and context onto the acoustic parameter space as a target trajectory. At the same time it automatically provides a variable metric near the target trajectory, against which the candidates in the data base are matched during unit-selection.

4. Building the 'Greenman' Voice

Unlike in Blizzard Challenge 2011, where all of the other Blizzard Challenge entrants were using a voice corpus supplied by Lessac, for 2012, we had to build a new voice, and did not have the advantage of being able to use one of our already existing TTS voices. The following briefly describes the steps that were taken to create the "Greenman" voice.

4.1. Transcription to Lessemes

Lessac Technologies has developed an automated rules based method more fully described in our Blizzard Challenge Workshop 2011 paper that allows us to automatically generate the Lesseme sequence from arbitrary text. This provides the input information for the synthesizer's back-end.

4.2. Pitch-Marking

We have observed that minimal pitch mismatches can cause noticeable synthesis artifacts. Based on this empirical experience, we feel that highly accurate pitch-marking is critical to high quality synthesis. Since we do not have the manpower to manually review and adjust pitch-marks, we put a significant portion of our technical effort over the past year into better automated pitch-marking.

Synthesis artifacts can often be minimized by slightly adjusting pitch around the join point, such that the mean pitch at the ends of the units to be joined is the same. However, herein is a dilemma: If the specific pitch marks are not accurate, then we can assume the adjustment of pitch will also be inaccurate, and there will be a noticeable synthesis glitch at the join point..

In our speech synthesizer, we make use of pitch marks that are usually specified as the first zero crossing from negative to positive signal amplitude after the point of excitation of the vocal tract.

For pitch marking we formerly used the software tool Praat on a standalone basis, but found it (for this voice and others) fairly inaccurate for certain sound classes. In particular, for the low F0 portions of the "Greenam" voice, Praat often specified too many pitch marks.

Basic approach to pitch marking

Using Burg's method, we compute 17 Parcor coefficients from the speech signal which is sampled at 16 kHz for each interval of 320 samples (20ms) weighted by a Hanning window, and generate the error signal in an overlap-add method proceeding 1/2 of the window length at each step. The residual or error signal shows peaks of energy near the points of glottal excitation, which are more readily picked out by instead using the Hilbert envelope of the signal. The Hilbert transform provides the complex valued analytic signal, whereby the real part is the signal itself, and the imaginary part the Hilbert transformed signal. The Hilbert envelope is the absolute value of the analytic signal, which is strictly positive and therefore simplifies identifying signal peaks. Pitchmarks are then found by searching in the speech signal for zero crossings with positive slope near the maximal peaks. This method usually works well for all fully phonated vowel sounds, but is not completely reliable for nasal sounds and the lateral L sound. The "Greenman" voice used for this challenge has some irregular phonations and occasionally has very low F0, which could easily be taken as double and triple excitation of the vocal tract, leading to incorrectly determined additional pitch markers.

Group delay time improves robustness

The problem of finding and eliminating the additional superfluous pitchmarks in each cycle can be overcome if the duration of the glottal cycle is known with greater confidence. A somewhat more reliable estimate of the glottal cycles can be obtained by using in addition to the magnitude of the error function, the local group delay function, computed for each sample from the error signal. This was proposed in by P. S. Murthy & B. Yegnanarayana [5]. The authors proposed a spectral method to compute the short term average group delay time from the error signal, which is computationally rather expensive. In our adaptation of this idea, we compute a similar function, starting from the residual signal. It is motivated by the fact that the differentiation of the Laplace transform of the signal with respect to instantaneous frequency is equivalent in the time domain to a multiplication of the signal by the (negative) time parameter. A simplification is obtained by taking into account that the amplitude spectrum of the residual signal is approximately flat, allowing one to avoid the spectral normalization step in the original article. In our procedure we take a short interval of duration T (usually 10ms) around the current sampling point and translate the center to the time t=0. The signal portion e(t) over this interval from -T/2 to +T/2 is then multiplied by -t, with $-T/2 < t < T/2$, and summed over the interval from -T/2 to T/2. The implementation is done by a fast convolution method with overlap add, which is faster than the calculation in the time domain. The resulting (quasi-) group delay time function, in the following denoted g(t), usually shows an almost periodic behavior, and readily exposes the periodicity of the original speech signal.

