A Unified Parser for Developing Indian Language Text to Speech Synthesizers

Arun Baby(✉), Nishanthi N.L., Anju Leela Thomas, and Hema A. Murthy

Computer Science and Engineering, IIT Madras, Chennai, India
{arunbaby,hema}@cse.iitm.ac.in, nlnishanthi@gmail.com,
anjuthomas95@gmail.com

Abstract. This paper describes the design of a language independent parser for text-to-speech synthesis in Indian languages. Indian languages come from 5–6 different language families of the world. Most Indian languages have their own scripts. This makes parsing for text to speech systems for Indian languages a difficult task. In spite of the number of different families which leads to divergence, there is a convergence owing to borrowings across language families. Most importantly Indian languages are more or less phonetic and can be considered to consist broadly of about 35–38 consonants and 15–18 vowels. In this paper, an attempt is made to unify the languages based on this broad list of phones. A common label set is defined to represent the various phones in Indian languages. A uniform parser is designed across all the languages capitalising on the syllable structure of Indian languages. The proposed parser converts UTF-8 text to common label set, applies letter-to-sound rules and generates the corresponding phoneme sequences. The parser is tested against the custom-built parsers for multiple Indian languages. The TTS results show that the accuracy of the phoneme sequences generated by the proposed parser is more accurate than that of language specific parsers.

Keywords: Indian languages · Text-to-speech synthesis (TTS) · Letter to sound (LTS) · Syllable · Common label set · Parser

1 Introduction

India is the second most populous nation with over 1.27 billion people. It has about 22 major languages, written in 13 different scripts, with over 1600 languages/dialects. Further, it is only about 65 percent of this population that is literate, that too primarily in the vernacular. Less than 5 % is English literate thus marginalizing most of the Indian society. Speech interfaces, especially in the vernacular, are enablers in such an environment.

The objective of this paper is to build technology that will enable the building of TTS systems in any Indian language quickly. Hidden Markov model (HMM), a statistical parametric based approach which is found effective in synthesizing speech is employed here [12]. The objective of text to speech (TTS) synthesis

P. Sojka et al. (Eds.): TSD 2016, LNAI 9924, pp. 514–521, 2016.
DOI: 10.1007/978-3-319-45510-5_59

system is to convert an arbitrary input text to its corresponding speech output. Text processing and speech generation are the two major components of a TTS system. The text processing component converts graphemes into a sequence of phonemes while the latter proffers the produced sequence of phonemes to the synthesizer to generate the speech waveform. Determining the appropriate sequence of sounds is very crucial for natural and intelligible speech.

Syllable structures across Indian languages can be similar or vary. Traditional approaches in converting text to speech for a given language make use of language specific parsers. Such approaches use specific rules of a given language and build parsers that are highly customised. This makes the task of creating individual parsers for new languages difficult.

Parsers that work for more than one language focuses on structurally related languages such as English and French or English and German [1]. Bilingual parsers built in Indian multilingual context are also available [9]. This paper introduces a unified parser that can handle Indian languages which are free-word-order and are also morphologically rich. The main challenges are finding the rules for different languages and incorporating the context-sensitive rules. A unified parser is designed that systematically identifies the invariant properties of different Indian languages. The UTF-8 text is converted to a sequence of labels. A common label is first defined for all the languages. Rules that are peculiar to a language are treated as exceptions.

Text to speech synthesis systems built using the common parser show that the parsers are as good as if not better than custom built parsers. Lex and Yacc [4] stands in good stead to build rule-based language parsers as these employ rule-based method for token matching. This paper tries to capture the similarities and resolve the differences in rules across multiple Indian languages so that lexical rules can handle occurrences of all native sentences and pass it to a synthesizer.

Section 2 describes the characteristics of Indian languages. Section 3 gives a brief overview of Letter To Sound (LTS) rules and discusses the design of the common label set. Section 4 discusses the design of the parser. Section 5 details the exceptions in parsing Indian language text. Section 6 discusses experiments and results. Section 7 concludes the work and provides future scope of the work.

2 Characteristics of Indian Languages

Most of the Indian languages can be broadly classified into two language families:

- Indo-Aryan languages
- Dravidian languages

The former is the largest and is spoken mostly in North India while the latter is predominant in South. These classes of languages share some common features and the geographical proximity of the regions where these languages have been spoken, have resulted in significant borrowings [5]. Indian languages are characterized by character set, which is termed as aksharas [7]. These aksharas are the fundamental linguistic units of the writing system in Indian languages [3]. According to the properties of aksharas, syllable boundaries can be marked at vowels at regular intervals for a given sequence of phones. This finding is typically followed in building TTS systems for Indian languages [2].

