Godambe *et al. EURASIP Journal on Audio, Speech, and Music Processing* (2016) 2016:6
DOI 10.1186/s13636-016-0084-y

EURASIP Journal on Audio, Speech, and Music Processing
a SpringerOpen Journal

RESEARCH Open Access

Developing a unit selection voice given audio without corresponding text

Tejas Godambe[1], Sai Krishna Rallabandi[1], Suryakanth V. Gangashetty[1], Ashraf Alkhairy[2] and Afshan Jafri[3*]

Abstract

Today, a large amount of audio data is available on the web in the form of audiobooks, podcasts, video lectures, video blogs, news bulletins, etc. In addition, we can effortlessly record and store audio data such as a read, lecture, or impromptu speech on handheld devices. These data are rich in prosody and provide a plethora of voices to choose from, and their availability can significantly reduce the overhead of data preparation and help rapid building of synthetic voices. But, a few problems are associated with readily using this data such as (1) these audio files are generally long, and audio-transcription alignment is memory intensive; (2) precise corresponding transcriptions are unavailable, (3) many times, no transcriptions are available at all; (4) the audio may contain dis-fluencies and non-speech noises, since they are not specifically recorded for building synthetic voices; and (5) if we obtain automatic transcripts, they will not be error free. Earlier works on long audio alignment addressing the first and second issue generally preferred reasonable transcripts and mainly focused on (1) less manual intervention, (2) mispronunciation detection, and (3) segmentation error recovery. In this work, we use a large vocabulary public domain automatic speech recognition (ASR) system to obtain transcripts, followed by confidence measure-based data pruning which together address the five issues with the found data and also ensure the above three points. For proof of concept, we build voices in the English language using an audiobook (read speech) in a female voice from LibriVox and a lecture (spontaneous speech) in a male voice from Coursera, using both reference and hypotheses transcriptions, and evaluate them in terms of intelligibility and naturalness with the help of a perceptual listening test on the Blizzard 2013 corpus.

Keywords: Unit selection synthesis, Found data, Unsupervised TTS, LibriSpeech, Phone recognition, Confidence measures, Data pruning

1 Introduction

1.1 Motivation

Unit selection speech synthesis is one of the techniques for synthesizing speech, where appropriate units from a database of natural speech are selected and concatenated [1–3]. Unit selection synthesis can produce natural-sounding and expressive speech output given a large amount of data containing various prosodic and spectral characteristics. As a result, it is used in several commercial text-to-speech (TTS) applications today.

1.1.1 Overhead of data preparation for building general-purpose synthetic voices

Building a new general-purpose (non-limited domain) unit selection voice in a new language from scratch includes a huge overhead of data preparation, which includes preparing phonetically balanced sentences, recording them from a professional speaker in various speaking styles and emotions in a noise-free environment, and manually segmenting or correcting the automatic segmentation errors. All of it is time consuming, laborious, and expensive, and it restricts rapid building of synthetic voices. A free database such as CMU ARCTIC [4] has largely helped to rapidly build synthetic voices in the English language. But CMU ARCTIC is a small database, contains only a few speakers data, and is not prosodically rich (contains short declarative utterances only). Today, (1) a large amount of audio data has become available on the web in the form of audiobooks, podcasts, video lectures, video blogs, news bulletins, etc, and (2) thanks to technology, we can effortlessly record and store large amounts of high-quality single speaker audio such as lecture, impromptu, or read speech. Unlike

*Correspondence: AfshanJafri@gmail.com
[3]King Saud University, Riyadh, Saudi Arabia
Full list of author information is available at the end of the article

CMU ARCTIC, these data are rich in prosody and provide a plethora of voices to choose from, and their use can significantly ease the overhead of data preparation thus allowing to rapidly build general-purpose natural-sounding synthetic voices.

1.1.2 Problems with using found data for building synthetic voices

Now, the questions to be asked are whether we can readily use such data to build expressive unit selection synthetic voices [5] and will the synthesis be good? In this paper, we try to answer these questions. There are a few problems related to it such as the following: (1) the audio files are generally long and audio-text alignment becomes memory intensive; (2) precise corresponding transcriptions are unavailable; (3) often, no transcriptions are available, and manually transcribing the data from scratch or even correcting the imprecise transcriptions is laborious, time consuming, and expensive; (4) the audio may contain bad acoustic (poorly articulated, dis-fluent, unintelligible, inaudible, clipped, noisy) regions as the audio is not particularly recorded for building TTS systems; and (5) if we obtain automatic transcripts using a speech recognition system, the transcripts will not be error free.

1.1.3 Previous works on long audio alignment

Earlier works have addressed the abovementioned first and second issues of long audio alignment in the following three ways: (1) audio-to-audio alignment, (2) acoustic model-to-audio alignment, and (3) text to text alignment. Each method has its advantages and limitations.

1. *Audio-to-audio alignment*: Here, the text is converted to speech using a TTS system, and the synthesized speech is aligned with the audio [6, 7]. This method requires the existence of a TTS system.
2. *Acoustic model-to-audio alignment*: Here, acoustic models are aligned with the audio. In [8], a modified Viterbi algorithm to segment monologues was used. Their method assumed a good (at least 99 %) correspondence between speech and text, required manual intervention to insert text at the beginnings and endings of monologues, did not handle mispronunciations, and propagated an error in one segment to subsequent segments. In [9], a Java-based GUI to align speech and text was released. They also used acoustic models, assumed good correspondence between audio and text, and required manual intervention.
3. *Text-to-text alignment*: Here, a full-fledged automatic speech recognition (ASR) system including an acoustic and language model is used. Basically, long files are chunked into smaller segments based on silence. Hypothesis transcriptions are obtained for these smaller segments. In [10], they proposed a method where a search is made to see where the sequence of words in the reference and hypothesis transcriptions match. The stretch where they match is aligned with the audio using the Viterbi algorithm. This process is repeated until a forced alignment is done for each audio chunk. The process is practically difficult to implement and relies on correctness of the reference and hypothesis transcriptions. In [11], a finite state transducer-based language model instead of N-grams was used. In [12], they used a phone-level acoustic decoder without any phonotactic or language model and then found the best match within the phonetic transcripts. This approach was inspired by the fact that the data to be aligned could have a mixture of languages. But phonetic alignment is less robust than that at the word level. In [13], they quantified the number of insertions, substitutions, and deletions made by the volunteer who read the book "A Tramp Abroad" by Mark Twain and proposed a lightly supervised approach that accounts for these differences between the audio and text. Their method, unlike the forced-alignment approach in [11] which uses beam pruning to identify erroneous matches, could find also the correct sequence and not only the best match in terms of the state sequence between the text and audio chunk. In [14], a dynamic alignment method to align speech at the sentence level in the presence of imperfect text data was proposed. The drawback of this method is that it cannot handle phrase reordering within the transcripts.

