TTS for Low Resource Languages: A Bangla Synthesizer

Alexander Gutkin, Linne Ha, Martin Jansche, Knot Pipatsrisawat, Richard Sproat
Google, Inc.
1600 Amphitheatre Parkway, Mountain View, CA
{agutkin,linne,mjansche,thammaknot,rws}@google.com

Abstract

We present a text-to-speech (TTS) system designed for the dialect of Bengali spoken in Bangladesh. This work is part of an ongoing effort to address the needs of under-resourced languages. We propose a process for streamlining the bootstrapping of TTS systems for under-resourced languages. First, we use crowdsourcing to collect the data from multiple ordinary speakers, each speaker recording small amount of sentences. Second, we leverage an existing text normalization system for a related language (Hindi) to bootstrap a linguistic front-end for Bangla. Third, we employ statistical techniques to construct multi-speaker acoustic models using Long Short-Term Memory Recurrent Neural Network (LSTM-RNN) and Hidden Markov Model (HMM) approaches. We then describe our experiments that show that the resulting TTS voices score well in terms of their perceived quality as measured by Mean Opinion Score (MOS) evaluations.
Keywords: Text-to-Speech, Under-resourced languages, Bangla

1. Introduction

Developing a text-to-speech (TTS) system is a major investment of effort. For the best concatenative unit-selection systems (Hunt and Black, 1996), many hours of recording are typical, and one needs to invest in careful lexicon development, and complex rules for text normalization, among other things. All of this requires resources, as well as curation from native-speaker linguists.

For low-resource languages it is often hard to find relevant resources, so there has been much recent work on methods for developing systems using minimal data (Sitaram et al., 2013). The downside of these approaches is that the quality of the resulting systems can be low and it is doubtful people would want to use them.

We are therefore interested in approaches that minimize effort, but still produce systems that are acceptable to users. This paper describes our development of a system for Bangla, the main language of Bangladesh and a major language of India, and in particular the speech, lexicon and text normalization resources, all of which we are planning to release, under a liberal open-source license.

A core idea is the use of *multiple ordinary speakers*, rather than a single professional speaker (the normal approach). There are two main justifications. First, voice talents are expensive, so it is more cost-effective to record ordinary people; but these quickly get tired reading aloud, limiting how much they can read. We thus need multiple speakers for an adequate database. Second, there is an added benefit of privacy: we can create a natural-sounding voice that is not identifiable as a specific individual.

Unit selection (Hunt and Black, 1996) is a dominant approach to speech synthesis, but it is not suitable when working with multiple speakers, one obvious reason being that the system will often adjoin units from different speakers, resulting in very unnatural output. Instead we adopt a statistical parametric approach (Zen et al., 2009). In statistical parametric synthesis the training stage uses multiple speaker data by estimating an averaged representation of various acoustic parameters representing each individual speaker. Depending on the number of speakers in the corpus, their acoustic similarity and ratio of speaker genders, the resulting acoustic model can represent an average voice that is very humanlike yet cannot be identified as any specific recorded speaker.

This paper is organized as follows: We describe the crowdsourcing approach to assemblying the speech database in Section 2.. The TTS system architecture is introduced in Section 3.. Next, experimental results are presented in Section 4.. While this paper mostly focuses on Bangla spoken in Bangladesh, we also describe the initial experiments with the West Bengali dialect of Bangla spoken in India. Finally, Section 5. concludes the paper and discusses venues for future research.

2. Crowdsourcing the Speakers

We were familiar with collecting data from multiple speakers from data collection efforts for automatic speech recognition (Hughes et al., 2010). There, our goal was at least 500 speakers, of varying regional accents in different recording environments, recorded using mobile phones. For TTS, very different criterion is conventional: a professional standard dialect speaker in a recording studio. But this is expensive and cannot scale if one wants to cover the world's many low-resource languages.

New statistical parametric synthesis methods (Zen et al., 2009) allow for building a voice from multiple speakers, but one still needs speakers that are acoustically similar. To achieve this, we held an audition to find Bangla speakers with compatible voices. Fifteen Bangladeshi employees at Google's Mountain View campus auditioned. From that sample, we sent a blind test survey to 50 Bangladeshi Googlers to vote for their top two preferences. Using the top choice – a male software engineer from Dhaka – as our reference, we chose 5 other male Dhaka speakers with similar vocal characteristics.

