WSTCSMC 2020

The First Workshop on Speech Technologies for Code-Switching in Multilingual Communities

Proceedings of the Workshop

October 30 and 31, 2020

Introduction

Code-switching (CS) is the phenomenon by which multilingual speakers switch back and forth between their common languages in written or spoken communication. CS is pervasive in conversational communications across multilingual communities. Processing CS speech efficiently poses a great variety of challenges to Speech Processing algorithms: speech recognition and synthesis, language modeling, dialogue systems and chatbots, speech analyzers, and many others. In this workshop, we aim to create a space for speech researchers interested in discussing all these challenges and identifying a path forward in this research area.The workshop program includes papers discussing code-switching analysis from multilingual corpora, language identification and code-switching detection from voice utterances, design of code-switched voice services, and different aspects of code-switched speech recognition.

Another component of the workshop is the Shared Task on Code-switched Spoken Language Identification (LID) of CS Data. The shared task focused on three language pairs (Gujarati-English, Telugu-English and Tamil-English), and consisted of two subtasks: utterance-level identification of monolingual vs. code-switched utterances and frame-level identification of language in a code-switched utterance. A total of five teams submitted their system's output: five for the first subtask, and two for the second. All shared task systems will be presented during the workshop.

We would like to thank all authors who submitted their contributions to this workshop and all shared task participants for taking on the challenge. We received a total of 19 regular workshop submissions of which we accepted 14 for publication, all of them as workshop talks. We also thank the program committee members for their help in providing meaningful reviews. Lastly, we thank the Interspeech 2020 organizers for the opportunity to put together this workshop, Microsoft for their tech support and Speech Ocean for generously sharing their data for the Shared Task.

See you all remotely, at WSTCSMC 2020!

Workshop co-chairs,

Kalika Bali
Alan W. Black
Rupesh Kumar Mehta
Sunayana Sitaram
Thamar Solorio
Victor Soto
Emre Yilmaz

Workshop Co-Chairs:

Kalika Bali, Microsoft Research (India)
Alan W. Black, Carnegie Mellon University (USA)
Rupesh Kumar Mehta, Microsoft (India)
Sunayana Sitaram, Microsoft Research (India)
Thamar Solorio, University of Houston (USA)
Victor Soto, Amazon Alexa AI (USA)
Emre Yilmaz, National University of Singapore (Singapore)

Shared Task Co-Chairs:

Sanket Shah, Microsoft Research (India)
Sandeepkumar Satpal, Microsoft (India)
Vinay Krishna, Microsoft (India)

Program Committee:

Basil Abraham, Microsoft (India)
Gustavo Aguilar, University of Houston (USA)
Kalika Bali, Microsoft Research (India)
Astik Biswas, Oracle (India)
Barbara Bullock, University of Texas (USA)
Khyathi Raghavi Chandu, Carnegie Mellon University (USA)
Febe De Wet, Stellenbosch University (South Africa)
Satarupa Guha, Microsoft (India)
Julia Hirschberg, Columbia University (USA)
Leijing Hou, China Mobile Research (China)
Grandee Lee, National University of Singapore (Singapore)
Hung-yi Lee, National Taiwan University (Taiwan)
Haizhou Li, National University of Singapore (Singapore)
Sakriani Sakti, Nara Institutes of Science and Technology (Japan)
Jacqueline Serigos, George Mason University (USA)
Sunayana Sitaram, Microsoft Research (India)
Sunit Sivasankaran, INRIA (France)
Thamar Solorio, University of Houston (USA)
Victor Soto, Amazon Alexa AI (USA)
Brij Mohan Lal Srivastava, INRIA (France)
Almeida Jacqueline Toribio, University of Texas (USA)
Ewald Van der Westhuizen, Stellenbosch University (South Africa)
Ngoc Thang Vu, University of Stuttgart (Germany)
Shinji Watanabe, Johns Hopkins University (USA)
Haihua Xu, National University of Singapore (Singapore)
Emre Yilmaz, National University of Singapore (Singapore)
Shiliang Zhang, Peking University (China)

Invited Speakers:

Barbara E. Bullock, University of Texas (USA)
Almeida Jacqueline Toribio, University of Texas (USA)

Technical Support:

Rashmi K Y, Microsoft (India)
Naveen Kumar, Microsoft (India)
Vyshak Jain, Microsoft (India)
Nadeem Shaheer, Microsoft (India)

Table of Contents

Workshop Program

Friday, October 30, 2020

20:30–21:30 Opening remarks and Keynote

20:30–21:30 *Points of Connection Between Linguistics and Speech Technology with Regard to Code-Switching*
Barbara E. Bullock and Almeida Jacqueline Toribio

21:30–21:40 Break

21:40–22:40 Paper Session 1

21:40–21:55 *A Study of Types and Characteristics of Code-Switching in Mandarin-English Speech*
Leijing Hou, Ying Liu, Yingying Gao and Junlan Feng

21:55–22:10 *Malayalam-English Code-Switched: Speech Corpus Development and Analysis*
Sreeram Manghat, Sreeja Manghat and Tanja Schultz

22:10–22:25 *Understanding Forced Alignment Errors in Hindi-English Code-mixed Speech – a Feature Analysis*
Ayushi Pandey, Pamir Gogoi and Kevin Tang

22:25–22:40 *Q & A*

22:40–22:50 Break

22:50–23:00 Sponsor Talk

22:50–23:00 *Microsoft*
Basil Abraham

23:00–23:40 Paper Session 2

23:00–23:15 *Mere Account Mein Kitna Balance Hai? - on Building Voice Enabled Banking Services for Multilingual Communities*
Akshat Gupta, Sai Krishna Rallabandi and Alan W Black

23:15–23:30 *Investigating Modelling Techniques for Natural Language Inference on Code-Switched Dialogues in Bollywood Movies*
Anjana Umapathy, Sharanya Chakravarthy and Alan W Black

23:30–23:40 *Q & A*

Saturday, October 31, 2020

20:30–21:45 Shared Task Session

20:30–20:35 *Opening Remarks*
Sunayana Sitaram

20:35–20:45 *First Workshop on Speech Processing for Code-switching in Multilingual Communities: Shared Task on Code-switched Spoken Language Identification*
Sanket Shah, Sunayana Sitaram and Rupesh Mehta

20:45–20:55 *Vocapia-LIMSI System for 2020 Shared Task on Code-switched Spoken Language Identification*
Claude Barras, Viet Bac Le and Jean-Luc Gauvain

20:55–21:05 *Exploiting Spectral Augmentation for Code-Switched Spoken Language Identification*
Pradeep R, Sundeep Teki and Hemant Misra

21:05–21:15 *On Detecting Code Mixing in Speech Using Discrete Latent Representations*
Sai Krishna Rallabandi and Alan W Black

21:15–21:25 *Language Identification for Code-Mixed Indian Languages in the Wild*
Parav Nagarsheth and Jehoshaph Akshay Chandran

21:25–21:35 *Utterance-level Code-Switching Identification Using Transformer Network*
Krishna D N and Ankita Patil

21:35–21:45 *Q & A*

21:45–22:00 Break

22:00–22:10 Sponsor Talk

22:00–22:10 *Speech Ocean*
Yufeng Hao

Saturday, October 31, 2020 (continued)

22:10–23:10 Paper Session 3

22:10–22:25 *Learning Not to Discriminate: Task Agnostic Learning for Improving Monolingual and Code-switched Speech Recognition*
Gurunath Reddy M, Sanket Shah, Basil Abraham, Vikas Joshi and Sunayana Sitaram

22:25–22:40 *Multilingual Bottleneck Features for Improving ASR Performance of Code-Switched Speech in Under-Resourced Languages*
Trideba Padhi, Astik Biswas, Febe De Wet, Ewald Van der westhuizen and Thomas Niesler

22:40–22:55 *The ASRU 2019 Mandarin-English Code-Switching Speech Recognition Challenge: Open Datasets, Tracks, Methods and Results*
Xian Shi, Qiangze Feng and Lei Xie

22:55–23:10 *Q & A*

23:10–23:20 Closing Remarks

15

A Study of Types and Characteristics of Code-Switching in Mandarin-English Speech

Leijing Hou, Ying Liu*, Yingying Gao, Junlan Feng*

China Mobile Research

{houleijing, liuyingzn, gaoyingying, fengjunlan}@chinamobile.com

Leijing Hou and Ying Liu contributed equally to this work

Abstract

In this paper, we are motivated to systematically study the code-switching behavior with a large Mandarin-English speech corpus. Our analysis is organized hierarchically into three levels: phonemes, words and context of the code-switched segments. On each level, we investigate the characteristics of the distribution, pronunciation, syntax, and semantics of CS segments. We run quantitive evaluation to measure how these factors influence CS and reveal certain commonalities. The findings in this study are complementary to direct CS modeling and can be used to further guide the enhancement of acoustic and language model design.

Index Terms:code-switching, characteristics at code-switching points

1. Introduction

Code-switching(CS) denotes a shift from one language to another within an utterance or a discourse. With rapid progress of globalization, this phenomenon occurs increasingly frequent in our daily life. It poses extra challenges to automantic speech recognition(ASR), since most off-the-shelf ASR systems today are monolingual and cannot handle code-switched speech well.

Substantial research efforts recently have been made to bridge this gap from system integration, pronunciation, language, and acoustic modeling perspectives. [1] proposes to run multiple ASR engines in parallel with an language identification(LID) system and compositionally uses scores from all these systems to decode CS speech. [2] uses a English speech recogniser and a Malay recogniser in the first pass and rescored the produced hypothesis to get the final result without a LID system. The choice of phone set for CS ASR is important too. [3] applied International Phonetic Alphabet(IPA), Bhattacharya distance and discriminative training to combine phone sets for Mandarin-English. [4] describes similar approaches to combine phone sets for Hindi-English speech recogniser. For language modeling, there is sure more text data comparing to speech especially on social media sites. However the patterns of them found online is very different from speech in real life. [5] proposes to apply Bert and GAN models to generate CS data, which is then to augment the monolingual corpus to retrain the language model for ASR. [6] obtains the best result by training a RNN LM with interleaved monolingual data in English and Spanish followed by CS data. For acoustic modelling, [7] develops a cross-lingual phonetic acoustic model for Cantonese-English speech. More recently, researchers have explored end-to-end acoustic modeling for CS speech. [8] trained a CTC-based model using monolingual data and fine-tune with CS data for Mandarin-English speech. [9] uses transfer learning from monolingual models.

The above efforts all obtain promising results to lower CS speech word error rate. However, the gap is still substantial. In this paper, we are motivated to systematically study the nature and types of the CS phenomena in speech. We believe this line of study is complementary and helpful to direct acoustic, language, and pronunciation modelling for CS ASR.

As we all observe, CS in general is not a simple and random activity. There are certain underlying linguistic structures and tendencies that are accepted as appropriate. A few studies in the past have explored various aspects of CS. Kraus et al. investigated textural features that trigger CS with the SEAME [10] data corpus [11]. Their study shows CS segments are most often triggered by determiners in Mandarin and switched by nouns in English. Preeti et al. studies the lexical and prosodic cues for CS with a Hindi-English code-switching corpus [12]. Heike et al. explores Brown word clusters, open class words and clusters of open class word embeddings with a Mandarin-English code-switching corpus [13]. [14] reveals that examples of highly-switched speech are primarily composed of frequent spans of short length, reflecting the bursty nature of switching and the lower span entropy of frequent CS. [15] discovers two constraints of CS by quantitative analysis: Free Morpheme Constraint and Equivalence Constraint that function simultaneously. The Free Morpheme constraint specifies that it is possible to switch between full sentences as well as any constituent within the sentence if a free morpheme is present in a constituent. The Equivalence Constraint specifies that language switches generally occur at points where there is no violation of syntactic rules of the participating languages. Most of these analyses conducted are based on a relatively small corpus. Some focused on finding a general underlying regularity of CS. Some are more into one or two very specifics of CS.

In this paper, we conduct our study on a large CS Mandarin-English speech corpus made by DataTang with around 630K utterances that has recently become available. We propose to systematically analyze CS on three levels : Phoneme, Word, and Context of the CS segments. We target our efforts to answer the following questions: 1) How does the pronunciation of an English word or the comprised phonemes influence the way it is code-switched into Mandarin? 2)What types of words and phrases are more likely to be switched in? 3)What is the generality of the context surrounding these CS segments? 4)Suggestion for ASR given these studies?

The rest of the paper is organized as follows. Section 2 describes the *DataTang* corpus. Section 3 elaborate on our analysis. We discuss our observations and reflections in Section 3. In Section 4, we conclude our study.

2. The Corpus

The Datatang Mandarin-English CS corpus consists of 1030 hours of audio data and 629,946 corresponding transcriptions.

The audio data were recorded from Mandarin speakers who have learned English as a second language for more than ten years. The recordings take place in natural situation and are switched spontaneously between Chinese and English. The corpus contains 24,958 unique English words, 46,000 unique Chinese words and totally 6.7 million tokens. Code-mixing Index(CMI) is adopted to evaluate the CS degree of the corpus [16]. The average CMI(C_{avg})of the corpus is 16.55. And the average number of CS points per utterance P_{avg} is 1.67. The average length of per utterance is 10.66 in Chinese characters or English words. Examples in this corpus are provided in Table 1.

Table 1: *Code-switching sentence examples in DataTang corpus.*

ID	Transcriptions
trans1	特别(especially) 配(match) 我的(my) college bag
trans2	在(at) gold coast 的(#auxiliary word) 一日游(one-day tour)
trans3	哎(ah) time cures everything 真的(really) 很(very) 美好(good)
trans4	成为了(became) 这个(this) 国家的(country's) 真正(real) 意义上的(meaningful) global citizen

3. Our Analysis

In this section, we elaborate on our analysis of CS on three levels: phoneme, word and context of the CS segments.

3.1. Phoneme

The investigation on phoneme aims at answering the first question presented at the end of Introduction –from the pronunciation perspective. First, we look into the distribution of English phonemes in Mandarin-English CS segments and English alone. Second, we study how pronunciations of English and Mandarin phonemes and their differences influence CS behavior.

3.1.1. Distribution

To compare the phoneme distribution, we choose to use the Carnegie Mellon University(CMU) Pronouncing Dictionary [17], which includes 134,000 words, as the English lexicon dictionary. We use the Google Books N-grams [18] as our source to obtain the frequency of phonemes in English corpus. We train a phoneme-level language model with the CS segments in our corpus and the probability of uni-gram is used as the occurrence frequency. In our analysis, 39 English phones are considered disregarding the stress variations. Figure 1 illustrates the distribution of these 39 uni-phones and their bi-grams in our CS corpus and in Google books. The y-axis represents the logarithmic frequency of uni-phone and bi-phone sequences. As we observe, there is no apparent difference on overall tendency between CS and English alone.

We move on to study the specific phonemes. We divide the 39 phones into four bins as shown in Figure 2, respectively corresponding to top frequent phonemes (top 50%) in both CS and native English corpus (Q_1 quadrant), CS top 50%-Native bottom 50%(Q_2 quadrant), CS bottom 50%-Native bottom 50% (Q_3 quadrant) and CS bottom 50%-Native top 50% (Q_4 quadrant). The majority of phones, 31 out of 39 phones, fall into Q_1, Q_3, which means their popularity are similar in CS and in English. However, there are 8 phonemes in Q_2 and Q_4. For instance, *DH(ð)* is popular in English words such as *rhythm* and *gather*, however is not as frequent in Mandarin-English CS speech. One possible reason is that pronunciation 'ð' is relatively hard for Mandarin speakers to pronounce.

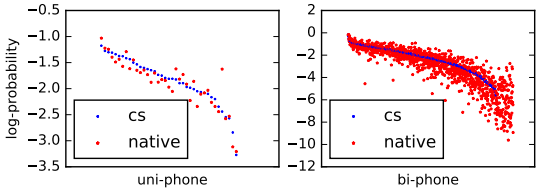

Figure 1: *Distribution of uni-phone and bi-phone in English and CS.*

3.1.2. Pronunciation

In this section, we propose to investigate if pronunciation difference between English and Mandarin phones has an effect on CS behavior. We use 66 Mandarin phonemes without regarding to tones of Mandarin in our study. We employ two methods to measure the similarity between phones. First, we use IPA phone mapping [19] to determine similar phone pairs in two languages. Second, we adopt a data-driven clustering method proposed in [20–23] on the audio recordings to find the phone pairs with shortest Bhattacharyya distance, which is a distance measure between two multivariate Gaussian distributions. Figure 2 left highlights 16 English phones that has mappings in Mandarin according to IPA. 12 of them resides in top right quadrant. Figure 2 right highlights English phones with similar Mandarin counterparts according to Bhattacharyya distance. This analysis shows a mixed story.

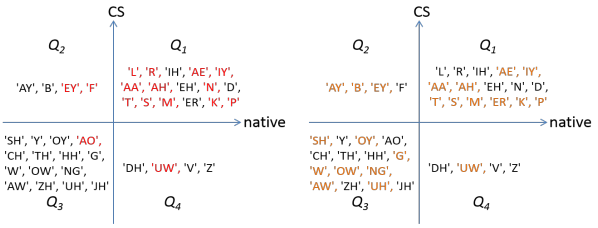

Figure 2: *Four quadrants distribution of 39 English phones, in which the left highlights the phones that has mappings in Mandarin according to IPA, and the right highlights the phones with similar Mandarin counterparts according to Bhattacharyya distance.*

Q_1 quadrant in Figure 2 shows the frequently appeared phones both in CS and English, most English phones have completely same phonetic pronunciation with Mandarin ones in IPA. And the clustered similar phone pairs share 9 phones with the IPA ones. For Q_3, the clustering mapping has many phone-pairs. It maybe because of the limited using of the words containing them both in native English and CS scenes. This limited observation shows pronunciation similarity aligns positively with the frequency of CS.

3.2. Word

In this section, we conduct our analysis of CS on the word level. There are a few of aspects we look into: distribution of code-switched English words, the complexity of them in terms of

their pronunciations, as well as syntactic and semantic commonalities they share.

3.2.1. Distribution

In order to measure the variation in word distribution, we first extract embedded English words in our corpus as CS words and words in the CMU dictionary as native English words. Word frequency is obtained from Google Books and numeric conversion in logarithmic is been done due to the huge span of frequency values. Figure 3 shows the boxplot of word distribution. The y-axis represents the logarithmic frequency. It can be seen that frequency of CS words is less spread. Furthermore, the range value of native words' log-frequency is 3.77 ± 2.69 and median is 4.01. As for CS words, the range value is 4.95 ± 2.05 and with median of 5.25. The commonly embedded words in CS are high-frequency ones used by native English speakers.

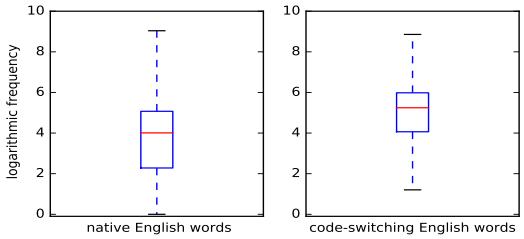

Figure 3: *Boxplots for (a) word frequency of native English words, (b) word frequency of code-switching words.*

3.2.2. Pronunciation

Our heuristic hypothesis is that if a popular English noun or verb phrase can be pronounced at much lower cost comparing to their counterparts in Mandarin, then it is more likely to be embedded. For example, *IBM* consists of 3 syllables and can be easily uttered by native Chinese. Its translation 国际商用机器公司 includes 8 syllables with tones. *IBM* is way more frequently used than its translation in Mandarin conversations. We measure the relative pronunciation complexity by the number of syllables involved on both sides. The ratio between 国际商用机器公司 and *IBM* is 8:3. The average ratio of the embedded CS segments in our corpus is 1.5. For the 50 most frequent CS English words, the ratio is 1.7. While for the 50 most frequent words in Google Books, the ratio is 1.5. This to some extent matches our hypothesis. The average number of syllables in CS English words is 2.18. The story is similar with the number of phonemes. 91.78% of the CS words consists of 2 to 7 phonemes (above 90%). And CS words with 3 to 6 phonemes are about 73.91%. The odd to have a CS word with more than 11 phonemes is about 0.61% .

3.2.3. POS of CS Words

Secondly, we implement a Part of Speech(POS) analysis on the code-mixed utterances to find out what type of words in terms of POS are more likely to be embedded. Since each CS sentence involves more than one language, we use the Stanford log-linear POS tagger for Chinese and English separately as described in [24], and the tags are derived from the Penn Treebank POS tag set for Chinese and English respectively [25], [26]. Furthermore, we compare the distribution of POS tags between the embedded English words in CS corpus and the ones in monolingual English corpus to figure out how much the POS distribution is impacted by the syntactic rule from the embedded language and how much is caused by the CS phenomenon. The Wiki news corpus [27] is downloaded and used as the monolingual corpus. Table 2 shows the top 5 POS tags of English words in CS corpus and Wiki corpus. We can see the most common POS tags in CS corpus are nouns(NN and NNS), which is close to half, followed by adjectives(JJ), and the percentage of other POS tags is relatively low. In the Wiki corpus, the percentage of ranking in the top five is relatively balanced, and the most common POS tag is proper nouns(NNP). This may be determined by the explanatory function of Wiki corpus to some extend. We have enough reason to consider that the proper nouns are also a large type to be switched, since most of them do not have corresponding translations in Mandarin. Here the NNP for CS corpus is not in the top 5 might be related with the design of the corpus which wants to cover the contents from different fields. Nevertheless, the chi-square test is carried out and the result shows that the distribution of POS tags in the CS corpus and Wiki corpus is significantly different(pValue <0.001).

Table 2: *Top 5 English POS tags in CS corpus and mono-lingual Wiki corpus.*

CS segments		Wiki corpus	
Tag	**proportion**	**Tag**	**proportion**
NN	48.65%	NNP	16.88%
JJ	12.80%	NN	14.95%
NNS	11.91%	IN	14.30%
VB	2.84%	DT	11.56%
IN	2.46%	JJ	6.90%

3.2.4. Semantic Clustering

We are motivated to find out what types of words are more likely to be embedded in terms of their meaning. We extract word embeddings directly from the Bidirectional Encoder Representations from Transformers (Bert) basic model [28] for all CS words in our corpus. They are clustered via a simple K-Means procedure into 10 bins. For visualization, we apply t-distributed Stochastic Neighbor Embedding (t-SNE) project word vectors into 2 dimensions. To assist observation, inflected forms of a given word are considered as the same word. The top 100 frequent CS words respectively for Noun are shown in Figure 4. For an example, we can observe that nouns related to *food* including *chicken, cream, salad, pizza, cake, cheese* are among top ones. Other top clusters are related to *music* in purple, *weather* in yellow, etc. Top clusters for verbs and adjectives are quite different related to *travel* and feeling.

3.3. Context

In this section, we examine the context of CS, especially the factors to trigger CS including primary categories of the proceeding words, the syntactic structure of the sentence, and the relative position that CS appears.