The above works on long audio alignment addressing the first and second problems with found data generally prefer reasonable transcripts and mainly focus on (1) less manual intervention, (2) mispronunciation detection, and (3) segmentation error recovery. In this work, we used a large vocabulary public domain ASR system to obtain transcripts, followed by confidence measure-based data pruning which together address the five issues with the found data and also ensure the above three points. We used posterior probability obtained from the ASR system and unit duration as confidence measures for data pruning. Posterior probability helps detecting mislabeled and bad acoustic regions while unit durational measure helps detect unnaturally short or long units which may have high posterior probability values but they can make words unintelligible or sound hyper-articulated, respectively. Thus, both these confidence features are directly related to the intelligibility and naturalness of speech. For proof of concept, we built voices in English language using an audiobook (read speech) in a female voice from LibriVox and a lecture (spontaneous speech) in a male voice from Coursera, using both the reference and hypotheses transcriptions, and evaluate them in terms of intelligibility and naturalness with the help of a perceptual listening test on the Blizzard 2013 corpus.

1.2 System overview

Figure 1 shows the architecture of the entire system. Each module in Fig. 1 is explained in detail in the succeeding sections. First, the ASR system accepts the audio data and produces corresponding labels. Then, data pruning using confidence measures takes place. The pruned audio and label data form the unit inventory for the TTS system. During synthesis time, the TTS system accepts normalized text, takes into account the duration and phrase break information predicted by a statistical parametric speech synthesizer trained using the same audio data and hypothesized transcriptions, and chooses an appropriate sequence of units that minimizes the total of the target and concatenation costs. The output of the TTS system is an audio file.

1.3 Experiments and evaluation

In the first experiment, we compare the recognition performance of an ASR system trained on LibriSpeech data against those trained on TTS data (Olive and lecture speech). In the second experiment, we check the effectiveness of posterior probability as a confidence measure to prune bad data. In the third experiment, we check the effect of pruning using a combination of posterior probability and unit duration on intelligibility and naturalness of the synthesized voice.

The rest of the paper is organized as follows. Section 2 describes data preparation. Section 3 gives ASR and TTS system development details. Section 4 discusses data pruning using confidence measures. Section 5 explains the experiments and evaluation. We conclude the paper in the Section 6.

2 Data preparation

2.1 Data used for building the ASR system

In this paper, we built three ASR systems, one of which was built using LibriSpeech data [15]. LibriSpeech is a fairly recently made available continuous speech corpus in English language, which is prepared by collating parts of several audiobooks available at the LibriVox website. It contains two parts: 460 h of clean speech and 500 h of speech data containing artificially added noise. We used 460 h of clean speech[1] to build acoustic models, a 3-gram language model[2] pruned with a threshold of 3×10^{-7} to generate the lattices, and a higher order 4-gram language model[3] to rescore the lattices and find the 1-best Viterbi path for the ASR system.

The other two ASR systems (both the acoustic and language models) were built using the TTS data described in the next subsection.

2.2 Data preparation for building the TTS system

Details of the audio used for building voices are given in Tables 1 and 2. One is an audiobook (read speech) in a female voice downloaded from LibriVox, and the other is a lecture (spontaneous speech) in a male voice downloaded from Coursera. The audio files were converted to 16-kHz WAV format and power normalized. Before downloading the audio, we checked that the voice quality and speech intelligibility of the speakers are good and that the audio has not been recorded in a noisy background. For the audiobook, we also made sure that it is not a part of the 460-h clean speech LibriSpeech corpus which was used for training the ASR system so that we can simulate the situation that the found audiobook data is unseen by the ASR system. For the lecture speech, a few lectures contained TED talks and voices from other speakers, both of which were removed. The audiobook and lecture audio files were long and could not be directly used for decoding as memory shortage problems can arise while running

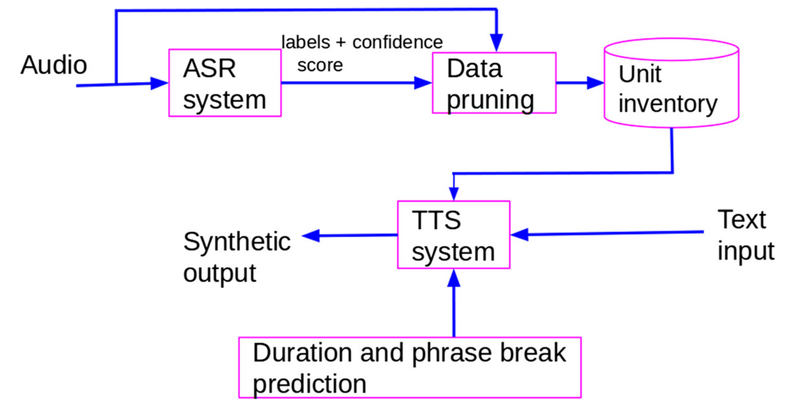

Fig. 1 Architectural block diagram of the complete system

Table 1 Details of the audiobook used for building the unit selection voice

Name of the audiobook	Author	Read by	Running time
Olive (voice 1)	Dinah Maria Mulock CRAIK	Arielle Lipshaw	14:03:13

the Viterbi algorithm. So, silence-based chunking of the long audio files is usually performed. In this work, however, we obtained audio chunks using the open source tool Interslice [8] as we also wanted to obtain chunks of corresponding reference transcripts for building the TTS system using reference transcripts for comparing it with the TTS system built using the hypothesis transcripts. The start and end of each spoken chapter of the audiobooks generally contain metadata such as "This is a LibriVox recording" and the reader's name which are not present in the text chapters downloaded from Project Gutenberg. Since Interslice requires an agreement between the text and speech, we manually checked and added/deleted text at the start and end of each chapter. The same process was carried out even for the lecture speech. Since Interslice does not have a mechanism to prevent the propagation of segmentation error to subsequent segments, we also manually verified the agreement between the start and end of the resulting speech and text chunks before using them for building the TTS systems.

3 ASR and TTS systems

3.1 ASR system development details

Lately, ASR systems have become much more accurate and robust thanks to deep neural networks (DNNs) [16–18]. We used scripts provided with the Kaldi toolkit [19] for training DNN-based ASR systems and the IRSTLM tool [20] for building language models. Kaldi is based upon finite-state transducers, and it is compiled against the OpenFst toolkit [21].

3.1.1 Acoustic modeling

Figure 2 shows the flow of the steps followed for training acoustic models.

