A Semi-Automatic Method for Transcription Error Correction for Indian Language TTS Systems

Swetha Tanamala, Jeena J Prakash and Hema A Murthy
Department of Computer Science and Engineering, IIT Madras, India
swethat, jeenaj, hema@cse.iitm.ac.in

***Abstract*—Database for building text-to-speech (TTS) synthesis systems consist of text sentences and the corresponding spoken waveform. Errors in transcription is an important factor that leads to poor synthesis quality. Mismatches between recorded speech and the corresponding text transcriptions occur mainly due to addition, deletion and substitution of words or phrases in the text or speech data. Presently, such mismatches are identified and corrected manually after listening to individual speech waveforms.**

In this work, this process of transcription error correction is made semi-automatic by flagging off the error units at phone level, based on the log-likelihood values of each phone after forced Viterbi alignment. A user interface is developed that highlights the errors. The errors only need to be manually corrected. TTS systems are built for two languages with the complete dataset and with the dataset after removing the data with wrong units. Result of listening test conducted showed that there is a significant improvement in synthesis quality.

I. Introduction

Text-to-speech (TTS) synthesis is the process of converting text input to speech output. Two major state-of-art TTS systems include unit selection synthesis (USS) and hidden Markov model based parametric speech synthesis systems (HTS) [1] [2]. In USS for Indian languages, speech is synthesized by the concatenation of sub-word unit, syllables, based on the best context that suits the context in the sentence to be synthesized. Best context is selected based on linguistic and acoustic similarity. Whereas, in HTS, hidden Markov model (HMM) based phone models are built by initializing and re-estimating context independent mono-phone and context dependent penta-phone HMMs. During synthesis, speech parameter sequences are generated using appropriate context dependent continuous density HMMs of the sequence of phones in the sentence to be synthesized. In any TTS system, appropriate mapping between the text and speech must be captured by the TTS system for producing natural and intelligible speech output.

The training database used for building a TTS system consists of text sentences and the corresponding recorded speech waveforms. The database used for training should be error free because quality of the synthesized output depends on the quality of database used for training the TTS system. Care is taken in selecting text sentences, choosing the speakers, studio etc. Proper instructions are also given to the artist to reduce errors during recording. In spite of these precautions, there exist mismatches in the text transcription and speech data, which affects the quality of speech output. Presently, these transcription mismatch errors are identified and corrected manually by listening to individual speech wave files and verifying with corresponding transcriptions. This is a tedious task which involves several man hours, that is still prone to errors, especially when the data required is large.

In this paper, a semi-automatic approach is proposed for identifying and correcting transcription errors at phone level. Mismatches between the text and wavefiles are accounted for using the likelihood scores obtained for phones after forced alignment. A graphical user interface (GUI) is developed that highlights the units that are in error.

Indian languages belonging to two different families, Indo-Aryan and Dravidian are used for the work. A semi-automatic tool for detection of errors in transcription is developed for twelve Indian languages, eight belonging to Indo-Aryan and four belonging to Dravidian languages. TTS systems are built for two of these languages, one Indo-Aryan and one Dravidian, with the complete dataset used for the study and with the dataset after removing the sentences with transcription errors.

The rest of the paper is organized as follows. Section II explains the data preparation part. Section III discusses the proposed approach for transcription error detection and correction. Section IV explains the GUI developed. Section V details the experiments performed, and briefly describes TTS system built. Section VI concludes the work done.

II. Data preparation

This section describes various procedures involved in data preparation. The details of the dataset used is given in Section II-A. Although Indian languages are more or less phonetic, there are instances where the spoken form and written form do not match. Hindi has schwa deletion at syllable boundaries, while Tamil has no representation in the script for voiced consonants. We briefly discuss letter to sound (LTS) rules in Section II-B. Segmentation of the waveform is discussed in Section II-C.