A further improvement of the signal g(t) is obtained by performing an amplitude normalization in the time domain, explained for the example portrayed in Fig. 4.

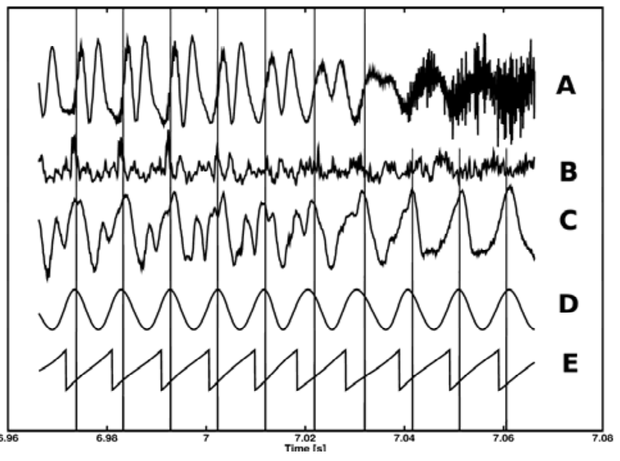

***Figure 4:** Pitchmarks derived with Group Time Delay*

The signal labeled A in figure 4 is a small portion of speech signal. In this example it is a 100 ms portion of speech near the end of the word "serves", showing the transition into the voiced "s" at the end of the word. The signal labeled B is the magnitude (Hilbert envelope) of the error signal. The error signal (not shown) was used to compute the group delay time function g(t), labeled C in the figure. The signal labeled D is obtained by lowpass filtering of g(t) and dividing by its Hilbert envelope. Taking again the Hilbert transform of the signal shown as D, the phase function E is obtained. It is portrayed here as a saw-tooth function by taking its modulo 2π, but is actually computed as an unrolled phase, which also allows us to detect phase jumps that sometimes occur in rapid transitions of the speech signal. We thus have a clean definition of the instantaneous pitch period duration. To obtain pitch marks, within each interval of phase change by

2π, we look for the maximal point of excitation in the magnitude of the residual (B) for the nearest zero crossing with positive slope.

Improve pitch marking with neural network

The method described above is a slight improvement over using Praat, but to make it 100 percent reliable takes a lot of additional tweaking.

For a given voice it is usually possible to find problems by inspection and build special code that reduces the errors, for example by inventing rules governing the selection among ambivalent pitch markers. Usually a lot of additional threshold parameters need to be tweaked to get satisfactory results.

An improvement can be made by making use of a neural network that learns constraints for generating pitch marks, and thus learns to predict the rough position of pitch marks directly from the speech signal.

Usually, the mistakes made by a conventional pitch marking algorithm, including our own, reveal themselves only by close inspection of a large number of samples. However, with a sufficiently accurate pitch marking algorithm it is usually possible to find by direct inspection of the results a large set of speech samples for which pitch marking is error-free.

We then train a neural network based on only these examples. The neural network we are using is a standard multi-layer perceptron with two hidden layers and one input layer. It makes use of hyperbolic tangent output non-linearities. The neural network receives as input several consecutive samples from the speech signal, several samples from the error signal and its magnitude, and in the case of the feed forward network, the Parcor coefficients used for computing the residual signal. The technique is explained by an example shown in Figure 5.