1. *Feature extraction*: First, 13 dimensional Mel frequency cepstral coefficients (MFCCs) [22] are extracted. A Hamming window of 25-ms frame size and 10-ms frame shift was used. Then, cepstral mean subtraction is applied on a per-speaker basis. Then, MFCCs are appended with the velocity and acceleration coefficients.
2. *Training monophone system*: A set of context-independent or monophone Gaussian mixture model-hidden Markov model (GMM-HMM) acoustic models were trained on the above features.
3. *Training triphone system with LDA + MLLT*: MFCCs without the deltas and acceleration coefficients were spliced in time taking a context size of seven frames (i.e., ± 3). These features were de-correlated, and their dimensionality was reduced to 40 using linear discriminant analysis (LDA) [23]. Further de-correlation was applied on resulting features using maximum likelihood linear transform (MLLT) [24] which is also known as global semi-tied covariance (STC) [25]. The resulting features were used to train triphone acoustic models.
4. *Training triphone system with LDA + MLLT + SAT*: Then, speaker normalization was applied on above features using feature-space maximum likelihood linear regression (fMLLR), also known as constrained MLLR (CMLLR) [26]. The fMLLR was estimated using the GMM-based system applying speaker-adaptive training (SAT) [26, 27]. A triphone system was again trained with these resulting features.
5. *Training DNN system*: A DNN-HMM system with p-norm non-linearities [28] was trained on top of the SAT features. Here, GMM likelihoods are replaced with the quasi-likelihoods obtained from DNN posteriors by dividing them by the priors of the triphone HMM states.

Table 2 Details of the lecture speech used for building the unit selection voice

Name of the course	Instructor	Running time
Introduction to Public Speaking (voice 2)	Dr. Matt McGarrity	≈ 12 h

3.1.2 Lexicon

A lexicon was prepared from the most frequent 200,000 words in the LibriSpeech corpus. Pronunciations for around one third of them were obtained from CMU-dict. The pronunciations for the remaining words were generated using the Sequitur G2P toolkit [29].

3.1.3 Language modeling

We used the IRSTLM toolkit [20] for training language models. A modified Kneser-Ney smoothing was used [30, 31].

3.1.4 Decoding

First, hypothesis transcriptions were produced using the spliced MFCC features. These transcriptions were then used to estimate the fMLLR transforms as explained above. The accuracy of the hypothesis transcriptions obtained after SAT was much better than that before SAT.

Decoding of the audiobooks was done in two passes. In the first pass, lattices containing competing alternative hypothesis were generated, while in the second pass, Viterbi decoding was applied to find the 1-best hypothesis. While decoding the Olive and lecture data with the ASR

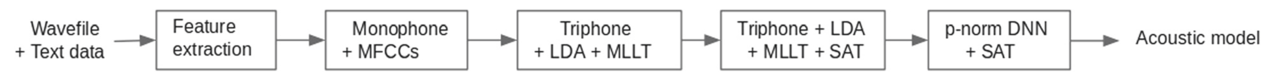

Fig. 2 Steps followed for acoustic modeling

systems trained on themselves, the same 3-gram language model was used for both lattice generation and 1-best Viterbi decoding. In contrast, while decoding the Olive and lecture data with the LibriSpeech ASR system, a 3-gram inexpensive language model pruned with a threshold of 3×10^{-7} was used for lattice generation and a higher order 4-gram language model was used to re-score the language model likelihoods in the lattice, re-rank the set of alternative hypotheses, and find the 1-best hypothesis.

Even though we required phone labels for the audio for building the TTS system, direct phone decoding was not performed as it normally leads to high errors. Rather, word decoding was performed first, and then, word lattices were converted to phone lattices using the lexicon lookup.

3.2 TTS system development details

For synthesis, we made modifications to the TTS system submitted to the Blizzard challenge 2015 [32].

3.2.1 Feature extraction and unit inventory preparation

1. *Unit size*: Units of different sizes ranging from frame-sized units [33, 34], HMM-state sized [35, 36], half-phones [1], and diphones [37] to syllables [38] and to much larger and non-uniform units [39] have been investigated in the literature. We used a quinphone as a context to select appropriate phone-level units. While earlier works mostly used diphone units, where only the previous phone was used as context, the current use of a quinphone is a superset of such selection. Thus, the quinphone context is quite powerful than the regular diphone context. The actual unit used in synthesis is a context-sensitive "phone." This is quite a standard process in unit selection now. Please refer to [2] where the units are called as phones, although sufficient context is used in choosing them. A backoff context (triphone and diphone) is used when a quinphone context is not met. This not only ensures fewer joins and consequently fewer signal processing artifacts, this also leads to good prosody.
2. *Acoustic feature extraction*: Log-energy, 13 dimensional MFCCs, and fundamental frequency (F_0) were extracted for every wave file. A frame size of 20- and 5-ms frame shift was used. The F_0 were extracted using the STRAIGHT tool [40]. The durations and posterior probabilities of the phones which we use as the confidence measure were obtained from the Kaldi decoder.
3. *Preparing the catalog file*: A catalog or dictionary file (which is basically a text file) was prepared which contained the list of all units (including monphone to quinphone) and the attributes of each unit such as duration, start and end times, the duration zscore of each unit type, F_0, log-energy, MFCC values of boundary frames, and posterior probability scores (computed as the minimum of posterior probabilities of phones in that unit). This file was used during synthesis time to compute the target and join costs explained below.
4. *Pre-clustering units*: Pre-clustering is a method that allows the target cost to be effectively pre-calculated. Typically, units of the same type (phones, diphones, etc.) are clustered based on acoustic differences using decision trees [37]. In this work, we clustered units of a type (i.e., units containing the same sequence of phones) on the basis of their positions in words (such as beginning, internal, ending, and singleton), as such clustering implicitly considers acoustic similarity. Units in a cluster are typically called as "candidate" units.

3.2.2 Steps followed at synthesis time

Figure 3 shows the flow of the steps described below.

1. *Text normalization*: A test sentence was first tokenized, punctuations were removed, non-standard words such as time and date were normalized, and abbreviations and acronyms were converted to full forms.
2. *Phonetic analysis*: Each word was broken into a sequence of phones using lexicon. A grapheme-to-phoneme converter [29] was used to convert out-of-vocabulary words and proper names into a phone sequence.
3. *Prediction of phrase-break locations*: Using the audio and automatic transcriptions obtained from the ASR systems, we built statistical parametric voices using the Clustergen synthesizer [41] in the Festival framework [42, 43]. In the current implementation, we took help of the phrase-break locations predicted using classification and regression trees (CART) [44] in Clustergen. For each text input, we first synthesized a statistical parametric voice. A pause unit of appropriate duration was placed at the predicted phrase-break locations during concatenation.