A. Dataset used

The database used for the work consists of text sentences and corresponding spoken waveforms recorded at sentence level. The data is recorded in a noise free studio environment by professional native speakers of the corresponding language. The utterances are recorded in a noise-free studio environment at a sampling rate of 48KHz, 16 bits per sample. A subset of Indic database is used for the experiments [3]. Twelve Indian languages, eight Indo-Aryan and four Dravidian languages are used for the study. The details of the dataset used is given in Table I. The first eight belong to the Indo-Aryan and next four belong to the Dravidian group of languages. Both male and female data are considered.

TABLE I: Database of languages

Language	Gender	Duration (in hours)	No. of phones
Assamese	male	12.95	52
Bengali	female, male	5.01, 10.05	52
Bodo	female	4.76	44
Gujarathi	female, male	10.33, 10.92	52
Hindi	female, male	5.18, 5.16	60
Manipuri	female, male	10.14, 10.61	50
Odia	female	4.55	52
Rajasthani	male	9.82	58
Kannada	female, male	3.99, 3.43	50
Malayalam	female, male	8.19, 9.70	58
Tamil	male	10.55	43
Telugu	male	4.24	50

B. Letter to sound rules

Letter to sound (LTS) rules are written to convert a grapheme notation of a word into its phonetic representation. Rules are written to break a word into its constituent syllables and phones. A unified parser for Indian languages is used for applying LTS rules [4]. Phones are represented in common label set (CLS) format [6].

C. Segmentation

The speech waveform is segmented at syllable and phone level using hybrid segmentation [7]. It is an automatic segmentation algorithm which uses signal processing cues in tandem with machine learning for segmenting speech data. In this method, HMM based forced Viterbi alignment is performed on speech data to get phone boundaries. This is done after initializing using flat start and then re-estimating the phone boundaries. The phone segmentation obtained using HMMs is inaccurate. Group delay (GD) based processing of short term energy (STE) and spectral flux is used to correct these boundaries at syllable level. Within the syllable boundary, HMM based forced alignment is again performed to get the phone boundaries.

III. Transcription error correction

Good quality TTS system, in terms of naturalness and intelligibility, require good mono-phone and full context HMM models. As discussed in Section I, an important cause for poor phone models is errors in text transcriptions, that is, mismatches between text and speech data in the database used. This, in turn, affects the synthesis quality of TTS systems.

Such mismatches occur in the database because of errors in text or because of errors made by the speaker while reading. This results in insertion, deletion and substitution of words or sub-word units (syllables or phones). Current day TTS system correct such errors manually. The text transcriptions are usually written in native scripts. In this case, language experts must be available for accurate transcription correction. It is not easy to get native listeners for all languages for this purpose. Therefore the text transcriptions in native script is transliterated using any available transliterator first. Individual speech wave files are listened to and verified with the transliterated text transcriptions. Appropriate corrections are then made in the text manually. The quality of some speech wave files will be poor because of bad recording. Such wave files are discarded and the sentences are re-recorded.

Manual identification and correction of transcription errors is a tedious task and also prone to human error. The procedure will be even more tough if non-native listeners are involved in error identification with transliterated text. Also, transliterators are not available for all Indian languages. In this case, either of the following two approaches are followed: (1) use different transliterators for different languages or (2) use same transliterator, but use the one available for similar language. For example, if transllitrator for Rajasthani is not available, use Hindi transliterator for transliterating Rajasthani text. Both of the approaches have its own limitations. Different transliterators may follow different representations for a single sound which make the procedure more prone to error and makes it time consuming especially because the listener is non-native. In the second approach, where the transliteration is approximated, the transliteration will not be an exact match, which in turn affects transcription error detection.

“Indic TTS” is a project for developing TTSes for 13 Indian languages. The details of the languages and databases is given in [3]. The transcription errors in the database of these lanuages are identified and corrected manually, mostly by non-native listeners because native listeners were not available for most of the languages. Initially it took about one man year to do the task completely manually for a single language. The earlier effort on giving syllable boundary cues reduced this effort to a few man months [5] and also improved the consistency in transcription. When the transcription is accurate hybrid segmentation performs very well. The proposed approach is to augment hybrid segmentation to correct errors in text which further reduced the time for transcription error correction.