Our experience with crowd-sourced ASR data collection taught us the importance of good data collection tools. ChitChat is a web-based mobile recording studio that allows audio data to be collected and managed simply. Each speaker is presented with a series of sentences assigned to them for recording. The tool records at 48 kHz, detecting audio clipping to ensure quality, and ambient noise prior to recording each sentence, with a high noise level triggering an alert preventing further recording. Audio can be up-

loaded to the server or stored locally for later uploading. A sample screenshot of the ChitChat UI is shown in Figure 1.

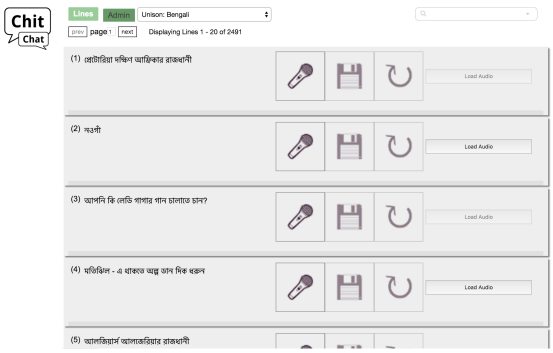

Figure 1: Screenshot of ChitChat UI.

For the recordings we used an ASUS Zen fanless laptop with a Neumann KM 184 microphone, a USB converter and preamp, together costing under US$2000. We recorded our volunteers over 3 days in an anechoic studio in June 2015. Each recorded about 250 phrases, averaging 45 minutes, mined from Bangla and English Wikipedia. Volunteers were first instructed on the "bright" style of voice we were interested in. After a supervised practice run of 10–15 minutes, the remainder was recorded independently while being observed remotely using ChitChat's admin features. Recordings were stopped if the voice sounded tired or mouth-dry. The sessions yielded about 2000 utterances.

3. System Architecture

A typical parametric synthesizer pipeline consists of training and synthesis parts. Similar to Automatic Speech Recognition (ASR) pipeline (Gales and Young, 2008), the training process consists of two steps: data preparation and acoustic model training (Zen and Sak, 2015). During the data preparation step one extracts a parametric representation of the audio from the speech corpus. A typical acoustic representation includes spectral, excitation and fundamental frequency parameters, and pertinent linguistic parameters are extracted as well, which take into account linguistic and prosodic contexts for the current phoneme. Once acoustic and linguistic parameters are extracted, during the acoustic model training stage we use machine learning techniques to estimate faithful statistical representations of the acoustic and linguistic parameters extracted by the previous step.

3.1. Phonology and Lexicon

As with any TTS system, our Bangla system requires a phoneme inventory and a grapheme-to-phoneme conversion system. While the latter might be done with simple grapheme-to-phoneme rules, Bangla spelling is sufficiently mismatched with the pronunciation of colloquial Bangla to warrant a transcription effort to develop a phonemic pronunciation dictionary.

Consider the Bangla word for *telescope*, which is transcribed in IT3 transliteration (Prahallad et al., 2012) as `d uu ra b ii k shha nd a` and in IPA as /dur.bik.k^{h}ɔn/ (Figure 2).

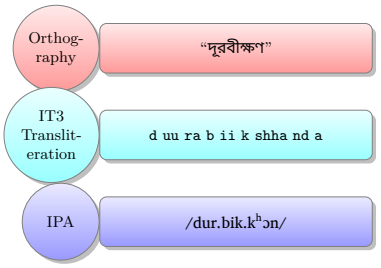

Figure 2: Transcriptions for Bangla word for *telescope*.

In this example there are several mismatches between the actual pronunciation and what we would expect on the basis of the spelling: the orthographic long vowel diacritics transliterated as `uu` and `ii` are actually pronounced as the short vowels /u/ and /i/, and the cluster `k shh` is actually pronounced /k.k^{h}/. The final letter transcribed as `nd a` has an inherent vowel, which is not pronounced in this case, but in other cases would be /o/ or /ɔ/. Indeed, the determination of the pronunciation of the inherent vowel (as /null/, /o/ or /ɔ/) is a major issue in Bangla.

The ambiguity of the inherent vowel between /o/ and /ɔ/ gives rise to systematic homographs in the verb paradigm. For example, the frequently occurring verb form transliterated as `ka r e` can be pronounced /ko.re/ (perfect participle) or /kɔ.re/ (3rd person familiar, simple present). This ambiguity is systematic in Bangla and generally affects verb stems with open-mid vowels. We deal with these and other homographs in a rule-based disambiguation component as part of text normalization.