Indian languages are syllable-timed and a large number of syllables are common across Indian languages [7]. Approximating to the nearest syllable is possible even if the syllable as such is not available [8]. Accounting for the acoustic phonetic properties of different languages, this paper primarily focuses on the generation of phonemes sequences of the form C*VC*, where C is a consonant and V is a vowel.

3 Acoustics and Phonetics

Despite having different scripts there exist a relationship between orthography and sound that is common in the Indian languages. Exploiting this fact, Letter To Sound (LTS) rules and the common label set are introduced.

3.1 LTS

LTS rules are a set of hand-crafted rules that define the relationship between graphemes and phonemes for each language. Most Indian languages consist of about 50 sounds - 15–18 vowels and 35–38 consonants. Appropriateness of using the letter to sound rules is language dependent and requires considerable effort. The rules vary with the context as well and hence the LTS is modified with precise data making it more appropriate and flawless.

3.2 Common Label Set

The acoustic similarity among the same set of phones of different languages suggests the possibility of a compact and common set of labels [10,11]. The common label set is defined using the Latin 1 script. The common label set uses a standard set of labels for speech sounds that are commonly used in Indian languages. The notations of labels and rules for mapping are detailed in [10]. The paper make uses of this label set such that the native script is largely recoverable from the transliteration.

4 Parsers

Each language has its own set of grammar and syllabification rules. Non-phonetic languages like English use Classification and Regression Tree (CART) [6] to predict the phonetic transcription. This approach needs a large dictionary of words and its correct phonetic transcription. Since Indian languages are more or less phonetic in nature, building a rule-based parser is possible.

4.1 Structure of Parser and Its Parsing

The primary task of a parser is to segment the text for TTS systems. However, the parser cannot handle the raw text as it is available in news websites, blogs, documents etc. Standardizing the text input by removing the unwanted characters like special characters and emoticons is essential. Once the text is standardized, next phase is language identification. The first character of the word is taken and compared with the Unicode range to detect the language. Identifying whether the language belongs to Aryan or Dravidian is also vital owing to vast differences in pronunciation. For parsing the word, the sequence of graphemes is mapped to labels in the common label set. Having mapped the input to common label set, the next step is to obtain appropriate pronunciation for each of these labels. Although Indian languages are more or less phonetic, occasionally, the one-to-one correspondence between the spoken and written form is absent. These exceptions are handled by rules detailed in Sect. 5.

5 Parsing Indian Language Text

The main issue with parsing is the identification of vowel deletion points, syllable boundaries and the manner of applying rules. The unified parser uses the following set of rules.

Schwa Deletion Rules: Phonetically, schwa is a short neutral vowel sound */a/* which is associated with a consonant. For Aryan languages, the implicit mid central vowel (schwa), in each consonant of the script, is obligatorily deleted in certain context while uttering. This is known as schwa deletion or Inherent Vowel Suppression (IVS). Identifying which schwas are to be deleted and which are to be retained makes the process of schwa deletion complex. This is obvious for a native speaker, but for machine processing this decision depends on language specific rules. IVS rules are performed on Free Consonants/Semivowels (FCS) in a word. FCS refers to the consonants/semivowels in a word that do not have a vowel sound adjacent to it in written form. Example: In चटचटाहट, the first and third occurrences of ट are FCS whereas the second occurrence is not. Following are the known IVS rules.

1. Characters present in the first position of a word, never undergo IVS. Example: चदा (**c a** dxh aa)
2. Characters in final position always undergo IVS. Example: चदाकर (c a dxh aa k a **r**)
3. No two successive characters undergo IVS. Example: चटचटाहट (c a **tx c a** tx aa h a tx)
4. No two vowels come together.
5. The remaining FCS in a word that are not processed by rules 1 and 2 is processed in left-to-right order. IVS occurs for an FCS if its successor in the word is (i) not the last character of the word or (ii) a vowel other than

$'a'$. Application of this rule leads to erroneous parsing of a subset of words. Example: Application of the rule yields the following output.
पागलपन (p aa g a l p a n)
ताजमहल (t aa j a m h a l)
पागलपन is parsed correctly whereas ताजमहल is not. This single rule would not be able to handle such contradictorily parsed words.
This paper proposes 2 new rules - AB (1) and AB (2) - to solve such parsing problems.

AB (1) Rule: A free semivowel at the second position of a word starting with a vowel never undergoes IVS whereas a free consonant at the second position of a word starting with a vowel always undergoes IVS.