3.3.1. Triggering Words

POS tags are the primary categories of words according to their function in a sentence. Chinese Penn Treeback POS tag set is substantially from English. A brief description can be found in [29]. Table 3 lists the top 5 frequent parts of speech for the

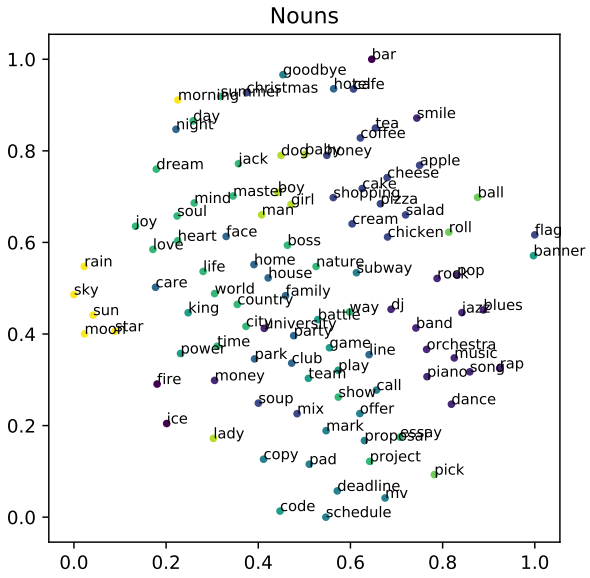

Figure 4: *Semantic clusters for top 100 embedded nouns.*

words proceeding CS. *BA* is a tag mainly for Chinese character 把, which plays a syntactic role to emphasize the following noun or pronoun as the main objective of an action. Coordinating conjunction and Measure words are usually followed by a noun or pronoun. In our CS corpus, these words have high chance to trigger CS . For instance, 38.92% of *BA* words are followed by CS. The last column shows the probability for other top frequent tags. This result echos our above conclusion that Nouns are the top category to be code-switched. In addition, Adverbial particle is also quite common in front of CS, which is in concert with the observation that Adjective and Verbs are the second and third tags only to Nouns. SB and DEV words are very likely leading to CS verbs or adjectives.

Table 3: *Top 5 Chinese POS tags for the words triggering CS.*

Chinese Pos Tag	Description	Chance To Trigger CS
BA	Preposition to emphasize	38.92%
CC	Coordinating conjunction	34.87%
SB	Passive indicator	32.58%
M	Measure word	28.04%
DEV	Adverbial particle	25.39%

3.3.2. Parsing Structure

We parse the CS sentences to capture if there is general global syntactic rules that may influence CS. The Stanford Parser is adopted as described in [30]. We list the top 5 syntactic relationships between the CS segments and their contexts in Table 4. We can observe that most major dependencies between CS and the context are Noun Compound Modifier, Direct Object, Dependent and Adjectival Modifier. These observations are expected considering the results of the POS analysis on the CS segments or the words triggering them. CS follows certain constraints of syntactic dependency from the matrix language, in accord with the Matrix Language Frame(MLF) theory to a certain degree [31]. Table 4 shows the specific frequency and the percentage that they are related to CS.

Table 4: *Top5 syntactic dependencies between CS segments and their contexts.*

Syntactic Role	Description	Chance of CS
nmod	noun compound modifier	58.76%
dobj	direct object	50.47%
dep	dependent	43.24%
amod	adjectival modifier	41.65%
cc	coordination	36.94%

3.3.3. CS Position

Additionally, we check where CS tends to appear in an utterance. The starting position of a CS segment is normalized by the length of the sentence. The relative position, ranging from 0.0 to 1.0, are divided into three bins: (0.0,0.3],(0.3,0.7],(0.7,1.0], which approximately represent the head, the body and the tail of a sentence. Overall in our CS corpus, 20.81% of the chance CS appears in the head, 43.18% in the body and 36.00% in the tail.

4. Discussion

We all know people don't code-switch randomly. The underlying constraints are multi-facts in multi-layers. We design our analysis in a hierarchical way for three levels: phonemes, words and context to specifically quantify how much they co-relate with CS. Our analysis results on each layer is obtained with a specific Mandarin-English corpus: DataTang. It would be very interesting to extend these experiments to other CS corpus for Mandarin-English and other CS langague pairs. It is one of the directions we will continue our research on. We hope our proposed analysis framework can be used systematically analyze CS behavior in more languages.

Specific results that we provide in each section can potentially help algorithm and system design to cope with CS. We reviewed some of these previous work in Introduction section. These metrics we proposed on three layers can be used as features to enhance CS data generation, CS language model adjustment, CS naturalness evaluation as well as end-to-end CS speech ASR.

5. Conclusion

This work present a systematic analysis for the characteristics of CS on three levels: phoneme, word and context. We look into distribution and pronunciation of phonemes. We highlight which English phonemes are more frequently embedded into Mandarin and how pronunciation difference on phoneme level influences the CS behavior. The overall distribution is plotted and don't show overall difference. For the word level, we investigate four aspects, overall distribution, pronunciation, syntactic role and semantics. We run comparison study comparing CS and mono-lingual English. Our experimental results have addressed our motivating questions posed in Introduction. For the context level, we study how positions of CS, triggering words as well as the sentence parsing structure affects CS. Our analysis code and the obtained results will be open-sourced once the paper gets accepted.

6. References

[1] J. Weiner, N. T. Vu, D. Telaar, F. Metze, T. Schultz, D.-C. Lyu, E.-S. Chng, and H. Li, "Integration of language identification into a recognition system for spoken conversations containing code-switches," in *Spoken Language Technologies for Under-Resourced Languages*, 2012.

[2] B. H. Ahmed and T.-P. Tan, "Automatic speech recognition of code switching speech using 1-best rescoring," in *2012 International Conference on Asian Language Processing*. IEEE, 2012, pp. 137–140.

[3] N. T. Vu, D.-C. Lyu, J. Weiner, D. Telaar, T. Schlippe, F. Blaicher, E.-S. Chng, T. Schultz, and H. Li, "A first speech recognition system for mandarin-english code-switch conversational speech," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2012, pp. 4889–4892.

[4] S. Sivasankaran, B. M. L. Srivastava, S. Sitaram, K. Bali, and M. Choudhury, "Phone merging for code-switched speech recognition," 2018.

[5] Y. Gao, J. Feng, Y. Liu, L. Hou, X. Pan, and Y. Ma, "Code-switching sentence generation by bert and generative adversarial networks," *Proc. Interspeech 2019*, pp. 3525–3529, 2019.

[6] 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 *Proceedings of the 14th International Conference on Natural Language Processing (ICON-2017)*, 2017, pp. 65–74.

[7] J. Y. Chan, H. Cao, P. Ching, and T. Lee, "Automatic recognition of cantonese-english code-mixing speech," in *International Journal of Computational Linguistics & Chinese Language Processing, Volume 14, Number 3, September 2009*, 2009.

[8] G. I. Winata, A. Madotto, C.-S. Wu, and P. Fung, "Towards end-to-end automatic code-switching speech recognition," *arXiv preprint arXiv:1810.12620*, 2018.

[9] C. Shan, C. Weng, G. Wang, D. Su, M. Luo, D. Yu, and L. Xie, "Investigating end-to-end speech recognition for mandarin-english code-switching," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6056–6060.

[10] H. Adel, N. T. Vu, F. Kraus, T. Schlippe, H. Li, and T. Schultz, "Recurrent neural network language modeling for code switching conversational speech," 2013.

[11] V. H. Do, X. Xiao, E. S. Chng, and H. Li, "A phone mapping technique for acoustic modeling of under-resourced languages," in *International Conference on Asian Language Processing*, 2012.

[12] P. Rao, M. Pandya, K. Sabu, K. Kumar, and N. Bondale, "A study of lexical and prosodic cues to segmentation in a hindi-english code-switched discourse," 09 2018, pp. 1918–1922.

[13] H. Adel, N. T. Vu, K. Kirchhoff, D. Telaar, and T. Schultz, "Syntactic and semantic features for code-switching factored language models," *IEEE/ACM Transactions on Audio Speech & Language Processing*, vol. 23, no. 3, pp. 431–440.

[14] G. A. Guzmán, J. Ricard, J. Serigos, B. E. Bullock, and A. J. Toribio, "Metrics for modeling code-switching across corpora." in *INTERSPEECH*, 2017, pp. 67–71.

[15] S. Poplack, "Syntactic structure and social function of code-switching, vol. 2," *Centro de Estudios Puertorriquenos,[City University of New York]*, 1978.

[16] B. Gambäck and A. Das, "Comparing the level of code-switching in corpora," in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, 2016, pp. 1850–1855.

[17] R. L. Weide, "The cmu pronouncing dictionary," *URL: http://www.speech.cs.cmu.edu/cgi-bin/cmudict*, 1998.

[18] J.-B. Michel, Y. K. Shen, A. P. Aiden, A. Veres, M. K. Gray, J. P. Pickett, D. Hoiberg, D. Clancy, P. Norvig, J. Orwant *et al.*, "Quantitative analysis of culture using millions of digitized books," *science*, vol. 331, no. 6014, pp. 176–182, 2011.

[19] "The ipa mapping," *URL: http://en.wikipedia.org/wiki/Arpabet*.

[20] X. Xiao, E. S. Chng, H. Li *et al.*, "A phone mapping technique for acoustic modeling of under-resourced languages," in *2012 International Conference on Asian Language Processing*. IEEE, 2012, pp. 233–236.

[21] F. Stahlberg, T. Schlippe, S. Vogel, and T. Schultz, "Towards automatic speech recognition without pronunciation dictionary, transcribed speech and text resources in the target language using cross-lingual word-to-phoneme alignment," in *Spoken Language Technologies for Under-Resourced Languages*, 2014.

[22] N. F. Chen, B. P. Lim, M. Hasegawa-Johnson *et al.*, "A many-to-one phone mapping approach for cross-lingual speech recognition," in *2016 IEEE RIVF International Conference on Computing & Communication Technologies, Research, Innovation, and Vision for the Future (RIVF)*. IEEE, 2016, pp. 120–124.

[23] X. Xiao, E. S. Chng, H. Li *et al.*, "Context dependant phone mapping for cross-lingual acoustic modeling," in *2012 8th International Symposium on Chinese Spoken Language Processing*. IEEE, 2012, pp. 16–20.

[24] K. Toutanova, K. Dan, C. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network," 2003.

[25] M. Marcus, B. Santorini, and M. Marcinkiewicz, "Building a large annotated corpus of english: The penn treebank," *Computational Linguistics*, vol. 19, no. 2, pp. 313–330, 1994.

[26] N. Xue, F. XIA, F.-D. CHIOU, and M. PALMER, "The penn chinese treebank: Phrase structure annotation of a large corpus," *Natural Language Engineering*, vol. 11, no. 2, pp. 207–238.

[27] Wiki, "Index of /enwiki/latest/," https://dumps.wikimedia.org/enwiki/latest.

[28] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding."

[29] penn treebank, "chinese-penn-treebank-part-of-speech-tagset," https://www.sketchengine.eu/chinese-penn-treebank-part-of-speech-tagset/.

[30] R. Levy and C. Manning, "Is it harder to parse chinese, or the chinese treebank?" in *Meeting on Association for Computational Linguistics*, 2003.

[31] K. Namba, "An overview of myers-scotton' s matrix language frame model," *Senri International School (SIS) Educational Research Bulletin*, vol. 9, pp. 1–10, 2004.

610

Malayalam-English Code-Switched: Speech Corpus Development and Analysis

Sreeram Manghat[1], *Sreeja Manghat*[1], *Tanja Schultz*[2]

[1]IEEE Graduate Member
[2]University of Bremen, Germany

sreeram9@ieee.org, sreejamanghat@ieee.org, tanja.schultz@uni-bremen.de

Abstract

In multilingual communities, code-switching is a common phenomenon and is predominant in most of the low resource languages as well. Many of the Indic languages fall under the category of low resource languages. Malayalam is a Dravidian language and is the official language of the Indian state Kerala. High literacy rate and media exposure in the state of Kerala makes Malayalam-English a good fit for code-switched speech research. This paper reports the first stage development of code-switched speech corpus for Malayalam-English using a dual stage approach. The speech corpus contains 20 hours of conversational speech from 42 speakers. The database has inter-sentential, intra-sentential and intra-word code-switching. Along with the description of the database creation, annotation scheme and analysis of the database is presented in this paper. The corpus will serve as a data resource for our ASR and speech synthesis research in code-switching.

Index Terms: code-switching, speech corpus, database development, low resource languages, Malayalam

1. Introduction

Multilingual speakers are those who can use more than one language and outnumber monolingual speakers [1]. The primary language acquired and used during childhood is commonly known as the mother tongue. Factors like education, technology, migration for job, cultural openness, and interaction for commercial, scientific and technological purposes can contribute to the need of communicating in more than one language and create bilingualism or multilingualism.

Code-switching is the use of more than one language by a speaker within a conversation or utterance. Previous research has shown that there is an increase in percentage of speakers around the world with code-switching in speech [2]. In most of the cases, code-switching can be seen as mixing of language elements of non-native language in the native language. Code-switching is often considered as speaker dependent and occurs within utterance positions where the syntactic rules of the native language are not violated [3]. There are common code-switching patterns shared among speakers as well [4].

India is the world's second largest country in terms of number of languages [5]. Hindi and English are the official languages of the country by its constitution [6]. The increased use of media like TV channels, Radio and IT enabled technologies in this modern globalized society motivates the use of English language in daily life. Kerala is a state in south India with a population of more than 35 million. Malayalam is the native language of more than 96% people in Kerala. Malayalam is a Dravidian language and is indigenous to the Indian sub-continent [7]. Kerala state ranks highest in literacy rate [8] and majority of the people in the state have school level or higher level education. Malayalam and English are the official languages of the state Kerala. English is taught in school starting from the primary classes itself, thus giving exposure to English language from a very early school age. This exposure to English language and media exposure can be considered as one of the major factors for high levels of Malayalam-English code-switching.

Due to the increase in the presence of code-switching in the conversational speech, Malayalam-English is a good fit for code-switched speech corpus development and research. To the best of our knowledge, we could not find any Malayalam-English code-switched speech corpus. This paper describes our two stage approach used for speech corpus development as part of our ongoing speech research. The analysis done on the speech data collected in stage two is also presented.

2. Related Work

Previous researches have shown good progress in the processing of code-switching data in the text form. Earlier there were only few formal frameworks to analyze code-switched text [9] and now there are works on code-switched data done in a large scale. Due to the influence of social media, we get larger text corpora of multilingual nature and advances in research help us have better language identification [10] and prediction of code-switch points [11]. Most of the research studies focus on text rather than speech corpus. Difficulty in database collection for code-switched speech and lack of efficient multiple language identification systems could be considered as one of the major factors for fewer works in code-switched speech. Most studies in code-switched speech corpus are limited to certain language pairs and led to improvement of code-switch language modeling [12] and code-switch automatic speech recognition (ASR) system developments [13] in those language pairs. There is a survey [14] on different language pairs in which code-switched speech corpus is developed.

Some of the major non-Indic language pairs in which studies were done include Cantonese – English [15], Mandarin – English [16] and Spanish-English [17]. A two method approach [18] was used to create Chinese-English code-switched dataset. First method was to collect text database from the websites and used a machine translation system in the second method. Code-switching due to insertion of English words was frequent in code-switched language-balanced speech corpora containing English-isiZulu, English-isiXhosa, English-Setswana and English-Sesotho language pairs [19].

A method used for Japanese-English code-switching speech corpus development [20] was by utilizing Japanese and English TTS systems. A Swiss accented bilingual French-German speech database was created [21] that contain accented speech and dialects.

Some of the code-switched speech corpus in Indic languages includes Hindi – English [22], Tamil-English, Telugu-English, and Gujarati-English [14]. However, to the best of our knowledge, the number of databases for code-switching in Indic languages is limited.

Some speech corpora are created by manually interconnecting mono-lingual segments, whereas most of the remaining databases are created by making the speakers read the scripts given to them. A system was developed for automatic detection of language boundaries in code mixed texts found in social media by taking English – Hindi and English – Bengali Facebook messages [23].

In an interview conducted for 9 University students whose native language was Hindi and they used English as much as they used their native language [22], it was found that all the students did intra-sentential code-switching. 33% of words were English words. Another database for code switched speech of crowd workers was created using HALEF system for Hindi – English and English – Spanish with more than 700 dialogs. The system was designed to create mainly intra-sentential code-switched speech between the workers and the machine [24]. There is also another ongoing effort in developing Hindi-English speech corpus [25]. A study showed increase in the code switching behavior of aphasics [26] when compared to the neurologically normal Malayalam – English bilingual speakers. There was no much difference in the quality of code-switch when compared to those code-switches in normal subjects.

A phonetically balanced speech corpus for Hindi-English read data [27] was created by selecting sentences that contained triphones lower in frequency than a predefined threshold.

One of the primary requirements in any speech research especially like ASR is the availability of database. There is an increasing trend for code-switching among bilingual speakers. Malayalam-English being a language pair with high level of code-switching, the development of speech corpus can contribute to the code-switched speech research. To the best of our knowledge, there is no code-switched speech corpus available for Malayalam-English code-switched speech.

3. Corpus design methodology

Malayalam being a low resourced language, corpus development for code-switched speech is done using an approach which ensures maximum code-switching in the minimum collected data. A two stage approach is used for the corpus development. In both the stages, speech data was collected and analyzed. The primary aim of speech data collected in stage one is to help in identifying the nature of speakers for stage two data collection so that we get stage two speech corpus with maximum code-switching. Stage one informs the speech data collection in stage two.

3.1. Stage One

Stage one of data collection was done to identify the extent and nature of code-switching among the different category of speakers. The analysis results are used to identify the type of speakers for stage two, so that the speech corpus contains maximum code-switching. In stage one, speech data was collected through conversation, interview and questionnaire. The topics of speech were random daily situations and current trends, and this ensures that the code-switched data collected is natural as much as possible. A total of 60 speakers with Malayalam as native language was part of stage one of corpus development. Speakers belong to the age group of 18-60 and 60% of the speakers were male. Speakers are fluent in Malayalam and English speaking fluency varied from low to very high. Figure 1 shows the contribution of each component in stage 1 data collection. The interviews and observations were maximum of 5 minutes in length. The observation was done by recording the natural conversation between two people.

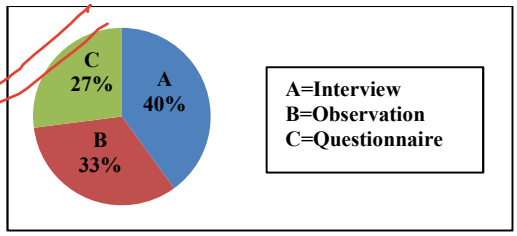

Figure 1: *Distribution of each type of data collection method*

The analysis results of stage one data collection shows that the Malayalam-English code-switched speech data collected contain inter-sentential code-switching, intra-sentential code-switching as well as intra-word code-switching. Other than borrowed words, many tend to always use certain English words such as 'school, newspaper, barber' instead of their corresponding Malayalam word. Also, it is to be noted that this data was from natural environment with varied environmental settings and was only used for the purpose of optimal speaker identification for stage two. Only basic transcription was done manually in English script with markers for code-switching.

Table 1: *Amount of code-switch events for different groups*

	Group 1 (R_n=1)	**Group 2 (R_n=2)**	**Group 3 (R_n=3)**	**Group 4 (R_n=4)**
Inter-sentential	45%	36%	31%	28%
Intra-sentential	32%	28%	22%	18%
Intra-word	2%	4%	3%	2%

To identify a subset of participants who do maximum code-switching, we analyzed the amount of code-switch events with regard to four speaker parameters - Time spent outside the state where speaker was born (Group 1), English language used at work (Group 2), English language used at home (Group 3) and Educational background (Group 4). Group 1 is the group of speakers who spent minimum of 5 years outside the state where the speaker was born. Group 4 consists of speakers who have minimum graduation level education and their medium of education was English. Table 1 shows the percentage of code-switching in the collected data. While grouping, only one parameter was considered at a time and there are utterances without code-switching. There are speakers who are part of more than one group. We also found that some of the common factors which affected this code-switching are ease of substitution, influence of technology and emotional expression.

A confidence score index was developed to identify whether a new speaker selected randomly can contribute significantly for the code-switched speech corpus development, once we know the background parameters of the speaker. The speakers with maximum confidence score among the available ones can be selected for speech corpus development.

Each of the parameters is given ranks (R_n) 1 to 4 as shown in Table 1. For a speaker k, the confidence score CS_k is calculated as below:

$$CS_k = \Sigma (C_R / R_n)$$

where R_n is the rank of the parameter and C_R is the extent of code switching in that category for speaker k. Every speaker will have a maximum score of 1 for each parameter. For example, if a speaker has never stayed outside the native state, C_R for that parameter is zero. A speaker spending 5 years or more outside the native state will have C_R as 1. The same formula can also be extended for separately calculating the confidence score for inter-sentential as well as intra-sentential code switching.

Also, it was noted that the nature and type of code-switching done by the speakers with confidence score less than one was to a large extend the subset of those with confidence score more than one. This further ensures the speaker selection using the developed confidence score to have lesser bias.

3.2. Stage two

In stage two, the Malayalam English code-switched speech corpus is created. The category of speakers was selected considering the analysis results from stage one. The speakers selected for stage two have a minimum confidence score of 1.25. 42 speakers were part of stage two data collection. All the speakers selected for stage two have minimum high school level education and Malayalam is their first language. The speakers are in the age group of 18-60 and 55% of the speakers are male. The recordings were interviews and of general topics. The questions were related to current trends, background of the speaker as well as their profession. For this purpose, a professional interviewer was engaged who can conduct conversations and interviews of different topics depending on the speaker. This will help in ensuring the response to be natural as much as possible.

Table 2: *Types of code-switching*

Type	Example
Inter-sentential	എന്റെ വീട് കേരളത്തിൽ ആണ്. I Love my city. (My house is in Kerala. I love my city)
Intra-sentential	ഞാൻ ഇന്നലെ cinema കണ്ടു. (I saw a movie yesterday)
Intra-word (English-Malayalam)	directorന്റെ (Director's)
Intra-word (Malayalam-English)	കുട്ടികൾs (kids)

The speech was recorded at 44.1kHz sampling rate and 16 bit resolution. The UTF-8 code orthographic transcription is used for both English and Malayalam script. In stage two of data collection, a total of 20 hours of speech data is obtained from 42 speakers. Table 2 show the types of code-switching found in the speech data collected during stage one. It was found that English-Malayalam intra-word code-switching is predominant when compared to Malayalam-English intra-word code-switching. In addition to this, data was also collected from social media using python script and Twitter API. We collected 20,000 words from Twitter which contains intra-word code-switching and slang words. The script obtained was in English. This ensures to have maximum number of code-switched words in the database.

3.3. Transcription and Annotation

The speech data was manually transcribed into utterances by engaging three bilingual transcribers who are fluent in English as well as Malayalam. The file format of transcription is compatible for ASR system development. Each utterance has a file ID, speaker ID, utterance ID, start time and stop time. Each speaker data was typically 25 to 35 minutes speech in length. The software audacity was used for labeling the utterances. Figure 2 shows a sample utterance which contains Malayalam, English-Malayalam intra-word code-switching and English parts. ***Exam-inte,*** meaning 'of exam' is an example of English-Malayalam intra-word code-switched word.

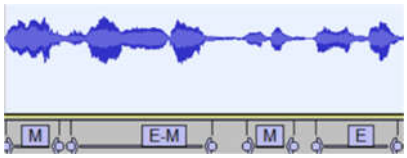

Figure 2: *Example of a typical code-switched utterance with notation M for Malayalam and E for English*

Malayalam words are transcribed in Malayalam script and English words are transcribed in English script. The English and Malayalam part of intra-word code-switched words are transcribed in their respective scripts. Out of vocabulary words are transcribed in English script. Accent category is also marked during transcription. Each speech data took a transcription rate of 30 to 40 times the real time (xRT) and validation effort of 10xRT. Each transcription is verified twice for timing errors, typo errors, language boundary errors as well as code-switch specific transcription error in each language.

