Interspeech 2018
2-6 September 2018, Hyderabad

Code-switching in Indic Speech Synthesisers

Anju Leela Thomas, Anusha Prakash, Arun Baby, Hema A. Murthy

Indian Institute of Technology Madras, India

hema@iitm.ac.in

Abstract

Most Indians are inherently bilingual or multilingual owing to the diverse linguistic culture in India. As a result, code-switching is quite common in conversational speech. The objective of this work is to train good quality text-to-speech (TTS) synthesisers that can seamlessly handle code-switching. To achieve this, bilingual TTSes that are capable of handling phonotactic variations across languages are trained using combinations of monolingual data in a unified framework. In addition to segmenting Indic speech data using signal processing cues in tandem with hidden Markov model-deep neural network (HMM-DNN), we propose to segment Indian English data using the same approach after NIST syllabification. Then, bilingual HTS-STRAIGHT based systems are trained by randomizing the order of data so that the systematic interactions between the two languages are captured better. Experiments are conducted by considering three language pairs: Hindi+English, Tamil+English and Hindi+Tamil. The code-switched systems are evaluated on monolingual, code-mixed and code-switched texts. Degradation mean opinion score (DMOS) for monolingual sentences shows marginal degradation over that of an equivalent monolingual TTS system, while the DMOS for bilingual sentences is significantly better than that of the corresponding monolingual TTS systems.

Index Terms: code-switching, Indian languages, text-to-speech synthesis, HMM-DNN segmentation, HTS-STRAIGHT

1. Introduction

Indians proficient in more than one language can code-switch without any hesitation, more or less preserving the phonotactics of each language. The objective of this work is to build text-to-speech (TTS) synthesisers such that they are capable of handling multilingual text and are able to code-switch seamlessly, especially in the context of communities where speakers are equally fluent in multiple languages. This is particularly true for Indians where children are exposed to 2-3 languages simultaneously from childhood[1], namely the mother tongue, English, and a regional language. Influence of all three languages on each other can be perceived in the seamlessness of code-switching. The objective of this paper is to achieve this effect in code-switched synthesis systems using only monolingual recordings.

Code-switching is the alternation between multiple languages in a single conversation [2]. It is more of an inter-sentential phenomenon whereas code-mixing is an intra-sentential phenomenon. Code-mixing may also include the embedding of words from other languages [3]. In this paper, we use the term code-switching to refer to both code-switching and code-mixing, unless stated explicitly. According to [2], code-switching is characteristic of bilingual and multilingual communities. Further, migration across states, urbanization and globalization have led to a rise in code-switching. The most evident code-switching is between any native language and English. Movies, advertisements and (pop) songs have also made code-switching popular. The patterns of code-switching also vary, depending on proficiency, region and extent of urbanization.

In a previous attempt to synthesise multilingual text, we performed the synthesis using a monolingual voice after phone mapping. It was observed that the transitions during code-switching were not smooth. This is primarily because the phonotactics of the code-switched language is not learned by the synthesiser. In [4], multilingual speech corpus of a single speaker was used to train TTSes. But recording a multilingual corpus is an expensive process.

Most of the current efforts in this domain synthesise multilingual text using systems trained on a mixture of monolingual data. In polyglot synthesis, multiple monolingual data are combined together to build an average voice, and the voice is then adapted to the target language/speaker [5]. In [6], the polyglot synthesiser is used to synthesise text in a new language by mapping the phones in the new language to the phones in the trained data. [7] experiments with phone sharing, context-dependent state sharing and state mapping across languages. In [8], pitch and duration of English words are modified in bilingual text. [9] experiments with various front-end modules to obtain a bilingual voice from monolingual corpus of Mandarin and English. Mixed-lingual speech synthesis systems are trained based on different grapheme to phoneme conversion techniques, acoustic and prosodic modeling [10]. [11] attempts to obtain correct pronunciation of Indic words in navigation instructions. While [3, 11] deal with multilingual text in the Romanized script, [7, 9] synthesise multilingual text with words in their native script. In [12], experiments are conducted by code-mixing in the same script and mixed scripts.

In this work, multilingual text is assumed to be UTF-8. The trained TTS systems can be directly used for language learning applications, webpages with multilingual text[2], etc. Monolingual data across languages are mixed to train a code-switchable bilingual

[1]According to Census report 2001, more than a quarter of India's population was bilingual and less than one-tenth was at least trilingual [1].

