The NTUT Blizzard Challenge 2013 Entry

Yuan-Fu Liao and Jiun-Yan Pan

Department of Electronic Engineering, National Taipei University of Technology, Taipei, Taiwan
yfliao@ntut.edu.tw

Abstract

This paper describes our HMM-based speech synthesis system (HTS) [1] submitted to Blizzard Challenge 2013 [2]. The focus of this entry is to build a TTS without using any provided information and speedup the training procedures by parallel processing. In this system, the input text is tagged by Stanford parser [3] and transformed into phone sequences by Flite's letter to sound module [4]. Then all utterances are force-aligned using a phone recognizer trained using TIMIT [5] corpus. To consider the relationship between neighboring sentences, the linguistic features beyond sentence level are extracted including the (1) number and forward and backward positions of sentences in a paragraph and (2) punctuation marks (PMs) of current and surrounding sentences. Moreover, deterministic annealing expectation and maximization (DAEM) [6] and minimum generation error (MGE) [7] criterions are used to initialize and fine-tune the HTS models, respectively.

Index Terms: speech synthesis, HMM, HTS, audiobook

1 Introduction

This paper describes our HTS-based speech synthesis system submitted to Blizzard Challenge 2013, the open evaluation that compares the performance of different TTS systems with a common speech database

Although, this is not our first English TTS, it is the first one that was built all by ourselves without using any other provided information (except utterance segmentation). In our sytem, many open source toolkits including Stanford parser (for part-of-speech (POS) tagging), Flite's letter to sound module (transform an sentence into phone sequences), HTK (mono-phone recognizer) and HTS (for voice building) were adopted.

Moreover, since the whole voice building procedure is very time-consuming, some HTS and HTK commands were executed in parallel to take the advantage of modem multi-core CPUs training environment including (1) HCopy for feature extraction, (2) HHEd for state clustering, (3) HERest for context-dependent model training, and (4) HVite and HSMMAlign for forced-alignment. The parallel processing mechanism is implemented using modified Perl scripts (Perl's multi-threading feature) and a job queuing system (Sun Grid Engine (SGE)).

In the following sections, the linguistic features extraction frontend, the voice building backend and finally the evaluation results will be described in more detail.

2 Linguistic Cue Extraction Frontend

Figure 1 shows the block diagram of the linguistic cue extraction front-end. It has three major modules including (1) a parser (POS tagging), (2) a letter to sound and (3) a phone aligner. This frontend produces label files and feeds them to the HTS-based voice building backend.

In the following sub-sections, some sub-modules of the frontend and the question set for state clustering will be briefly introduced.

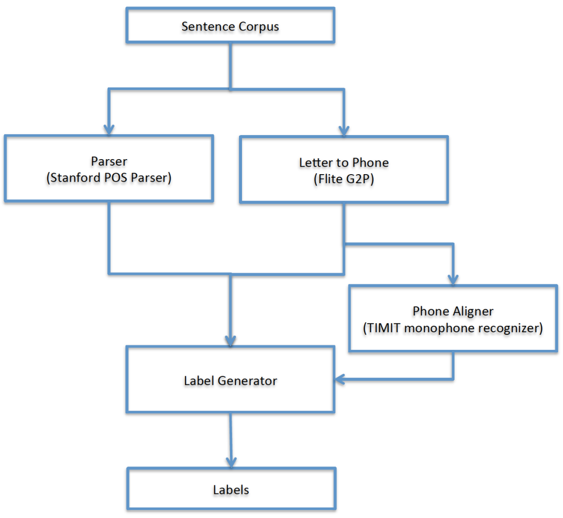

Figure 1: The block diagram of the linguistic cues extraction frontend.

2.1 POS tagging

Stanford NLP parser was adopted here to generate the POS tag of each word in a sentence. The parser is a Java implementation of probabilistic context-free grammars (PCFGs) [8] parser that outputs in total 36 different POS tags. The list of the POSs is as follows:

- "CC","CD","DT","EX","FW","IN","JJ","JJR","JJS","MD","NN","NNP","NNPS","NNS","PDT","POS","PRP", "PRP$","RB","RBR","RBS","RP","SYM","TO","UH", "VB","VBD","VBG","VBN","VBP","VBZ","WDT"," WP","WP$", "WRB", "PM"

The detail explanation of the POS categories could be found in [3] and a typical parsering result is shown as follows:

- She had two sisters to be benefited by her elevation;
- She/PRP had/VBD two/CD sisters/NNS to/TO be/VB benefited/VBN by/IN her/PRP$ elevation/NN ;/PM

Moreover, PMs were further separated into 8 different tags including ".", ",", ":", "?", "!", "-", ".--" and ".." in order to keep the context relationship between neighboring clauses and sentences .