Annotation scheme

The annotation process includes marking of language boundaries and code-switching boundaries. In the annotation scheme, each speaker is given a speaker ID. The ID starts with either 'F' or 'M' for speaker gender and a three-digit speaker number. At the moment, manual annotation is done with utterance time stamps marked. Each transcription was first verified by the same transcriber who did the transcription and then it was verified manually by another transcriber. Transcribers were given a set of guidelines. Like, during the process of transcription, initial transcription was done in English script since it will make the initial process faster. Later, Malayalam words and Malayalam part of the intra-word code-switched words are manually transliterated with the help of an online transliteration tool. Words like borrowed words, names of person and place are transcribed in English script.

Table 3: *Transcription errors and agreement*

Error	Agreement
Kanam (കനം) and Kanam (കാനം)	Transcribe kanam (കാനം) as Kaanam (C-V-V)
Manju (മഞ്ജു) and Manju (മഞ്ഞു)	Transcribe manju (മഞ്ഞു) as manjju
map (മാപ്) and map (മാപ്പ്)	Trascribe map (മാപ്പ്) as mapp
Manam (മനം) and manam (മണം)	Transcribe manam (മണം) as man1am

One of the major errors that was encountered during transcription is during conversion of Malayalam words from English script to Malayalam script. For example, the Malayalam word "കാളം (kaalam)" in the initial English script can be kaalam (കാളം) or kalam (കളം) where both കാളം and കളം are valid dictionary words. Another situation is where we do not have exact transliteration in English script. ശ (IPA:ʃa) and ഷ (IPA:ʂa) have same transliteration "sha" in English. In such case, ഷ is represented as sh1a. These words were verified again. Some

of the cases are shown in Table 3 and inter-transcriber agreement was made to avoid similar errors in future transcription.

During transcription, sentence and word boundaries are marked. Apart from this, prosodic annotation like pause duration is also marked with custom labels given to the transcriber. Interruptions or pauses include pauses within words and between words. The pause can be short pause or long pause and these are separately marked. At this stage, only three common accents are considered and they are south Kerala accent (SK), central Kerala accent (CK) and Malabar/North Kerala accent (NK). This is the first level of annotation and we expect to add more specifiers like tags for parts of speech (POS) and emotions as the speech research progresses.

4. Analysis of the corpus

Analysis was done on the Malayalam-English speech corpus collected during stage two of our corpus development. The utterances contain English words, Malayalam words and intra-word code-switched words.

Table 4: Statistics of speech corpus

Total number of utterances	22640
English words	32%
Malayalam words	66%
Intra-word code-switched words	2%

.

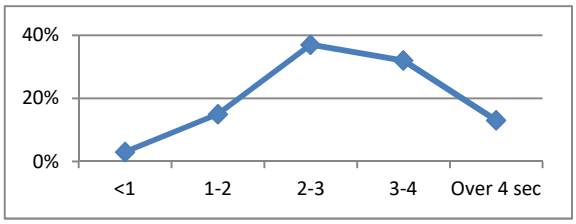

Figure 3: *Distribution of utterance length*

Table 4 shows the total number of utterances and the distribution of different types of words in the speech data. It is seen that two third of the total number of words are Malayalam words. Utterances were of different lengths as shown in Figure 3 and most number of utterances was of average length 3 seconds.

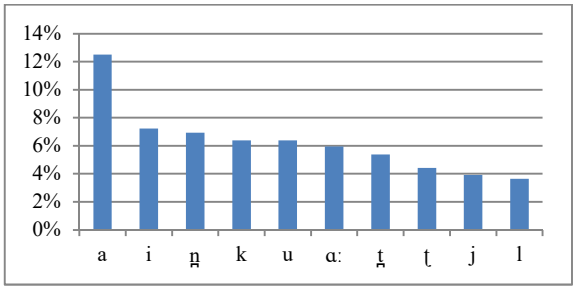

Figure 4: *Frequency of Malayalam phonemes*

Phoneme analysis was done on the phonetic dictionary obtained using the developed code-switched grapheme to phoneme system [28]. The data taken for this analysis is the transcript of the speech corpus. Neither Malayalam nor English are phonetic subset of each other. However there are total of 25 overlapping phonemes. Figure 4 and Figure 5 shows the ten most frequent Malayalam phonemes and English phonemes respectively, found in the database collected in stage two. Among these prominent phonemes, it can be seen that there are 3 overlapping phonemes (*i, k, l*) as well.

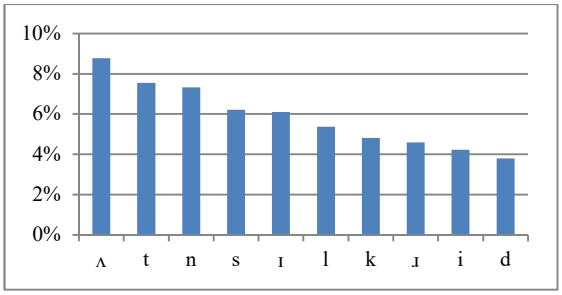

Figure 5: *Frequency of English phonemes*

As shown in Table 5, it is observed that, English-Malayalam intra-word code-switching is very predominant when compared to Malayalam-English code-switched words. English Malayalam intra-word code-switched words contribute to the 95% of the total number of code-switched words.

Table 5: *Intra-word code-switching*

	Example	**Percentage**
M-E	കുട്ടികൾ**s** (children)	5%
E-M	director**ൻറെ** (Director's)	95%

Table 6 shows the different types of utterances in the speech corpora. Non-word utterances are those which include speech elements like sigh, laugh, um etc. It is seen that intra-sentential code-switched utterances appear almost three times more often of that of utterances with only English words. This is due to the usage of many common words and phrases directly from English even though it might have a corresponding word or phrase in Malayalam.

Table 6: *Types of utterances*

Utterances with only Malayalam words	21%
Non-word utterances	6%
Utterances with only English words	17%
Utterances with at least one intra-sentential code-switch	47%
Utterances with at least one intra-word code-switch	6%
Utterances with at least one intra-sentential code-switch as well as intra-word code-switch	3%

5. Conclusion

A Malayalam-English code-switched speech corpus is developed using a two stage approach. Primary analysis was done on the stage one data to identify the type of speaker who can contribute with maximum code-switched speech data. Depending on this analysis, a method of confidence score to choose the optimal speaker for a practical type of data collection has been presented. Using the information obtained from stage one and confidence score, speakers were selected for stage two of corpus development. Transcription and basic annotation process is presented. The speech corpus contains inter-sentential code-switching, intra-sentential code-switching and intra-word code-switching. Primary analysis like type of utterances, distribution of utterance length and frequency of phonemes was done on the speech corpus developed. We expect to use this speech corpus as the primary dataset for our ongoing speech research.

6. References

[1] G. R. Tucker, *A Global Perspective on Bilingualism and Bilingual Education*, CMU, 1999.

[2] B. E. Bullock and A. J. Toribio, "The Cambridge Handbook of Linguistic Code-switching", Cambridge University Press, 2009.

[3] N. T. Vu, H. Adel and T. Schultz, "An investigation of code-switching attitude dependent language modeling," *International Conference on Statistical Language and Speech Processing*, SLSP, pp 297-308, 2013.

[4] S. Poplack, "Sometimes i'll start a sentence in Spanish y termino en Espanol: Toward a typology of code-switching," *Linguistics*, vol. 18, no. 7-8, pp. 581–618, 1980.

[5] C. Moseley, *Encyclopedia of the World's Endangered Languages*, Routledge, 2008.

[6] *"*Constitution of India*", Available at: https://www.india.gov.in/sites/upload_files/npi/files/coi_part_full.pdf* [Accessed: 10 September 2019].

[7] "Census of India 2011", *Available at: http://censusindia.gov.in* [Accessed: 21 September 2019].

[8] G. Ramachandra, "6. Ideas of India (section IX)". *India After Gandhi: The History of the World's Largest Democracy*, Pan Macmillan, pp. 117–120, 2011.

[9] A. K. Joshi, "Processing of sentences with intra-sentential codeswitching", *Proceedings of the 9th conference on Computational linguistics*-Volume 1. Academia Praha, pp. 145– 150, 1982.

[10] U. Barman, A. Das, J. Wagner and J. Foster, "Code mixing: A challenge for language identification in the language of social media", *EMNLP 2014*, vol. 13, 2014.

[11] T. Solorio and Y. Liu, "Learning to predict code-switching points", *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, pp. 973–981, 2008.

[12] H. Adel, N. T. Vu and T. Schultz, "Combination of recurrent neural networks and factored language models for code-switching language modeling", *ACL* (2), pp. 206–211, 2013.

[13] N. T. Vu, D. C. Lyu, J. Weiner, D. Telaar, T. Schlippe, F. Blaicher, E. S. Chng, T. Schultz and H. Li, "A first speech recognition system for mandarin-english code-switch conversational speech," in *Acoustics, Speech and Signal Processing (ICASSP 2012)*, pp. 4889–4892, 2012.

[14] S. Sitaram, K. R. Chandu, S. K. Rallabandi and A. W. Black, "A Survey of code-Switched Speech and language Processing", in arXiv, arXiv:1904.00784v3 [cs.CL], 2020

[15] J. Y. Chan, P. Ching and T. Lee, "Development of a cantonese-english code-mixing speech corpus" in *Interspeech 2005*, pp. 1533–1536, 2005.

[16] D. C. Lyu, T. P. Tan, E. S. Chng and H. Li, "SEAME: a Mandarin-English Code-switching Speech Corpus in South-East Asia", *Language Resources and Evaluation*, Interspeech 2010, Japan, 2010, pp. 1986–89.

[17] V. Soto, N. Cestero and J. Hirschberg, "The Role of Cognate Words, POS Tags and Entrainment in Code-Switching". in *Interspeech 2018*, pp.1938-1942, 2018.

[18] H. Shen, C. Wu, Y. Yang and C. Hsu, "CECOS: A Chinese-English code-switching speech database," in *International Conference on Speech Database and Assessments (Oriental COCOSDA 2011)*, Taiwan, pp. 120-123, 2011.

[19] E. Westhuizen and T. Niesler, "A First South African Corpus of Multilingual Code-switched Soap Opera Speech", in *Proceedings of the 11th edition of the Language Resources and Evaluation Conference (LREC 2018)*, 2018.

[20] S. Nakayama, T. Kano, Q. T. Do, S. Sakti and S. Nakamura, "Japanese-English Code-Switching Speech Data Construction," *2018 Oriental COCOSDA - International Conference on Speech Database and Assessments*, Miyazaki, Japan, 2018, pp. 67-71.

[21] D. Imseng, H. Bourlard, H. Caesar, P. N. Garner, G. Lecorvé and A. Nanchen, "MediaParl: Bilingual mixed language accented speech database," *2012 IEEE Spoken Language Technology Workshop (SLT)*, Miami, FL, 2012, pp. 263-268.

[22] A. Dey and P. Fung, "A Hindi-English Code-Switching Corpus", in *Proceedings of the 9th edition of the Language Resources and Evaluation Conference (LREC 2014)*, p.2410-2413, 2014.

[23] D. Amitava and G. Björn, *Code-Mixing in Social Media Text: The Last Language Identification Frontier?*. Revue TAL 2015, 54(3):41-64, 2015.

[24] V. Ramanarayanan and D. S. Oeft, "*Jee haan, I'd like both, por favor*": Elicitation of a Code-Switched corpus of Hindi-English and Spanish-English Human – Machine Dialog, in proceedings of: *Interspeech 2017*, 2017.

[25] S. Ganji, K. Dhawan and R. Sinha, "IITG-HingCoS corpus: A Hinglish code-switching database for automatic speech recognition", Speech Communication Journal. Volume 110, July 2019, Pages 76-89, 2019.

[26] S. Chengappa, K. E. Daniel and S. Bhat, "Language mixing and switching in malayalam - english bilingual aphasics", *Asia Pacific Disability Rehabilitation Journal*, Vol. 15 No. 2, 2004.

[27] A. Pandey, B. M. L. Srivastava, R. Kumar, B. T. Nellore, K. S. Teja and S. V. Gangashetty, "Phonetically balanced code-mixed speech corpus for Hindi-English automatic speech recognition", in *Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC 2018), Japan, 2018, pp.1480-1484.*

[28] S. Manghat, S. Manghat and T. Schultz, "Malayalam-English Code-Switched: Grapheme to Phoneme System", accepted at: *Interspeech 2020*, Shanghai, 2020.

1115

Understanding forced alignment errors in Hindi-English code-mixed speech – a feature analysis

Ayushi Pandey[1*], *Pamir Gogoi*[2*], *Kevin Tang*[2]

[1]Sigmedia, ADAPT Centre, School of Engineering, Trinity College Dublin, Ireland
[2]Department of Linguistics, University of Florida, Gainesville, FL, U.S.A., 32611-5454

pandeya@tcd.ie, pgogoi@ufl.edu, tang.kevin@ufl.edu

Abstract

Forced alignment methods have recently seen great progress in the fields of acoustic-phonetics studies of low-resource languages. Code-mixed speech however, presents complex challenges to forced-alignment techniques, because of the longer phonemic inventory of bilingual speakers, the nature of accented speech, and the confounding interaction of two languages at a frame level. In this paper, we use the Montreal Forced Aligner to annotate the Phonetically Balanced Code-Mixed read-speech corpus (7.4 hours; 113 speakers) in 3 different training environments (code-mixed, Hindi and English). Additionally, we present an analysis of alignment errors using phonological and data-driven features using Random Forest and Linear mixed effects models. We find that contextual influence of neighbouring phonemes influences the error in alignment most significantly, when compared against any other features. Many of the alignment errors by phonological features can be explained by their acoustic distinctiveness. Additionally, the amount of training data by phone type also contributed to lowering their respective error rates.

Index Terms: code-mixing, code-switching, forced alignment, error analysis, speech recognition

1. Introduction

Forced alignment techniques have become increasingly popular in the analysis and description of speech data. After successful performance of forced aligners on large scale monolingual speech resources [1, 2, 3], a variety of non-traditional speech resources [4, 5, 6] have also been explored in the past decade. However, code-mixed speech may present a challenge for automatic speech alignment, given the complex interaction of two languages at the frame level within the same utterance.

Code-mixed speech is frequently observed in bilingual and multilingual communities all across the world. Speakers exhibit alternation either within the utterance (code-mixing) or at a clausal or phrasal level (code-switching). In light of its relevance and complexity, techniques on processing [7, 8, 9, 10] code-mixed speech have actively been explored in the past few years. Until a few years ago, prevalent practices in the domain have included two-pass approaches; first detecting language and then recognizing using appropriate acoustic models. At the same time, adapting monolingual speech resources have also seen active investment [11, 12]. More recently, dedicated neural network architectures [13, 14, 8], augmentation of existing datasets [10] and several fine-grained problems [15, 16] in speech recognition are being unearthed, underscoring the relevance of these studies even further.

*authors of equal contribution

With the rise in computational speech processing, theoretical questions surrounding this phenomena may also arise, as would a discussion on their tools and techniques. In this paper, we present a step in this direction. We compare the performance of Montreal Forced Aligner [1] against gold-standard annotations over isolated English words from the Phonetically Balanced Code Mixed corpus of Hindi-English read speech [17]. We present results in three word-level training environments; a) with pooled Hindi and English data, b) with monolingual Hindi data, and c) within corpus monolingual English data. Finally, using Random Forests and Linear Mixed Effects models, we present an analysis of forced alignment errors, against a set of phonological and data-driven predictors.

The organization of the paper is as follows: Section 2 describes the acoustic data, the pronunciation lexicon, and the gold-standard annotation procedure. Section 3 details the experimental procedure, with descriptions of the training datasets for alignment, and the analytical models for evaluation. Section 4 discusses the results of the analysis, and presents comparison between each type of experiment over different models. Section 5 concludes the paper.

2. Data

2.1. Acoustic data

The Phonetically Balanced Code Mixed (PBCM) corpus [17] contains 6,941[1] phonetically balanced sentences, recorded at IIIT-Hyderabad. Sentences for this corpus were compiled using an optimal selection procedure from selected sections of a prominent Hindi newspaper, Dainik Bhaskar. The speech data for this corpus was recorded by 113 native Hindi speakers (58 male, and 55 female), all of whom were fluent in English.

	Hindi	**English**
word (types)	4790	3754
word (tokens)	54961	18839
phoneme (types)	73	52
phoneme (tokens)	194672	97137

Table 1: *Distribution of Hindi-English words and phonemes in the PBCM corpus.*

2.2. Pronunciation data

The PBCM corpus was originally transcribed using the Wx notation [18], which is a popular transcription metric for Indian languages, especially for NLP and related purposes. For conducting acoustic-phonetic studies however, we compared the pronunciations generated by Espeak, Epitran and Wx. We

[1]more phonetically balanced sentences were added after the original publication, which reports 6,126 utterances

found Espeak (http://espeak.sourceforge.net/) to be the best pronunciation scheme, particularly for cases where pronunciation was not predictable by orthography. Some errors, however, still persisted. Errors of nasalization and syllabification (irregular schwa insertion) were corrected through a combination of manual edits and phonological rules that ensured homorganic nasal-consonant clusters.

These corrections could still not accommodate the speaker-specific variation between the **/ʤ-z/** and **/p^h-f/** contexts, which were not always distinct in the orthography. Manual inspection revealed very little speaker variation in the /p^h-f/ context, with most speakers preferring the /f/ variant. But for the /ʤ-z/ context, speakers were found to compensate for the orthographic inconsistency, resulting in variant pronunciations of the ʤ words. To overcome the problem of lesser represented phonemes, bootstrapping techniques are prevalent in the forced alignment literature [5, 6, 19, 20, 21]. In such models, target phonemes in the lexicon are mapped to more frequent and closely resembling phonemes. Therefore, we decided to create several bootstrapped versions of each of the variants (in /ʤ-z/ and /p^h-f/) lexicon, and allowed the Montreal Forced Aligner [1] to pick the most appropriate variant. The pronunciation selected by the best performing bootstrapping model was chosen to be listed in the lexicon. Thus, we created a variant free lexicon where the /p^h-f/ variation was mapped entirely to /f/ and each pronunciation of the /ʤ/ words were given a unique identity.

2.3. Gold-standard alignment data

After the development of a variant-free lexicon, each word in the dataset was manually given a) Hindi, b) English and c) part-Hindi tags, by two Hindi speakers (one fluent, one native). Part-Hindi tags referred to switches between Hindi and English at a morphological level [22] (e.g, *amerik-i*, for American). Such words were removed from the analysis. Among the English words, 10% of the words were selected for gold-standard annotation. To maintain consistent speaker variation, 6 sentences from each speaker's set of sentences were chosen. Word and phoneme level boundaries for each of these words were manually annotated and cross-evaluated by the two Hindi speakers. Forced alignment results obtained from each training environment (discussed Section 3.1) will be analyzed against the words in this dataset.

3. Experimental setup

This section describes the experimental procedure for the analysis of force aligned English words. First, we discuss the 3 training environments using each of which, the isolated English words were aligned. We then introduce the phonological and data driven features that were used as predictors for the evaluation of the forced alignment.

3.1. Acoustic models

As a preliminary step, the complete PBCM corpus was used as training as well as alignment data. This initial alignment resulted in word and phoneme boundaries for every sentence in the corpus. Using this dataset, and the TextGridTools [23] package, we separated the data into word level chunks. Subsets of this data were created in the following way:

- **Code-mixed Words (CoM-W):** All the extracted words from the aligned sentences were used as training environment for this experiment. This included Hindi words, English words, as well as those English words that carried Hindi inflection (for example: "amerik-i"). The purpose of this sub-experiment was to maximize coverage for each phoneme within the training dataset, and pool Hindi and English word-level data for the alignment.
- **Hindi Words (Hin-W)** From the aforementioned word-level dataset, monolingual Hindi words were separated and a new dataset was created. The purpose of this sub-experiment was to assess the reliability of forced alignment in absence of any English data. This type of setup also probes the question of the accuracy of alignment when all the phonemes are well represented, but the phonotactic information of English words is withdrawn.
- **English Words**: In the same vein as the previous setup (Hin-W), monolingual English words separated from the CoM-W dataset, and an English-only model was created. The purpose of this setup was to evaluate the role of monolingual English data in identical training and alignment environments.

Each of these datasets was then sent to align the isolated English words dataset (Eng-W trained and aligned on itself), using the speaker-adapted triphone model. From the obtained forced alignment timestamps, Midpoint (the central timestamp between left and right boundary) for each phoneme were extracted. Similarly, this Midpoint value was extracted for the gold-standard annotations as well. The absolute difference between the Gold-standard Midpoint and the Forced-Aligner Midpoint will serve as the main dependent variable for each of the subsequent analyses.

3.2. Predictors

In this subsection, we discuss each of the features (both phonological, and data-driven) that were used as predictors to evaluate the accuracy of the forced alignment. To consistently maintain the number of features per phoneme, all phonemes in the corpus were uniquely specified for each of these features.

3.2.1. Consonant-specific feature set

The following set of features were specified for each consonant in the inventory:

- Manner of articulation
- Place of articulation
- Voicing type

Existing literature on error evaluation of speech recognition models and forced aligners have shown that some of these features are recognised better than others [24], [6]. Mel-frequency cepstral coefficients (MFCCs) are the standard acoustic speech signal representations in speech recognition, and indeed in forced alignment models. [24] compared that the performance of an MFCC-based recognition system with an articulatory based system trained on German speech, some articulatory features were suboptimally encoded by MFCCs, such as labial, coronal, dental, palatal, velar, fricative, –round, high, back and –voice. Similarly, [6] finds that a cross-lingual forced alignment of non-English speech using English models performed better on natural classes of stops, fricatives, nasals and affricates. These results supported our motivation for separating our consonants and vowels into sets of natural classes of place, manner and voicing features. In general acoustic terms, consonant specific features, fricatives and stops were hypothesized to be well aligned given their prominent acoustic characteristics. For instance, the boundaries of fricatives are marked by a section of

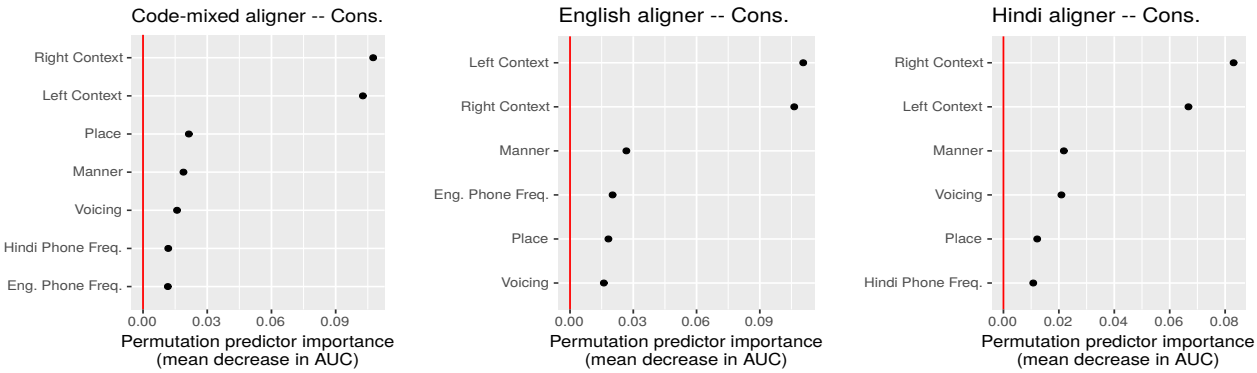

Figure 1: *Variance importance for predicting Error$_{consonant}$ using consonant specific features across the CoM-W, Hin-W and Eng-W training environments*

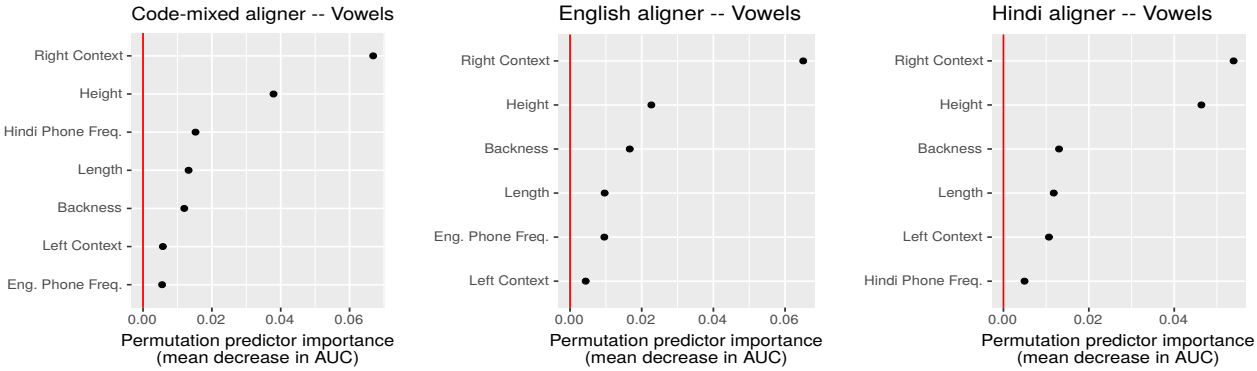

Figure 2: *Variance importance for predicting Error$_{vowel}$ using consonant specific features across the CoM-W, Hin-W and Eng-W training environments*

noise and those of stops are marked by a clear burst in most cases, suggesting better alignments, while approximants poorly so, given their dynamic transitions into adjacent vowels.