[2]Examples of webpages with multilingual text in native scripts: www.tn.gov.in/, www.ilearntamil.com/time-learn-tamil-through-english/, www.achhikhabar.com/2012/04/17/10-tips-to-become-self-confident-in-hindi/

Table 1: *Examples of code-switching*

Example	Languages	Phonetic transcription (in CLS)	Meaning
Good morning! आप कैसे हो?	Hindi+English	g-u-dx m-ax-r-n-i-ng aa-p k-ei-s-ee h-o	Good morning! How are you?
நான் shopping போக ready	Tamil+English	n-aa-nd sh-aa-p-i-ng p-oo-g-a r-ee-dx-ii	I am ready to go shopping
அண்ணா , ये கடை कब खुलेगा?	Hindi+Tamil	a-nx-nx-aa y-ee k-a-dx-ai k-a-b kh-u-l-ee-g-aa	Brother, when will this shop open?

TTS system. Segmentation of Indic speech data is obtained using signal processing cues in tandem with hidden Markov model-deep neural network (HMM-DNN) [13]. In this work, we extend this technique to segment Indian English data. This is motivated by the fact that Indian English is syllable-timed [14]. HTS-STRAIGHT voices (HTS: HMM-based speech synthesis systems, STRAIGHT: Speech Transformation and Representation using Adaptive Interpolation weiGHTed spectrum) [15, 16] are trained from a mixture of monolingual corpora in a language pair. Experiments are conducted by varying the order in which the data is presented for training acoustic models. For each language pair '*L1+L2*', three different TTSes are built based on the order of pooled data while training– (1) *L1* followed by *L2* (*L1-L2*), (2) *L2* followed by *L1* (*L2-L1*), and (3) random order of data across *L1* and *L2* (*Random*). The motivation for randomizing the order of pooled data is to minimize the bias in learning model parameters towards a particular language. This is further motivated by the results obtained in [17]. It is observed that alternating between languages for training in language identification and modeling tasks results in better performance.

Three languages have been considered in this work, namely, Hindi, Tamil, and English. Hindi and Tamil belong to different language groups– Indo-Aryan and Dravidian, respectively. In addition to training Indian language+English systems, bilingual Tamil+Hindi systems are built. These systems are used to synthesise a set of monolingual, code-mixed and code-switched texts. Subjective evaluations are conducted to compare the performance of bilingual TTSes with relevant monolingual TTSes.

A brief description of the variation in phonotactics across languages is given in Section 2. Section 3 describes the proposed technique to build the bilingual TTS systems for code-switching. Experiments and results are detailed in Section 4 and the work is concluded in Section 5.

2. Motivation

Some examples of code-switching are given in Table 1. Phonetic transcriptions are replaced by common label set (CLS) representation [18] to arrive at a common set of labels across all the language pairs. From Example 2 (Tamil+English), it is evident that the sequence of English words in the code-switched text and its translated version is different. At a fundamental level, it is the variation in phonotactics across languages that motivates this work. Phonotactics is the sequencing of phones that are allowed in a language. Phone combinations valid in one language may not be valid in other languages.

Phonotactic differences are more evident across language groups (Indo-Aryan and Dravidian). For example, Dravidian languages are characterised by their agglutinative nature [19]. Language-specific phones also affect the phonotactics of languages. Complex phone clusters (for example- strength, twelfth) that occur in English are rare in Indian languages [19]. Indian languages are *akshara*-based and have simpler phone clusters [19]. Phone clusters such as *bh* (aspirated *b*), *sr*, etc. are quite rare in English; similarly, phone clusters such as *ous* and *ion* are rare in Indian languages [20]. English does not have geminates [21], whereas Indian languages are replete with geminates.

It is vital to preserve the phonotactics in languages while code-switching, even though a common label set may be used for synthesis. An approach to achieving this is to train TTS synthesisers from different monolingual speech data as described in the next section.

3. Proposed approach

The modules involved in building a code-switchable TTS synthesiser are dealt at three levels– (1) parsing words in a monolingual/multilingual text to their corresponding phone and syllable sequences, (2) segmenting the Indic and English speech data for training, and (3) building monolingual (baseline) and bilingual voices.