2.2 Letter to Sound

Flite's letter to sound module was adopted here to transform a sentence into phone sequence. Following the Flite's design, 39 phonemes defined in ARPAbet were chosen as the basic units. The list of the phonemes is as follows:

- "aa","ae","ao","aw","ax","ay","b","ch","d","dh","eh","er","ey","f","g","hh","ih","iy","jh","k","l","m","n","ng","ow","oy","p","r","s","sh","t","th","uh","uw","v","w","y","z", "zh"

In this procedure, each word was transformed by first looking a modified Carnegie Mellon University (CMU) dictionary [9] (as distributed as part of Festival). Then decision-tree-based letter-to-sound rules were applied for those words that could not be found in the lexicon. Moreover, the CMU lexicon was pruned by removing those words which the letter to sound rule models get correct.

However, due to some programming issues, in our system, no stress marks were produced and utilized in our TTS system.

2.3 Forced-Alignment

Although, context-dependent tri-phone models are usually used to improve automatic speech recognition's (ASR's) performance, for the end of force-alignment, mono-phone models may be already enough. Therefore, in order to prepare the necessary segmentation information for voice building backend, an English mono-phoneme recognizer trained using the TIMIT corpus was applied.

Following the letter to sound module, 39 phonemes were chosen as the basic units. Each phone was represented as a three-state (left-to-right) hidden Markov model (HMM). And 39 dimensional mel-frequency cepstral coefficients (MFCC) features (13-dimension MFCCs and their first and second order derivatives) were used as the spectral parameters (with 25 ms frame window and 10-ms frame shift). The numbers of mixtures were iteratively increased to 32 and 64 for all phones and "sil" model, respectively.

Moreover, in order to compensate speaker and channel mismatch between training and test phases, mean subtraction, variance normalization and ARMA filtering (MVA) [10] frontend was performed on a per-utterance basis.

The following table shows the performance of the English recognizer on the TIMIT test set.

Correction	Substation	Deletion	Insertion	Error
76.31	19.59	4.10	13.07	36.76

Table 1: Performance of the mono-phone recognizer trained and tested on TIMIT corpus.

2.4 Question Set for Clustering

The question set used for clustering all the context-dependent phones is composed of 3 layers as listed in Table 1. It is worth noting that for audiobook tasks, linguistic features beyond sentence level are added including (1) PMs of current and surrounding sentences and (2) the number and forward and backward positions of sentences in a paragraph. Moreover, in our system, 8 different PM groups including period, comma, question exclamation and colon, etc., are considered.

Layer	Question
Phone	the names and types of current and surrounding phones (5-gram); the number and forward and backward position of a phone in a word
Word	the part-of-speech (POS) of current and surrounding words; the number and forward and backward position of a word in a sentence
Sentence	the punctuation mark (PM) of current and surrounding sentences; the number and forward and backward position of a sentence in a paragraph

Table 2: Hierarchical structure of linguistic cues/questions for decision tree-based context-dependent phone model clustering.

3 Voice Building Settings

HTS version 2.2 was adopted to build the voice. In this section, the voice building settings and procedures including (1) the audiobook database, (2) speech signal representation, (3) training procedure and (4) parallel training procedure are briefly given in the following sub-sections.

3.1 Audiobook Database

Due to resource constraints, only English task 2 (2013-EH2) is considered in this entry. The goal is to build a voice from provided segmented audio for the books Black Beauty and Mansfield Park.

These two audiobook data is kindly provided by The Voice Factory, from a single female speaker, provided as approximately 19 hours of non-compressed wav files. The wav files have been segmented into sentences and aligned with the text by Lessac Technologies, Inc.

No external data is used for building our system. And the corpus have been pruned a little bit to avoid some label file generation problems (due to some programming issues).

3.2 Speech Representation

34-order mel-generalized cepstrum (MGC) [11] and fundamental frequency, F0 was extracted using A Robust Algorithm for Pitch Tracking (RAPT) [12] algorithm as the spectral and excitation parameters (with 5ms frame shift). Besides, MGCs and F0 features and their first and second order derivatives were generated to form a 105-dimentional feature vector for each speech frame.