In this paper, the semiautomatic tool proposed earlier in [5] is replaced with the hybrid segmentation (assuming correct transcriptions). This is augmented with log-likelihood scores at the phone-level, where poor scores are highlighted. For this, the text is converted into a sequence of phones and the speech waveform is segmented at phone level. As discussed in Section II, a unified parser for Indian languages is used for getting sequence of phones from the text, and hybrid segmentation algorithm is used for segmenting the speech wave file at phone level. Log-likelihood scores are calculated for all phones in the database after performing forced Viterbi alignment on phone boundaries during hybrid segmentation. These log-likelihood values are normalized and a threshold is set based on the mean and standard deviation of log-likelihood scores. The phones for which the log-likelihood scores are not within a predefined range (with respect to the mean log likelihood score for each phone) are flagged as wrongly marked units. The flowchart showing the procedure followed for detecting transcription errors is shown in Figure 1.

Phones of Indian languages are represented in common label set (CLS) format, in this work. CLS format is available for 13 Indian languages as part of a common framework for building Indian language TTS systems. In CLS format, same sounds in different languages are represented using a common label. This is developed by exploiting the acoustic similarity of phones in all languages. The notations and rules of mapping is detailed in [6]. A part of CLS is shown in Figure 2. This representation provides an added advantage to non-native people because the mapping is same for similar sounds in all languages. Additionally, it is not required to use different transliterators or approximate transliterations from other similar language transliterators.

IV. Error detection tool

A graphical user interface (GUI) is developed for easy detection and correction of transcription errors, using Java Swing and Awt components on Netbeans Integrated Development Environment (IDE). This error detection tool is similar to the semi-automatic labeling tool "DON Label" [5], previously used for manual correction of syllable boundaries after group delay based segmentation [8]. The tool gives options to the users for selecting language and a threshold on log-likelihood value. Threshold can be chosen by the users based on the degree of tolerance of error acceptable. The tool provides the option to load and play audio (speech) wave files, load label files, select and play specific segments of wave file etc. There is also an option to play the speech file up to marked segment and

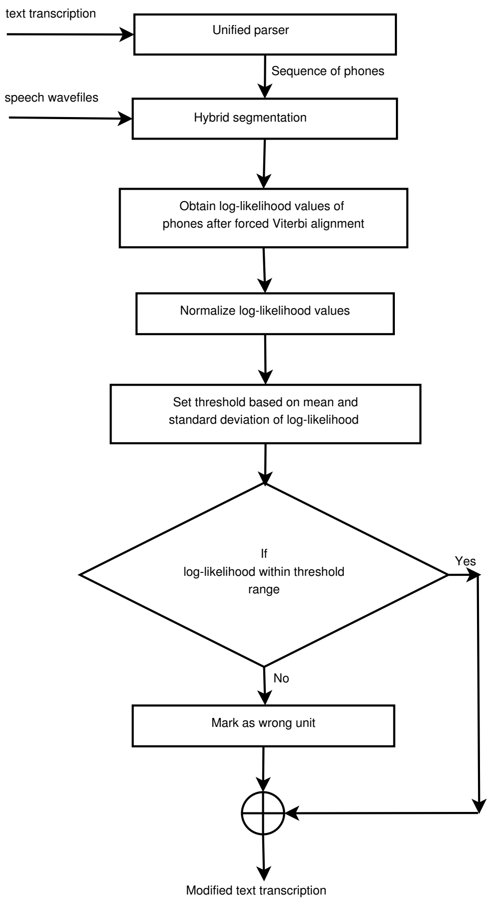
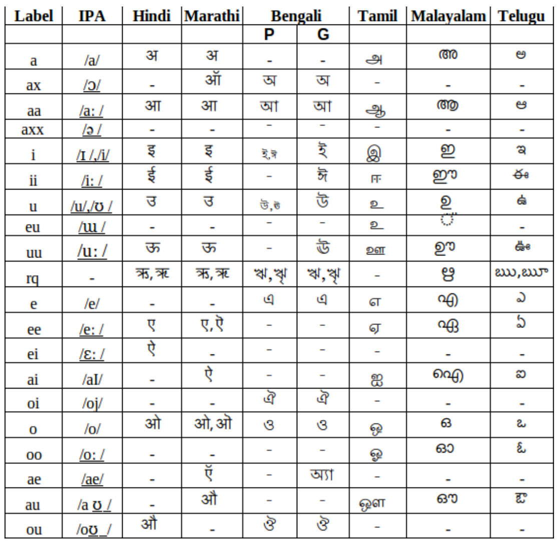