3.1. Parsing words in a multilingual text

An overview of obtaining the phone and syllable sequences of a multilingual (or monolingual) text is illustrated in Figure 1. To parse Indic words, the unified parser is used [22]. The unified parser is a language-independent parser for Indian languages that uses a set of pre-defined rules to convert Indic words to corresponding syllable and phone sequences. The parser automatically recognises the script of the language based on its Unicode range. The sequences thus obtained are in terms of the CLS representation [18]. Phonetic transcriptions of Indic words in Table 1 have been obtained using the unified parser.

Since the unified parser is designed for Indian languages, English words are handled separately. When an English word is encountered, and if it is present in the CMU pronunciation dictionary [23], the word is parsed in terms of CMU phones. Stress markers are removed from this representation. Then, the NIST syllabification software is used to generate the syllable sequence [24]. The syllabification output is in terms of CMU phones.

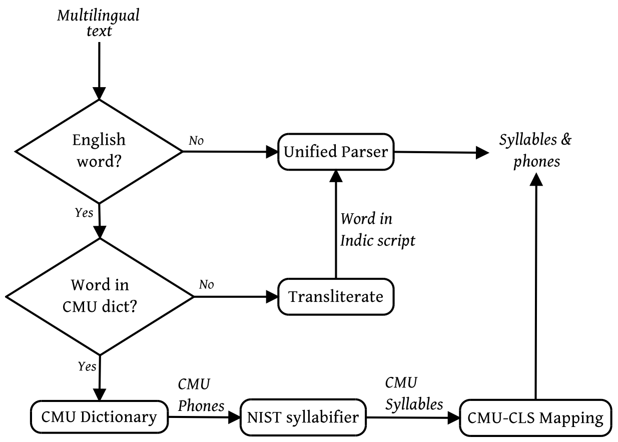

Figure 1: *Parsing multilingual text*

Table 2: *Obtaining syllable sequence of 'ABANDONED'*

Word: ABANDONED
From CMU dictionary: AH0 B AE1 N D AH0 N D
After removing stress markers and converting to lowercase: ah b ae n d ah n d
NIST syllabification output:
1 /# ['0 ah] [b '0 ae n] [d '0 ah n d] #/
After cleaning up: [ah] [b ae n] [d ah n d]
After CMU-CLS mapping:
[a] [b axx n] [dx a n dx]

This is first cleaned up. If a syllable consists of a single consonant, it is included either with the previous syllable or the next syllable. This is done semi-automatically. The constituent phones in CMU format are then mapped to labels in CLS using a pre-defined label mapping. The label mapping is prepared manually by finding the closest match between CMU and CLS phones. This label mapping need not be one-to-one. For example, the CMU phone *er* is mapped to a sequence of labels in CLS- *a* followed by *r*. An example of this parsing is illustrated in Table 2. Syllables are demarcated by square brackets.

On the other hand, English words not present in the CMU dictionary are transliterated to the script of the other Indic language in the language pair using Google Transliterate. These transliterations are then corrected manually before using the unified parser to obtain syllables and phones. This is done only in the training phase. For the current work, we have chosen test sentences whose constituent English words are present in the CMU dictionary.

Instead of using a transliterator, a generic solution would be to use a suitable grapheme-to-phoneme converter such as the classification and regression tree (CART) available in the Festival framework [25].

3.2. Segmenting speech data

With syllable and phone labels, phone alignments of speech wavefiles are obtained using signal processing cues in tandem with deep learning techniques [13]. First, syllable level boundaries are determined with the help of spectral cues [26]. The syllable boundaries are then corrected in an iterative manner for HMM-DNN-based segmentation. Finally, phone level boundaries are obtained from the HMM-DNN alignment. Segmentation is performed for each dataset separately. The phone alignments are pooled together (or used directly) to train a bilingual (or monolingual) TTS system.

3.3. Training the synthesiser

3.3.1. Monolingual systems

From the phone-level segmentation of each monolingual data, an HTS-STRAIGHT system is trained [15, 16]. A common question set (CQS) is used for tree-based clustering to handle unseen models [18]. We consider these monolingual TTSes as baseline systems.

3.3.2. Bilingual systems

A bilingual system is built from a mixture of monolingual data. Speech data corresponding to languages in a given pair are pooled together (as described in Section 1) and the bilingual TTS is trained similarly to the monolingual case.