3.2.2. Vowel-specific feature set

The following set of features were specified for each vowel in the inventory:

- Vowel height
- Vowel frontness/backness
- Vowel length

Among these, height and backness were expected to influence the alignment quality more significantly, especially with high and back vowels given the findings by [24].

3.2.3. Data driven feature set

The following set of features were specified for each phoneme in the inventory, regardless of its segment type (consonant or vowel). However, the phone frequency are relevant only selectively for particular training environments. For example, the frequency of English phone is not relevant for training environments with Hindi monolingual words.

- Right boundary
- Left boundary
- Phone frequency in Hindi Words
- Phone frequency in English Words

Here, the right and left boundary indicated whether the target phone is surrounded by a vowel, a consonant or is a boundary phone. Due to coarticulatory effects, and the triphone modeling inherent in the Montreal Forced Aligner, the influence of the neighbouring environment was hypothesized to be strong on the alignment quality. Similarly, it was important to analyze the error in terms of the frequency of the phone in the Training environment.

3.3. Statistical models

This subsection gives an overview of the statistical models developed for use in the later part of the analysis. We use two statistical models: Random Forests, and Linear mixed effects model to analyze the Error under each training environments.

3.3.1. Random forest

Random forests have emerged as powerful tools in estimating the importance of individual features in the prediction of a dependent variable. To minimize the effect of speaker variation on alignment errors, the predicted variable Error was passed through a per speaker z-score normalization. Then, using the Party package [25, 26] in *R*, a random forest model was used to analyze Error for each of the vowel and consonant type. The entire set of *relevant* features in each case was used as predictor variables for each type of segment (vowel or consonant) and experiment. Therefore, for example:

$$Error_{vowel} \sim Height + Backness + Roundedness + Left.Context + Right.Context + Eng.Ph.Freq + Hin.Ph.Freq + Length$$

Here, equation 1 represents the model equation for Error in vowels in the CoM-W context. All the vowel specific features have been specified in the model, in addition to the global features. Their individual influence is described in the Results section, and can be seen in Figure 2.

3.3.2. Linear mixed effects model

Random forest and linear mixed effects models are complementary statistical methods [27]. While random forest will provide the *overall* importance of the variables of interest (factorial or continuous), mixed-effects model will be used to highlight how these variables affect the amount of alignment errors, while being able to capture random-effect factors such as the speakers and words in the sample. The model structure will be largely the same as 3.3.1 but with the addition of by-speaker and by-word random intercepts.

$$Error_{vowel} \sim Height + Backness + Roundedness + Left.Context + RightContext + Eng.Ph.Freq + Hin.Ph.Freq + Length + (1|Speaker) + (1|Word)$$

4. Results

This section presents the results obtained from comparing the Error computed using the absolute difference in Midpoints per phoneme, from the gold-standard annotations and the forced-aligned annotations. Error thus obtained will be modeled as a *predicted* variable using RandomForest and Linear Mixed Effects modeling. RandomForest and will be presented for vowels and consonants separately.

4.1. Descriptive Statistics

Table 2 displays the error tolerance levels (in msec) for each of the training environments (CoM-W, Hin-W and Eng-W). This means that, when trained on code-mixed word level data, 52.58% phoneme boundaries matched gold-standard annotations within $<$10 ms. A comparison across rows (different training models) shows that CoM-W was the best performing model, with the greatest coverage of phonemes in the $<$10 ms range.

Table 2: *Comparison of tolerance (in msec) of the three models*

	Tolerance (msec)				
	$<$10	**$<$20**	**$<$30**	**$<$40**	**$<$50**
CoM-W	52.58	83.00	94.54	97.33	98.94
HIN-W	50.75	80.37	92.56	96.14	98.16
ENG-W	51.89	82.71	94.12	97.27	98.86

4.2. Random Forest

Figures 1 and 2 display the relative importance of each feature for predicting error. The deviance from the red axis indicates the relative strength of each feature in predicting the error in alignment. Across all the training environments (CoM-W, Hin-W, Eng-W), the Right Context and the Left Context appear to be a lot more influential in the error prediction, compared with the phonological features of the target phone. This reflects the nature of consonants being encoded partly in surrounding phones especially vowels, e.g. stops. Since about a third (34%) of consonants in the corpus are stop consonants, which rely on transitional cues, it is possible that co-articulation may be governing this observation. Consistent patterns like these are not quite so clear among other features. The representation of the target phone in the training corpus is significant, but once again, does not influence the error as much. This indicates that simply increasing the individual presence of a phoneme may not be helpful enough, unless its supporting context is present. The vocalic context appears to be largely influenced by its Right boundaries, but not as much as by its left boundaries. This suggests that the vowels are better supported by their left boundaries, likely from an onset consonant, compared to a coda consonant, a well-established phonetic universal. Let us explore these effects in a more granular fashion in the next subsection.

4.3. Linear Mixed-Effects Regression

Using the lmerTest [28] package in *R*, we predicted the Error variable using fully specified models for vowels and consonants separately. Similar to the observations obtained in Random Forest model, we found a significant effect of surrounding phones ($p < 0.001$) on the alignment error in the consonantal context. For each of the training environments, non-boundary phones have higher error than boundary phones. Reverse trends were observed in [6], where non-boundary phones showed better alignment. However, the comparison with our results is not straightforward: because the analysis in [6] was conducted using monophone based aligners (HMAlign and P2FA), and Montreal Forced Align is a speaker-adapted triphone acoustic model. Through our LMER analysis, we found increased error for *consonants* due to both their left and right context, but for *vowels*, but b) vowels appear to be influenced largely by their Right boundaries across all training environments. While the case for consonants is not so clear, there is evidence that the onset (left boundary) cues for the vowel are perceptually more significant, than offset (right boundary) cues [29]. The presence of aspiration causes the formant transitions to be more stable across the initial and steady-state portion of the vowels [30, 31], perceptually supporting the recognition of the vowel.

5. Conclusion

In this paper, we conducted forced alignment on code-mixed speech from the PBCM Hindi-English read speech corpus. We created 3 types of acoustic models, code-mixed words (CoM-W), Hindi words (Hin-W) and English words (Eng-W) from the PBCM corpus. A variant-free Hindi-accented lexicon used was consistent across all the training datasets. We found that despite having only half the number of phonemes in the training corpus, the monolingual English word model performs better than the monolingual Hindi word model. This suggests that despite having increased phoneme representation, we may not achieve better alignment quality, if phonotactic information is absent. Similarly, in the RandomForest and linear mixed effect model analysis, we found that contextual information was most significant in influencing the errors of alignment.

6. References

[1] M. McAuliffe, M. Socolof, S. Mihuc, M. Wagner, and M. Sonderegger, "Montreal Forced Aligner: trainable text-speech alignment using Kaldi," in *Proceedings of the 18th Conference of the International Speech Communication Association*, 2017, pp. 498–502.

[2] I. Rosenfelder, J. Fruehwald, K. Evanini, and J. Yuan, "Fave (forced alignment and vowel extraction) program suite," *URL http://fave. ling. upenn. edu*, 2011.

[3] K. Gorman, J. Howell, and M. Wagner, "Prosodylab-aligner: A tool for forced alignment of laboratory speech," *Canadian Acoustics*, vol. 39, no. 3, pp. 192–193, 2011.

[4] T. Knowles, M. Clayards, M. Sonderegger, M. Wagner, A. Nadig, and K. H. Onishi, "Automatic forced alignment on child speech: Directions for improvement," in *Proceedings of Meetings on Acoustics 170ASA*, vol. 25, no. 1. Acoustical Society of America, 2015, p. 060001.

[5] K. Tang and R. Bennett, "Unite and conquer: Bootstrapping forced alignment tools for closely-related minority languages (Mayan)," in *Proceedings of the 19th International Congress of Phonetic Sciences, Melbourne, Australia 2019*, S. Calhoun, P. Escudero, M. Tabain, and P. Warren, Eds. Canberra, Australia: Australasian Speech Science and Technology Association Inc., 2019, pp. 1719–1723. [Online]. Available: https://assta.org/proceedings/ICPhS2019/papers/ICPhS_1768.pdf

[6] C. DiCanio, H. Nam, D. H. Whalen, H. Timothy Bunnell, J. D. Amith, and R. C. García, "Using automatic alignment to analyze endangered language data: Testing the viability of untrained alignment," *The Journal of the Acoustical Society of America*, vol. 134, no. 3, pp. 2235–2246, 2013. [Online]. Available: https://doi.org/10.1121/1.4816491

[7] S. Sitaram, K. R. Chandu, S. K. Rallabandi, and A. W. Black, "A survey of code-switched speech and language processing," *arXiv preprint arXiv:1904.00784*, 2019.

[8] S. Sivasankaran, B. M. L. Srivastava, S. Sitaram, K. Bali, and M. Choudhury, "Phone merging for code-switched speech recognition," 2018.

[9] 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 *Proceedings of the 14th International Conference on Natural Language Processing (ICON-2017)*, 2017, pp. 65–74.

[10] E. Yılmaz, H. v. d. Heuvel, and D. A. van Leeuwen, "Acoustic and textual data augmentation for improved asr of code-switching speech," *arXiv preprint arXiv:1807.10945*, 2018.

[11] K. Bhuvanagirir and S. K. Kopparapu, "Mixed language speech recognition without explicit identification of language," *American Journal of Signal Processing*, vol. 2, no. 5, pp. 92–97, 2012.

[12] A. Pandey, B. M. L. Srivastava, and S. V. Gangashetty, "Adapting monolingual resources for code-mixed Hindi-English speech recognition," in *2017 International Conference on Asian Language Processing (IALP)*. IEEE, 2017, pp. 218–221.

[13] E. Yilmaz, H. v. d. Heuvel, and D. A. van Leeuwen, "Investigating bilingual deep neural networks for automatic speech recognition of code-switching frisian speech," 2016.

[14] A. Biswas, F. de Wet, E. van der Westhuizen, E. Yilmaz, and T. Niesler, "Multilingual neural network acoustic modelling for asr of under-resourced english-isizulu code-switched speech." in *Interspeech*, 2018, pp. 2603–2607.

[15] B. M. L. Srivastava and S. Sitaram, "Homophone identification and merging for code-switched speech recognition." in *Interspeech*, 2018, pp. 1943–1947.

[16] S. Rallabandi, S. Sitaram, and A. W. Black, "Automatic detection of code-switching style from acoustics," in *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*, 2018, pp. 76–81.

[17] A. Pandey, B. M. L. Srivastava, R. Kumar, B. T. Nellore, K. S. Teja, and S. V. Gangashetty, "Phonetically balanced code-mixed speech corpus for Hindi-English automatic speech recognition," in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[18] V. Chaitanya, R. Sangal, and A. Bharati, *Natural language processing: a Paninian perspective*. Prentice-Hall of India, 1996.

[19] L. M. Johnson, M. Di Paolo, and A. Bell, "Forced alignment for understudied language varieties: Testing Prosodylab-Aligner with Tongan data," *Language Documentation & Conservation*, vol. 12, pp. 80–123, 2018.

[20] T. Kempton, "Cross-language forced alignment to assist community-based linguistics for low resource languages," March 6–7 2017, paper presented at ComputEL-2, Honolulu.

[21] E. Kurtic, B. Wells, G. J. Brown, T. Kempton, and A. Aker, "A corpus of spontaneous multi-party conversation in Bosnian Serbo-Croatian and British English," in *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, N. Calzolari, K. Choukri, T. Declerck, M. U. Doğan, B. Maegaard, J. Mariani, A. Moreno, J. Odijk, and S. Piperidis, Eds. Istanbul, Turkey: European Language Resources Association (ELRA), May 2012.

[22] M. Diab and A. Kamboj, "Feasibility of leveraging crowd sourcing for the creation of a large scale annotated resource for hindi english code switched data: A pilot annotation."

[23] H. Buschmeier and M. Wlodarczak, "Textgridtools: A textgrid processing and analysis toolkit for python," in *Conference proceedings of the 24th conference on electronic speech signal processing (ESSV 2013)*, 2013.

[24] K. Kirchhoff, "Integrating articulatory features into acoustic models for speech recognition," *Phonus 5, Institute of Phonetics, University of the Saarland*, pp. 73–86, 2000.

[25] C. Strobl, A.-L. Boulesteix, A. Zeileis, and T. Hothorn, "Bias in random forest variable importance measures: Illustrations, sources and a solution," *BMC Bioinformatics*, vol. 8, no. 25, 2007. [Online]. Available: http://www.biomedcentral.com/1471-2105/8/25

[26] C. Strobl, A.-L. Boulesteix, T. Kneib, T. Augustin, and A. Zeileis, "Conditional variable importance for random forests," *BMC Bioinformatics*, vol. 9, no. 307, 2008. [Online]. Available: http://www.biomedcentral.com/1471-2105/9/307

[27] S. A. Tagliamonte and R. H. Baayen, "Models, forests, and trees of york english: Was/were variation as a case study for statistical practice," *Language variation and change*, vol. 24, no. 2, pp. 135–178, 2012.

[28] A. Kuznetsova, P. B. Brockhoff, and R. H. B. Christensen, "lmerTest package: Tests in linear mixed effects models," *Journal of Statistical Software*, vol. 82, no. 13, pp. 1–26, 2017.

[29] R. Wright *et al.*, "A review of perceptual cues and cue robustness," *Phonetically based phonology*, vol. 34, p. 57, 2004.

[30] K. N. Stevens, A. S. House, and A. P. Paul, "Acoustical description of syllabic nuclei: An interpretation in terms of a dynamic model of articulation," *The Journal of the Acoustical Society of America*, vol. 40, no. 1, pp. 123–132, 1966.

[31] R. Ahmed and S. Agrawal, "Significant features in the perception of (hindi) consonants," *The Journal of the Acoustical Society of America*, vol. 45, no. 3, pp. 758–763, 1969.

1616

Mere account mein kitna balance hai? - On building voice enabled Banking Services for Multilingual Communities

Akshat Gupta, Sai Krishna Rallabandi and Alan W Black

Department of Electrical and Computer Engineering, Carnegie Mellon University
Language Technologies Institute, Carnegie Mellon University
akshatgu@andrew.cmu.edu, {srallaba,awb} @cs.cmu.edu

Abstract

Tremendous progress in speech and language processing has brought language technologies closer to daily human life. Voice technology has the potential to act as a horizontal enabling layer across all aspects of digitization. It is especially beneficial to rural communities in scenarios like a pandemic. In this work[1] we present our initial exploratory work towards one such direction - building voice enabled banking services for multilingual societies. Speech interaction for typical banking transactions in multilingual communities involves the presence of filled pauses and is characterized by Code Mixing. Code Mixing is a phenomenon where lexical items from one language are embedded in the utterance of another. Therefore speech systems deployed for banking applications should be able to process such content. In our work we investigate various training strategies for building speech based intent recognition systems. We present our results using a Naive Bayes classifier on approximate acoustic phone units using the Allosaurus library [1].

1. Data Collection Tool

We have created a dataset of multilingual queries using a dummy banking app [2]. It involves a two stage setup. In the first stage, speech data is crowdsourced from fluent Hindi speakers. Each speaker plays the role of a 'user' and interacts with the automated banking system. This resulted in 100+ task-based dialogs in Hinglish, with five distinct intents. In the second stage, each speaker is asked to acoustically translate their interaction into another language. This way, we obtain pseudo parallel data in two languages from the same speaker. We believe that creating a pseudo parallel dataset will allow us to design semi supervised approaches in the future. Our dataset currently has speech data from six Indian languages - Hindi, Marathi, Gujarati, Punjabi, Telugu and Bhojpuri.

2. Methodology and Results

We chose spoken intent classification as the first task. For this task, we have 25 utterances containing speech samples of 11 people, out of which 4 were female. We have 5 intents in the dataset - **Send Money** (11 utterances), **Check Balance** (9 utterances), **Check Last Transaction** (3 utterances), **Withdraw Money** and **Deposit Money** (1 utterance each) .

We first convert audio into phones using the Allosaurus library[1]. These phones are then used for intent classification. We employ Naive Bayes' Classifier with add-1 smoothing and absolute discounting. We use cross validation where we leave out 2 audio samples for testing. The intents *Withdraw Money* and *Deposit Money* were not used for testing as we only have one sample for both, but are included in the training set. Thus the testing is only done for three intents, but an intent could be classified into any of the 5 classes. The results for Naive Bayes Classifier for add-1 smoothing are shown in Table 1.

Model	# Unique N-grams	Test Accuracy
Unigram	38	0.56
Bigram	292	0.48
Trigram	543	0.17

Table 1: *N-gram classification accuracy with Add-1 smoothing*

Model	Delta	Test Accuracy
Unigram	5	0.69
Bigram	1	0.61
Trigram	1	0.30
Combination	(5, 1, 1)	**0.83**

Table 2: *Classification accuracy with Absolute Discounting*

It can be observed that the accuracy decreases with increasing N for the different N-gram models. We hypothesize that this happens due to the relatively small size of the dataset in our initial exploratory work. We posit the distribution characteristics to be more uniform across N-grams in our future experiments with full dataset.

Further we have also performed absolute discounting with different delta values for each of the Ngram models. The test accuracies improved significantly. We also combined unigram, bigram and trigram models with equal weights with their respective best performing delta values, which gave us the best result as shown in Table 2.

3. Conclusion

In this paper, we present our initial exploration towards acoustic intent classification using a Naive Bayes classifier on approximate acoustic phone units. Our results indicate that our system can be employed to build real life banking applications in multilingual scenarios.

4. References

[1] X. Li *et al.*, "Universal phone recognition with a multilingual allophone system," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 8249–8253.

[2] https://awb.pc.cs.cmu.edu/.

[1] This is an extended abstract.

1718

Investigating Modelling Techniques for Natural Language Inference on Code-Switched Dialogues in Bollywood Movies

Anjana Umapathy[1]*, Sharanya Chakravarthy*[1]*, Alan W Black*[1]

[1]Language Technologies Institute, Carnegie Mellon University

aumapath@cs.cmu.edu, sharanyc@cs.cmu.edu, awb@cs.cmu.edu

1. Abstract

Code-switching is the intertwined usage of multiple languages in an utterance or conversation, and is commonly seen in informal conversations among polyglots. Natural Language Inference (NLI) is the task of detecting whether a premise entails a hypothesis, or contradicts it. The performance of Machine Learning models on NLI provides an indication of their ability to understand natural language. Unsupervised pre-trained transformer models have advanced the state of the art in NLI and various other Natural Language Understanding (NLU) tasks in recent years. Our work focuses on various techniques that extend the capability of popular transformer models trained on monolingual text to detect entailment in code-switched settings.

For our task, we utilize the dataset introduced by [1], which leverages Bollywood movie scripts containing code-switched Hindi-English (Hinglish) text. Although this data is textual, it is written to be performed as speech. Premises are in the form of multiple utterances from a conversation, with each utterance preceded by the name of the speaker. The creation of hypotheses based on dialogue-like premises transforms the task from one of textual entailment to one of conversational entailment. The dataset contains 2,240 labelled premise-hypothesis pairs, with an 80:20 train-test split. As there is large variance in the training examples, we employ cross-validation with 8 splits to better understand the performance of the models.

We present a comparison of linguistic, data augmentation and architecture based approaches to conversational entailment in code-switched text. The rest of this abstract outlines our methodology, results and analysis.

Transformer [2] based models such as BERT [3] and RoBERTa [4], pre-trained on large monolingual corpora, have advanced the state of the art on the SNLI [5] and MultiNLI [6] datasets. We fine-tune $\text{BERT}_{\text{BASE}}$ and mBERT for the sentence-pair classification task, and these models serve as points of comparison for our other approaches. We observe that the performance of $\text{BERT}_{\text{BASE}}$ is comparable to that of mBERT despite the large difference in the magnitude of Hindi text used for pre-training the two models. Additionally, we fine-tune mBERT for masked language modeling on code-switched Hinglish text, and obtain mod-mBERT as described in [7].

Due to the limited amount of code-switched NLI data available for fine-tuning, we augment the dataset with monolingual examples from the SNLI [5], XNLI [8] and MPE [9] datasets. Romanized transliterations of Hindi sentence-pairs from the XNLI dataset provide additional NLI data in Romanized Hindi while SNLI examples do the same in English. We observe that the addition of this external data results in an improvement in performance over the baseline. We further investigate the impact of different data selection strategies by varying the amount of data used for augmentation and combining data from multiple external sources.

mBERT is pre-trained on monolingual Hindi data in the Devanagari script. However, the majority of Hindi words in the code-switched dataset are out of vocabulary since they are in Romanized form. To better leverage mBERT's knowledge of Hindi, we detect and transliterate Hindi tokens to Devanagari, and use this modified version of the dataset for fine-tuning. We also provide a comparison of various strategies for obtaining monolingual translations of the data in the absence of a translation model trained on a large parallel corpus of code-switched and monolingual data. We find that errors in language identification and transliteration propagate to the translations, deterring the performance. However, augmenting the original data with the translations obtained, reduces the impact of these errors.

An analysis of the dataset shows that a number of instances require the model to have an understanding of the speaker of each turn in the conversation. Unexpectedly, we find that removing the speaker names from the dataset does not cause a significant drop in the accuracy of predictions. We believe this indicates that BERT models could benefit from reinforcing speaker information. To this end, we augment the existing dataset with examples which differ only in the names of the speakers. In order to tackle instances where the hypothesis is constructed by swapping roles [1], we add contradiction and entailment examples to the dataset by varying the speaker names. Many of these modifications better the accuracy of the models.

Apart from the data augmentation and modification approaches detailed thus far, we modify the architecture used to encode the conversational NLI data. Since transformer models are typically used on single sentences, we seek to obtain better representations of the multi-turn premises using an architecture that combines a bidirectional LSTM network with BERT embeddings of each utterance. While such approaches have been used in other dialogue based tasks, we believe that their utility here is limited by the amount of training data available.