text transcription
Unified parser
Sequence of phones
speech wavefiles
Hybrid segmentation
Obtain log-likelihood values of phones after forced Viterbi alignment
Normalize log-likelihood values
Set threshold based on mean and standard deviation of log-likelihood
If log-likelihood within threshold range
Yes
No
Mark as wrong unit
Modified text transcription

Fig. 1: Flow chart



Label	IPA	Hindi	Marathi	Bengali		Tamil	Malayalam	Telugu
				P	G			
a	/a/	अ	अ	-	-	அ	അ	అ
ax	/ɔ/	-	ऑ	অ	অ	-	-	-
aa	/a: /	आ	आ	আ	আ	ஆ	ആ	ఆ
axx	/ə /	-	-	-	-	-	-	-
i	/ɪ /,/i/	इ	इ	ই,ঈ	ই	இ	ഇ	ఇ
ii	/i: /	ई	ई	-	ঈ	ஈ	ഈ	ఈ
u	/u/,/ʊ /	उ	उ	উ,ঊ	উ	உ	ഉ	ఉ
eu	/ɯ /	-	-	-	-	உ	്	-
uu	/u: /	ऊ	ऊ	-	ঊ	ஊ	ഊ	ఊ
rq	-	ऋ,ॠ	ऋ,ॠ	ঋ,ৠ	ঋ,ৠ	-	ഋ	ఋ,ౠ
e	/e/	-	-	এ	এ	எ	എ	ఎ
ee	/e: /	ए	ए,ऐ	-	-	ஏ	ഏ	ఏ
ei	/ɛ: /	ऐ	-	-	-	-	-	-
ai	/aI/	-	ऐ	-	-	ஐ	ഐ	ఐ
oi	/oj/	-	-	ঐ	ঐ	-	-	-
o	/o/	ओ	ओ,ओ	ও	ও	ஒ	ഒ	ఒ
oo	/o: /	-	-	-	-	ஓ	ഓ	ఓ
ae	/ae/	-	ऍ	-	অ্যা	-	-	-
au	/a ʊ /	-	औ	-	-	ஔ	ഔ	ఔ
ou	/oʊ /	औ	-	ঔ	ঔ	-	-	-

Fig. 2: Common label set mapping

to play the speech file from marked segment to end of the speech file. It also provides the facility to zoom in and zoom out the entire waveform and its corresponding text transcription. There are three different panels that shows wavefiles, spectrogram and the transcriptions. Individual speech wave files and the corresponding label file can be selected by the user. Label files has the sequence of phones in the text transcription with their start and end time stamps obtained after hybrid segmentation. In the panel that displays label file, the phones that has log-likelihood values outside the range of the selected threshold are highlighted with a different color.

The thresholds are chosen such that false alarms are permitted. This is to ensure that no text transcription errors are missed. It is observed that listening to the highlighted phones alone is adequate for correction of transcription errors. Boundaries of such segments can be corrected after listening to these segments and few other boundaries in their vicinity. Occasional false alarms are ignored.

V. Experiments and Results

Log-likelihood values of a phone belonging to a segment of speech is obtained for all phones in the database after forced Viterbi alignment of phones during segmentation. There are two main reasons for poor log-likelihood values of phones: (1) error in transcription and (2) error in segmentation. Such phones are considered as wrong phones.

Threshold is set based on the mean and standard deviation of log-likelihood values of individual phones, as $\mu \pm \mathrm{k}\sigma$. Experiments are performed with different values of k. With lower values of k, more number of units are detected as wrong. But this includes a lot of false alarms also. The number of false alarms decreases with increase in the value of k. But with higher values of k with minimum false alarm, the method fails to detect the phones that are actually wrong. That is, true negatives starts appearing with higher values. The value of k should be chosen in such a way that all wrong phones are detected, but with minimum number of false alarms. And, if there are true negatives, transcription error remains as undetected and hence will not be corrected in the database.