3.3 Training Procedures

The voice building steps are showed in Fig. 3. Two advanced algorithm, DAEM and MGE training algorithm supported by HTS version 2.2 are utilized in order to build a better voice. The numbers of iterations for DAEM and MGE are experimentally set to 10 and 50, respectively.

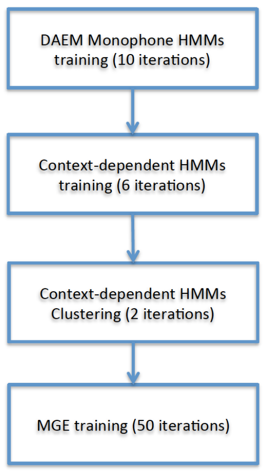
DAEM Monophone HMMs training (10 iterations)
Context-dependent HMMs training (6 iterations)
Context-dependent HMMs Clustering (2 iterations)
MGE training (50 iterations)

Figure 2: The block diagram of the HTS training procedure

3.4 Parallel Processing

The whole training procedure is very time-consuming; therefore, some HTK and HTS commands were executed in parallel to take the advantage of modem multi-core CPUs training environment. They are (1) HCopy for feature extraction, (2) HHEd for clustering, (3) HERest for context-dependent model training, HVite and HSMMAlign for forced-alignment.

Parallel processing of HCopy, HERest, HVite and HSMMAlign commands are done by executing modified Perl scripts in a job queuing system (Sun Grid Engine, SGE is adopted here) to:

- Divide the list of data files
- Send the divided jobs to the processors
- Check for job completion
- Combine the results

The detail explanation of the integration of the Perl script and job queuing mechanism could be found in [13].

On the other hand, for HHEd procedure, the “JM” command of HHEd was used to modify the standard training Perl script in order to run state clustering in parallel (using Perl’s multi-threading features) [14]. In the end, the computation time of HHEd could be greatly reduced to about 1/5 (for 5-state phone models).

It is also worthy noting that for the MGE procedure, the HTS command “HMgeTool” already could fully utilize the power of multi-core CPUs.

3.5 Computation

The whole system was run in a Linux machine equipped with a 3.2 GHz Intel Core i7 2930K, 6-core/12 threading CPU and 64 GB memory. The job queuing system implemented using Sun Grid Engine, was configured with 12 slots and a task was divided into 30 sub-jobs to fully occupy all CPU cores at the same time.

During the parallel processing procedure, HTS ate more than 50 GB memory. Especially The most memory consumption procedures are the state clustering (HHEd) and full-context model re-estimation (HERest). Finally, the total required time to finish all procedures is about 3 days.

4 Evaluation Results

All participants of 2013-EH2 task were asked to synthesize 2409 utterances including 209 book paragraphs, 100 address, 200 book, 100 novel, 500 reportorial and 900 semantically unpredictable sentences (SUS) sentences. And, several hundred online-volunteers, speech experts and paid listeners were involved in the listening test.

Figure 3 and 4 shows similarity and naturalness evaluation results of all participants of 2013-EH2 task. The average mean opinion scores (MOSs) of our entry were 2 and 1 (marked with symbol “G”) for similarity and naturalness, respectively. These results show our system is not good enough, especially for naturalness measurement.

These may due to the lack of stressed syllable cues in our letter to sound processing. Besides, there is also no prosodic model in our system, especially, the pause durations between neighboring sentences were not treated differently. This may further degrade the naturalness of the synthesized voice for audiobook applications.

Another possible room for improvement is to use a larger corpus in the future, for example the Wall Street Journal (WSJ) corpus [15-16], to train a better mono-phone recognizer for more precise forced-alignment. Moreover, pronunciation variation should also be well considered during forced-alignment. Because, it was found that DAEM algorithm may produce worse initial model than standard HInit command.

Finally, advanced speech representation other than MGCs should be adopted, such as STRAIGHT [17] features, to improve the similarity of our synthesized speech.