Several of the approaches outlined above surpass the baseline provided by [1], achieving accuracies in the range of 58-63%. However, combinations of these approaches do not yield further improvements, thus indicating the scope for new avenues of research in this area. In our experiments with varying the amount of training data used, we find that the performance of the model is better than chance even with a quarter of the already limited data. This serves to show that merely quadrupling the amount of training data will not solve the problem of code-switched conversational NLI, making this an interesting research problem.

2. References

[1] S. Khanuja, S. Dandapat, S. Sitaram, and M. Choudhury, "A new dataset for natural language inference from code-mixed conversa-

* Equal contribution

tions," *arXiv preprint arXiv:2004.05051*, 2020.

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[4] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," 2019.

[5] S. Bowman, G. Angeli, C. Potts, and C. D. Manning, "A large annotated corpus for learning natural language inference," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 632–642.

[6] A. Williams, N. Nangia, and S. Bowman, "A broad-coverage challenge corpus for sentence understanding through inference," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, 2018, pp. 1112–1122. [Online]. Available: http://aclweb.org/anthology/N18-1101

[7] S. Khanuja, S. Dandapat, A. Srinivasan, S. Sitaram, and M. Choudhury, "GLUECoS: An evaluation benchmark for code-switched NLP," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, Jul. 2020, pp. 3575–3585. [Online]. Available: https://www.aclweb.org/anthology/2020.acl-main.329

[8] A. Conneau, R. Rinott, G. Lample, A. Williams, S. R. Bowman, H. Schwenk, and V. Stoyanov, "Xnli: Evaluating cross-lingual sentence representations," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018.

[9] A. Lai, Y. Bisk, and J. Hockenmaier, "Natural language inference from multiple premises," *arXiv preprint arXiv:1710.02925*, 2017.

1923

First Workshop on Speech Processing for Code-switching in Multilingual Communities: Shared Task on Code-switched Spoken Language Identification

Sanket Shah[1], *Sunayana Sitaram*[1], *Rupeshkumar Mehta*[2]

[1]Microsoft Research India
[2]Microsoft Corporation

{t-sansha, sunayana.sitaram, rupesh.mehta}@microsoft.com

Abstract

Code-switched speech and language processing is challenging due to the paucity of publicly-available datasets for research. We describe a shared task on language identification from speech organized as part of the First Workshop on Speech Technologies for Code-switching in Multilingual Communities. The shared task consisted two sub-tasks 1. Spoken Language Identification at the utterance level, where the goal was to classify an utterance as monolingual or code-switched 2. Spoken Language Identification within a code-switched utterance, where the goal was to identify languages at a frame-level. We released a dataset consisting of 60 hours of data in three language pairs - Tamil-English, Telugu-English and Gujarati-English. Six teams participated in the utterance level task, while two teams participated in the frame-level task. Team VocapiaLIMSI used a combination of i-vector and phonotactic-based models to achieve the best performance across languages on both tasks. We hope that this dataset, which is now available for research purposes will encourage research in code-switched spoken language identification.

Index Terms: spoken language identification, code-switching, shared task

1. Introduction

Code-switching, which is the use of two or more languages in a single conversation or utterance, is a challenging phenomenon for Speech and Natural Language Processing systems to handle. Recently, there has been significant progress made in code-switching Speech and NLP research [1], however, the lack of sufficient data and resources in code-switched language pairs compared to monolingual resources still remains a hurdle.

A well known technique for handling code-switching is to first identify the language of part of an utterance, and then use monolingual systems to process each corresponding fragment. Such Language Identification (LID) systems have also been used in conjunction with monolingual or multilingual systems to aid in processing. So far, the focus of code-switching research has mainly been on LID in text, in which words in a code-switched utterance are individually labeled with language tags. Much of this research has been spurred due to several shared tasks on code-switched text LID. Inspired by this, we conducted a shared task on Language Identification from speech, which has been relatively less studied.

Our shared task consisted of two sub-tasks. Task A was an utterance-level LID task, in which a system had to classify an utterance as being monolingual or code-switched. Task B was a frame-level LID task, in which a system had to classify a code-switched utterance with language labels at the frame-level. For both tasks, systems only had access to raw audio at test time, without accompanying text transcriptions of the utterances.

For the shared task, we released a code-switched speech corpus with 60 hours of speech from three language pairs: Tamil-English, Telugu-English and Gujarati-English. This dataset is available for research use and is the first dataset of code-switched speech in these language pairs. More details about the dataset can be found in Section 4. We used a CNN-BLSTM-based model to build baselines for both tasks, which is described in Section 5. Participants were provided with a blind test set to evaluate their systems on. We describe the techniques used by participating teams in the shared task, along with results in Section 6.

2. Related work

Language Identification for code-switched utterances from text has been well studied and a comprehensive overview can be found in [1]. The first and second workshops on Computational Approaches to Code Switching conducted a shared task on Language Identification for several language pairs [2, 3].

While LID for speech has been a well established area of research, intra-utterance LID from speech for code-switching has been relatively less studied. However, there have been initial attempts to solve this problem, mainly in the context of using LID systems in conjunction with an Automatic Speech Recognition (ASR) system.

In [4], the authors show that humans exploit prosodic cues to detect code switching in speech and can anticipate switch points even in noisy speech. In [5, 6] the authors investigate the effectiveness of using retrained multilingual DNNs and augmenting the data for detecting the language. In [7, 8] authors employ word based lexical information, build HMM-based acoustic models followed by an SVM based decision classifier to identify code-switching between Northern Sotho and English [9]. It may be useful for an ASR system to be able to detect the code-switching style of a particular utterance, and be able to adapt to that style through specialized language models or other adaptation techniques. [10] classify code-switched corpora by code-switching style and show that features extracted from acoustics alone can distinguish between different kinds of code-switching in a single language.

Previously, we conducted a low resource ASR challenge [11] in which we released 150 hours of speech data for three languages: Tamil, Telugu and English. In this challenge, we release code-switched data for the same language pairs so that the research community can make use of this data together to build robust speech systems for multilingual scenarios.

3. Challenge Rules

As mentioned before, the shared task consisted of two subtasks:

- Task A: Utterance-level identification of monolingual vs.

code-switched utterances

- Task B: Frame-level identification of language in a code-switched utterance.

Registered participants were sent a link to download data sets in all three languages. We released 16 hours of training data and around 2 hours of test data for each language, details of which can be seen in section 4. We also released baseline Accuracy (ACC) and Equal Error Rate (EER) numbers evaluated on the 2 hour test sets along with instruction on how to replicate the baselines. Details of the baseline and evaluation metrics can be found in section 5. Participants were allowed to use monolingual data that we released in our previous ASR challenge [12], but no other external data was allowed to be used.

Testing was conducted in April 2020, in which we tested all submitted systems over a period of 3 days. Participants were sent links to 4 hours of blind test audio data in each language for Task A and 2 hours of blind test audio data in each language for Task B. Participants ran their systems on the blind test audio data and submitted hypothesis files via email to an automated scoring system which calculated the ACC and EER and sent it back to them as a reply to their email. Each participating team was allowed 3 attempts per language per task, so each team could submit up to 18 models in all for evaluation. The automated scoring system was created using Microsoft Flow [1].

4. Data

4.1. Data

The data released for the challenge was provided by SpeechOcean.com and Microsoft. It consisted of phrasal (recorded as read-out phrases) and conversational speech in Tamil-English, Telugu-English and Gujarati-English. The transcription consisted of English words in Roman script and the other language's words in native script, although there was some cases of cross-transcription. Some example code-switched utterances are shown in figure 1.

4.1.1. Task A

For training data, each character in the transcript was replaced its corresponding language tag by looking up the language it was transcribed in i.e., 'T' for Telugu or Tamil and 'G' for Gujarati. As a part of training data, participants were allowed to use data from Low Resource Automatic Speech Recognition Challenge [11] along with the data released as a part of this challenge. The test and blind test sets consisted of monolingual and code-switched utterances and their corresponding class (0 and 1 respectively). The data description for task A can be seen in table 1.

Table 1: *Description of train, test and blind test data for Task A*

	Train	***Test***	***Blind Test***
ta-en	16 hrs (8928 utt.) (CMI:0.17)	4 hrs (2223 utt.)	4 hrs (2235 utt.)
te-en	16 hrs (8226 utt.) (CMI:0.22)	4 hrs (2149 utt.)	4 hrs (2133 utt.)
gu-en	16 hrs (8620 utt.) (CMI:0.16)	4 hrs (2091 utt.)	4 hrs (2102 utt.)

[1] https://flow.microsoft.com/

The Code Mixing Index [13] is a metric that measures the amount of code-switching in a corpus by using word frequencies. We report the CMI of our code-switched train and test sets in parentheses in Table 1 and Table 2. Gujarati and Tamil have a relatively lower CMI of 17% while the Telugu dataset has a CMI of 22%.

4.1.2. Task B

Since Task A was an utterance-level task, each utterance had a single label. In task B, each utterance was labeled with language information at the frame-level. In the training data, each character in the transcript was replaced its corresponding language tag i.e., 'T' for Telugu or Tamil and 'G' for Gujarati. For the test and blind test sets, we generated language tags for every 200 ms of the CS audio. Further details of the data for task B can be found in table 2.

Table 2: *Description of train, test and blind test data for Task B*

	Train	***Test***	***Blind Test***
ta-en	16 hrs (8928 utt.) (CMI:0.17)	2 hrs (1135 utt.) (CMI:0.18)	2 hrs (1120 utt.) (CMI:0.17)
te-en	16 hrs (8226 utt.) (CMI:0.22)	2 hrs (1047 utt.) (CMI:0.22)	2 hrs (1033 utt.) (CMI:0.22)
gu-en	16 hrs (8620 utt.) (CMI:0.16)	2 hrs (1080 utt.) (CMI:0.17)	2 hrs (1078 utt.) (CMI:0.17)

5. Evaluation and Baselines

5.1. Evaluation

We used two standard metrics for evaluation: Accuracy (ACC) and Equal Error Rate (EER). They are defined as follows.

- Accuracy

$$Accuracy = \frac{N}{T}$$

Where,
N is the total no. of correctly predicted data samples
T is the total no. of data points

- Equal Error Rate (EER)

$$EER = \frac{FRR + FAR}{2}$$

$$FRR = \frac{TFR}{T}$$

$$FAR = \frac{TFA}{T}$$

where,
EER Equal Error Rate
FRR False Rejection Rate
FAR False Acceptance Rate
TFR Total No. of False Rejects
TFA Total No. of False Accepts
T Total No. of Datapoints

5.2. Baselines

We used a CNN-BLSTM model based on Deepspeech2 [14] as the baseline system for both tasks.

We denote our training monolingual datasets (X_1^L, Y_1^L),...,(X_n^L, Y_n^L) where $L \in$ {TE/TA/GU}, code-switched

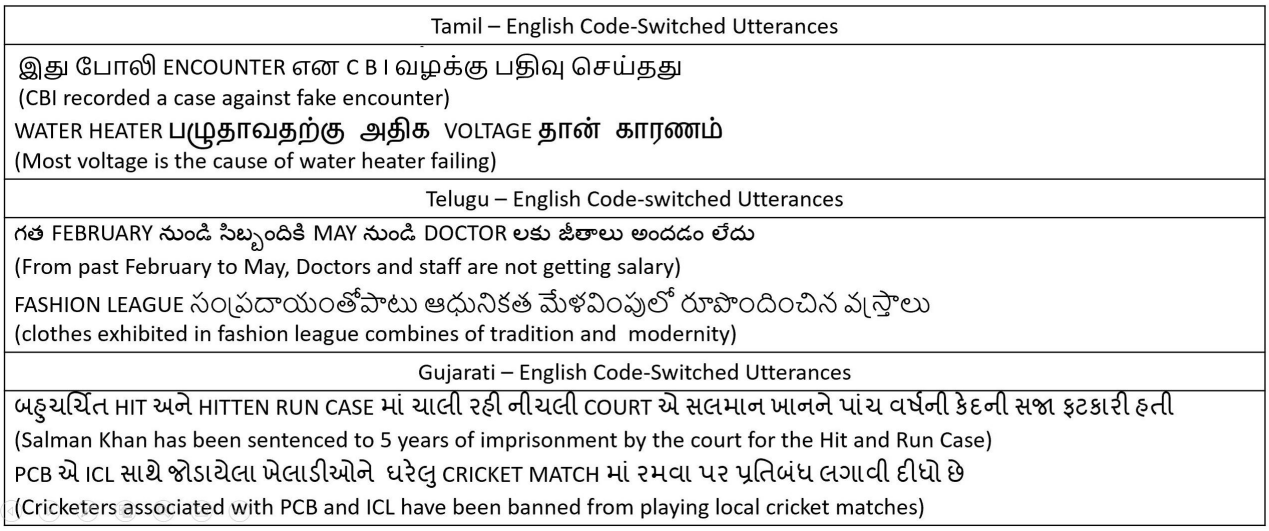

Figure 1: *Some example code-switched utterances from the dataset.*

datasets (X_1^{CS}, Y_1^{CS}),...,(X_n^{CS}, Y_n^{CS}) where $CS \in$ {TE-EN/TA-EN/GU-EN}. The labels Y are language tags i.e., 'T' for Telugu or Tamil and 'G' for Gujarati. Our baseline model consists of two Convolution Neural Network (CNN) layers followed by five bidirectional long-short term (BLSTM) layers of 1024 dimension. Further, the frame-wise posterior distribution $P(Y|X)$ is conditioned on the input X and calculated by applying a fully-connected layer and a softmax function.

$$P(Y|X) = \text{Softmax}(Linear(h)) \quad (1)$$

where h is the hidden state from BLSTM. For task A, based on the softmax output, we determine whether the audio is monolingual or code-switched, while for task B, We use the 200ms frame length and predict the language tag for the same. The model parameters are trained using the Connectionist Temporal Classification (CTC) [15] criterion. We use the SGD optimizer with 3e-4 learning rate. We trained the model for 40 epochs for Task A and 60 epochs for Task B, with mini-batch size equal to 64 per GPU.

Baseline results for task A and task B are provided in table 3 and 4 respectively.

Table 3: *Baseline results for Task A*

Class 1	***Class 2***	***ACC***	***EER***
ta-en	ta-mono	74.0%	13.0%
te-en	te-mono	71.2%	14.4%
gu-en	gu-mono	76.8%	11.6%

Table 4: *Baseline results for Task B*

	ACC	***EER***
ta-en	77.6%	6.5%
te-en	76.5%	6.7%
gu-en	76.7%	6.7%

6. Systems and Results

Six teams participated in task A, while two teams participated in task B. In this section, we describe the systems built by participating teams and compare their results to the baseline numbers. The teams which participated in the evaluation were VocapiaLIMSI, Swiggy, AnnotateIt, Sizzle, Ground Zero and CMU. Participants spanned academic institutions, industry labs and startups.

Tables 5 and 6 show results from participating teams on both tasks. For task A, in case of Gujarati, no team was able to beat the baseline in terms of ACC and EER. In case of Telugu, four teams (VocapiaLIMSI, Swiggy, CMU and Sizzle) outperformed the baseline. In case of Tamil, two teams (VocapiaLIMSI and Swiggy) outperformed the baseline. For task B, the VocapiaLIMSI system outperformed the baseline for all three language pairs, while the Swiggy team came close to but could not outperform the baseline. Next, we describe the approaches that the various teams took for solving both tasks.

Team VocapiaLIMSI [16] achieved the best scores for both the tasks, and across all the languages. They proposed two acoustic modeling approaches, one being i-vector modeling of the audio segments and phonotactic modeling focusing on sequences of language-independent phone units. The i-vector system characterizes the language of an utterance with vectors obtained by projecting the speech data onto a total variability space while the phonotactic approach relies on the idea that the phonetic sequence in an audio sample is characteristic of the language used. Team VocapiaLIMSI [16] also experimented with combining the scores from both the systems and found that a linear combination of the scores from both the systems gave the best performance.

Team Swiggy [17] modified the spectral augmentation approach and proposed a language mask that discriminated language ID pairs, leading to a noise robust spoken LID system. The authors proposed a temporal masking approach in which they first map language transcripts to speech frames. The number of temporal masks were determined based on the number of English words present in the language transcripts. Time segments corresponding to the English words were masked. Once the masking was done, they then extracted features and passed

Table 5: *ACCs and EERs for top performing models for task A*

	Team	***ACC***	***EER***
tamil	Baseline	74.0%	13.0%
tamil	VocapiaLIMSI	79.8%	10.1%
tamil	Swiggy	78.6%	10.6%
tamil	CMU	73.6%	13.2%
tamil	Sizzle	69.1%	15.5%
tamil	Ground Zero	67.1%	17.0%
tamil	AnnotateIt	61.8%	19.1%
telugu	Baseline	71.2%	14.4%
telugu	VocapiaLIMSI	79.3%	10.3%
telugu	Swiggy	79.0 %	10.5%
telugu	CMU	73.9%	13.0%
telugu	Sizzle	71.3%	14.3%
telugu	Ground Zero	67.2%	16.4%
telugu	AnnotateIt	59.6%	20.2%
gujarati	Baseline	76.8%	11.6%
gujarati	VocapiaLIMSI	75.3%	12.3%
gujarati	Swiggy	73.0%	13.5%
gujarati	AnnotateIt	66.1%	16.9%
gujarati	Ground Zero	55.1%	22.4%
gujarati	CMU	49.9%	25.0%
gujarati	Sizzle	48.4%	25.7%

them to an end-to-end CNN-LSTM system and train the system using CTC loss function at the output layer.

Team CMU [18] proposed a multi-task learning framework where the primary task is language detection and secondary task is audio reconstruction. Considering a speech corpus X consisting of languages $\{l_1, .., l_n\}$, where each l_i comprise of difference speakers. $x_1, ..., x_n$ denotes acoustic frames of X. x_i can be monolingual or code-switched and Y denotes the labels. The authors trained a model to learn the join distribution between $\{x, y\}$. The process was mediated using latent discrete random representation. To ensure that the latent representations correspond to speech utterances, they adopted this multi-task learning framework.

Team AnnotateIt [19] used a two-stage training and inference model. In the front-end, the authors trained a monolingual speech recognition model using the monolingual speech corpus described in the previous section. In the second stage, they used the inference from the first stage to train a linear classifier for the binary language identification task: monolingual vs. code-switched.

Team Sizzle [20] proposed a convolutional encoder in combination with the transformer architecture for utterance-level

Table 6: *ACCs and EERs for top performing models for task B*

	Team	***ACC***	***EER***
tamil	Baseline	77.6%	6.5%
tamil	VocapiaLIMSI	78.8%	6.5%
tamil	Swiggy	76.1%	7.4%
telugu	Baseline	76.5%	6.7%
telugu	VocapiaLIMSI	79.6%	6.3%
telugu	Swiggy	76.1%	7.4%
gujarati	Baseline	76.7%	6.7%
gujarati	VocapiaLIMSI	77.7%	6.9%
gujarati	Swiggy	76.1%	7.5%

code-switching detection. The authors used the convolutional encoder to process audio features and pass these features to a transformer based network with multi-head self attention layers. Finally, the authors, aggregated the frame level features from the transformer network to get utterance level features to predict the class label.

In summary, approaches that used the audio directly and did not rely on an ASR system seemed to work best, however this may be due to the small amount of data used to train the ASR. Future work in this direction could include using all the languages in a multilingual setup for spoken language identification.

7. Conclusions

In this paper, we described the first shared task conducted for spoken language identification of code-switched speech. The shared task consisted of two sub-tasks - an utterance-level task to classify speech into monolingual or code-switched and an intra-utterance classification task, in which a code-switched utterance had to labeled with languages at the frame-level. We released, for the first time, code-switched data in three language pairs: Tamil-English, Telugu-English and Gujarati-English for the shared task. The data released as part of this shared task is available for future research use.

Six teams participated in the shared task, however we had significantly less participation in task B compared to task A. The best performing systems that beat the baseline included VocapiaLIMSI and Swiggy, which used approaches based on i-vectors, phonotactic models and temporal masking. The second task of our shared task, which is the frame-level identification of code-switching remains under-explored, and we hope that future research will focus on this problem.

8. Acknowledgements

The authors would like to thank Sandeepkumar Satpal and Vinay Krishna from Microsoft for help with creating the data for the shared task.

9. References

[1] S. Sitaram, K. R. Chandu, S. K. Rallabandi, and A. W. Black, "A survey of code-switched speech and language processing," *arXiv preprint arXiv:1904.00784*, 2019.

[2] T. Solorio, E. Blair, S. Maharjan, S. Bethard, M. Diab, M. Ghoneim, A. Hawwari, F. AlGhamdi, J. Hirschberg, A. Chang *et al.*, "Overview for the first shared task on language identification in code-switched data," in *Proceedings of the First Workshop on Computational Approaches to Code Switching*, 2014, pp. 62–72.

[3] G. Molina, F. AlGhamdi, M. Ghoneim, A. Hawwari, N. Rey-Villamizar, M. Diab, and T. Solorio, "Overview for the second shared task on language identification in code-switched data," *arXiv preprint arXiv:1909.13016*, 2019.

[4] P. E. Piccinini and M. Garellek, "Prosodic cues to monolingual versus code-switching sentences in english and spanish," in *Proceedings of the 7th Speech Prosody Conference*, 2014, pp. 885–889.

[5] E. Yılmaz, H. van den Heuvel, and D. van Leeuwen, "Code-switching detection using multilingual dnns," in *2016 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2016, pp. 610–616.

[6] E. Yılmaz, H. v. d. Heuvel, and D. A. van Leeuwen, "Code-switching detection with data-augmented acoustic and language models," *arXiv preprint arXiv:1808.00521*, 2018.

[7] D.-C. Lyu, R.-Y. Lyu, C.-L. Zhu, and M.-T. Ko, "Language identification in code-switching speech using word-based lexical model," in *2010 7th International Symposium on Chinese Spoken Language Processing*. IEEE, 2010, pp. 460–464.

[8] D.-C. Lyu, T.-P. Tan, E.-S. Chng, and H. Li, "An analysis of a mandarin-english code-switching speech corpus: Seame," *Age*, vol. 21, pp. 25–8, 2010.

[9] K. R. Mabokela, M. J. Manamela, and M. Manaileng, "Modeling code-switching speech on under-resourced languages for language identification," in *Spoken Language Technologies for Under-Resourced Languages*, 2014.

[10] S. Rallabandi, S. Sitaram, and A. W. Black, "Automatic detection of code-switching style from acoustics," in *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*, 2018, pp. 76–81.

[11] B. M. L. Srivastava, S. Sitaram, K. Bali, R. K. Mehta, K. D. Mohan, P. Matani, S. Satpal, K. Bali, R. Srikanth, and N. N. , "Interspeech 2018 low resource automatic speech recognition challenge for indian languages," in *SLTU*, August 2018. [Online]. Available: https://www.microsoft.com/en-us/research/publication/interspeech-2018-low-resource-automatic-speech-recognition-challenge-for-indian-languages/

[12] B. M. L. Srivastava and S. Sitaram, "Homophone Identification and Merging for Code-switched Speech Recognition," *Proeedings of Interspeech 2018*, 2018.

[13] B. Gambäck and A. Das, "On measuring the complexity of code-mixing," in *Proceedings of the 11th International Conference on Natural Language Processing, Goa, India*, 2014, pp. 1–7.