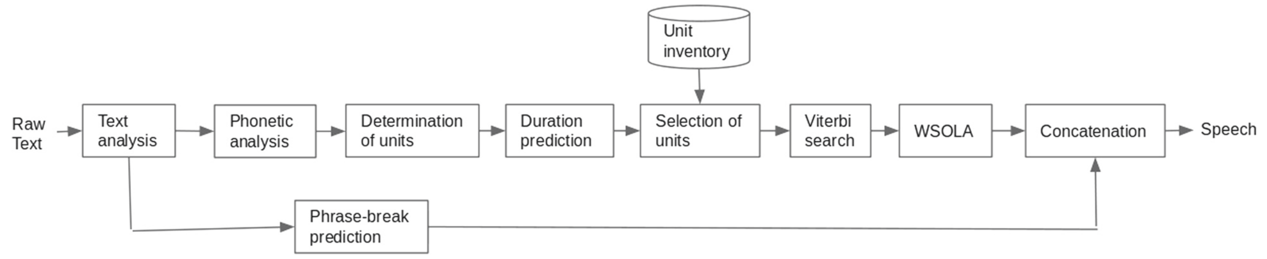

Fig. 3 Steps followed at synthesis time

4. *Determination of units*: For every word in the test sentence, we first searched for units of maximum length which are quinphones or units of length equal to the length of the word if the word is comprised of fewer than five phones. If units of maximum length were not found, then we searched for units of length which is one less than the maximum length, and so on. In short, we joined units of maximum length when they were available in the database or used backoff units of shorter lengths. This approach resulted in fewer joins and a more natural and faster synthesis.
5. *Predicting the durations of units*: In the current implementation, we used the CART-based duration prediction module in Clustergen. For each text input, we first synthesized a statistical parametric voice and used the predicted phone/word durations to select units close to the predicted durations.
6. *Selection of units*:
   - *Target cost computation*: Target cost indicates how close a database unit is to the desired unit. The difference between duration predicted by the CART module in Clustergen [41] and duration of candidate units in the database was used as the target cost.
   - *Join cost computation*: Join cost indicates how well the two adjacently selected units join together. The join cost between two adjacent units u_{i-1} and u_i was calculated using the following equation, which is a linear weighted combination of the distance between log energy, fundamental frequency F_0 (extracted using STRAIGHT tool), and MFCCs of frames near the joining of u_{i-1} and u_i. In the following equation, the symbols α, β, and γ respectively denote the weights for log energy, F_0, and MFCC.

$$\text{Join\_cost} = \alpha C_{F_0}(u_{i-1}, u_i) + \beta C_{\text{log\_energy}}(u_{i-1}, u_i) + \gamma C_{\text{MFCC}}(u_{i-1}, u_i) \tag{1}$$

     Following [32, 45], we used four context frames while computing the distance between the log energies and F_0 of u_{i-1} and u_i, as it helped minimize perceived discontinuities.
   - *Viterbi search*: The Eq. 2 below explains the way the total cost is computed. The term $\mathrm{T}_{\text{dist}}(u_i)$ is the difference between the duration of unit u_i and the predicted duration, and the term $\mathrm{J}_{\text{dist}}(u_i, u_{i-1})$ is the join cost of the optimal coupling point between candidate unit u_i and the previous candidate unit it is to be joined to. W_1 and W_2 denote the weights given to target and join costs, respectively. N denotes the number of units to be concatenated to synthesize the sentence in question. We then used a Viterbi search to find the optimal path through candidate units that minimized the total cost which is the sum total of target and concatenation costs.

$$\text{Total cost} = \sum_{i=1}^{N} W_1 \mathrm{T}_{\text{dist}}(u_i) + W_2 \mathrm{J}_{\text{dist}}(u_i, u_{i-1}) \tag{2}$$

7. *Waveform similarity overlap addition (WSOLA)*: We used an overlap addition-based approach for smoothing the join at the boundaries. Specifically, the cross-correlation formulation of WSOLA [46] was used. The algorithm was reformulated in order to first find a suitable temporal point for concatenating the units at the boundary. This ensured that the concatenation is performed at a point where maximal similarity exists between the units. In different words, this ensured that sufficient signal continuity exists at the concatenation point. For this, cross-correlation between the units was used as a measure of similarity between the units. Next, the units were joined at the point of maximal correlation using a cross-fade technique [33] which further helped remove the phase discontinuities. The number of frames used to calculate the correlation was limited by the duration of the available subword unit. In the current framework, we used the two boundary frames of the individual units to calculate the cross-correlation.

4 Data pruning using confidence measures

In the context of TTS systems, data pruning involves the removal of spurious units (which may be a result of mislabeling or bad acoustics) and units that are redundant in terms of prosodic and phonetic features. Pruning spurious units improves the TTS output [37, 47–51] while pruning redundant units reduces database size thus enabling portability [52–54] and real-time concatenative synthesis [2, 55, 56]. In this work, we focus on removing spurious units and not redundant units.

4.1 Previous works to prune spurious units

In [37], each unit is represented as a sequence of MFCC vectors, and clustering using decision tree proceeds based on questions related to prosodic and phonetic context; each unit is then assessed for its frame-based distance to the cluster center. Units which lie far from their cluster centers are termed as outliers and hence pruned. In [47], this evaluation is done on the basis of an HMM framework: only instances which have the highest HMM scores are retained to represent a cluster of similar units. Confidence features such as log-likelihood ratio [48], transcription confidence ratio [49], and generalized posterior probability [50] have also been used for pruning. We used posterior probability [50, 57] and unit duration [51] obtained from the ASR system as confidence measures.

4.2 Relevance of posterior probability and unit duration as confidence features

Posterior probability helps detecting mislabeled and bad acoustic regions while unit durational measure helps detecting unnaturally short or long units (which may have high posterior probability values), but they can make words unintelligible or sound hyper-articulated, respectively. Thus, both these confidence features are directly related to the intelligibility and naturalness of speech.

4.3 Other advantages of posterior probability

Other motivations to use posterior probability as confidence feature are (1) posterior probability, by definition, tells the correctness or confidence of a classification; (2) it has been shown to work consistently better than the other two formulations of confidence measures, which are the confidence measure as a combination of predictor features and the confidence measures posed as an utterance verification problem [58]; and (3) posterior probability becomes more reliable when robust acoustic [28] and language models are used (as in this case) [59].