4. Experiments and results

4.1. Speech datasets

Speech data used in the experiments consists of four monolingual datasets- Hindi speech of a native male Hindi speaker (mHin) and English speech of the same speaker (IE_H); Tamil speech of a native male Tamil speaker (mTam) and English speech of the same speaker (IE_T). Each dataset is about 5 hours in duration. Transcriptions corresponding to audio waveforms are in UTF-8 format. The datasets are part of the Indic TTS database [27].

4.2. Experiments

Monolingual TTSes (mHin, IE_H, mTam, IE_T) are trained for each dataset using 5 hours of data. Bilingual TTSes are trained using 2.5 hours of *L1* and 2.5 hours of *L2* for a language pair. The bilingual TTS systems built are listed in Table 3. Synthesised speech output of each TTS is passed through a low pass filter with cut-off frequency 8kHz to remove noise.

The language pair Hindi+Tamil is a special case as we are combining speech data of different speakers. There is no constraint to synthesise a specific voice; we have normalized synthesised speech of the bilingual TTS to get an average voice.

Table 3: *Bilingual TTS systems*

Language pairs	*L1-L2*	*L2-L1*	*Random*
Hindi+English	HE	EH	HER
Tamil+English	TE	ET	TER
Hindi+Tamil	HT	TH	HTR

4.3. Evaluation

A degradation mean opinion score (DMOS) test is conducted to evaluate the performance of the TTS systems. Listeners in the test are asked to rate the quality of synthesised speech on a scale of 1-5 (5 being the highest)

Table 4: *DMOS for Hindi+English (global average)*

mHin	IE_H	HE	EH	HER
3.21	2.54	3.51	3.42	3.58

Table 5: *Categorized results of Hindi+English DMOS*

Text	TTS				
type	mHin	IE_H	HE	EH	HER
CM	3.36	2.25	3.75	3.24	3.56
CS	2.95	2.43	3.47	3.54	3.77
Eng	2.05	3.43	3.46	3.40	3.61
Hin	4.48	2.07	3.34	3.51	3.40

Table 6: *DMOS for Tamil+English (global average)*

mTam	IE_T	TE	ET	TER
2.59	2.58	3.62	3.78	3.81

Table 7: *Categorized results of Tamil+English DMOS*

Text	TTS				
type	mTam	IE_T	TE	ET	TER
CM	3.09	2.07	3.27	3.63	3.51
CS	1.91	2.16	3.39	3.59	3.50
Eng	1.33	4.49	4.34	4.20	4.42
Tam	4.01	1.60	3.49	3.69	3.82

based on continuity and naturalness. For Hindi+English (or Tamil+English) DMOS test, native Hindi (or native Tamil) speakers who are proficient in English have been used. For Hindi+Tamil test, evaluators are native Tamil speakers who are proficient in Hindi. The performance of each system is evaluated on different text types: monolingual (Hin for Hindi/Tam for Tamil/Eng for English), code-switched (CS) and code-mixed (CM) texts. Test sentences of Blizzard Challenge 2014-15 [28, 29] and sentences from the web, that include both monolingual and bilingual texts have been used.

For each DMOS evaluation, 20 listeners evaluated 5 synthesised outputs per system per text type, i.e., each person evaluated 20 synthesised speech samples of each system. The evaluation results are presented in Tables 4, 6 and 8 for Hindi+English, Tamil+English, Hindi+Tamil evaluations, respectively. We refer to these scores as the global average. A categorized version of the scores are given in Tables 5, 7 and 9; each row of a table corresponds to a text type and each column corresponds to a TTS system.

In most cases, monolingual texts are best synthesied by the corresponding monolingual systems, as expected. For example, in Table 7, system mTam performs better than the other TTSes for text type Tam. However, in Table 9, for text type Hin, system mHin is slightly degraded compared to systems TH and HTR. This is not a significant difference and may also be a result of native Tamil speakers evaluating monolingual Hindi utterances. This anomaly is also present for systems IE_E, EH and HER on text type Eng in Table 5. This requires further analysis. For Indian Language+English systems, the performance of monolingual Indic TTS (system mHin in Table 5 and system mTam in Table 7) is in this descending order: Indic_text>CM>CS>Eng; for monolingual English TTSes (IE_H, IE_T), this order is reversed. This is because CM texts for Table 5 and Table 7 have more Indic words than English words, and CS texts are more balanced. However, in Table 9, for both systems mHin and mTam, synthesised CM utterances are better than synthesised CS utterances. This is because CM texts have more or less the same number of Hindi and Tamil words. CS texts, although balanced, are longer compared to CM texts, which may have led to a drop in DMOS scores.