The sometimes complex and often unpredictable correspondence between surface orthography and phonemic pronunciation motivates the need for a hand curated pronunciation dictionary. While we are aware of preceding efforts, e.g. the speech corpus described in (Alam et al., 2010; Habib et al., 2011), we were unable to find sufficient existing resources that are available for commercial use. In an effort to partially remedy this situation, we have made our own pronunciation dictionary available[1] under a permissive open-source license.

Our phonological representation closely follows the description of Bangladeshi colloquial Bangla in (Ud Dowla Khan, 2010). It uses 39 segmental phonemes, much fewer than the 47 phonemes used by (Prahallad et al., 2012). A team of five linguists transcribed more than 65,000 words into a phonemic representation. This includes mostly words in Bengali script, as well as nearly 5,000 words in Latin script, including English words and foreign entity names. The transcription effort utilized a version of our phonemic transcription tools (Ainsley et al., 2011) and quality control methodology (Jansche, 2014). Our transcribers were further aided by the output of a pronunciation model, which was used to pre-fill the

[1] `https://github.com/googlei18n/language-resources/tree/master/bn/data`

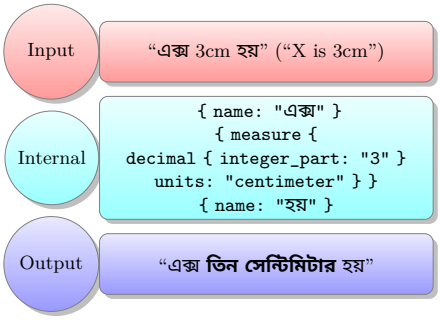

Figure 3: Example of a basic Kestrel workflow. The output corresponding to the measure phrase is shown in bold.

transcriptions of words so that transcribers could focus on correcting transcriptions, rather than entering them from scratch. The pronunciation model also provides important clues about the consistency and inherent difficulty of transcription. Trained on 37,000 words, the pronunciation model achieves a token-level accuracy of 81.5%. Further analysis revealed that the most frequent phoneme-level error made by the pronunciation model was the realization of the inherent vowel (confusing /null/, /o/ and /ɔ/), followed by the confusion of /s/ and /ʃ/, as well as confusions involving semi-vowels. Without these errors the overall token-level accuracy would be 90.5%, which means that about half of the token-level errors are due to these difficult aspects of Bangla orthography and phonology.

In order to make our system available on mobile devices we employ LOUDS-based compression techniques (Fuketa et al., 2013) to encode the pronunciation lexicon into compressed representation of approximately 500 kB that is also fast enough for access.

3.2. Text Normalization

The first stage of text-to-speech synthesis is text normalization. This is responsible for such basic tasks as tokenizing the text, splitting off punctuation, classifying the tokens and deciding how to verbalize non-standard words, i.e. things like numerical expressions, letter sequences, dates, times, measure and currency expressions (Sproat et al., 2001). The Google text normalization system, Kestrel (Ebden and Sproat, 2015), handles several different kinds of linguistic analysis, but here we focus on the tokenization/classification and verbalization phases, which use grammars written in Thrax (Tai et al., 2011).

The stages of analysis in Kestrel are best illustrated by example shown in Figure 3. Consider Bangla input corresponding to English sentence “X is 3cm”. The Bangla Kestrel system will first tokenize this into three tokens, consisting of two ordinary words, and a measure expression. The output of this phase is represented using protocol buffers (Google, 2008), which can be represented serially (and omitting some irrelevant detail) in the “Internal” step of Figure 3.

The two ordinary words just have a name field, but the measure token has a couple of fields one specifying the unit and the other the amount. The output of the tokenizer/classifier is then passed to the verbalizer, which for the non-standard tokens decides how to verbalize them. In the case of the measure expression at hand this would verbalize to the natural language word sequence as shown in the “Output” step of Figure 3. The natural language string corresponding to the measure phrase is shown in bold.

Archive Type	Original (kB)	Compressed (kB)	Ratio
rewrite	117	22	×5.3
tokenize/classify	5,501	1,714	×3.2
verbalize	14,190	3,330	×4.3
total	19,808	5,066	×3.9

Table 1: FST grammars and their disk footprint (in kilobytes).