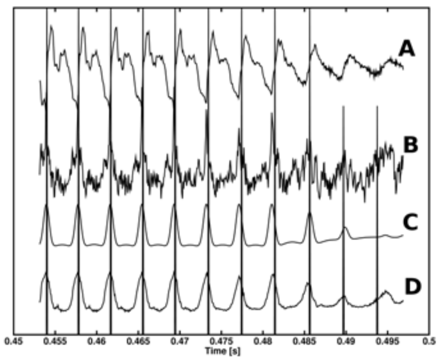

Figure 5: *Use of a Neural Network for pitchmarks*

In this figure 5, signal A is a short portion of speech signal, and signal B is the magnitude of the error function (Hilbert envelope). The pitch marks, which are already given here, are shown as vertical lines through the plot. During network training, a signal exemplified as C in the figure is computed as follows: For each pitch mark in the training set, the closest maximum of the error magnitude function is searched in a small interval of 2.5 ms before the pitch mark. At this position, which usually coincides with the actual local maximum of the magnitude signal, a bell shaped signal is generated and added to an array that is initialized with all zeros. The resulting signal C provides the learning target for the neural network. The neural network is trained to approximate the signal C, using the difference between its output and the learning target as a learning signal in a standard back propagation algorithm. After training the network output (signal D in Figure 5) can be treated as a spike signal not unlike the original residual magnitude signal, and pitch marking is then done in the usual way by finding the nearest zero crossing points. For a large data set, we set aside a manageable training set (typically 200 to 500 short prompts), for which pitch marks already exist by other means or from the methods described earlier, but excluding all examples with faulty pitch marks. The network is then trained on the selected training set. After training and after it has been verified that the neural network accurately reproduces the pitch marks from the training set, the neural network is used to pitch mark the remaining prompts.

4.3. Database creation

As our labeling and metrics for prosodic structure are different from methods commonly used; we modified the Festival feature functions to produce relevant linguistic features at segment, syllable, word, and phrase levels based on the Lessemes and prosodic breaks that the front-end provides as output. The end time of each unit came from the label files produced by automatic segmentation. Our segmentation procedures are based on a slightly modified version of the EHMM software that is part of speech tools and Festvox. Acoustic parameters were computed for each prompt, and a dimensionality reduction was obtained by principal component analysis. The resulting set of parameters were then used in building the HMM model for segmentation. For building the catalogue, all the linguistic features coming from the front-end analysis and the acoustic parameters were collected into a binary catalog file, which was then used to train the HME model off-line. The same binary catalogue is called by the synthesizer during run-time.

4.4. Synthesizer

While Lessemes help narrow the pool of candidates for unit selection and enable more precise targeting, labeling units with Lessemes can lead to the problem of non-existing or a sparse number of units of particular labels in the database, especially in a small database. We handle this problem by incorporating a set of fail-over rules. Whenever the target Lesseme has a very limited number of or even no matching candidates in the database, the fail-over rules look for closely matched Lessemes, e.g., those with a different inflection or pitch level, to include among the candidates for the target and join cost calculations. The target cost is computed as a weighted distance to the acoustic target trajectory that is generated by the HME model. The target penalty cost is derived from a logarithmic probability that can be computed for each candidate using the parameters provided by the HME model, namely target acoustic feature vector and covariances.

Similar to Kominek [6], our join cost calculation discourages joins between sonorant sounds. The join penalty varies depending on the types of sonorants being joined. For example, the join between two vowels gets a higher penalty than the join between a vowel and an onset lateral sound. We also make use of the HME output, namely the variance information, to modulate the spectral weights used in the join cost computation.

4.4. Concatenation

After the best units are selected, they are concatenated together in a process that works entirely in the time-domain. For this we no longer use Festival but instead built a separate module which receives only the information about the units to be concatenated. The concatenation of voiced sounds is done pitch synchronously, and some mutual adjustments of two sounds that are concatenated are made to increase the coherence and to reduce clicks and warbles. F0 modifications and duration modifications are also done independently of Festival in the concatenation module, using information that is transmitted to the concatenation module from the HME model.

5. Results and Discussion

Ten systems participated in the Blizzard Challenge 2012. In addition, the original speaker's voice was evaluated as a benchmark, or pseudo eleventh system (system A). System B was a benchmark Festival unit selection systems built by CSTR. The Lessac Hybrid Concatenation unit selection system was system F. During the online evaluation of the task, listeners were asked: (i) to judge how similar a system was to the original speaker; and (ii) to provide mean opinion scores (MOS) representing how natural or unnatural the utterances sounded. This task was done on different scales for both single sentences (0-5 scale as done in previous Blizzard Challenges), and short paragraphs (0-60 scale, new for Blizzard Challenge 2012), In addition each listener was asked to listen to synthesized semantically unpredictable sentences (SUS) and transcribe what they heard. The listeners included paid participants, volunteers, speech experts, plus native and non-native English speakers. Results for our system in comparison with natural recorded human speech, the benchmark Festival unit-selection system and others systems are presented below.

5.1. Naturalness and similarity to original speaker - sentences

A 5-point mean opinion scale (MOS) was used to evaluate both how natural synthesized speech sounds, and how similar synthesized speech sounds to the original voice when the synthesizer was reading aloud single sentences. This is the same methodology as used in previous Blizzard Challenges. With respect to naturalness of our synthesized speech, Lessac (system F) received a mean MOS score of 3.4 for all data and a median of 4. For similarity to the original speaker, we received a mean score of 3.4 and median of 3. Overall we were ranked in second place, based on pair-wise Wilcoxon signed rank tests. One system (system C) ranked higher on a statistically significant basis than our system and all others.