For CM and CS texts, bilingual TTSes perform better than monolingual TTSes in almost all cases. It is further observed that if monolingual system *L1* performs better than monolingual system *L2* on corresponding monolingual texts, then this shows up in the superior performance of *L1-L2* bilingual system over *L2-L1* bilingual system in the global average. For example, in Table 5, system mHin performs better than system IE_H for Hin and Eng text types, respectively. This is reflected in the global average- score of HE is better than that of EH (Table 4).

Table 8: *DMOS for Hindi+Tamil (global average)*

mHin	mTam	HT	TH	HTR
2.91	2.91	3.52	3.51	3.73

Table 9: *Categorized results of Hindi+Tamil DMOS*

Text	TTS				
type	mHin	mTam	HT	TH	HTR
CM	2.76	3.05	3.31	3.24	3.90
CS	2.38	2.39	3.43	3.17	3.37
Tam	2.28	4.00	3.53	3.36	3.39
Hin	4.20	2.18	3.83	4.28	4.25

When the global average is considered, *Random* systems (HER, TER, HTR) perform the best. *Random* TTSes are mostly degraded with respect to monolingual systems on corresponding monolingual texts, and are better for CM and CS texts; nevertheless, they are more or less robust overall. According to [30], the languages involved in code-switching systematically interact with each other. We conjecture that the *Random* TTS is able to capture these interactions better. Sample synthesised speech can be found at this link– `www.iitm.ac.in/donlab/is2018/code.php`.

The CMU pronunciation dictionary has American English pronunciations which are different from Indian English pronunciations. With a better word to phone conversion of English words to suit native Indian pronunciations, bilingual systems can be further improved.

5. Conclusion

Bilingual TTSes have been developed for smooth code-switching in the Indian context. HMM-DNN segmentation along with signal processing cues have been used to segment Indian English data and Indic speech data. It is conjectured that randomizing the order of data before voice building better captures the systematic interactions between languages. We plan to extend this work to include more language pairs and also attempt at developing trilingual (eventually multilingual) TTS systems.

6. References

[1] The Times of India, "Indiaspeak: English is our 2nd language," www.timesofindia.indiatimes.com/india/Indiaspeak-English-is-our-2nd-language/articleshow/5680962.cms.

[2] P. Auer, *Code-switching in conversation: Language, interaction and identity.* Routledge, 2013.

[3] S. Sitaram, S. K. Rallabandi, and S. Black, "Experiments with cross-lingual systems for synthesis of code-mixed text," in *Speech Synthesis Workshop*, 2015, pp. 76–81.

[4] C. Traber, K. Huber, K. Nedir, B. Pfister, E. Keller, and B. Zellner, "From multilingual to polyglot speech synthesis," in *European Conference on Speech Communication and Technology*, 1999.

[5] J. Latorre, K. Iwano, and S. Furui, "Polyglot synthesis using a mixture of monolingual corpora," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2005, pp. 1–4.

[6] J. Latorre, K. Iwano, and S. Furui, "Cross-language synthesis with a polyglot synthesizer," in *INTERSPEECH*, 2005, pp. 1477–1480.

[7] H. Liang, Y. Qian, and F. K. Soong, "An HMM-based Bilingual (Mandarin-English) TTS," in *Speech Synthesis Workshop*, 2007, pp. 137–142.

[8] Y. Zhang and J. Tao, "Prosody modification on mixed-language speech synthesis," in *International Symposium on Chinese Spoken Language Processing (ISCSLP)*, 2008, pp. 1–4.

[9] M. Chu, H. Peng, Y. Zhao, Z. Niu, and E. Chang, "Microsoft Mulan-a bilingual TTS system," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2003, pp. I–264–I–267.

[10] S. Rallabandi and A. W. Black, "On building mixed lingual speech synthesis systems," in *INTERSPEECH*, 2017, pp. 52–56.