The Thrax grammar development environment allows the grammar developer to easily write rules that, for example, transmute units, like “centimeter”, into other languages (like Bangla), as well as write rules that describe how to read numbers. See, again (Ebden and Sproat, 2015) for a few examples of rules used in the Google English and Russian TTS systems, and (Tai et al., 2011) and (Sproat, 2015) more generally for how to develop grammars in Thrax.

For our Bangla system we benefited from already having a grammar for verbalizing numbers (used in ASR), and in addition we had a well worked-out set of Kestrel grammars for the related language Hindi. Our target is Bangladesh, where very few people speak Hindi, but Bangla is also spoken in West Bengal in India. We therefore asked an Indian speaker of Bangla to translate all the Hindi content (about 1,500 strings) in our Kestrel grammars into Bangla. The Hindi grammars were then converted using the Bangla translations. Inevitably some tweaking of the result was required. To date the grammars have required six changes to fix problems, and we are continuing to fix text normalization issues as they arise. Given this relatively small number of problems, we believe that bootstrapping a system from a closely related language can be a reasonable approach if one is short of engineering resources to devote to the new language.

The various components of the normalization system are efficiently represented in our system as archives of finite state transducers (FSTs) (Allauzen et al., 2007). There are three main FST archives: the *rewrite grammar* handles the basic rewriting of the incoming text and necessary unicode normalization, the *tokenizer and classifier* grammar is responsible for text tokenization and detection of critical verbalization categories. Finally the *verbalization grammar* converts main verbalization categories into natural language text (Ebden and Sproat, 2015). In the final system each grammar archive is losslessly compressed. The sizes of various Thrax FSTs before and after compression (and the corresponding compression ratios) are given in Table 1.

The Bangla Kestrel grammars will be released along with the voice data. Also, in order for these to be useful, have developed a lightweight version of Kestrel called Sparrowhawk. This has been released as open-source and is in the process of being integrated with Festival.

3.3. Synthesizer

The synthesis stage consists of two steps: First, a sentence is decomposed into corresponding linguistic parameters and acoustic model is used to predict a sequence of optimal acoustic parameters that correspond to linguistic ones. Second, the signal processing component, a vocoder, is used to reconstruct speech from the acoustic parameters (Kawahara et al., 2008). In our system we use the state-of-the-art Vocaine algorithm (Agiomyrgiannakis, 2015) for the vocoding stage.

We have explored two acoustic modeling approaches. It is important to note in both approaches that we train all the speakers together and that the statistical nature of the acoustic modeling has the effect of averaging out the differences between the speakers in the original dataset. While the resulting acoustic parameters do not represent any particular person they can still nevertheless be used to reconstruct naturally sounding speech.

The first approach uses Hidden Markov Models (HMMs), and is a well-established parametric synthesis technique (Yoshimura et al., 1999) In this approach we model the conditional distribution of an acoustic feature sequence given a linguistic feature sequence using HMMs.

One of the main limitations of HMMs is the frame independence assumption: HMM models typically assume that each frame is sampled independently despite concrete phonetic evidence for strong correlations between consecutive frames in human speech. One promising alternative approach that provides an elegant way to model the correlation between neighboring frames is Recurrent Neural Networks (RNNs) originally proposed in (Tuerk and Robinson, 1993). RNNs can also use all the available input features to predict output features at any given frame. In RNN-based approaches a neural network acoustic model is trained to map the input linguistic parameters to output acoustic parameters. In our work we use Long Short Term Memory (LSTM) architecture that has excellent properties for modeling the temporal variation in acoustic parameters and especially long-term dependencies between them (Fan et al., 2014; Zen and Sak, 2015). LSTM models can be quite compact, making them particularly suitable for deployment on mobile devices.

4. Experiments

4.1. Experimental Setup

We experimented with a multi-speaker Bangla corpus totaling 1,891 utterances (waveforms and corresponding transcriptions) from five speakers selected during crowdsourcing process described in Section 2.. The script contains total of 3,681 unique Bangla words which are covered by 40 monophones from Bangla phonology given in Section 3.1.. Phone-level alignments between the acoustic data and its corresponding transcriptions have been generated using HMM-based aligner bootstrapped on the same corpus.

In order to account for phonemic effects such coarticulation the monophones were expanded using the full linguistic context. In particular, for each phoneme in an utterance we take into account its left and right neighbors, stress information, position in a syllable, distinctive features and so on, resulting in 271 distinct contexts. Expanding monophones in this fashion resulted in 21,917 unique full-context models to estimate.