The error detection tool provides the option to select the value of k among 0.1, 0.25, 0.5, 0.75, 1, 1.25 and 1.5. Appropriate value can be chosen based on the acceptable tolerance of error. The percentage of total number of occurrences of phones marked as wrong units for different values of threshold for male and female dataset of all languages is shown in Figures 3 and 4 respectively. It can be seen that for k=0.1 and k=0.25, number of false alarms are high since 70-90% of phones are detected as wrong. High values like k=1.5 detects only less than

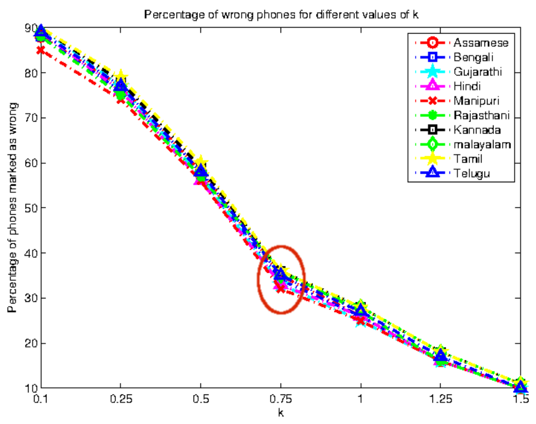


Fig. 3: Percentage of phones detected as wrong phones with different thresholds for male data of all languages. *Red circle in the figure shows the value of k below which no transcription errors are missed.*

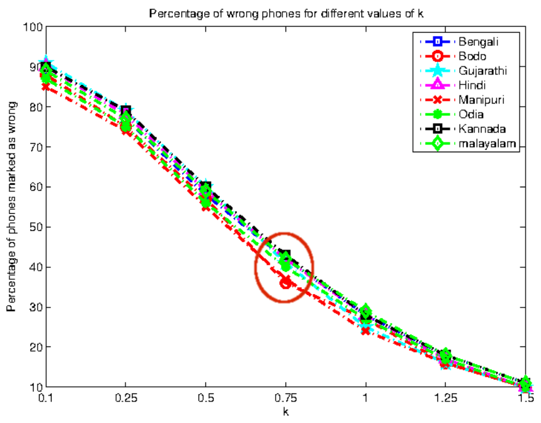


Fig. 4: Percentage of phones detected as wrong phones with different thresholds with female data of all languages. *Red circle in the figure shows the value of k below which no transcription errors are missed.*

10% of phones as wrong. Manual verification also proved that there are many true negatives with this threshold. For k=0.75 around 30-40% and 40-50% of phones are identified as wrong units for male and female dataset respectively. Manual verification of individual files shows that no transcription errors are missed with this threshold. For any value of $k \leq 0.75$, there are no true negatives, but false alarms increases as k decreases. The threshold can be chosen based on the percentage of error acceptable.

Figure 5 shows a segment from a Hindi utterance after highlighting wrong units with a threshold of $\mu \pm 0.75\sigma$. The first, second and third pane respectively shows the transcription, spectrogram and wavefile corresponding to the utterance. In this sentence, the word “जिस तरह ” (*jis tarah*) is repeating twice in the text transcription where

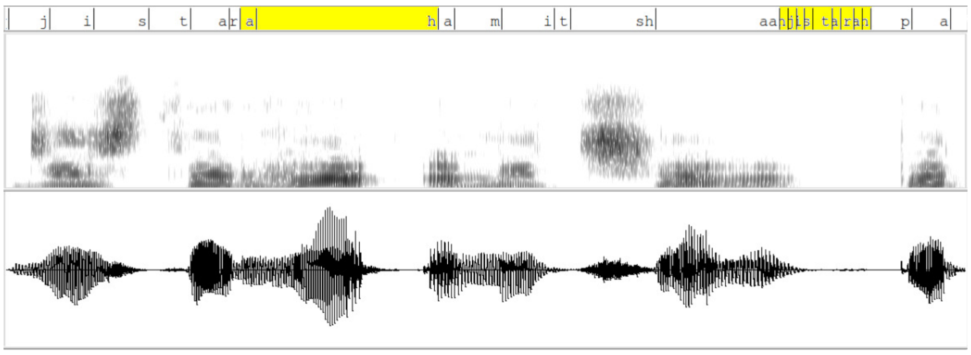


Fig. 5: Segment of Hindi speech in which transcription error is highlighted in the error detection tool