[14] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen, J. Chen, J. Chen, Z. Chen, M. Chrzanowski, A. Coates, G. Diamos, K. Ding, N. Du, E. Elsen, J. Engel, W. Fang, L. Fan, C. Fougner, L. Gao, C. Gong, A. Hannun, T. Han, L. V. Johannes, B. Jiang, C. Ju, B. Jun, P. LeGresley, L. Lin, J. Liu, Y. Liu, W. Li, X. Li, D. Ma, S. Narang, A. Ng, S. Ozair, Y. Peng, R. Prenger, S. Qian, Z. Quan, J. Raiman, V. Rao, S. Satheesh, D. Seetapun, S. Sengupta, K. Srinet, A. Sriram, H. Tang, L. Tang, C. Wang, J. Wang, K. Wang, Y. Wang, Z. Wang, Z. Wang, S. Wu, L. Wei, B. Xiao, W. Xie, Y. Xie, D. Yogatama, B. Yuan, J. Zhan, and Z. Zhu, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ser. ICML'16. JMLR.org, 2016, p. 173–182.

[15] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd International Conference on Machine Learning*, ser. ICML '06. New York, NY, USA: Association for Computing Machinery, 2006, p. 369–376. [Online]. Available: https://doi.org/10.1145/1143844.1143891

[16] J.-L. G. Claude Barras, Viet-Bac Le, "Vocapia-limsi system for 2020 shared task on code-switched spoken language identification," 2020.

[17] H. M. Pradeep Rangan, Sundeep Teki, "Exploiting spectral augmentation for code-switched spoken language identification," 2020.

[18] A. W. B. Sai Krishna Rallabandi, "On detecting code mixing in speech using discrete latent representations," 2020.

[19] J. A. C. Parav Nagarsheth, "Language identification for code-mixed indian languages in the wild," 2020.

[20] A. P. Krishna D N, "Utterance-level code-switching identification using transformer network," 2020.

2428

Vocapia-LIMSI System for 2020 Shared Task on Code-switched Spoken Language Identification

Claude Barras[1], *Viet-Bac Le*[1], *Jean-Luc Gauvain* [1,2]

[1]Vocapia Research, Orsay, France
[2]Université Paris-Saclay, CNRS, LIMSI, Orsay, France

barras@vocapia.com, levb@vocapia.com, gauvain@limsi.fr

Abstract

This paper describes the systems submitted by Vocapia Research and LIMSI for the shared task on Code-switched Spoken Language Identification, organized in the conjunction with the First Workshop on Speech Technologies for Code-switching in Multilingual Communities 2020. Our primary system combines an acoustic approach based on i-vector modeling of audio segments with a phonotactic approach that focuses on sequences of language-independent phone units. Both modeling approaches provided comparable performance, and a gain was obtained by a simple linear combination of their scores, showing their complementarity. One of our submissions obtained first rank for all combinations of tasks and language pairs. For the utterance-level detection task (task A), an F-measure of 76.0% was obtained with our combined system for which the average accuracy on the development set was 83.3%. For the frame-level detection task, the average accuracy was 81.2% on the development set and 78.7% on the evaluation set. However, a detailed analysis reveals a very high rejection of the 200ms code-switched frames, which comprise only 12% of the corpus. This shows that a more precise modeling of code-switched segments is needed for an accurate segmentation.

Index Terms: language identification, code-switching, phonotactic model

1. Introduction

Code-switching is very usual in multilingual communities. In formal situations, a speaker may choose one language according to the situation. For such speech data, automatic language identification can be performed at the speaker turn or document level before further content processing. However, in more spontaneous cases, short code-switched segments may occur in the middle of a sentence. This type of code-switching is much harder to detect and adversely affects the speech transcription, since words in the alternative language will be missing from the vocabulary of the speech-to-text system. Although code-switching has been studied in the linguistic community for many years, it has recently started attracting growing interest in the speech technology domain with the collection of several code-switching corpora for Cantonese-English [1], Mandarin-English [2], Frisian-Dutch [3], Hindi-English and Spanish-English [4], South African languages [5], Egyptian Arabic-English [6] or CanVEC Vietnamese-English [7]. This interested has also resulted in special sessions at Interspeech conferences since 2017, covering various language pairs (e.g., Mandarin-English [8], Hindi-English [9], isiZulu-English [10], English-Spanish [11], French-Algerian Arabic [12] or Frisian-Dutch [13, 14]) and addressing linguistic analysis, speech synthesis [15], code-switching detection [14], language modeling [11, 16] or automatic transcription [17, 18, 19, 20].

In this paper, we describe the systems submitted by Vocapia Research and LIMSI laboratory for the shared task on Code-switched Spoken Language Identification (LID), which was organized in conjunction with the First Workshop on Speech Technologies for Code-switching in Multilingual Communities 2020[1]. Our system combines an acoustic approach based on the i-vector modeling of audio segments and a phonotactic approach focusing on sequences of language-independent phonetic units. The outputs of the two component systems were also submitted individually to obtain contrastive results.

The next section summarizes the two evaluation tasks and experimental conditions. Section 3 describes the component i-vector and phonotactic systems. Their performance on the development and evaluation data is presented in Section 4, before a conclusion.

2. Task description

A short summary of the tasks and evaluation plan is given here along with some characteristics of the corpus. A complete description is available from the link in footnote 1.

2.1. Subtasks

Two subtasks were proposed: (Task A) utterance-level identification of monolingual vs. code-switched utterances; (Task B) frame-level language identification in code-switched utterances, where frames are 200ms contiguous audio segments. For each task, three language pairs were considered with one primary Indian language among Gujarati, Telugu and Tamil and English as a possible code-switching target. The primary language was known a priori.

2.2. Evaluation metric

The primary evaluation metric for the evaluation was the accuracy rate, defined as the ratio of correctly predicted samples over the total number of samples. Task A is a sentence-level binary classification task (code-switched vs. non code-switched utterance), while task B requires a frame-level labeling with 3 classes (silence, primary language or English). There is no specific cost function, so the prior distribution of the classes is the factor governing the relative weight of each error type.

In this paper, we also report three other metrics: recall and precision rates expressed as the number of correctly detected code-switched samples (utterances for task A or frames for task B) over the number of expected or hypothesized samples, respectively; the F-measure is defined as the harmonic mean of

[1]https://www.microsoft.com/en-us/research/event/workshop-on-speech-technologies-for-code-switching-2020/

the recall and precision; and the false positive and false negative rates expressed as the number of false positive (resp. negative) hypothesis over the total number of positive (resp. negative) code-switched samples.

The evaluation plan also proposed an EER metric defined, for task A, as the total number of false rejects and false accepts divided by twice the total number of sentences. Being redundant with the accuracy rate, this metric is not reported in the paper.

2.3. Corpus

The training and evaluation data is composed of sets of sentences with durations ranging from 2 and 20 seconds, and an average duration of 6.8 sec. For task A, about 8000 training sentences and 1000 development sentences without code-switching were provided for each target language (Gujarati, Tamil and Telugu) along with a similar number of code-switched sentences. The detail is provided in Table 1.

Table 1: *Number of sentences without or with code-switching (resp. no CS and CS) in the training and dev. sets for task A.*

	Training		Development	
Language	no CS	CS	no CS	CS
Gujarati	8161	8619	1012	1079
Tamil	8965	8978	1123	1135
Telugu	8766	8225	1088	1047

For task B, only code-switched sentences were provided with the corresponding 200ms frame-level annotation: 8000 sentences for training (about 15 hours) and 1000 sentences for development (about 2 hours) per language. On average, 21% of the frames are labelled as silence, 12% as English and the remaining 67% as either Gujarati, Tamil or Telugu. Table 2 shows the detail for the training and development sets.

Table 2: *Cumulated duration (hh:mm) of primary language, English code-switched and silent frames (resp. P, CS and SIL) in the training and development sets for task B.*

	Training			Development		
Language	P	CS	SIL	P	CS	SIL
Gujarati	10:39	2:08	3:24	1:20	0:16	0:25
Tamil	10:52	1:47	3:32	1:21	0:15	0:26
Telugu	10:53	1:50	3:27	1:21	0:14	0:25

Note that the additional monolingual datasets provided for the three languages were not used to develop the submitted systems described in this paper.

3. Submitted systems

In this section, we describe the two types of systems developed for the detection of code-switched utterances or frames and used for the evaluation: one based on the acoustic (i-vector) approach and other on a phonotactic approach. In addition we also submitted results obtained by combining the outputs of these two.

3.1. i-vector acoustic modeling

The i-vector framework [21] has been successfully applied in Speaker Verification [22, 23] and Language Identification [24] tasks. The i-vector system characterizes the language of an utterance with vectors obtained by projecting the speech data onto a *total variability space*. The approach is generally formulated as follows: $S = m + Tw$ where w is called an *i-vector*, T is a matrix representing the *total variability space*; and m and S represent Gaussian supervectors (GSV) obtained from a language-independent and a language dependent model respectively. The language-independent model is also called *Universal Background Model* (UBM).

In our implementation, the input features of the i-vector language identification system are 40 dimensional phonetic bottleneck features. For each frame, a 32 ms window and a 10 ms offset are used to extract 32-band Mel scale spectrogram concatenated with log-pitch, delta-log-pitch and voicing probability. Then, TRAP-DCT features [25] are estimate on 100 ms windows (11 frames), retaining the first 6 coefficients including the DC component [26]. The resulting TRAP-DCT features with 210 dimensions (35x6) are input to a bottle-neck DNN that has 3 hidden layers with 2000 units and 1 bottle-neck hidden layer with 400 units. Each hidden layer is followed by a non-linearity p-norm unit [27] which reduces the dimension of the layer to 200 and 40, respectively. The phonetic bottle-neck DNN was trained on about 1000 hours of English Broadcast Data. The bottle-neck features are extracted without cepstral mean or variance normalization (CMVN).

The full covariance GMM with 2048 components, the UBM, and an i-vector extractor are estimated using the training data using the Kaldi toolkit [28]. A 600-dimension i-vector is extracted for each training utterance or segment. The i-vector length is normalized to unity [23]. A language-specific i-vector is obtained by averaging the normalized i-vectors for each training utterance [23].

During the test phase, an i-vector is extracted for each test utterance or segment, and is processed to compensate for session variability. Different techniques can be used to compute the test utterance scores. Multi-class logistic regression (MLR) [29] is used in this work. The use of probabilistic linear discriminant analysis (PLDA) [30] such as applied in [22, 23], was explored for the NIST 2015 Language Recognition Evaluation (LRE15) [31] but since the MLR method provided better results on Broadcast data on an internal LID dataset it was adopted here. The MLR model is estimated on all training utterances/segments using an expectation-maximization algorithm.

For Task A, and for each of the 3 Indian languages, an i-vector is extracted for each training utterance (code-switched or non-code-switched) and then a logistic regression (LR) classifier (positive and negative code-switched classes) is estimated on these i-vectors. In the test phase, an i-vector is extracted for each test utterence and scored using the LR model. No voice activity detection (VAD) is performed on the training and test sentences before acoustic features extraction in this task.

For task B, all audio files of the training, developement and evaluation data are analyzed using 600ms-long overlapping windows with a 200ms step (a frame). The label of the segment was associated to the frame at the center of the window, with e.g. the [0-600ms] window corresponds to the frame [200-400ms]. For each of 3 the Indian languages, an MLR classifier is estimated on all training i-vectors (one/frame) for each class (silence, native language and English). In the test phase, an i-vector is extracted for each test frame and scored using the MLR model. For this task, the use of an explicit i-vector class for silence trained on the target dataset significantly improved the silence frame detection performance over using either GMM or DNN pretrained VAD models.

3.2. Phonotactic identification

The phonotactic approach to automatic language identification relies on the idea that the phonetic sequence in an audio sample is characteristic of the language used. It has been shown over time to perform very competitively compared to purely acoustic approaches, from phone-based acoustic decoding [32] to parallel phone recognizers [33] and phone lattices, as presented to LRE15 [31]; RNN-based phonotactic models were also shown as very efficient [34]. Similar to [31], phone decoders using phonetic models from several languages are used to decode the training data and to estimate phone n-gram statistics on the resulting phone lattices for each target class; then, given a new utterance, expectation of its phonetic log-likelihood is computed according to each target models, resulting in a set of posterior scores. The implementation relies on the VoxSigma Software Suite, a commercial product from Vocapia.

For task A and for each target language, a pair of phonotactic models was estimated separately using either the positives code-switched sentences or the negative non-code-switched sentences. The development/evaluation utterances were then scored against the matching language pair. Given the global balance between positive and negative samples in the data set, a 0.5 decision threshold was chosen, i.e. the class with the highest posterior score was selected.

To train models for task B, the frames labeled as silence in the reference annotation were discarded and a pair of models (one for English and one for the primary language) were trained for each target language on the associated training audio segments. For development and evaluation, the frame-level VAD from the i-vector module, as described at the end of previous sub-section, was first applied and frames automatically labelled as silence were discarded prior to further processing. The remaining frames were scored against both the code-switched and non code-switched models. Similar to the i-vector system, each frame was extended to a 600ms audio segment centered around it for training or scoring. As explained in Section 2, the distribution of code-switched and native speech frames is unequal, so a subset of the training data was kept apart and used for to optimize the decision threshold for the code-switched class.

3.3. Systems combination

Since i-vector and phonotactic modeling capture different, and potentially complementary, information, the two models were combined for the primary submission. For both tasks, a linear combination of the posterior scores of the phonotactic and the i-vector models was used, using weights optimized on the development data.

Table 3: *Task A: accuracy (%) by language on the development / on the evaluation data sets for the phonotactic, i-vector and combined systems. Accuracy of the organizer's baseline system is also given on the dev set. Best score on evaluation data in bold.*

Language	baseline	phonotactic	i-vector	combined
Gujarati	76.8	79.1 / **75.4**	84.1 / 57.3	84.3 / 68.8
Tamil	71.2	77.4 / 76.6	79.1 / 76.4	82.2 / **79.8**
Telugu	74.0	79.0 / 77.1	78.8 / 78.3	81.5 / **79.4**
Average	74.0	78.5 / 76.4	80.6 / 70.7	83.3 / 76.0

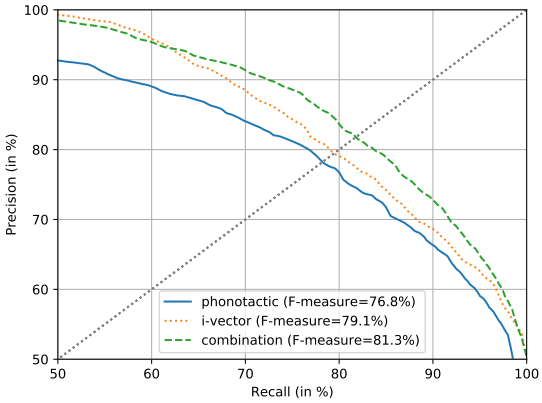

Figure 1: *Task A: recall and precision of monolinugal vs. code-switched utterance detection on the dev set, for the phonotactic, i-vector and combined systems averaged across all languages. F-measure is given for the chosen operating point.*

4. Experiments and results

Figure 1 shows the recall and precision for the code-switched utterance detection (Task A) on the development data set, as computed globally across the three languages for the combined system and the individual phonotactic and i-vector systems. The combination weight was set to 0.4 for the phonotactic scores and 0.6 for the i-vector scores. We can see that the i-vector system generally performs better than the phonotactic system, and that their combination provides a gain for almost all operating points. This is confirmed by looking at the detailed accuracy by language in Table3. On the development data, the gap between the phonotactic and i-vector models is especially high for Gujarati (79.1 vs 84.1%), but even in this case the combination is slightly positive. The performance difference is less for Tamil (and even slightly better for the phonotactic models for Telugu), where their combination provides more than a 2% absolute gain in accuracy. On average for the three languages, the combined system achieves an accuracy of 83.3% on the development set, i.e. an error rate of 16.7%.

On the evaluation dataset, there is a very specific and dramatic degradation of performance for the i-vector system on Gujarati, dropping from 84.1% on the development set to 57.3% on the evaluation set, which carries over to a lesser extent in the combined system. This resulted from a shift of the Gujarati i-vector score distribution on the evaluation set compared to the development set (a 22% relative increase of the average score), which was not observed for the other languages. It is interesting that the phonotactic system proved to be much more robust, with a more limited reduction from 79.1% to 75.4%. For Tamil and Telugu, the performance on the evaluation data was slightly reduced compared with the development data, while keeping a 1-3% absolute gain in accuracy due to system combination. Overall, the combined system has an accuracy of 76.0%, i.e. an error rate of 24%.

For Task B, Figure 2 shows the balance between the false positive and false negative rates of code-switched frame detection as a function of the decision threshold. The axes are scaled by their standard normal deviates, and non-speech frames were excluded for the figure. The phonotactic and i-vector systems show similar behaviors and their combination brings an im-

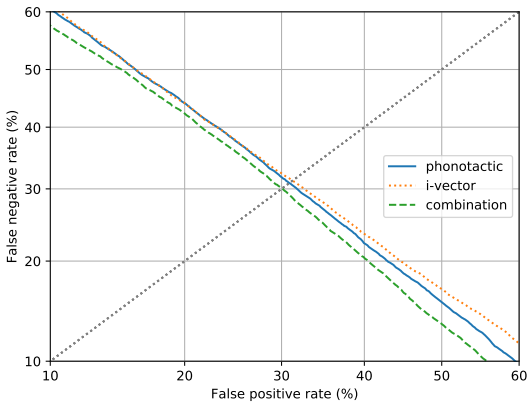

Figure 2: *Task B: false positive and false negative rates of code-switched frame detection computed on the development set and cumulated on all language pairs, for the phonotactic, i-vector and combined systems. Non-speech frames are excluded.*

Table 4: *Task B: accuracy (%) by language on the development / evaluation data sets for the phonotactic, i-vector and combined systems. Accuracy of the organizer's baseline system is also given on the dev set. Best score on evaluation in bold.*

Language	baseline	phonotactic	i-vector	combined
Gujarati	76.7	79.9 / 76.9	80.0 / 76.9	80.5 / **77.7**
Tamil	76.5	79.5 / 77.5	80.8 / 78.6	81.2 / **78.8**
Telugu	77.6	80.0 / 78.9	81.4 / 78.9	81.8 / **79.6**
Average	76.9	79.8 / 77.8	80.7 / 78.1	81.2 / 78.7

Table 5: *Task B: frame-level confusion matrix for the combined system on the development set; Columns for reference, rows for hypothesis. SIL stands for non-speech, CS for code-switched English, P for primary language ie. Gujarati, Tamil or Telugu.*

hyp \ ref	SIL	CS	P
SIL	14,097 (69.7%)	248 (1.9%)	1,797 (2.5%)
CS	445 (2.2%)	3,016 (22.6%)	2,885 (4.0%)
P	5,675 (28.1%)	10,068 (75.5%)	68,089 (93.5%)

provement across all operating points. The discrimination between code-switched and non code-switched frames appears to be a difficult task with an equal error rate above 30%. The actual accuracy rates shown in Table 4 confirm this behavior both on the development and evaluation data sets. The combined system accuracy rates averaged across the 3 languages are 81.2% and 78.7%, respectively.

Given the relatively low prior of the code-switched samples at about 12%, the decision thresholds of the systems were optimized according to the evaluation primary metric towards a very low false positive (or false alarm) rate for code-switching and thus a very high negative (or miss) rate. The confusion matrix in Table 5 between the three target classes (silence, code-switched English or primary language) of the combined system, cumulated for all languages on the development set, shows that only 3016 code-switched frames out of 13332, i.e. 22.6%, were correctly labelled, leaving room for improvement. Conversely, identification was correct for 68,089 out of 72,771 speech frames in the primary language, i.e. 93.5%.

One important factor impacting code-switching detection should be the length of the segments; furthermore, code-switched segment shorter than the 600ms analysis window will provide a very sparse information to the phonotactic or i-vector modelling. We show on Figure 3 the histogram of code-switched segments according to their duration: 56% of the segments last only 200 or 400ms. We also show the accuracy of our combined system on the development set cumulated for all languages, in the situation where code-switching would account for half of the spoken content, corresponding to the equal-error-rate configuration. As expected, the accuracy increases with duration, raising from 63% for 200ms segments to 74% for 1.2sec segments.

5. Conclusion

The shared task on Code-switched Spoken Language Identification allowed us to compare different approaches for the utterance-level and frame-level detection of code-switched speech, thanks to the annotated corpora provided in three language pairs. For each combination of task and language pair, one of our systems was ranked first among the submitted systems. In general, both the phonotactic and i-vector acoustic modeling obtained comparable performances, and a simple linear combination brought a further improvement showing their complementarity.

For the utterance-level detection task, an F-measure of about 80% was obtained. Compared to the provided baseline system with a 74% accuracy average across the three languages, our combined system had an 83.3% accuracy on the development data. Seen conversely, the error rate was reduced from 26% to 16.7%. On the evaluation set, its performance was lower, 76.0%, caused by a distribution shift of the i-vector scores for one of the language pairs. The phonotactic scores were less affected by this distribution mismatch and the phonotactic modeling appears to be more robust than the i-vector approach.

For the frame-level annotation task, the accuracy appears to be of the same order of magnitude, with an average of 81.2% on the development set and 78.7% on the evaluation set. However, a detailed analysis revealed a very high rejection of the code-switched frames, which amount to only 12% of the corpus. This shows that a more precise modeling of the code-switched segments is needed for an accurate segmentation. For this aim, following metrics more specifically fitted to the code-switching detection task as e.g. the ones proposed by Guzmán et al. [35], would certainly be beneficial.

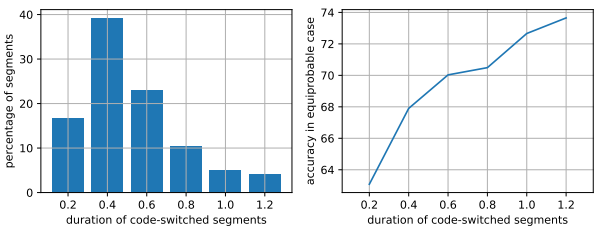

Figure 3: *Task B: histogram of code-switched segments as a function of their duration (left) and accuracy in code-switching frame detection in a equiprobable setting (right), cumulated over all languages for the combined system on the dev set.*

6. References

[1] J. Y. C. Chan, P. C. Ching, and T. Lee, "Development of a Cantonese-English Code-Mixing Speech Corpus," in *Interspeech 2005*, Lisbon, Portugal, Sep. 2005, p. 4.

[2] D.-C. Lyu, T.-P. Tan, E. S. Chng, and H. Li, "SEAME: A Mandarin-English Code-Switching Speech Corpus in South-East Asia," in *Interspeech 2010*, Makuhari, Chiba, Japan, Sep. 2010, p. 4.

[3] E. Yilmaz, M. Andringa, S. Kingma, J. Dijkstra, F. V. der Kuip, H. V. de Velde, F. Kampstra, J. Algra, H. van den Heuvel, and D. van Leeuwen, "A Longitudinal Bilingual Frisian-Dutch Radio Broadcast Database Designed for Code-Switching Research," in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*. Paris, France: ELRA, May 2016.

[4] V. Ramanarayanan and D. Suendermann-Oeft, "Jee haan, I'd like both, por favor: Elicitation of a Code-Switched Corpus of Hindi–English and Spanish–English Human–Machine Dialog," in *Interspeech 2017*. ISCA, Aug. 2017, pp. 47–51.

[5] E. V. der westhuizen and T. Niesler, "A First South African Corpus of Multilingual Code-switched Soap Opera Speech," in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. Miyazaki, Japan: ELRA, May 2018.