The speech data was downsampled from 48 kHz to 22 kHz, then 40 mel-cepstral coefficients (Fukada et al., 1992), logarithmic fundamental frequency (log F0) values, and 5-band aperiodicities (0–1, 1–2, 2–4, 4–6, 6–8 kHz) (Zen et al., 2007) were extracted every 5 ms. The output features of LSTM-RNNs were phoneme-level durations. The output features of the acoustic LSTM-RNNs were acoustic features consisting of 40 mel-cepstral coefficients, log F0 value, and band 5 aperiodicity. To model log F0 sequences, the continuous F0 with explicit voicing modeling approach (Yu and Young, 2011) was used; voiced/unvoiced binary value was added to the output features and log F0 values in unvoiced frames were interpolated.

We built three parametric speech synthesis systems. The first configuration is an HMM system, which fits well on a mobile device (Gutkin et al., 2010). This system is essentially similar to the one described in (Zen et al., 2007). We also build two LSTM-RNN acoustic models that are essentially the same apart from the number of the input features. The LSTM-RNN configuration with fewer (270) features is slightly smaller, portable (we excluded one feature that is resource-intensive to compute) and fast enough to run on a modern mobile device. In addition, for the embedded configuration we use audio equalizer to boost the audio volumes on the device. No dynamic range compression is employed for this configuration. Further details of LSTM-RNN configurations are given in (Zen and Sak, 2015). For all the configurations, at synthesis time, predicted acoustic features were converted to speech using the Vocaine vocoder (Agiomyrgiannakis, 2015).

To subjectively evaluate the performance of the above configurations we conducted a mean opinion score (MOS) tests. We used 100 sentences not included in the training data for evaluation. Each subject was required to evaluate a maximum of 100 stimuli in the MOS test. Each item was required to have at least 8 ratings. The subjects used headphones. In the MOS tests, after listening to a stimulus, the subjects were asked to rate the naturalness of the stimulus in a 5-scale score (1: Bad, 2: Poor, 3: Fair, 4: Good, 5: Excellent). 13 native Bangladeshi Bangla speakers participated in the experiment. Each participant had an average of minute and a half to rate each stimuli.

4.2. Results and Discussion

The results of MOS evaluations are shown in Table 2. In addition to regular MOS estimate we also report robust MOS estimate which is a mean opinion score computed using trimmed means (smallest and largest value are removed before computing a mean response for each stimuli).

The MOS scores reported in Table 2 indicate that the three multi-speaker configurations are acceptable to the evaluators both in terms of naturalness and intelligibility – all the scores centering around the median between “Fair” and “Good”. The embeded LSTM-RNN configuration is preferred over server LSTM-RNNs. Since the number of input features for both models only differs by one, we hypothesize that the quality difference is due to the use of an audio

Model Type	5-scale MOS	5-scale Robust MOS
Server LSTM-RNN	3.403±0.098	3.424±0.101
Embedded LSTM-RNN	3.519±0.102	3.526±0.106
HMM	3.430±0.091	3.394±0.102

Table 2: Subjective 5-scale MOS scores: regular (MOS) and trimmed (Robust MOS) estimates for speech samples produced by LSTM-RNN and HMM configurations.

Model Type	5-scale MOS	5-scale Robust MOS
Server LSTM-RNN	2.879±0.107	2.871±0.110
Embedded LSTM-RNN	3.406±0.092	3.428±0.096
HMM	3.691±0.104	3.682±0.113

Table 3: West Bengali subjective 5-scale MOS scores: regular (MOS) and trimmed (Robust MOS) estimates for speech samples produced by LSTM-RNN and HMM configurations.

equalization post-processing step which is employed in the embedded LSTM-RNN system.

Interestingly enough, the HMM system did reasonably well: according to regular MOS score it is second behind the embedded LSTM-RNN. According to the robust MOS scores, the HMM system comes out worst out of the three systems but it is not very far behind the server LSTM. The difference in robust MOS scores between the two systems is 0.03, which is not very significant. We hypothesize that this is due to the size of the training corpus – HMM configuration may generalize reasonably well on a small dataset, whereas LSTM-RNNs may struggle with a small amount of data because there are too many parameters to estimate. In order to verify this hypothesis we performed an experiment described in Section 4.2.1. below.