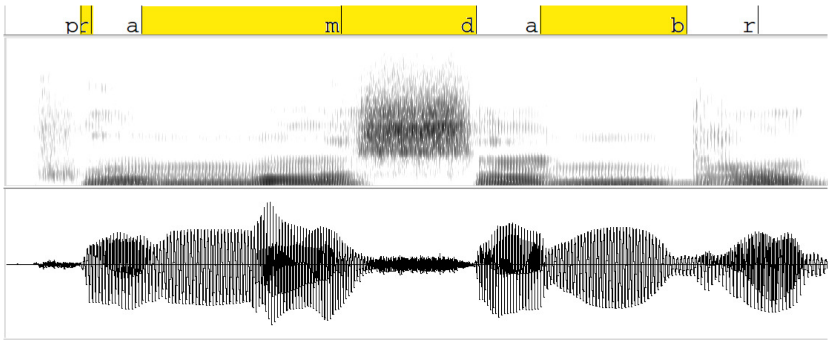


Fig. 6: Segment of Bodo speech in which transcription error is highlighted in the error detection tool

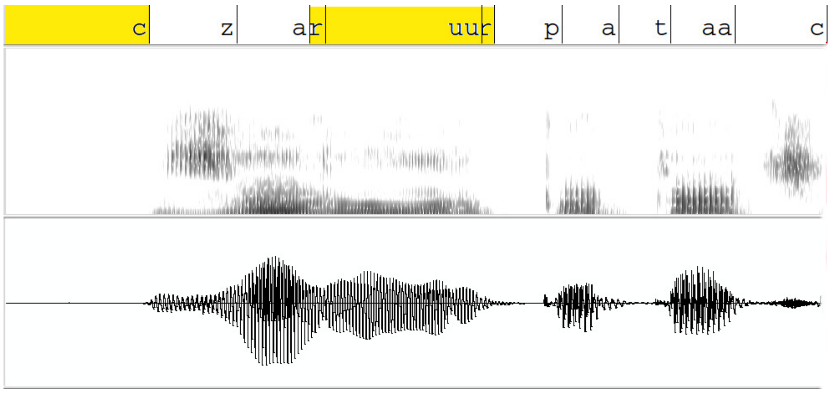


Fig. 7: Segment of Hindi speech in which segmentation error is highlighted in the error detection tool

as it is not repeating in the speech utterance. All phones in the second occurrence of the word are highlighted in yellow color because the log-likelihood value of these phones belonging to the segment of speech is very low because of transcription error. Phones *r, a and h* in the first occurrence of the word are highlighted because of low log-likelihood values due to segmentation error. Figure 6 shows a similar segment with transcription error from a Bodo utterance. In this sentence, the word “प्रमोसंबर” (*pramosambar*) is written as “प्रम द ब्र” (*pram da bra*) in the transcription. The unit *r* is highlighted because of segmentation error, *m* and *b* are highlighted because of missing phones and *d* is highlighted because of substitution of phone *s*. The corrections are made to the text after listening to the highlighted segment.

As discussed before, the wrong units highlighted in the tool includes errors in segmentation also. For example, Figure 7 shows a segment of speech “सोच ज़रूर पता” (*soc zaruur pataa*). The segments corresponding to the phones *c*, *r* and *uu* are highlighted because the boundaries are wrong. Such segmentation errors can be corrected manually by listening to individual segments that are highlighted and moving the boundaries manually.