[6] I. Hamed, M. Elmahdy, and S. Abdennadher, "Collection and Analysis of Code-switch Egyptian Arabic-English Speech Corpus," in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. Miyazaki, Japan: ELRA, May 2018.

[7] L. Nguyen and C. Bryant, "CanVEC - the Canberra Vietnamese-English Code-switching Natural Speech Corpus," in *Proceedings of The 12th Language Resources and Evaluation Conference*. Marseille, France: ELRA, May 2020, pp. 4121–4129.

[8] P. Guo, H. Xu, L. Xie, and E. S. Chng, "Study of Semi-supervised Approaches to Improving English-Mandarin Code-Switching Speech Recognition," in *Interspeech 2018*. ISCA, Sep. 2018, pp. 1928–1932.

[9] S. Ganji and R. Sinha, "A Novel Approach for Effective Recognition of the Code-Switched Data on Monolingual Language Model," in *Interspeech 2018*. ISCA, Sep. 2018, pp. 1953–1957.

[10] E. van der Westhuizen and T. Niesler, "Synthesising isiZulu-English Code-Switch Bigrams Using Word Embeddings," in *Interspeech 2017*. ISCA, Aug. 2017, pp. 72–76.

[11] V. Soto and J. Hirschberg, "Improving Code-Switched Language Modeling Performance Using Cognate Features," in *Interspeech 2019*. ISCA, Sep. 2019, pp. 3725–3729.

[12] D. Amazouz, M. Adda-Decker, and L. Lamel, "Addressing Code-Switching in French/Algerian Arabic Speech," in *Interspeech 2017*. ISCA, Aug. 2017, pp. 62–66.

[13] E. Yılmaz, H. van den Heuvel, and D. V. Leeuwen, "Exploiting Untranscribed Broadcast Data for Improved Code-Switching Detection," in *Interspeech 2017*. ISCA, Aug. 2017, pp. 42–46.

[14] Q. Wang, E. Yılmaz, A. Derinel, and H. Li, "Code-Switching Detection Using ASR-Generated Language Posteriors," in *Interspeech 2019*. ISCA, Sep. 2019, pp. 3740–3744.

[15] S. Rallabandi and A. W. Black, "Variational Attention Using Articulatory Priors for Generating Code Mixed Speech Using Monolingual Corpora," in *Interspeech 2019*. ISCA, Sep. 2019, pp. 3735–3739.

[16] G. Lee, X. Yue, and H. Li, "Linguistically Motivated Parallel Data Augmentation for Code-Switch Language Modeling," in *Interspeech 2019*. ISCA, Sep. 2019, pp. 3730–3734.

[17] B. M. L. Srivastava and S. Sitaram, "Homophone Identification and Merging for Code-switched Speech Recognition," in *Interspeech 2018*. ISCA, Sep. 2018, pp. 1943–1947.

[18] E. Yılmaz, S. Cohen, X. Yue, D. A. van Leeuwen, and H. Li, "Multi-Graph Decoding for Code-Switching ASR," in *Interspeech 2019*. ISCA, Sep. 2019, pp. 3750–3754.

[19] A. Biswas, E. Yılmaz, F. de Wet, E. van der Westhuizen, and T. Niesler, "Semi-Supervised Acoustic Model Training for Five-Lingual Code-Switched ASR," in *Interspeech 2019*. ISCA, Sep. 2019, pp. 3745–3749.

[20] H. Seki, T. Hori, S. Watanabe, J. L. Roux, and J. R. Hershey, "End-to-End Multilingual Multi-Speaker Speech Recognition," in *Interspeech 2019*. ISCA, Sep. 2019, pp. 3755–3759.

[21] D. Najim, K. Patrick, D. Reda, D. Pierre, and O. Pierre, "Front-End Factor Analysis for Speaker Verification," *IEEE Trans. on Acoustics, Speech and Signal Process.*, vol. 19, no. 4, pp. 788–798, 2011.

[22] P. Kenny, "Bayesian speaker verification with heavy-tailed priors." in *Proc. Odyssey*, 2010, p. 14.

[23] D. Garcia-Romero and C. Y. Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems." in *Proc. Interspeech*, 2011, pp. 249–252.

[24] D. Martınez, O. Plchot, L. Burget, O. Glembek, and P. Matejka, "Language recognition in ivectors space," *Proc. Interspeech*, pp. 861–864, 2011.

[25] H. Hermansky and S. Sharma, "Traps – classifiers of temporal patterns." in *Proc. ICSLP*, 1998.

[26] P. Schwarz, P. Matejka, and J. Cernocky, "Towards lower error rates in phoneme recognition." in *TSD*, Sep 2004, pp. 456–472.

[27] D. P. X. Zhang, J. Trmal and S. Khudanpur, "Improving deep neural network acoustic models using generalized maxout networks." in *Proc. IEEE ICASSP*, Adelaide, May 2014.

[28] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The Kaldi Speech Recognition Toolkit," in *IEEE Workshop on Automatic Speech Recognition and Understanding*, Dec. 2011.

[29] D. van Leeuwen and N. Brummer, "Channel-dependent gmm and multi-class logistic regression models for language recognition," in *Proc. Odyssey*, 2006.

[30] S. J. Prince and J. H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *IEEE 11th Int. Conf. on Computer Vision. ICCV 2007.*, 2007, pp. 1–8.

[31] G. Gelly, J.-L. Gauvain, L. Lamel, A. Laurent, V. B. Le, and A. Messaoudi, "Language Recognition for Dialects and Closely Related Languages," in *Proc. Odyssey*, Jun. 2016, pp. 124–131.

[32] L. F. Lamel and J.-L. Gauvain, "Language identification using phone-based acoustic likelihoods," in *Proc. IEEE ICASSP*, Adelaide, Apr. 1994.

[33] M. A. Zissman, "Comparison of four approaches to automatic language identification of telephone speech," *IEEE Trans. on Speech and Audio Process.*, vol. 4, no. 1, pp. 31–44, 1996.

[34] B. M. L. Srivastava, H. Vydana, A. K. Vuppala, and M. Shrivastava, "Significance of neural phonotactic models for large-scale spoken language identification," in *2017 International Joint Conference on Neural Networks (IJCNN)*. Anchorage, AK, USA: IEEE, May 2017, pp. 2144–2151.

[35] G. Guzmán, J. Ricard, J. Serigos, B. E. Bullock, and A. J. Toribio, "Metrics for Modeling Code-Switching Across Corpora," in *Interspeech 2017*. ISCA, Aug. 2017, pp. 67–71.

2933

Exploiting Spectral Augmentation for Code-Switched Spoken Language Identification

Pradeep Rangan, Sundeep Teki*, and Hemant Misra*

Applied Research Department, Swiggy (Bundl Technologies India Limited)

[pradeep.r1, sundeep.teki, hemant.misra] @swiggy.in

Abstract

Spoken language Identification (LID) systems are needed to identify the language(s) present in a given audio sample, and typically could be the first step in many speech processing related tasks such as automatic speech recognition (ASR). Automatic identification of the languages present in a speech signal is not only scientifically interesting, but also of practical importance in a multilingual country such as India. In many of the Indian cities, when people interact with each other, as many as three languages may get mixed. These may include the official language of that province, Hindi and English (at times the languages of the neighboring provinces may also get mixed during these interactions). This makes the spoken LID task extremely challenging in Indian context. While quite a few LID systems in the context of Indian languages have been implemented, most such systems have used small scale speech data collected internally within an organization. In the current work, we perform spoken LID on three Indian languages (Gujarati, Telugu, and Tamil) code-mixed with English. This task was organized by the Microsoft research team as a spoken LID challenge. In our work, we modify the usual spectral augmentation approach and propose a language mask that discriminates the language ID pairs, which leads to a noise robust spoken LID system. The proposed method gives a relative improvement of approximately 3-5% in the LID accuracy over a baseline system proposed by Microsoft on the three language pairs for two shared tasks suggested in the challenge.

Index Terms: spoken language identification, code-mixing, spectral augmentation, noise-robustness, speech recognition

1. Introduction

In recent years, advances in artificial intelligence (AI) have significantly expanded the degree to which individuals can interact with technology utilizing just their voice [1]. However, these systems often require explicit information about the language of the users. The ability to dynamically process a single input speech stream consisting of different languages would expand the usefulness of existing voice-based systems and open up a wide array of additional functionalities. The spoken language identification (LID) research addresses this issue by exploring how to extract information from the audio signal and use it to predict the spoken language. LID is used in several applications such as multilingual translation systems or emergency call routing, where the response time of a fluent native operator might be critical [2].

Over the years, researchers have utilized many prosodic and acoustic features to construct machine learning models for LID systems [3]. Several prosodic and acoustic features are based on phonemes, which become the underlying features that drive the performance of the statistical models [4]. If two languages have many overlapping phonemes, then identifying them becomes a challenging task for a classifier. Subsequent LID systems relied on acoustic modelling [5]. In particular, guided by the advances on speaker verification, the use of i-vector feature extractors as a front-end followed by diverse classification mechanisms became popular as acoustic LID systems [6] [7]. The extensive feature engineering with i-vectors results in very complex systems, with an increasing number of computational steps in their pipeline [8].

Approaches solely based on applying neural networks on input features like mel-frequency cepstral coefficients (MFCC) show that they reach state-of-the-art results, while being less complex [9]. Current research on language identification systems using deep neural networks (DNN) mainly focuses on using different forms of long short term memory (LSTMs), working on input sequences of transformed audio data. The resulting sequence features are fused together and used to classify the language of the input samples. In [10] [11], a DNN based architecture was proposed for extracting spatial features from the log-mel spectrograms of raw audio using convolutional neural networks (CNNs) and then using recurrent neural networks (RNNs) for capturing temporal features to identify the language.

Automatic spoken LID systems are particularly relevant for multilingual countries such as India. An LID system recognising 27 Indian languages was implemented using Gaussian mixture model (GMM) and MFCC features [12]. Another LID system for Bengali, Hindi, Telugu, Urdu, Assamese, Punjabi and Manipuri languages was implemented that used feed forward neural networks trained with two hours of speech data of each of the seven languages [13]. Most LID systems for Indian languages were trained and tested with either speech of professional news readers [2] or with small scale speech data collected by researchers [14] [15], that may not be available for public use. Till recently, the lack of publicly available Indian language speech databases was a limitation for a scientific study of various acoustic analysis and modeling techniques. Recently, a phonetically balanced speech corpus of Hindi-English code-mixed speech was developed by IIT Guwahati [16], and is explicitly designed for automatic speech recognition (ASR) paradigm, but not for the LID task.

Microsoft organized a shared task on Code-switched spoken language identification (LID) in three language pairs – Gujarati-English, Telugu-English and Tamil-English [17]. The shared task consists of two subtasks:(1) utterance-level identification of monolingual vs. code-switched utterances, and, (2) frame-level identification of languages in a code-switched utterance.

Spectral augmentation is a popular technique to improve the noise robustness of an ASR system [18] [19]. It involves three main steps namely time warping, frequency masking, and tem-

*These authors contributed equally to this work

poral masking. In this paper, we have explored the spectral augmentation [20] for detecting the LID information. The spectral augmentation approach randomly chooses the position of the different masks. However, in our work we select the temporal masks based on the number and position of the English segments in a code-switched utterance. The clean and the masked spectrograms are provided as an input to the CNN-BiLSTM encoder network trained with CTC loss function. The variation of frequency and temporal masks in the recent spectral augmentation [20] are found to provide discriminative power to identify the language information in a code-mixed speech data.

The organization of the remaining sections of the paper are as follows: In Section 2, the importance of spectral augmentation for LID is discussed. The proposed framework for spoken LID is explained in Section 3. Section 4 provides the experimental set-up of the proposed spectral augmentation based spoken LID. We analyze the impact of the proposed approach on spoken LID performance in Section 5. Section 6 describes the conclusions and future directions.

2. Spectral Augmentation

In this section, the related work on spectral augmentation for different speech processing applications is discussed. Data augmentation is a popular method for improving robustness and training of neural networks [18] . It's been used successfully in several domains starting from image classification to molecular modelling [21]. The fundamental principle is to increase the amount of training samples by creating multiple variants of the dataset. The data augmentation method is typically applied to create an additional training data for ASR to improve the performance. For instance, in [22], the data was augmented for low resource speech recognition tasks. Vocal Tract Length Normalization has been explored for data augmentation in [23]. Speech perturbation has been applied on raw audio for LVCSR tasks in [19]. The use of an acoustic room simulator has been adopted in [22]. Data augmentation is used to spot the important keywords from the speech utterance [24]. Perceptually, human listeners show remarkable tolerance to a variety of spectrotemporal manipulations of the input sound signal during segregation of speech-in-noise recognition tasks [25] [26] [27]. Inspired by such studies in human auditory perception and cognitive neuroscience, and the recent success of augmentation techniques in the speech and vision domains, SpecAugment, an augmentation method that operates on the log mel spectrogram of the input audio (rather than the raw audio) was proposed in [20]. This method is simple and computationally inexpensive, as it directly acts on the log mel spectrogram as if it was an image.

Spectral Augmentation includes the following steps:

1. **Time warping**: Given a log mel spectrogram with τ time steps, it is viewed as an image where horizontal and vertical axes represents time (s) and the frequency (Hz) respectively. A random point along the horizontal line passing through the center of the image within the time steps (W, τ - W) is to be warped either to the left or right by a distance w chosen from a uniform distribution from 0 to the time warp parameter W along that line.
2. **Frequency Masking**: It is applied so that f consecutive mel frequency channels [f_0, f_0 + f) are masked, where f is first chosen from a uniform distribution from 0 to the frequency mask parameter F, and f_0 is chosen from [0, v - f). v is the number of mel frequency channels.
3. **Temporal Masking**: It is applied so that t consecutive time steps [t_0, t_0 + t) are masked, where t is first chosen from a uniform distribution from 0 to the time mask parameter T, and t_0 is chosen from [0, τ - t).

Figure 1 shows examples of the individual augmentations applied to a single speech input (PartBGujarati/Dev/Audio/000060438.wav). The log mel spectrograms are normalized to have zero mean value, and thus setting the masked value to zero is equivalent to setting it to the mean value.

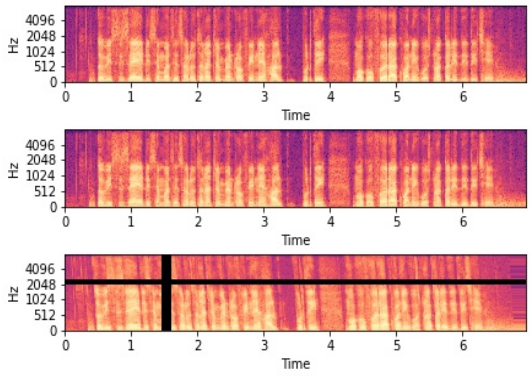

Figure 1: *Augmentations applied to the base input. The figures (from top) depict the log mel spectrogram of the base input with no augmentation, (time warp, frequency and time masking applied)*

The spectral augmentation approach is shown to be successful on Noise robust ASR [18]. For instance, the selection of different masks in the spectral augmentation can discriminate between the clean speech and the noisy speech. In code-mixed speech spoken in India, it is generally observed that the native speaker tries to speak in her native language, and occasionally uses English words. The process of masking the English words can lead to building a mono-lingual corpus. In order to build an efficient spoken LID system, the DNN may first learn to discriminate between the code-mixed speech and the mono-lingual speech. Since the temporal masking positions can be altered with respect to different languages, we sought to mask out the English words in the code-switched speech.

3. Proposed Work

Figure 2 shows the CNN-LSTM system that uses CTC loss function at the output layer. The fundamental end-to-end CTC

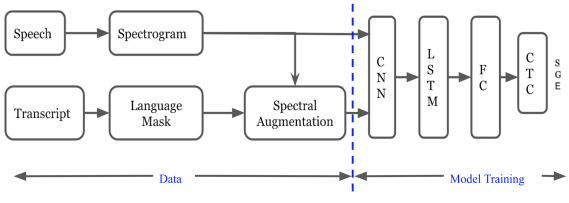

Figure 2: *End-to-End spoken LID using CNN-LSTM model*

system receives the 2-D spectrogram as shown in the bottom of the figure. The CNN layer is effectively used in speech processing applications as they are capable of modeling temporal invariance for variable length utterances [28]. The BiLSTM

models are trained on the convolved features to capture long-term sequential context. Later, the dense layer or the fully connected (FC) layer is connected to the output layer. The output is produced through a softmax function computing a probability distribution over the target labels. The target labels used in Figure 2 are S, G and E which corresponds to Silence, Gujarati and English language ID, respectively. To accelerate the training procedure, Batch Normalization [1] is applied on hidden layers. In our work, we feed the clean spectrogram, and the augmented spectrogram to the end-to-end LID system. However, the spectral augmentation is obtained by proposing an appropriate temporal mask termed as 'language mask' in the paper, and explained in the next subsection.

3.1. Language Transcript based Temporal mask

The conventional temporal mask(s) used in the spectral augmentation is selected at a random position that follows a uniform distribution from 0 to the time mask parameter T. In our work, at the time of training, the temporal masks are obtained from the language ID transcripts. The steps followed to obtain the temporal mask from the language transcript are as follows:

1. Initially, the number of characters in the transcript is computed (l_1).
2. Find the language segments from the transcript. Ex: If transcript = 'SSSGGGEEGG', then the language segments are 'SSS', 'GGG', 'EE', and 'GG', where 'S', 'E' and 'G' corresponds to the silence, English, and Gujarati speech segments, respectively.
3. Obtain the maximum number of frames based on the length of the characters in the transcript (N_f). Since the language label is created for every 200 ms for the shared task, every language ID character corresponds to 200 ms of speech content.
4. We calculate the number of temporal masks by finding the number of speech segments corresponding to the English speech. Since, the non-predominant language in the code-mixed speech data is English, we mask the English words from the spectrogram.
5. Obtain the start (t_0), and end time frame index (t_1) for every English speech segment.
6. The t_0 to t_1 consecutive time steps [t_0, t_1] are masked.

Figure 3 shows a toy example to augment the spectrogram based on the language mask. The spectrogram is obtained for the speech signal 'PartB/Gujarati/000010183.wav' as shown in Figure 3 (a). The speech signal is sampled at 16 KHz with window size of 200 ms, and window stride of 100 ms. The language transcripts are mapped to the speech frames, and are shown in Figure 3 (b). For instance, there is an English phrase ('flight operation') in the speech signal from 3.21 s to 3.99 s (shown in black). The number of temporal masks are decided based on the number of English segments present in the language transcripts. In this case, the number of English segments are two, and are present from (a) 3.1 - 3.99 s, and (b) 4.69 - 4.79 s. The corresponding time segments are masked as shown in Figure 3 (c) .

4. Experiments

First, we introduce our experimental environment and the metrics used for the spoken LID task. Then, we show our results on the shared tasks. Following this, we show the results of our experiments on the proposed work.

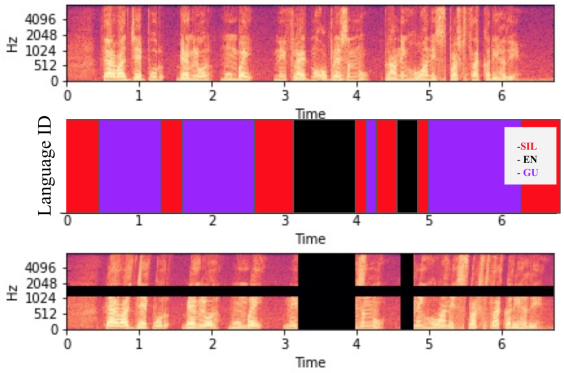

Figure 3: *Illustrative example of the proposed LID based temporal masking in spectral augmentation: (Top) Raw spectrogram, (Middle) Language information at different time instance, and (Bottom) Proposed spectral augmentation*

4.1. Data

The data set consists of three code-switched language pairs - Gujarati-English, Tamil-English and Telugu-English. The shared task for the workshop consists of two sub-tasks for spokenLID of code-switched audio. The two sub-tasks will consist of (1) Part A: Utterance-level LID, and (2) Part B: Frame-level LID .

Table 1: *Dataset description for the code-mixed spoken LID*

Data	Language	Number of Samples (Duration in hh:mm:ss)		
		Train	Dev	Test
Part-A	Gu-En	16780 (31:59:53)	2091 (3:59:55)	2156 (3:44:50)
	Ta-En	17943 (31:59:47)	2258 (4:00:05)	2258 (3:53:59)
	Te-En	16991 (31:59:50)	2135 (4:00:06)	2064 (3:42:15)
Part-B	Gu-En	8620 (15:59:56)	1080(1:59:58)	1078 (2:00:02)
	Ta-En	8982 (15:59:57)	1135 (1:59:59)	1129 (1:59:57)
	Te-En	8226 (15:59:56)	1047 (2:00:03)	1033 (1:59:55)

4.2. Training

The Microsoft's baseline system is made up of an end-to-end multi-layer model consistingof 5 layers of LSTM, each consisting of 1024 neurons. The model is based on deepspeech-2 [29]. The model is trained using the CTC loss function. The LID detector using CTC starts with two layers of 2D convolutions over both time and frequency domains with 32 channels, 41 × 11, 21 × 11 filter dimensions, and 2 × 2, 2 × 1 stride. Next, five Bi-LSTM layers with 1024 hidden units are followed by one fully connected linear layer with 5 softmax outputs blank, , Gujarati, English, Silence. The Bi-LSTM models have around 10.2 millions (M) parameters. The input sequence are values of spectrogram slices, 20 ms long, computed from Hamming windows with 10 ms frame shifts. The output (target) sequence was obtained directly from the letters of the word transcription. We used 100 epochs to train all the models used for further evaluation. The two variants of spectral augmentation are implemented in this paper: (1) spectral augmentation with random temporal, and frequency mask positions, and (2) spectral augmentation with the proposed language mask.

4.3. Decoding

Assuming an input sequence of length T , the output of the neural network will be $p(c|x_t)$ for $t = 1, ..., T$. Let $p(c|x_t)$ is a distribution over possible characters in the alphabet ψ (which includes the blank symbol) given audio input x_t. In order to recover a character string from the output of the neural network, as a first approximation, we take the argmax at each time step. Let $S = (s_1, ..., s_T)$ be the character sequence where

$$s_t = \arg\max_{c \in \psi} p(c|x_t) \quad (1)$$

The sequence S is mapped to a transcription by collapsing repeat characters and removing blanks. On the other hand, the beam search decoder uses the context information in generating the decoded sequence. There are two cases: either we extend the beam by a character c different from the last character, then there is no need for separating blanks in the paths, or the last character is repeated.

4.4. Evaluation

For task A, the predicted label (Monolingual or code- switched) file is submitted for a particular audio file in the blind test set. For task B, a frame-level (200ms) label for each frame in the audio is submitted. The LID system performance is evaluated using accuracy and Equal Error Rate (EER) as evaluation metrics.

$$Accuracy = \frac{N}{T} \quad (2)$$

where N and T are the total no. of correctly predicted data samples and the total no. of data points in the speech dataset respectively.

$$EER = \frac{FRR + FAR}{2} \quad (3)$$

where $FRR = TFR/T$, and $FAR = TFA/T$; FRR, FAR, TFR, TFA and T corresponds to false rejection rate, false acceptance rate, total no. of false rejects, total no. of false accepts and total number of datapoints, respectively.

5. Results and Discussion

The Deepspeech-2 model is trained on the baseline configurations and on the variants of spectral augmentation. In this section, we report the accuracies and EERs on the Test set for Task-A and Task-B.