Following the subjective listening tests, the native speakers used the system in real-life scenarios (e.g., as part of machine translation). Out of approximately 25 bugs reported most of them were pronunciation errors due to the errors in lexicon transcription (or missing pronunciations) or text normalization issues. No reported problems are related to the actual quality of acoustic models.

4.2.1. West Bengali Experiment

Bangla is also spoken in West Bengal in India. As part of our Bangla data collection efforts we assembled a corpus representing West Bengali dialect of Bangla using the same crowd-sourcing techniques (described in Section 2.). This dataset consists of 1,366 utterances recorded by 10 native Bangla speakers from West Bengal. Phonological and lexical configuration is mostly shared with the setup for Bangladesh.

Because this corpus is smaller (by 525 utterances) than the Bangladeshi corpus (described in Section 4.1.) we are interested to know how our parametric models fare on smaller corpora. To this end, we built three parametric synthesis systems (two of them based on LSTM-RNNs and one on HMM acoustic models). We then performed an MOS evaluation of these three configurations. The rater pool consists of 9 native speakers of Bangla from West Bengal. We used 100 sentences not included in the training data for evaluation. These are the same sentences that we used in evaluating the Bangladeshi Bangla dialect. Each subject was required to evaluate a maximum of 100 stimuli in the MOS test. Each item was required to have at least 8 ratings. The results of these experiments scores are shown in Table 3. Similar to Bangladeshi Bangla (Table 2) we report both MOS and trimmed (robust) MOS scores.

The best MOS score of 3.691±0.104 (corresponding robust MOS of 3.682±0.113) is displayed by the HMM system which seems to confirm our initial hypothesis that both LSTM-RNN configurations have too many parameters to be estimated reliably on this smaller dataset. Since the rater pool for West Bengali Bangla is different from the rater pool we used for Bangladeshi Bangla it is not easy to establish a correlation between different configurations used in the experiments. What is clear, however, is that there is definitely a room for improving the performance of LSTM-RNN configurations by, for example, performing regularization (Zaremba et al., 2014; Sak et al., 2014) or simply by decreasing the dimensionality of the input feature space.

5. Conclusion and Future Work

We described the process of constructing a multi-speaker acoustic database for Bangladeshi dialect of Bangla by the means of crowdsourcing. This database is used to bootstrap statistical parametric speech synthesis system that scores reasonably well in terms naturalness and intelligibility according to mean opinion score (MOS) criteria. We belive that the proposed approach will allow us to scale better to further under-resourced languages. While the results of our experiments are encouraging, there is still further research required into improving the scalability of the linguistic components: phonological definitions, lexica and text normalization. We would like to focus on this line of research next.

6. Acknowledgements

The authors would like to thank Khan Salam and Hyun-Jeong Choe for their help with this project.

7. Bibliographical References

Agiomyrgiannakis, Y. (2015). VOCAINE the vocoder and applications in speech synthesis. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4230–4234.

Ainsley, S., Ha, L., Jansche, M., Kim, A., and Nanzawa, M. (2011). A Web-Based Tool for Developing Multilingual Pronunciation Lexicons. In *INTERSPEECH*, pages 3331–3332.

Alam, F., Habib, S., Sultana, D. A., and Khan, M. (2010). Development of annotated Bangla speech corpora. In *Spoken Language Technologies for Under-resourced Languages (SLTU'10)*, volume 1, pages 35–41.

Allauzen, C., Riley, M., Schalkwyk, J., Skut, W., and Mohri, M. (2007). OpenFst: A general and efficient weighted finite-state transducer library. In *Proceedings*

of the Twelfth International Conference on Implementation and Application of Automata, (CIAA 2007), volume 4783 of *Lecture Nnotes in Computer Science*, pages 11–23, Prague. Springer.

Ebden, P. and Sproat, R. (2015). The Kestrel TTS text normalization system. *Natural Language Engineering*, 21(03):333–353.

Fan, Y., Qian, Y., Xie, F., and Soong, F. K. (2014). TTS synthesis with bidirectional LSTM based recurrent neural networks. In *Proc. Interspeech*, pages 1964–1968.

Fukada, T., Tokuda, K., Kobayashi, T., and Imai, S. (1992). An adaptive algorithm for mel-cepstral analysis of speech. In *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*, volume 1, pages 137–140. IEEE.

Fuketa, M., Tamai, T., Morita, K., and Aoe, J.-i. (2013). Effectiveness of an implementation method for retrieving similar strings by trie structures. *International Journal of Computer Applications in Technology*, 48(2):130–135.