[11] K. R. Chandu, S. K. Rallabandi, S. Sitaram, and A. W. Black, "Speech synthesis for mixed-language navigation instructions," in *INTERSPEECH*, 2017, pp. 57–61.

[12] S. Sitaram and A. W. Black, "Speech synthesis of code-mixed text," in *International Conference on Language Resources and Evaluation (LREC)*. European Language Resources Association (ELRA), 2016, pp. 3422–3428.

[13] A. Baby, J. J. Prakash, R. Vignesh, and H. A. Murthy, "Deep Learning Techniques in Tandem with Signal Processing Cues for Phonetic Segmentation for Text to Speech Synthesis in Indian Languages," in *INTERSPEECH*, 2017, pp. 3817–3821.

[14] S. Rupak Vignesh, A. Shanmugam, and H. A. Murthy, "Significance of pseudo-syllables in building better acoustic models for Indian English TTS," in *International conference on Acoustics, Speech and Signal processing (ICASSP)*, 2016, pp. 5620–5624.

[15] H. Zen, K. Tokuda, and A. W. Black, "Statistical parametric speech synthesis," *Speech Communication*, vol. 51, no. 3, pp. 1039–1064, November 2009.

[16] H. Kawahara, I. Masuda-Katsuse, and A. de Cheveigné, "Restructuring speech representations using a pitch-adaptive time-frequency smoothing and an instantaneous-frequency-based F0 extraction: Possible role of a repetitive structure in sounds," *Speech Communication*, vol. 27, no. 3–4, pp. 187 – 207, 1999.

[17] M. Choudhury, K. Bali, S. Sitaram, and A. Baheti, "Curriculum Design for Code-switching: Experiments with Language Identification and Language Modeling with Deep Neural Networks," in *International Conference on Natural Language Processing (ICON)*, 2017, pp. 65–74.

[18] B. Ramani, S. Lilly Christina, G. Anushiya Rachel, V. Sherlin Solomi, M. K. Nandwana, A. Prakash, S. Aswin Shanmugam, R. Krishnan, S. Kishore, K. Samudravijaya, P. Vijayalakshmi, T. Nagarajan, and H. A. Murthy, "A common attribute based unified HTS framework for speech synthesis in Indian languages," in *Speech Synthesis Workshop*, 2013, pp. 291–296.

[19] A. Prakash, J. J. Prakash, and H. A. Murthy, "Acoustic Analysis of Syllables Across Indian Languages," in *INTERSPEECH*, September 2016, pp. 327–331.

[20] A. Sen and K. Samudravijaya, "Indian accent text-to-speech system for web browsing," *Sadhana*, vol. 27, no. 1, pp. 113–126, 2002.

[21] S. Davis, *Geminates, The Blackwell companion to phonology, II: Suprasegmental and Prosodic Phonology.* Oxford University Press, 2011.

[22] A. Baby, N. N. L., A. L. Thomas, and H. A. Murthy, "A unified parser for developing Indian language text to speech synthesizers," in *International Conference on Text, Speech and Dialogue*, 2016, pp. 514–521.

[23] Carnegie Mellon University, "The CMU pronounciation dictionary," www.speech.cs.cmu.edu/cgi-bin/cmudict.

[24] M. Fisher, *Syllabification Software*, The Spoken Natural Language Processing Group, National Institute of Standards and Technology, U.S.A. [Online]. Available: www.nist.gov/file/65961

[25] A. Black, P. Taylor, and R. Caley, "The Festival speech synthesis system," `http://festvox.org/festival/`, 1998.

[26] S. A. Shanmugam and H. Murthy, "A hybrid approach to segmentation of speech using group delay processing and HMM based embedded reestimation," in *INTERSPEECH*, 2014, pp. 1648–1652.

[27] A. Baby, A. L. Thomas, N. N. L, and H. A. Murthy, "Resources for Indian languages," in *Community-based Building of Language Resources (International Conference on Text, Speech and Dialogue)*, 2016, pp. 37–43.

[28] K. Prahallad et al., "The blizzard challenge 2014," in *Blizzard Challenge workshop*, 2014.

[29] K. Prahallad et al., "The blizzard challenge 2015," in *Blizzard Challenge workshop*, 2015.

[30] A. K. Joshi, "Processing of sentences with intra-sentential code-switching," in *Conference on Computational linguistics-Volume 1.* Academia Praha, 1982, pp. 145–150.