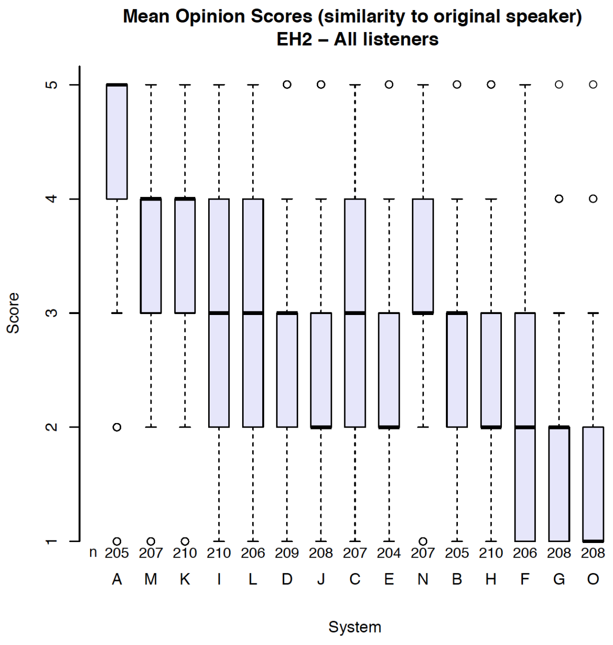

Figure 3. Similarity evaluation results (mean opinion scores (MOSs) reported by all listeners) of all participants of 2013-EH2 task.

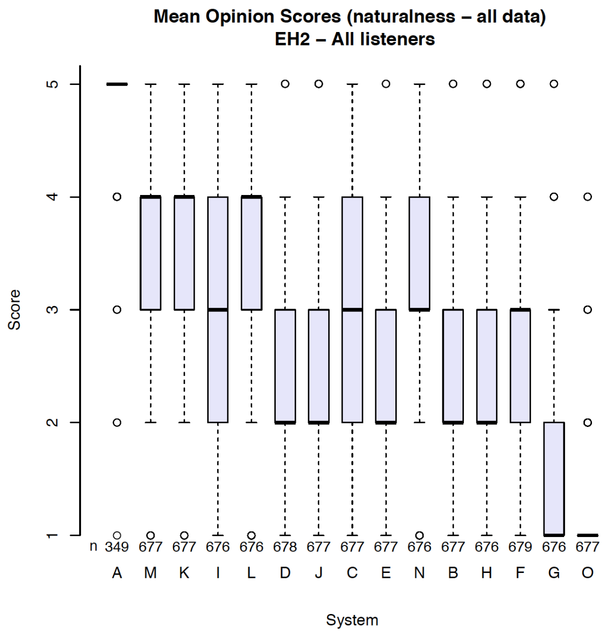

Figure 4. Naturalness evaluation results (mean opinion scores (MOSs) reported by all listeners) of all participants of 2013-EH2 task.

However, Figure 5 shows that paid listeners gave our system a little bit higher similarity score than other listeners. And Figure 6 indicates that all listeners felt that our system performed more natural on the novel utterances synthesis than other sentence styles.

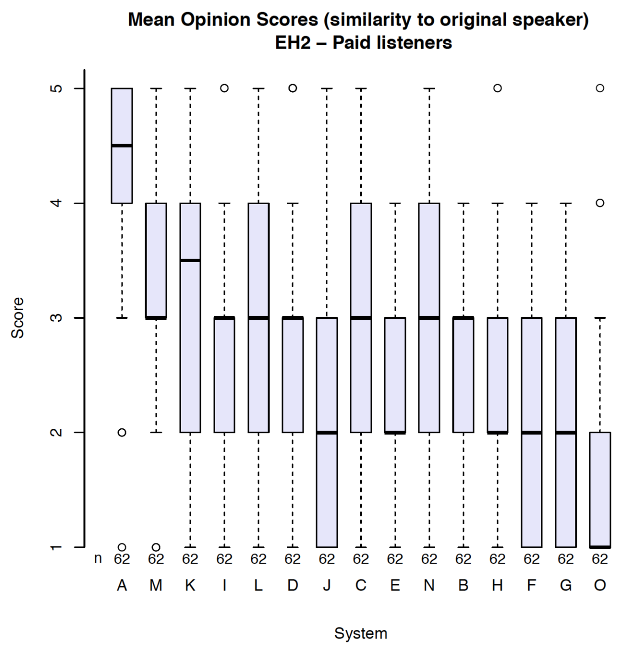

Figure 5. Similarity evaluation results (mean opinion scores (MOSs) reported by paid listeners) of all participants of 2013-EH2 task.