Gales, M. and Young, S. (2008). The application of hidden Markov models in speech recognition. *Foundations and trends in signal processing*, 1(3):195–304.

Google. (2008). Protocol Buffers. Google's Data Interchange Format. `http://code.google.com/apis/protocolbuffers`.

Gutkin, A., Gonzalvo, X., Breuer, S., and Taylor, P. (2010). Quantized HMMs for low footprint text-to-speech synthesis. In *INTERSPEECH*, pages 837–840.

Habib, S. M., Alam, F., Sultana, R., Chowdhur, S. A., and Khan, M. (2011). Phonetically balanced Bangla speech corpus. In *Proc. Conference on Human Language Technology for Development*, pages 87–93, May.

Hughes, T., Nakajima, K., Ha, L., Vasu, A., Moreno, P. J., and LeBeau, M. (2010). Building transcribed speech corpora quickly and cheaply for many languages. In *INTERSPEECH*, pages 1914–1917.

Hunt, A. J. and Black, A. W. (1996). Unit selection in a concatenative speech synthesis system using a large speech database. In *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*, volume 1, pages 373–376. IEEE.

Jansche, M. (2014). Computer-Aided Quality Assurance of an Icelandic Pronunciation Dictionary. In Nicoletta Calzolari, et al., editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 2111–2114, Reykjavik, Iceland, May. European Language Resources Association (ELRA). ACL Anthology Identifier: L14-1299.

Kawahara, H., Morise, M., Takahashi, T., Nisimura, R., Irino, T., and Banno, H. (2008). TANDEM-STRAIGHT: A temporally stable power spectral representation for periodic signals and applications to interference-free spectrum, F0, and aperiodicity estimation. In *ICASSP 2008. IEEE International Conference on*, pages 3933–3936. IEEE.

Prahallad, K., Elluru, N. K., Keri, V., Rajendran, S., and Black, A. W. (2012). The IIIT-H Indic Speech Databases. In *INTERSPEECH*.

Sak, H., Vinyals, O., Heigold, G., Senior, A., McDermott, E., Monga, R., and Mao, M. (2014). Sequence discriminative distributed training of long short-term memory recurrent neural networks. In *Interspeech*.

Sitaram, S., Palkar, S., Chen, Y.-N., Parlikar, A., and Black, A. W. (2013). Bootstrapping text-to-speech for speech processing in languages without an orthography. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 7992–7996. IEEE.

Sproat, R., Black, A. W., Chen, S., Kumar, S., Ostendorf, M., and Richards, C. (2001). Normalization of non-standard words. *Computer Speech & Language*, 15(3):287–333.

Sproat, R. (2015). OpenGrm Thrax Grammar Compiler. `http://www.openfst.org/twiki/bin/view/GRM/ThraxQuickTour`.

Tai, T., Skut, W., and Sproat, R. (2011). Thrax: An open source grammar compiler built on OpenFst. In *IEEE Automatic Speech Recognition and Understanding Workshop*.

Tuerk, C. and Robinson, T. (1993). Speech synthesis using artificial neural networks trained on cepstral coefficients. In *EUROSPEECH*, pages 1713–1716.

Ud Dowla Khan, S. (2010). Bengali (Bangladeshi Standard). *Journal of the International Phonetic Association*, 40(2):221–225.

Yoshimura, T., Tokuda, K., Masuko, T., Kobayashi, T., and Kitamura, T. (1999). Simultaneous modeling of spectrum, pitch and duration in HMM-based speech synthesis. pages 2347–2350.

Yu, K. and Young, S. (2011). Continuous f0 modeling for hmm based statistical parametric speech synthesis. *Audio, Speech, and Language Processing, IEEE Transactions on*, 19(5):1071–1079.

Zaremba, W., Sutskever, I., and Vinyals, O. (2014). Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*.

Zen, H. and Sak, H. (2015). Unidirectional long short-term memory recurrent neural network with recurrent output layer for low-latency speech synthesis. In *Acoustics, Speech and Signal Processing*.

Zen, H., Toda, T., Nakamura, M., and Tokuda, K. (2007). Details of the Nitech HMM-based speech synthesis system for the Blizzard Challenge 2005. *IEICE transactions on information and systems*, 90(1):325–333.

Zen, H., Tokuda, K., and Black, A. W. (2009). Statistical parametric speech synthesis. *Speech Communication*, 51(11):1039–1064.