4.4 Computation of posterior probability and unit durational zscore

In an ASR system, the posterior probability of a phone or a word hypothesis w given a sequence of acoustic feature vectors $O_1^T = O_1 O_2 .. O_T$ is computed (as given in Eq. 3) as the sum of posterior probabilities of all paths passing through w (in around the same time region) in the lattice. It is computed using a forward-backward algorithm over the lattice. In the equation below, W_s and W_e respectively indicate the sequence of words preceding and succeeding w in a path in the lattice.

$$\begin{aligned}
p(w|O_1^T) &= \\
&= \sum_{W_s} \sum_{W_e} p\left(W_s w W_e | O_1^T\right) \\
&= \frac{\sum_{W_s} \sum_{W_e} \left[p\left(O_1^{t_s}|W_s\right) p\left(O_{t_s}^{t_e}|w\right) p\left(O_{t_e}^T|W_e\right) p\left(W'\right)\right]}{p\left(O_1^T\right)} \\
&= \frac{\sum_{W_s} \sum_{W_e} \left[p\left(O_1^{t_s}|W_s\right) p\left(O_{t_s}^{t_e}|w\right) p\left(O_{t_e}^T|W_e\right) p\left(W'\right)\right]}{\sum_{W} \left[\sum_{W_s} \sum_{W_e} \left[p\left(O_1^{t_s}|W_s\right) p\left(O_{t_s}^{t_e}|w\right) p\left(O_{t_e}^T|W_e\right) p\left(W'\right)\right]\right]}
\end{aligned} \tag{3}$$

Normally, the phone hypotheses in the neighborhood of a low confidence phone are also affected. Hence, we discard all units containing even a single phone below a specified threshold. The posterior probability of a unit is calculated as the minimum of posterior probability of phones in that unit.

The unit durational zscore for every unit class is computed as in the following equation.

$$\text{zscore} = \frac{\text{duration} - \text{mean}}{\text{standard deviation}} \tag{4}$$

5 Experiments and evaluation

5.1 Experiment 1: checking the performance of ASR systems

We trained a p-norm DNN-HMM acoustic model for all three ASR systems trained on the Olive, lecture, and LibriSpeech data. In the first pass, lattices containing a competing alternative hypothesis were generated, while in the second pass, Viterbi decoding was applied to find the 1-best hypothesis. While decoding the Olive and lecture data with ASR systems trained on themselves, the same 3-gram language model was used for both lattice generation and 1-best Viterbi decoding. In contrast, while decoding the Olive and lecture data with the LibriSpeech ASR system, a 3-gram inexpensive language model pruned with a threshold of 3×10^{-7} was used for lattice generation and a higher order 4-gram language model was used to re-score the language model likelihoods in the lattice, re-rank the set of alternative hypotheses, and find the 1-best hypothesis.

Table 3 shows the word error rates (WERs) and phone error rates (PERs) given by the ASR systems trained on the Olive and lecture data and tested on the Olive and lecture data (TTS data), respectively. It also shows the WER and PER of the ASR system trained with the LibriSpeech data and tested on the Olive and lecture

Table 3 WERs and PERs of ASR systems trained on TTS data and LibriSpeech data

Training data for ASR	Test data to ASR	WER (%)	PER (%)
Olive	Olive	2.54	1.02
LibriSpeech	Olive	3.74	1.57
Lecture	Lecture	5.91	5.19
LibriSpeech	Lecture	21.93	10.20

data. Note, for computing WERs for both the Olive and lecture data, we respectively used the word-level transcriptions available at the Project Gutenberg website and those available at Coursera, as reference transcriptions. These transcriptions are reliable, *but not gold standard*. For computing the PERs, phone-level reference transcriptions were obtained by converting word-level reference transcriptions to phones using a lexicon lookup. A lexicon containing 200,000 words provided with the Kaldi setup was used for that purpose. We can observe the following things in Table 3:

1. As expected, the performance of the ASR system trained with LibriSpeech data is relatively poor, but it is still quite decent.
2. The performance gap between the ASR system trained on lecture data and LibriSpeech is big as compared to that between the ASR system trained on Olive and LibriSpeech data because LibriSpeech is a read speech corpus just like Olive, while lecture data is spontaneous data.
3. The performance of the ASR system trained on lecture data and tested on lecture data is slightly poorer than the ASR system trained on Olive data and tested on Olive data. The reason could be that there is relatively more uniformity in the Olive data (read speech) compared to lecture speech. Lecture speech is more spontaneous and contains a lot of filled pauses, dis-fluencies, fast spoken (at times unintelligible words), and also emphasized words.

5.2 Experiment 2: checking the effectiveness of posterior probability as a confidence measure

We saw that the ASR system trained with LibriSpeech data produces more accurate and reasonably accurate transcripts for Olive and lecture data, respectively. The incorrect phone hypotheses should not be a part of the unit inventory and need to be automatically removed to prevent them from corrupting a synthesized voice. We used the posterior probability given by the ASR system as a confidence measure to prune the erroneous data, where all phones below an optimal posterior probability threshold were pruned. In this experiment, we see to what extent our confidence measure is useful to automatically detect bad acoustics and incorrect hypotheses of the ASR system. Table 4 shows the PER as in Table 3 and its breakup in terms of percentage substitution, insertion, and deletion errors. Note that we can have posterior probabilities only for a hypothesis produced by the ASR system. The phone hypotheses could not be correct phones, substitutions, or insertions. So, deletions could not be detected by the confidence measure, but as their amount was small, we ignored them. The confidence measure is expected to truly reject (TR) as many substitution and insertion errors. Table 4 also shows the percentage of true acceptances (TA), false rejections (FR), true rejections (TR), and false acceptances (FA) obtained at the maximum and optimal posterior probability threshold value equal to 1.0 for all the four cases in Table 3. This threshold value is optimal in the sense that it yields the least number of false acceptances (which are the ASR system's erroneous phone hypotheses termed as correct and hence left unpruned by the confidence measure). We would want the least number of erroneous hypotheses/spurious phones, and hence, we prefer the least number of false acceptances. We observe that the posterior probability does a decent job to harness most of the correct data (as can be seen from the percentage of TAs) leaving just a small amount of erroneous data behind (as can be seen from the amount of false acceptances). Specifically, in the case of the lecture speech recognized by the ASR system trained with Librispeech data, we can see that 10.20 %PER is effectively reduced to $2.77 + 1.63 = 4.4\,\%$ (percentage of FAs) with the use of a confidence measure. There is also a sizeable amount of false rejections that we can see, but we could afford to lose that data since we were using large data for synthesis.