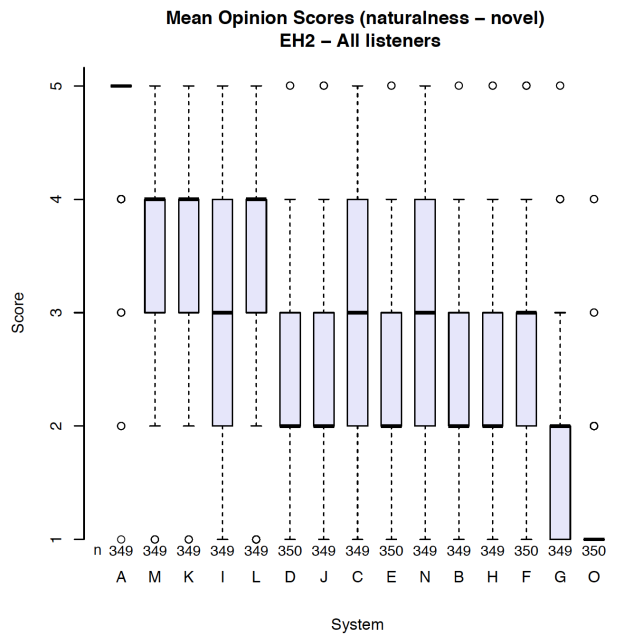

Figure 6. Naturalness evaluation results on novel set (mean opinion scores (MOSs) reported by all listeners) of all participants of 2013-EH2 task.

5 Conclusions and Future Works

This paper describes our HTS-based English TTS submitted to Blizzard Challenge 2013. This system was built all by ourselves with open source toolkits. Moreover, many command were executed in parallel to take the advantages of modem multi-core CPUs. However, there are still many rooms to improve our system. Especially, the letter to sound module need to be modified to output stressed syllable marks and a prosodic model should be incorporated to predict the pause positions and durations.

Acknowledgements

This work was partially supported by the National Science Council, Taiwan, under the projects with contract NSC 101-2221-E-027-129 and 102-2221-E-027-070.

References

[1]. HMM-based Speech Synthesis System, http://hts.sp.nitech.ac.jp/, Aug. 2013

[2]. Blizzard Challenge, http://www.synsig.org/index.php/Blizzard_Challenge, Aug. 2013

[3]. The Stanford Parser: A statistical parser, http://nlp.stanford.edu/software/lex-parser.shtml, Aug. 2012

[4]. Flite (festival-lite) is a small, fast run-time synthesis engine, http://www.festvox.org/flite/, Aug. 2013

[5]. TIMIT Acoustic-Phonetic Continuous Speech Corpus, LDC corpus, http://www.ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC93S1, Aug. 2013

[6]. Y. Itaya, H. Zen, Y. Nankaku, C. Miyajima, K. Tokuda, and T. Kitamura. Deterministic annealing EM algorithm in acoustic modeling for speaker and speech recognition. IEICE Trans. Inf. & Syst., E88-D(3):425–431, 2005.

[7]. Y.-J.Wu, R.-H.Wang, 2006. Minimum Generation Error Training for HMM-Based Speech Synthesis, In: Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing(ICASSP), vol. 1, pp. 889–892.

[8]. Dan Klein and Christopher D. Manning. 2003. Accurate Unlexicalized Parsing. Proceedings of the 41st Meeting of the Association for Computational Linguistics, pp. 423-430.

[9]. The CMU Pronouncing Dictionary, http://www.speech.cs.cmu.edu/cgi-bin/cmudict, Aug. 2013

[10]. C.-P. Chen, K. Filali, and J. Bilmes, MVA Processing of Speech Features, IEEE Transactions on Audio, Speech, and Language Processing, Volume:15 , Issue: 1 , 2007

[11]. Satoshi IMAI,Cepstral analysis synthesis on the mel frequency scale, Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '83, 1983.

[12]. D Talkin, A Robust Algorithm for Pitch Tracking (RAPT), Chapter 15, Speech Coding and Synthesis, Elsevier, 1995.

[13]. Bowon Lee, Parallel Processing of the HTK Commands, http://www.ifp.illinois.edu/~bowonlee/research/cluster/HTK_parallel.htm, Aug. 2013

[14]. Xuchen Yao and Feng Ding, Multi-process supporting patch for HTS-demo_CMU-ARCTIC-SLT, http://hts.sp.nitech.ac.jp/hts-users/spool/2008/msg00415.html, Aug. 2013

[15]. CSR-I (WSJ0) Complete, LDC corpus, http://www.ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC93S6A, Aug., 2013

[16]. CSR-II (WSJ1) Complete, LDC corpus http://www.ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC94S13A, Aug., 2013

[17]. STRAIGHT, a speech analysis, modification and synthesis system, http://www.wakayama-u.ac.jp/~kawahara/STRAIGHTadv/index_e.html, Aug. 2013