AB (2) Rule: The focus of this rule is on the substring of the word which is not processed by rules 1, 2 and AB (1). The inherent vowel sounds in this substring are named unmarked schwa. The rule proposes lexicographically ordered processing of FCS in this substring. An FCS is processed only if it is preceded by an unmarked schwa and succeeded by a vowel, vowel sound or unmarked schwa in the transliterated form. In this case, the predecessor (unmarked schwa) is deleted. If the successor is an unmarked schwa, it is marked as non-deletable in further iterations (marked schwa). Application of AB rule parses पागलपन and ताजमहल correctly. The process is illustrated in Tables 1 and 2. In Table 2, a^* represents unmarked schwa and â represents marked schwa.

Table 1. Pass 1 – apply rules 1, 2 and AB (1)

Word	Transliterated String	Rule 1	Rule 2	AB(1)
ताजमहल	taa j*a* m*a* h*a* l*a*	NA	taa j*a* m*a* h*a* l	NA
पागलपन	paa g*a* l*a* p*a* n*a*	NA	paa g*a* l*a* p*a* n	NA
अकबर	a k*a* b*a* r*a*	NA	a k*a* b*a* r	a k b*a* r
असफल	a s*a* f*a* l*a*	NA	a s*a* f*a* l	a sa f*a* l

Table 2. Pass 2 – apply AB (2) rule

Substring considered (with unmarked schwa)	Iteration 1			Iteration 2			Iteration 3		
	Character	Process?	Output	Character	Process?	Output	Character	Process?	Output
ja^* ma^* ha^*	j	No	ja^* ma^* ha^*	m	Yes	j mâ ha^*	h	No	j mâ ha^*
ga^* la^* pa^*	g	No	ga^* la^* pa^*	p	Yes	ga l pâ	l	No	ga l pa^*
ba^*	b	No	ba^*	-	-	-	-	-	-
sa^*	s	No	sa^*	-	-	-	-	-	-

Geminate Correction Rules: The term geminate in phonology refers to a long or doubled consonant sound, such as the */kk/* in the Hindi word पक्का that contrasts phonemically with its shorter or singleton counterpart पका. Such contrasts occur frequently in Indian languages. There exist other phonetic cues to geminates besides consonantal duration such as pitch and intensity differences. However, this paper will not focus on the phonological behavior of geminates. Focus is to keep the geminates together, that is, they are always grouped as a syllable, as the sound is distinct. Example: पक्का (p a)(**k k aa**)

Syllable Parsing Rules: Though each sound is mapped to a corresponding label in the common label set, the label set does not handle the implicit /*a*/ sound associated with each consonant of the script. Hence, a separate rule is written to add the /*a*/ sound to the labels of all consonants without a vowel modifier associated with it. Thereafter, schwa deletion is performed for Aryan languages alone. For Dravidian languages schwa deletion rules are not applied. The processed input text is split into a set of sub-syllables, both at vowel and halant positions. These sub-syllables are processed in last to first manner, to ensure that all the consonantal units are suffixed by a vowel. Necessary correction (if required) is done subsequently i.e., if the current unit does not possess a vowel sound, it is appended to the previous unit. For example, ताजमहल is syllabified as (t aa j)(m a)(h a l). This rule is significant in particular for chillaksharas in Malayalam that do not possess an inherent vowel. This rule is also considered while grouping geminates as syllables, as the first occurrence of the consonant does not possess an inherent vowel.

Language-Specific Rules: Only 80 %–95 % accuracy is achieved even after applying all the above rules. This is due to the fact that each language has certain specific rules which cannot be generalized. These language-specific rules are applied during parsing to obtain an accuracy of 95 %–100 %. A few examples of such rules for Tamil are shown in Fig. 1.

Grapheme	Phoneme in CLS	Rule	Phoneme in actual pronunciation	Example
க	k	If previous and next grapheme is a vowel	g	ஆகாயம் (aagayam)
ட	tx	If previous grapheme is ண	dx	வேண்டும் (weenxdxum)
ச	c	If previous grapheme is ஞ	j	பஞ்சம் (panjjam)
ப	p	If previous grapheme is ம	b	குடும்பம் (kudxumbam)
க	k	If previous grapheme is ங	g	திங்கள் (tinggalx)

Fig. 1. Language specific rules for Tamil

Agglutination is the process of combining words that are formed by stringing together morphemes. The combination is carried without changing the morphemes in either spelling or phonetics. Languages that uses the property of agglutination are called agglutinative languages. Unified parser handles even the agglutinative words that are common in Dravidian languages since it employs a rule-based approach.

6 Experiments and Results

Text to speech synthesis systems are built using the language specific parsers and unified parser for 11 Indian languages. Pairwise Comparison (PC) tests are performed by an average of 12 native listeners to evaluate the performance of the unified parser approach. PC tests reveal the effectiveness of the unified parser. As can be seen from Fig. 2, in most cases the unified parser and native parser have the same preference. Occasionally there is preference for the unified parser. This is primarily because a systematic approach to parsing across various languages has been taken. This has resulted in a consistent set of rules.