5.3 Experiment 3: checking the effect of pruning based on the posterior probability and unit duration on WER and MOS

There are several examples of fast unintelligible speech (in the case of common and short words such as "to," "the,"

Table 4 Performance of ASR systems and posterior probability confidence measure

Training data for ASR	Test data to ASR	%PER	%sub	%ins	%del	%FA	%TR	%FR	%TA
Olive	Olive	1.02	0.20	0.66	0.16	0.74	0.12	0.54	98.60
LibriSpeech	Olive	1.57	0.58	0.69	0.30	1.00	0.27	4.94	93.79
Lecture	Lecture	5.19	0.96	3.52	0.71	3.09	1.39	3.28	92.24
LibriSpeech	Lecture	10.20	3.27	5.30	1.63	2.77	5.80	19.61	71.82

Table 5 Posterior probability and duration zscore thresholds used to achieve different amounts of data pruning, for all four voices

Percent units used	Posterior probability and duration zscore thresholds			
	Test data = Olive		Test data = Intro. to Public Speaking	
	ASR trained on Olive data	ASR trained on LibriSpeech	ASR trained on lecture data	ASR trained on LibriSpeech
100	–	–	–	–
	1.00, ≈ 97 %	1.00, ≈ 92 %	1.00, ≈ 93 %	1.00, ≈ 65 %
50	1.00, ± 0.51	1.00, ± 0.70	1.00, ± 0.57	1.00, ± 0.98
30	1.00, ± 0.35	1.00, ± 0.45	1.00, ± 0.39	1.00, ± 0.60

"for," and "and" plus other short words) and unnaturally long or emphasized/hyper-articulated words particularly in lecture speech. Several instances of such words have a posterior probability value equal to 1.0, and are left unpruned. Hence, we also prune the units which are much deviant from their mean duration. In addition, pruning units based on duration allows us to prune many more units than is possible using posterior probability alone.

Table 5 shows, for all four voices, the different posterior probability and duration zscore thresholds used to achieve different amounts of unit pruning. The first number in every cell is the posterior probability threshold which is 1.0. The second entry in the second row indicates the percentage of units retained when only posterior probability based pruning is applied. This value is different (97 %, 92 %, 93 %, 65 %) in case of all four systems. No duration threshold was applied in this case. The second entry in third and fourth rows indicates the duration thresholds applied for performing duration based pruning in addition to posterior probability based pruning.

We used the hypotheses and the time stamps given by the ASR systems trained with Olive, lecture, and Librispeech data for synthesis. Even in the case of Olive and lecture, respectively, we used hypotheses of the ASR system instead of force-aligned reference transcriptions from Project Gutenberg and Coursera because the reference transcriptions are reliable, but not gold standard, and the ASR system trained and adapted to a single speaker generally gives better transcriptions and is able to detect the inconsistencies in the reference speech and text.

Table 5 contains 16 different combinations of posterior probability and duration zscore thresholds. We synthesized (for each combination in the table) 10 semantically unpredictable sentences (SUS) [60] and 10 news sentences from the Blizzard 2013 test corpus. So, 20 sentences were synthesized for each combination. In all, 320 sentences were synthesized. A few of the samples used for this experiment can be listened to at https://researchweb.iiit.ac.in/~tejas.godambe/EURASIP/. These sentences were randomly distributed among 16 listeners for perceptual test. So, each listener transcribed 10 SUS (from which we computed the WER indicating the speech intelligibility) and rated the naturalness of the news utterances on a scale of 1 (worst) to 5 (best) from which we calculated the mean opinion score (MOS).

Tables 6 and 7 respectively show the WER and MOS for all four voices synthesized using different amounts of pruned data. We can see that the WERs and MOS are quite good for all the four voices. We can observe the following things in Table 6.

1. The WERs are high for lecture speech than audiobook speech.
2. The WERs are high for unpruned data (as can be seen in the first row). They become slightly better in the second row when we use only units having a posterior probability value equal to 1.0 to synthesize the sentences. The improvement is maximum in the last column where the amount of pruned data having posterior probability less than 1.0 is the highest.
3. Selecting units close to mean duration (as in the third row) decreases WER even further, as short units which are much deviant from the mean duration are pruned.
4. The improvement in WER observed with duration pruning (difference in WERs of the second and third

Table 6 Word error rates for all four voices for different amounts of data pruning

Percentage units used	Word error rate (%)			
	Test data = Olive		Test data = lecture	
	ASR trained on Olive	ASR trained on LibriSpeech	ASR trained on lecture	ASR trained on LibriSpeech
100	14.25	17.21	22.13	28.17
	13.50	15.95	20.56	23.90
50	8.11	9.56	16.25	17.15
30	6.26	6.28	13.25	13.87

Table 7 MOS scores for all four voices for different amounts of data pruning

Percentage units used	Mean opinion score			
	Test data = Olive		Test data = lecture	
	ASR trained on Olive	ASR trained on LibriSpeech	ASR trained on lecture	ASR trained on LibriSpeech
100	3.49	3.52	3.18	2.91
	3.51	3.47	3.21	3.22
50	3.28	3.22	2.99	3.05
30	3.08	3.00	2.90	2.93

rows) is more than the difference in the WERs of first and second rows observed with pruning units having a posterior probability less than 1.0. This difference is more evident in the case of the lecture speech (which contains more units corresponding to fast speech than audiobook contributing to less intelligible speech). The WER further reduces when more units based on duration are pruned (even when only 30 % units are retained).

In the case of naturalness of speech in Table 7, we can observe the following things.

1. Voices built using an audiobook seem to be more natural than those built using lecture speech.
2. The MOS is almost the same for the first and second rows except the case of the last column where a noticeable improvement is observed in MOS.
3. The MOS decreases as we move down rows as it becomes difficult to find units having a duration close to predicted duration and which can also maintain continuity in terms of energy, F_0, and MFCCs.

6 Conclusions

Today, a large amount of audio data has become available to us via the web, and also, we can easily record and store a huge amount of audio data on handheld devices, etc. These data are rich in prosody and provide many voices to choose from, and their availability can help to rapidly build general-purpose unit selection voices. But, there are a few hurdles such as the unavailability of transcriptions or availability of imprecise transcriptions and the presence of speech and non-speech noises. In this paper, we built voices for an audiobook (read speech) in a female voice and a lecture (spontaneous speech) in a male voice using reference transcripts and a combination of automatic transcripts and confidence measure-based data pruning and showed that voices of comparable quality as that using reference transcripts can be rapidly built using found data.

Endnotes

[1]http://www.openslr.org/12/.

[2]http://www.openslr.org/resources/11/3-gram.pruned.3e-7.arpa.gz.

[3]http://www.openslr.org/resources/11/4-gram.arpa.gz.

Competing interests
The authors declare that they have no competing interests.

Acknowledgements
This project was funded by the National Plan for Science, Technology and Innovation (MAARIFAH), King Abdulaziz City for Science and Technology, Kingdom of Saudi Arabia, Award Number 10-INF1038-02.

Author details
[1]International Institute of Information Technology Hyderabad, Hyderabad, India. [2]King Abdulaziz City for Science and Technology, Riyadh, Saudi Arabia. [3]King Saud University, Riyadh, Saudi Arabia.