5.1. Naturalness and similarity to original speaker - paragraphs

A 60-point mean opinion scale (MOS) was used to evaluate both how natural synthesized speech sounds, and how similar synthesized speech sounds to the original voice. With respect to naturalness of our synthesized speech, Lessac received a mean and median MOS score of 32 for all data. Overall we were ranked in second place, based on pair-wise Wilcoxon signed rank tests. One system (system C) ranked higher on a statistically significant basis than our system and all others.

5.3. Word error rates

For the semantically unpredictable sentences (SUS) we received a median word error rate of 17% and a mean rate of 27%. The Wilcoxon signed rank test resulted in little information that would give a significant rank ordering of the different systems. Based on the Wilcoxon signed rank test, our word error rate is worse than natural recorded speech, and comparable to the other systems (worse than one, and better than three to a statistically significant degree). In other words, for nonsense sentences our system has similar word recognition rates as most other systems.

6. Conclusions

We have made good progress in producing near natural sounding synthesized human speech highly similar to the original speaker. We attribute much of this progress to our recent improvements in pitchmarking, and further enhancements to our HME approach.

We are also pleased that the listening results show that our system rendered short paragraphs perceptually better than single sentences. We believe this would likely be even more so the case for longer paragraphs, and book length texts.

We are also pleased that we were able to maintain our high ranking in this year's challenge; we have shown that the Lessac approach is fairly flexible. Blizzard 2011 was contested with the "Nancy" voice, which was supplied by Lessac. The Nancy voice corpus was carefully recorded using Lessac methods. The "Greenman" voice was not recorded using Lessac methods, yet we retained our high placement in Blizzard Challenge 2012.

The overall performance of our system as one of the best in the Blizzard Challenge (2nd, closely followed by another system) gives us some confidence in support of our general strategy to try to represent and capture in the synthesis model idiosyncratic properties of the original voice that are not directly represented by known explicit models. For the symbolic representation of speech sounds for synthesis we use a novel method that is a departure from traditional phonetics by introducing Lessemes, which carry both segmental and suprasegmental information and allow for much more fine grained tagging of speech. This tagging process is done fully automatically, starting from plain text. The processing done by the front-end results in a very rich stream of features that are encoded with the speech samples in the acoustic database. We use methods of machine learning to create a sufficiently comprehensive model of the voice without having to make too many assumptions about the nature of the relationship between acoustic parameters and perceived prosody.

Our hope is to demonstrate that since all of our voice building processes are fully automatic, and we do not rely on any manual pitch-marking, segmentation or labeling processes, Lessac techniques can be used to build multiple near natural human sounding synthetic voices quickly.

Participating in the Blizzard Challenge has proven very helpful for us in guiding further improvements of our technologies.

7. References

[1] Lessac, A., The Use and Training of the Human Voice: A Bio-Dynamic Approach to Vocal Life, McGraw-Hill, 1996.

[2] Nitisaroj, R. and Marple, G. A., "Use of Lessemes in text-to-speech synthesis", in M. Munro, S. Turner, A. Munro, and K. Campbell [Eds], Collective Writings on the Lessac Voice and Body Work: A Festschrift, Llumina Press, 2010.

[3] Jordan, M. I. and Jacobs, R. A., "Hierarchical Mixtures of Experts and the EM Algorithm", Neural Computation, 6:181-214, 1994.

[4] Ma, J., Xu, L. and Jordan, M. I., "Asymptotic Convergence Rate of the EM Algorithm for Gaussian Mixtures", Neural Computation, 12:2881-2900, 2000.

[5] P. Satyanarayana Murthy and B. Yegnanarayana. Robustness of group-delay-based method for extraction of significant instants of excitation from speech signals. Speech and Audio Processing, IEEE Transactions on, 7(6): 609–619, 1999.

[6] Kominek, J., Bennett, C., Langner, B. and Toth, A., "The Blizzard Callenge 2005 CMU Entry: A Method for Improving Speech Synthesis Systems", Proceedings of Interspeech 2005, 85-88.