A. *Text-to-speech (TTS) systems*

Phone based HTS systems are built for two languages Hindi and Telugu, one Indo-Aryan and one Dravidian language with the complete database, that is the database with transcription errors, and with the pruned database. Pruning is done at sentence level for this work. The sentences that have a large number of bad units as indicated by the tool are removed from the corpus for synthesis. By performing pruning at sentence level, it is ensured that only good units are considered while building context independent and context dependent phone models. HTS version 2.3 is used for building the systems.

The HTS systems built are compared by conducting two types of listening tests, degradation mean opinion score (DMOS) and word error rate (WER). DMOS listening test is conducted on 15 sentences with 10 speakers. Listeners are allowed to listen to the sentences only once. Different sentences synthesized using both systems are played randomly to the participants. Participants are asked to rate the systems on a scale of 1-5, 5 being the best and 1 being the worst system. In WER test, participants are asked to listen to and transcribe semantically unpredictable sentences (SUS). SUS sentences are generated using both the systems. The systems are evaluated by calculating WER. WER is calculated based on number of insertions, deletions and substitutions in the transcription. The results of DMOS and WER is shown in Table II and Table III respectively. System 1 refers to the system built with original dataset and system 2 refers to the pruned dataset. It can be seen from the table that the quality improved with the pruned database.

TABLE II: Degradation mean opinion scores (DMOS)

Language	System 1	System2
Hindi	3.08	3.63
Telugu	3.46	3.96

TABLE III: Word error rates (WER) (%)

Language	System 1	System2
Hindi	4	3.4
Telugu	12.5	6.16

VI. Conclusion

This paper proposes a semi-automatic approach for identifying and correcting transcription errors in TTS database at phone level. Wrong units (phones) in the database are identified based on the log-likelihood scores after performing forced Viterbi alignment during segmentation of speech waveforms. The study is carried out for twelve Indian languages. A graphical user interface is developed in which phones that are detected as wrong are highlighted with a different color. Corrections in the transcription are made after listening to only the highlighted units instead of listening to the complete utterance. This reduces the time and effort required, as well as the human prone error possibilities associated with manual error correction considerably.

HTS systems are built with complete dataset and the dataset after pruning off the sentences which has wrong units. These two systems are compared using DMOS and WER listening tests and results shows that the synthesis quality improves with the pruned dataset.

VII. Acknowledgement

The authors would like to thank the Department of Information Technology, Ministry of Communication and Technology, Government of India, for funding the project, "Development of Text-to-Speech synthesis for Indian Languages Phase II", Ref. no. 11(7)/2011HCC(TDIL).

References

[1] Hema A Murthy, et al. *Building Unit Selection Speech Synthesis in Indian Languages: An Initiative by an Indian Consortium.* Proceedings of COCOSDA, Kathmandu, Nepal (2010).

[2] Heiga Z, et al. *The HMM-based speech synthesis system (HTS) version 2.0.* SSW. 2007.

[3] Arun Baby, Anju L Thomas, Nishanthi N L and Hema A Murthy, *Resources for Indian languages*, CBBLR workshop, 19th International Conference on Text, Speech and Dialogue, 2016.

[4] Arun Baby, Nishanthi N L, Anju L Thomas, and Hema A Murthy, *A Unified Parser for Indian Languages*, 19th International Conference on Text, Speech and Dialogue, 2016.

[5] Deivapalan P, et al. *Donlabel: an automatic labeling tool for Indian languages.* Energy 2 (2008): 4.

[6] Ramani B, et al. *A Common Attribute based Unified HTS framework for Speech Synthesis in Indian Languages.*, Proc. of SSW8: 8th ISCA Speech Synthesis Workshop, pp. 291-296, Aug 31 - Sep 2, 2013.

[7] Aswin S Shanmugam , and Hema A Murthy. *A hybrid approach to segmentation of speech using group delay processing and HMM based embedded reestimation.* INTERSPEECH. 2014.

[8] Prasad V K, *Segmentation and Recognition of Continuous Speech.* PhD dissertation, Depart- ment of Computer Science and Engg., Indian Institute of Technology Madras, Chennai, India, May 2002.