Received: 12 September 2015 Accepted: 22 February 2016
Published online: 01 March 2016

References

1. M Beutnagel, A Conkie, J Schroeter, Y Stylianou, A Syrdal, in *Joint Meeting of ASA, EAA, and DAGA*. The AT&T next-gen TTS system (Citeseer, Berlin, Germany, 1999), pp. 18–24
2. AJ Hunt, AW Black, in *Proc. of ICASSP*. Unit selection in a concatenative speech synthesis system using a large speech database, vol. 1 (IEEE, Atlanta, Georgia, USA, 1996), pp. 373–376
3. S Ouni, V Colotte, U Musti, A Toutios, B Wrobel-Dautcourt, M-O Berger, C Lavecchia, Acoustic-visual synthesis technique using bimodal unit-selection. EURASIP J. Audio Speech Music Process. **2013**(1), 1–13 (2013)
4. J Kominek, AW Black, in *Proc. of Fifth ISCA Workshop on Speech Synthesis*. The CMU ARCTIC speech databases (ISCA, Pittsburgh, PA, USA, 2003). http://festvox.org/cmuarctic
5. H Harrod, How do you teach a computer to speak like Scarlett Johansson? (2014). http://goo.gl/xn5gBw. Accessed 15 Feb 2014
6. X Anguera, N Perez, A Urruela, N Oliver, in *Proc. of ICME*. Automatic synchronization of electronic and audio books via TTS alignment and silence filtering (IEEE, Barcelona, Spain, 2011), pp. 1–6
7. N Campbell, in *Proc. of ICSLP*. Autolabelling Japanese TOBI, vol. 4 (IEEE, Philadelphia, USA, 1996), pp. 2399–2402
8. K Prahallad, AW Black, Segmentation of monologues in audio books for building synthetic voices. IEEE Trans. Audio Speech Lang. Process. **19**(5), 1444–1449 (2011)
9. C Cerisara, O Mella, D Fohr, in *Proc. of Interspeech*. JTrans, an open-source software for semi-automatic text-to-speech alignment (ISCA, Brighton, UK, 2009), pp. 1823–1826
10. PJ Moreno, CF Joerg, J-M Van Thong, O Glickman, in *Proc. of ICSLP*. A recursive algorithm for the forced alignment of very long audio segments, vol. 98 (ISCA, Sydney, Australia, 1998), pp. 2711–2714
11. PJ Moreno, C Alberti, in *Proc. of ICASSP*. A factor automaton approach for the forced alignment of long speech recordings (IEEE, Taipei, Taiwan, 2009), pp. 4869–4872
12. G Bordel, M Peñagarikano, LJ Rodríguez-Fuentes, A Varona, in *Proc. of Interspeech*. A simple and efficient method to align very long speech

signals to acoustically imperfect transcriptions (ISCA, Portland, USA, 2012), pp. 1840–1843