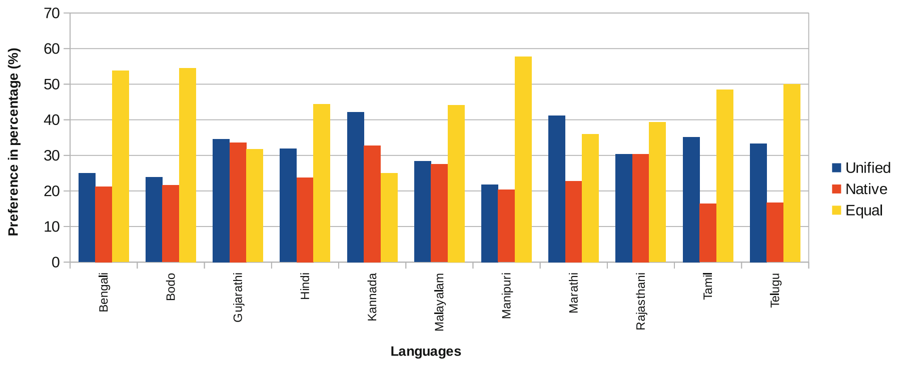

Fig. 2. Pairwise comparison test results

Experimental results show that the unified parser is more robust and accurate than the current systems that require similar supervision. This work is also the first attempt to introduce a unified approach, using the common label set, for parsing Indian languages. Unlike previous systems that require manually-constructed rules, this system requires much less knowledge of the native languages and can be easily scaled to other languages. To build new TTS systems for a new language, the mapping is made to the CLS. Existing language specific rules can be adapted. For example, Hindi and Rajasthani follows mostly the same set of rules.

7 Conclusion

The work presented in this paper is a step towards building an efficient pronunciation generator for Indian languages. Although the objective was primarily to unify various Indian language parsers, it is observed that the unified parser is more robust than custom built parsers. In order to build a parser for a new language, one needs to identify the language family, borrow the standard rule set. Exceptions may be handled incorporating language-specific rules.

References

1. Copestake, A., Flickinger, D.: An open source grammar development environment and broad-coverage English grammar using HPSG. In: Proceedings of LREC 2000, pp. 591–600 (2000)
2. Kishore, S., Kumar, R., Sangal, R.: A data driven synthesis approach for indian languages using syllable as basic unit. In: Proceedings of International Conference on NLP (ICON), pp. 311–316 (2002)
3. Lavanya, P., Kishore, P., Madhavi, G.T.: A simple approach for building transliteration editors for Indian languages. J. Zhejiang Univ. Sci. A **6**(11), 1354–1361 (2005)
4. Levine, J.R., Mason, T., Brown, D.: LEX & YACC. O'Reilly Media Inc., Sebastopol (1992)
5. Prakash, A., Reddy, M.R., Nagarajan, T., Murthy, H.A.: An approach to building language-independent text-to-speech synthesis for Indian languages. In: 2014 Twentieth National Conference on Communications (NCC), pp. 1–5. IEEE (2014)
6. Quinlan, J.R.: Induction of decision trees. Mach. Learn. **1**(1), 81–106 (1986)
7. Raghavendra, E.V., Desai, S., Yegnanarayana, B., Black, A.W., Prahallad, K.: Global syllable set for building speech synthesis in Indian languages. In: SLT 2008, pp. 49–52. IEEE (2008)
8. Raghavendra, E.V., Yegnanarayana, B., Black, A.W., Prahallad, K.: Building sleek synthesizers for multi-lingual screen reader. In: INTERSPEECH, pp. 1865–1868 (2008)
9. Raina, A.M., Mukerjee, A., Goyal, P., Shukla, P.: A unified computational lexicon for hindi-english code-switching. In: Proceedings International Conference on Natural Language Processing (ICON), Hyderabad, India, December 2004, pp. 19–22 (2004)
10. Ramani, B., Christina, S.L., Rachel, G.A., Solomi, V.S., Nandwana, M.K., Prakash, A., Shanmugam, S.A., Krishnan, R., Kishore, S., Samudravijaya, K., et al.: A common attribute based unified hts framework for speech synthesis in Indian languages. In: 8th ISCA Workshop on Speech Synthesis, pp. 311–316 (2013)
11. Singh, A.K.: A computational phonetic model for Indian language scripts. In: Constraints on Spelling Changes: Fifth International Workshop on Writing Systems (2006)
12. Tokuda, K., Yoshimura, T., Masuko, T., Kobayashi, T., Kitamura, T.: Speech parameter generation algorithms for HMM-based speech synthesis. In: Proceedings of the 2000 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2000, ICASSP 2000, vol. 3, pp. 1315–1318. IEEE (2000)