13. N Braunschweiler, MJ Gales, S Buchholz, in *Proc. of Interspeech*. Lightly supervised recognition for automatic alignment of large coherent speech recordings (ISCA, Makuhari, Chiba, Japan, 2010), pp. 2222–2225
14. Y Tao, L Xueqing, W Bian, A dynamic alignment algorithm for imperfect speech and transcript. Comput. Sci. Inf. Syst. **7**(1), 75–84 (2010)
15. V Panayotov, G Chen, D Povey, S Khudanpur, in *Proc. of ICASSP*. LibriSpeech: an ASR corpus based on public domain audio books (IEEE, Brisbane, Queensland, Australia, 2015), pp. 5206–5210
16. V Peddinti, D Povey, S Khudanpur, in *Proceedings of INTERSPEECH*. A time delay neural network architecture for efficient modeling of long temporal contexts (ISCA, Dresden, Germany, 2015), pp. 2440–2444
17. L Tóth, Phone recognition with hierarchical convolutional deep maxout networks. EURASIP J. Audio Speech Music Process. **2015**(1), 1–13 (2015)
18. P Motlicek, D Imseng, B Potard, PN Garner, I Himawan, Exploiting foreign resources for DNN-based ASR. EURASIP J. Audio Speech Music Process. **2015**(1), 1–10 (2015)
19. D Povey, A Ghoshal, G Boulianne, L Burget, O Glembek, N Goel, M Hannemann, P Motlíček, Y Qian, P Schwarz, et al, in *Proc. of ASRU*. The Kaldi speech recognition toolkit (IEEE, Waikoloa, HI, USA, 2011). EPFL-CONF 192584
20. M Federico, N Bertoldi, M Cettolo, in *Proc. of Interspeech*. Irstlm: an open source toolkit for handling large scale language models (ISCA, Brisbane, Australia, 2008), pp. 1618–1621
21. C Allauzen, M Riley, J Schalkwyk, W Skut, M Mohri, in *Implementation and Application of Automata*. Openfst: a general and efficient weighted finite-state transducer library (Springer, Berlin, Heidelberg, 2007), pp. 11–23
22. SB Davis, P Mermelstein, Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. IEEE Trans. Acoust. Speech Signal Process. **28**(4), 357–366 (1980)
23. RO Duda, PE Hart, DG Stork, *Pattern classification*. (Wiley, 2012)
24. RA Gopinath, in *Proc. of ICASSP*. Maximum likelihood modeling with Gaussian distributions for classification, vol. 2 (IEEE, Washington, USA, 1998), pp. 661–664
25. MJ Gales, Semi-tied covariance matrices for hidden Markov models. IEEE Trans. Speech Audio Process. **7**(3), 272–281 (1999)
26. MJ Gales, Maximum likelihood linear transformations for HMM-based speech recognition. Comput. Speech Lang. **12**(2), 75–98 (1998)
27. S Matsoukas, R Schwartz, H Jin, L Nguyen, in *DARPA Speech Recognition Workshop*. Practical implementations of speaker-adaptive training (Citeseer, Chantilly, VA, 1997), pp. 11–14
28. X Zhang, J Trmal, D Povey, S Khudanpur, in *Proc. of ICASSP*. Improving deep neural network acoustic models using generalized maxout networks (IEEE, Florence, Italy, 2014), pp. 215–219
29. M Bisani, H Ney, Joint-sequence models for grapheme-to-phoneme conversion. Speech Commun. **50**(5), 434–451 (2008)
30. R Kneser, H Ney, in *Proc. of ICASSP*. Improved backing-off for m-gram language modeling, vol. 1 (IEEE, Detroit, Michigan, USA, 1995), pp. 181–184
31. SF Chen, J Goodman, in *Proc. of ACL*. An empirical study of smoothing techniques for language modeling (Association for Computational Linguistics, Santa Cruz, California, USA, 1996), pp. 310–318
32. SK Rallabandi, A Vadapalli, S Achanta, S Gangashetty, in *Proc. of Blizzard Challenge 2015*. IIIT Hyderabad's submission to the Blizzard Challenge 2015 (ISCA, Dresden, Germany, 2015)
33. T Hirai, S Tenpaku, in *Fifth ISCA Workshop on Speech Synthesis*. Using 5 ms segments in concatenative speech synthesis (ISCA, Pittsburgh, PA, USA, 2004), pp. 37-42
34. Z-H Ling, R-H Wang, in *Ninth International Conference on Spoken Language Processing*. HMM-based unit selection using frame sized speech segments (ISCA, Pittsburgh, PA, USA, 2006)
35. RE Donovan, PC Woodland, in *Eurospeech Proceedings: 4th European Conference on Speech Communication and Technology*. Improvements in an HMM-based speech synthesiser, vol. 1 (ISCA, Madrid, Spain, 1995), pp. 573–576
36. X Huang, A Acero, J Adcock, H-W Hon, J Goldsmith, J Liu, M Plumpe, in *Spoken Language, 1996. ICSLP 96. Proceedings., Fourth International Conference On*. Whistler: a trainable text-to-speech system, vol. 4 (IEEE, Philadelphia, PA, USA, 1996), pp. 2387–2390
37. AW Black, PA Taylor, in *Proc. of Eurospeech*. Automatically clustering similar units for unit selection in speech synthesis (ISCA, Rhodes, Greece, 1997), pp. 601-604
38. SP Kishore, AW Black, in *Proc. of Interspeech*. Unit size in unit selection speech synthesis (ISCA, Geneva, Switzerland, 2003), pp. 1317–1320
39. H Segi, T Takagi, T Ito, in *Fifth ISCA Workshop on Speech Synthesis*. A concatenative speech synthesis method using context dependent phoneme sequences with variable length as search units (ISCA, Pittsburgh, PA, USA, 2004), pp. 115–120
40. H Kawahara, I Masuda-Katsuse, A De Cheveigne, Restructuring speech representations using a pitch-adaptive time-frequency smoothing and an instantaneous-frequency-based F0 extraction: possible role of a repetitive structure in sounds. Speech Commun. **27**(3), 187–207 (1999)
41. AW Black, in *Proc. of Interspeech*. Clustergen: a statistical parametric synthesizer using trajectory modeling (ISCA, Pittsburgh, PA, USA, 2006), pp. 1394–1397
42. A Black, P Taylor, R Caley, R Clark, K Richmond, S King, V Strom, H Zen, The festival speech synthesis system, version 1.4. 2 (2001). Unpublished document available via http://www.cstr.ed.ac.uk/projects/festival.html
43. P Taylor, AW Black, R Caley, in *Proc. of Eurospeech*. The architecture of the festival speech synthesis system (ISCA, Jenolan Caves, Australia, 1998), pp. 147-151
44. L Breiman, J Friedman, CJ Stone, RA Olshen, *Classification and regression trees*. (CRC press, 1984)
45. BSR Rajaram, KHR Shiva, R A G, in *Blizzard Challenge 2014*. MILE TTS for Tamil for Blizzard Challenge 2014 (ISCA, Singapore, 2014)
46. W Verhelst, M Roelands, in *Proc. of ICASSP*. An overlap-add technique based on waveform similarity (WSOLA) for high quality time-scale modification of speech, vol. 2 (IEEE, Minneapolis, Minnesota, USA, 1993), pp. 554–557
47. H Hon, A Acero, X Huang, J Liu, M Plumpe, in *Proc. of ICASSP*. Automatic generation of synthesis units for trainable text-to-speech systems, vol. 1 (IEEE, Seattle, Washington, USA, 1998), pp. 293–296
48. H Lu, Z-H Ling, S Wei, L-R Dai, R-H Wang, in *Proc. of Interspeech*. Automatic error detection for unit selection speech synthesis using log likelihood ratio based SVM classifier (ISCA, Makuhari, Chiba, Japan, 2010), pp. 162–165
49. J Adell, PD Agüero, A Bonafonte, in *Proc. of ICASSP*. Database pruning for unsupervised building of text-to-speech voices (IEEE, Toulouse, France, 2006), pp. 889–892
50. L Wang, Y Zhao, M Chu, FK Soong, Z Cao, in *Proc. of Interspeech*. Phonetic transcription verification with generalized posterior probability (ISCA, Lisbon, Portugal, 2005), pp. 1949-1952
51. J Kominek, AW Black, in *Fifth ISCA Workshop on Speech Synthesis*. Impact of durational outlier removal from unit selection catalogs (ISCA, Pittsburgh, PA, USA, 2004), pp. 155–160
52. H Lu, W Zhang, X Shao, Q Zhou, W Lei, H Zhou, A Breen, in *Proc. of Interspeech*. Pruning Redundant synthesis units based on static and delta unit appearance frequency (ISCA, Dresden, Germany, 2015), pp. 269–273
53. R Kumar, SP Kishore, in *Proc. of Interspeech*. Automatic pruning of unit selection speech databases for synthesis without loss of naturalness (ISCA, Jeju island, Korea, 2004), pp. 1377-1380
54. V Raghavendra, K Prahallad, in *Proc. of ICON*. Database pruning for Indian language unit selection synthesizers (ACL, Hyderabad, India, 2009), pp. 67–74
55. D Schwarz, G Beller, B Verbrugghe, S Britton, et al, in *Proceedings of the COST-G6 Conference on Digital Audio Effects (DAFx), Montreal, Canada*. Real-time corpus-based concatenative synthesis with catart (Citeseer, 2006), pp. 279–282
56. RE Donovan, in *Proc. of ICASSP*. Segment pre-selection in decision-tree based speech synthesis systems, vol. 2 (IEEE, Istanbul, Turkey, 2000), pp. 11937–11940
57. F Wessel, R Schlüter, K Macherey, H Ney, Confidence measures for large vocabulary continuous speech recognition. IEEE Trans. Speech Audio Process. **9**(3), 288–298 (2001)
58. H Jiang, Confidence measures for speech recognition: a survey. Speech Commun. **45**(4), 455–470 (2005)
59. NT Vu, F Kraus, T Schultz, in *Proc. of Spoken Language Technology Workshop*. Multilingual A-stabil: a new confidence score for multilingual unsupervised training (ISCA, Berkley, CA, USA, 2010), pp. 183–188
60. C Benoît, M Grice, V Hazan, The sus test: a method for the assessment of text-to-speech synthesis intelligibility using semantically unpredictable sentences. Speech Commun. **18**(4), 381–392 (1996)