Table 2 shows the % spoken LID Accuracy of the baseline model and the proposed spectral augmentation method. It can be observed that the proposed system outperforms as compared to that of the baseline model on Task-A.

Table 2: *Spoken LID accuracy on Task-A for different datasets*

Data_Test	% Acc [Baseline]	% Acc [SpecAug]
Gu-En	71.9	73.01
Ta-En	71.2	79.02
Te-En	74.0	78.65

For Task-B, we compare the performance of the code-mixed language recognizer using spectral augmentation with random temporal mask location, and the proposed language mask. Table 3 shows the performance comparison of the proposed work, and the DeepSpeech-2 model (Microsoft's Baseline) on the test set of Part-B with Greedy Search Decoding. It can be observed that the proposed language mask on the spectrogram is able to discriminate the language ID's within an utterance.

Table 3: *Performance comparison of the proposed work, and the DeepSpeech-2 model (Microsoft's Baseline) on the test set of Part-B by Greedy Search Decoder*

Language	% Acc [%EER] on the test set on Greedy search decoder		
	Baseline	SpecAug	SpecAug + Lang Mask
Gujarati	66.79 [9.71]	75.33 [7.78]	75.72 [7.53]
Tamil	72.17 [8.42]	74.80 [7.76]	75.02 [7.67]
Telugu	70.54 [8.65]	74.06 [7.88]	74.08 [7.87]

Table 4: *Performance comparison of the proposed work, and the DeepSpeech-2 model (Microsoft's Baseline) on the test set of Part-B by Beam Search Decoder*

Language	% Acc [%EER] on the test set on Beam search decoder		
	Baseline	SpecAug	SpecAug + Lang Mask
Gujarati	72.53 [8.54]	76.33 [7.52]	76.64 [7.36]
Tamil	73.89 [8.06]	75.80 [7.54]	76.06 [7.44]
Telugu	75.20 [7.68]	75.90 [7.47]	75.84 [7.47]

Table 4 shows the performance comparison of the proposed work, and the state-of-the-art methods on the test set of Part-B on Beam Search Decoder. The size of the beam width is varied from 5 to 20 in steps of 5. It is observed that the beam width size of 15 is optimal for all languages in the shared task.

5.1. Discussion

The research on spoken LID for code-switched speech is relatively new for Indian languages. Further, in India it is fairly common to mix words from English and Hindi along with the native language. Spectral augmentation has shown promising improvement on noise robust ASR, and on low resource languages. Instead of masking random positions in the spectrogram, the positions of the language transitions are masked. Due to this, the train speech corpus contains only the code- switched, and the mono-lingual utterances. As a result, the model learns to discriminate well between the code-switched utterances, and the mono-lingual utterances. Also, the positions of the proposed language mask captures the corresponding grammar in the utterance. Thereby the model learns about the different statistics of code-mixing. On the speech segments where there are small portions of non-dominant language is involved, the model captures these small language transitions.

6. Conclusion

Spoken LID is performed on the three Indian languages (Gujarati, Telugu, and Tamil) code-switched with English. We proposed a language mask from the speech transcripts, and incorporated it in the spectrogram. We observed that the proposed method is able to discriminates the languages in a code-mixed data. The proposed method gives a relative improvement in performance of approximately 3-5% in the LID accuracy over that of a baseline system proposed by Microsoft on three code-switched language pairs for two different shared tasks.

7. References

[1] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The

shared views of four research groups," *IEEE Signal processing magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[2] C. Kumar and H. Li, "Language identification for multilingual speech recognition systems," in *9th Conference Speech and Computer*, 2004.

[3] Y. Obuchi and N. Sato, "Language identification using phonetic and prosodic hmms with feature normalization," in *Proceedings.(ICASSP'05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, vol. 1. IEEE, 2005, pp. I–569.

[4] N. E. Safitri, A. Zahra, and M. Adriani, "Spoken language identification with phonotactics methods on minangkabau, sundanese, and javanese languages." in *SLTU*, 2016, pp. 182–187.

[5] R. Tong, B. Ma, D. Zhu, H. Li, and E. S. Chng, "Integrating acoustic, prosodic and phonotactic features for spoken language identification," in *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, vol. 1. IEEE, 2006, pp. I–I.

[6] J. Gonzalez-Dominguez, I. Lopez-Moreno, J. Franco-Pedroso, D. Ramos, D. T. Toledano, and J. Gonzalez-Rodriguez, "Multilevel and session variability compensated language recognition: Atvs-uam systems at nist lre 2009," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 6, pp. 1084–1093, 2010.

[7] P. A. Torres-Carrasquillo, E. Singer, T. Gleason, A. McCree, D. A. Reynolds, F. Richardson, and D. Sturim, "The mitll nist lre 2009 language recognition system," in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2010, pp. 4994–4997.

[8] D. Martinez, O. Plchot, L. Burget, O. Glembek, and P. Matějka, "Language recognition in ivectors space," in *Twelfth annual conference of the international speech communication association*, 2011.

[9] G. Gelly, J.-L. Gauvain, L. Lamel, A. Laurent, V. B. Le, and A. Messaoudi, "Language recognition for dialects and closely related languages." in *Odyssey*, vol. 2016, 2016, pp. 124–131.

[10] S. Shukla, G. Mittal *et al.*, "Spoken language identification using convnets," in *European Conference on Ambient Intelligence*. Springer, 2019, pp. 252–265.

[11] C. Bartz, T. Herold, H. Yang, and C. Meinel, "Language identification using deep convolutional recurrent neural networks," in *International Conference on Neural Information Processing*. Springer, 2017, pp. 880–889.

[12] S. G. Koolagudi, D. Rastogi, and K. S. Rao, "Identification of language using mel-frequency cepstral coefficients (mfcc)," *Procedia Engineering*, vol. 38, pp. 3391–3398, 2012.

[13] K. S. Rao, V. R. Reddy, and S. Maity, *Language identification using spectral and prosodic features*. Springer, 2015.

[14] C. Madhu, A. George, and L. Mary, "Automatic language identification for seven indian languages using higher level features," in *2017 IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES)*. IEEE, 2017, pp. 1–6.

[15] J. Chakraborty, S. Nath, S. Nirmala, and K. Samudravijaya, "Language identification of assamese, bengali and english speech." in *SLTU*, 2018, pp. 177–181.

[16] A. Pandey, B. M. L. Srivastava, R. Kumar, B. T. Nellore, K. S. Teja, and S. V. Gangashetty, "Phonetically balanced code-mixed speech corpus for hindi-english automatic speech recognition," in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[17] Microsoft, *Spoken LID Challenge*, 2020. [Online]. Available: https://www.microsoft.com/en-us/research/event/workshop-on-speech-technologies-for-code-switching-2020/

[18] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[19] L. Tóth, G. Kovács, and D. Van Compernolle, "A perceptually inspired data augmentation method for noise robust cnn acoustic models," in *International Conference on Speech and Computer*. Springer, 2018, pp. 697–706.

[20] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[21] E. J. Bjerrum, M. Glahder, and T. Skov, "Data augmentation of spectral data for convolutional neural network (cnn) based deep chemometrics," *arXiv preprint arXiv:1710.01927*, 2017.

[22] A. Ragni, K. Knill, S. P. Rath, and M. Gales, "Data augmentation for low resource languages," 2014.

[23] N. Jaitly and G. E. Hinton, "Vocal tract length perturbation (vtlp) improves speech recognition," in *Proc. ICML Workshop on Deep Learning for Audio, Speech and Language*, vol. 117, 2013.

[24] A. Raju, S. Panchapagesan, X. Liu, A. Mandal, and N. Strom, "Data augmentation for robust keyword spotting under playback interference," *arXiv preprint arXiv:1808.00563*, 2018.

[25] S. Teki, M. Chait, S. Kumar, K. von Kriegstein, and T. D. Griffiths, "Brain bases for auditory stimulus-driven figure–ground segregation," *Journal of Neuroscience*, vol. 31, no. 1, pp. 164–171, 2011.

[26] S. Teki, M. Chait, S. Kumar, S. Shamma, and T. D. Griffiths, "Segregation of complex acoustic scenes based on temporal coherence," *Elife*, vol. 2, p. e00699, 2013.

[27] S. Teki, N. Barascud, S. Picard, C. Payne, T. D. Griffiths, and M. Chait, "Neural correlates of auditory figure-ground segregation based on temporal coherence," *Cerebral cortex*, vol. 26, no. 9, pp. 3669–3680, 2016.

[28] Y. Miao, M. Gowayyed, X. Na, T. Ko, F. Metze, and A. Waibel, "An empirical exploration of ctc acoustic models," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 2623–2627.

[29] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International conference on machine learning*, 2016, pp. 173–182.

3438

On detecting code mixing in speech using discrete latent representations

Sai Krishna Rallabandi and Alan W Black

Language Technologies Institute, Carnegie Mellon University

{srallaba,awb} @cs.cmu.edu

Abstract

Code Mixing - phenomenon where lexical items from one language are embedded in the utterance of another - is relatively frequent in multilingual communities and therefore speech systems should be able to process such content. In this paper[1], we investigate approaches towards building systems capable of detecting code mixing from a speech utterance. We propose employing a conditional variational encoder decoder that extracts discrete latent representations while being optimized to detect if the input utterance is code mixed. We explore two ways of engineering the prior distribution to capture the linguistic space of the utterance. Through objective evaluation via metrics Accuracy and Equal Error Rate, we show that our approach significantly outperforms the baselines in the context of three Indian languages - Gujarati, Tamil and Telugu.

Index Terms: Latent variable models, variational inference, code mixing, language identification

1. Introduction

Code Mixing is a conversational phenomenon where linguistic units such as phrases, words and morphemes of one language are embedded into the utterance of another language [1, 2]. It is quite common in multilingual societies such as in India, Singapore where English has transitioned from the status of a foreign language to that of a second language. Today such mixing has manifested itself in various types of content ranging all the way from news articles through comments/posts on social media, leading to co-existence of multiple languages in the same sentence. In the context of devices such as Alexa/Siri, interfaces deployed in code mixed contexts should to be able to process mixed speech without ignoring the content from one of the languages. Identifying whether or not the utterance is mixed, and the participating languages in case of a mixed utterance can benefit the downstream speech and natural language models.

In this paper, we present an approach aimed at identifying if a given speech utterance is code mixed or not. Humans have been shown to perform language detection exploiting two types of information from the speech utterance[3]: *content* information such as phonetic repertoire, phonotactics and *style* information such as rhythm and intonation. Infants have been shown to rely on content information to discriminate between languages even before having gained lexical knowledge[3]. Inspired by this, we propose an approach aimed at first hypothesizing the linguistic space of the given speech utterance and then using this information to detect if the utterance is characterized by code mixing. In our approach, the acoustic phonetic information of the speech utterance corresponds to the content and the para linguistic information corresponds to style. To isolate both these types of information, we employ a conditional variational encoder decoder with latent stochastic variables and investigate two ways of constraining the prior space:(1) using a Uniform discrete prior and (2) designing prior space inspired by articulatory dimensions.

A typical optimization challenge observed while training latent stochastic variable models is referred to as KL-collapse (or) posterior collapse [4], wherein the decoder network marginalizes the learnt latent representation. Approaches to dealing this issue involve annealing the KL divergence loss [4, 5], weakening the generator [6] and ensuring the recall using bag of words loss. In [7], authors propose a principled solution using vector quantization in the latent space, effectively making the loss due to KL Divergence a hyperparameter. In addition, they show that the resultant discrete latent representations may correspond to linguistic units. Our approach builds on this observation and we hypothesize that the learnt discrete units must be different for monolingual and code mixed utterances. Extending [7, 8], we add additional constraints in the prior space forcing the latent representations to follow articulatory dimensions: The encoded representation is hashed to a latent code based on a articulatory prior bank designed using a discrete codebook. This coded representation(*content*) is fed both to our decoder to detect mixing as well as to the generator for reconstructing the input. To model speaker information(*style*), we investigate two approaches: (a) Using a discriminatory speaker encoder similar to [9] and (b) Using a generative speaker encoder based on [10].

Our contributions from this work are as follows: (1) We propose an approach to identify if a speech utterance is mixed by employing discovered latent units. The authors are not aware of other works employing such hypotheses for the task of language (or) mixing identification (2) We investigate the effectiveness of designing the latent prior space based on articulatory constraints. This paper is organized as follows: In section 2, we present some background followed by earlier work on exploiting acoustic units. This is followed by an explanation of our proposed approach in section 3. We present our experiments in section 4 followed by an analysis of the proposed approach. This is followed by conclusion in section 6.

2. Background

The goal of acoustic unit discovery is to come up with a set of units that represent a speech utterance allowing robust resynthesis. The elements of such a set also might conform to certain desirable characteristics such as being consistent and compact, i.e. that different inputs should have discriminant acoustic units, but expected variance such as speaker or dialect should produce the same acoustic units. There have been numerous attempts to discover such acoustic units in an unsupervised fashion. In [11], authors presented an approach to modify the speaker diarization system to detect speaker-dependent acoustic units. [12] proposed a GMM-based approach to discover speaker-independent subword units. [13] designed a stacked AutoEncoder using backpropagation and then cluster the representations at the bot-

[1]Our implementation can be found here: https://github.com/blackbawx/SILA

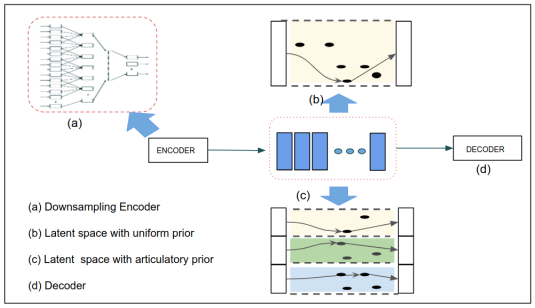

Figure 1: *Architecture of proposed approach. For brevity we have shown only three latent groups for the articulatory prior (b). In our model we have four latent groups with 2,3,4 and 7 classes respectively. The speaker embeddings are not shown here. (Best viewed in color)*

tleneck layer. To avoid quick transitions leading to repeated units, they employed a smoothing function based on transition probabilities of the individual states. [14] extended the structured VAE to incorporate the Hidden Markov Models as latent model. [7, 8] proposed VQ-VAE and argue that by vector quantization the posterior collapse problem could be circumvented. Our work is closely related to [7, 15] in that we follow a similar architecture. However, instead of reconstruction, our decoder performs a discriminative task of identifying if the input utterance is switched. We also engineer our prior space to reflect the articulatory constraints. Our view of decomposing the content and style information is in the same spirit as the approach in [16].

3. SILA: Switching Identification by Latent Articulation

Let us consider a speech corpus X consisting of mulitple languages where each language might comprise of multiple speakers. Let $x_1,...,x_n$ denote acoustic frames X. Note that x_i might be either monolingual or a code mixed utterance and let Y denote this information. Our model learns the joint distribution between $\{\mathbf{x}, \mathbf{y}\}$. We mediate this process using latent discrete random variables represented by Z. We employ two types of priors for Z to capture the *content* information: (1) Uniform discrete prior and (2) Prior space divided into multiple groups corresponding to articulatory dimensions. We hypothesize that this ensures the latent representations correspond to speech. To capture the *style* information, we employ pretrained speaker embeddings. Our model can be summarized by the following set of equations:

$$\begin{aligned} encoded_{1:T2} &= \mathbf{H}^{Encoder}(x_{1:T1}) \\ z_{1:T2} &= \mathbf{VQ}(encoded_{1:T2}) \\ y &= \mathbf{H}^{Decoder}(z_{1:T2}) \end{aligned} \tag{1}$$

At training time, parameters are learnt using Variational Inference. Specifically, we first draw latent code sequence(dropping the subscript for brevity) z from the current posterior represented by $\mathbf{H}^{Encoder}$. We then feed z into the decoder to optimize the likelihood and. During inference, we pass the speech utterance through encoder and obtain the latent units. The latent units are then fed to the decoder to obtain logits. The architecture of our proposed approach can be found in figure 1.

Table 1: *System Variants*

System	Spk Embedding Type	Spk Embedding Location
$\mathbf{SILA}_{uni}^{D+en}$	Discriminative	Encoder
$\mathbf{SILA}_{uni}^{D+de}$	Discriminative	Decoder
$\mathbf{SILA}_{arff}^{G+en}$	Generative	Encoder
$\mathbf{SILA}_{arff}^{G+de}$	Generative	Decoder

3.1. Model Components

3.1.1. Encoder

Our Encoder extracts temporal linguistic representation from the input to be fed to the variational layer. We refer to this representation as *content* and implement the encoder as a downsampler following the same architecture as in [7]. Additionally, we also add batch normalization to the output of encoder. We found that this improves the speed of training for some of the system variants we built.

3.1.2. Speaker Embeddings

In this work we employ pretrained speaker embeddings as the *style* information. For discriminative speaker embedding, we follow the procedure from [9] where the primary task was speaker identification[2]. For generative speaker embedding, we follow the procedure from [7] where the primary task was reconstruction of input. We extend the architecture to [10] and extract the 'top level' latent variable as the speaker embedding.

3.1.3. Decoder

Our decoder consists of an LSTM followed by an Attention block and a linear layer. The latent units z are first passed through decoder LSTM and then Attention block acts on the temporal sequence output by LSTM. We employ soft attention and implement it using dot product.

3.1.4. Latent Vector Quantization and Articulatory Priors

We investigate two separate prior space designs in our architecture. For the uniform discrete prior space, we follow the vector quantization procedure mentioned in [7]. For designing the articulatory prior space, we engineer our prior space to account for the phonetic information in the utterance by representing the prior as a discrete latent variable bank. Each discrete latent variable has a different set of states reflecting one of the articulatory dimensions. We divide the latent space into four groups, with 2, 4, 5 and 7 classes corresponding to the different articulatory dimensions as mentioned in Table 2.

3.2. Optimization and Model Interpretation

Our objective function is composed of four terms: (a) Classification loss characterized by cross entropy, (b) KL Divergence between the posterior output by encoder and the prior latent distribution, (c) Encoder penalty following [7] and (d) Vector Quantization penalty following [7]. Since we directly employ the pretrained embeddings for speaker we do not have the penalty terms corresponding to speaker embedding in the loss function. The KL Divergence being optimized in our model

[2]We have used the implementation of speaker encoder from https://github.com/CorentinJ/Real-Time-Voice-Cloning

Table 2: *Articulatory Features*

Feature name	Value	Details
vc	+ -	vowel or consonant
vlng	s l d a	vowel length
ctype	s f a n l r	consonant type
cplace	l a p b d v g	place of articulation

can be shown as follows. Consider the case of uniform discrete prior with K classes:

$$\begin{aligned}[q(z|x)||p(z)] &= \sum_{z \in Z} q(z|x) log[q(z|x)/p(z)] \\ &= q(k|x) log[q(k|x)/p(k)] \\ &= 1.log[1/(1/k)] \\ &= log(k)\end{aligned} \tag{2}$$

In the case of our model employing articulatory priors, the KL Divergence collapses to the sum of logarithm of the number of classes in each articulatory group. This can be shown as follows:

$$\begin{aligned}[q(z|x)||p(z)] &= \sum_{z \in Z} q(z|x) log[q(z|x)/p(z)] \\ &= \sum_{k \in K} q(k|x) log[q(k|x)/p(k)] \\ &= \sum_{k \in K} log(k)\end{aligned} \tag{3}$$

The advantage of formulating our model in this fashion is the presence of random variables that might capture the articulatory causal factors of variation in the speech utterance. Techniques aimed at this [17] have shown that it is possible to effectively disentangle the factors of variation using stochastic variables. Hence, we postulate that augmenting our model with appropriate prior distribution helps 'discover' latent linguistic units.

3.3. System Variants

We have built multiple variants of the proposed architecture. We denote the systems employing uniform prior and articulatory priors as **SILA**$_{uni}$ and **SILA**$_{arff}$ respectively. Each of these systems has two additional variants based on (a) the type of speaker embedding employed and (b) the position where speaker embedding is incorporated. A description of all the variants built can be seen in the table 1. The motivation for using the speaker embedding at the encoder is to capture the speaker specific mixing. On the other hand, providing the speaker embedding at the decoder allows the latent representation to capture linguistic units which are speaker independent and hence more representative of the speech utterance.

3.4. Hyperparameters

The architecture of our model is a modified version of the architectue proposed in [7]. As our content encoder, we use dilated convolution stack of layers which downsample the input audio by 64. The speech signal was power normalized and squashed to the range (-1,1) extracting the mel spectrum. We feed this spectrum to to the content encoder. For both discriminative and generative speaker embedding types, we obtain 128 dimensional utterance encoding. The quantizer acts as a bottleneck and performs vector quantization. Quantization is implemented using minimum Euclidean distance in the embedding space. The number of latent classes was chosen to be 512 and length of each class was 256 in the case of uniform prior. For the articulatory prior, the length of latent vectors was 64. Assuming $z_e(x)$ denotes the encoder output in the latent space, then the input of decoder $z_d(x)$ will be obtained by argmin of $d(e_j, z_e(x))$, where d is a similarity function of two vectors. The weight for encoder penalty was annealed from 0.001 to 0.01 over 2K timesteps and was maintained 0.01 there after. The decoder is a simple bidirectional LSTM with attention. It takes the output from the quantizer and is trained using cross entropy to predict if the utterance is mixed or not.

Table 3: *Accuracy for various systems in all the three languages: Gujarati, Tamil and Telugu.*

System	Gujarati	Tamil	Telugu
SILA$_{uni}^{D+en}$	77.7	72.4	76.8
SILA$_{uni}^{D+de}$	77.1	70.2	**77.0**
SILA$_{arff}^{G+en}$	**78.2**	**73.1**	76.9
SILA$_{arff}^{G+de}$	76.6	72.4	76.1

4. Experimental Setup

4.1. Data

We have used data released as part of shared task on detection of code mixing from speech [18]. Data included speech utterances for three Indian languages Gujarati, Tamil and Telugu. The labels for each utterance indicated if the utterance exhibits code mixing. No transcriptions or speaker labels have been provided as part of the official data release. The release included train-dev-test splits and therefore we have followed the same splits.

4.2. Baselines

An official baseline was released along with data [18]. Official baseline employed acoustic model similar to DeepSpeech and used CTC loss to optimize the network. We have built the following systems as baselines:

4.2.1. CBHG Baselines

This class of baselines were built using CBHG(1-D convolution bank + highway network + bidirectional GRU) from Tacotron[19] as the acoustic model and a linear layer as the decoder. This baseline was trained using cepstral representation. We have experimented with 80 dimensional Mel features used typically in speech synthesis as well as 39 dimensional MFCCs used in speech recognition.

4.2.2. Downsampling Baselines

This class of baselines used downsampling encoder from [7] as the acoustic model and a linear layer as the decoder. We have experimented with different rates of downsampling from 2 through 64. We have also experimented with different input representations of audio: cepstral and raw audio. For systems using raw audio, we have also performed μ law quantization using 256 levels.

4.2.3. Experts Baselines

We built Mixture of Experts baselines extending the approach proposed in [20]. Specifically, we train a number of expert models with soft parameter sharing implemented by gating mechanism. We hypothesized that each expert could capture information relevant to one language, or the individual linguistic units. We experimented with 2 through 8 experts where each expert was hypothesized to track the characteristics of a language and/or mixture of languages.

Table 4: *Accuracy and Equal Error Rates on Dev Set for Various Systems in all the three languages: Gujarati, Tamil and Telugu. BL - Baseline*

System	Gujarati	Tamil	Telugu
Official BL	76.8 / 11.6	71.2 / 14.4	74.0 / 13.0
CBHG BL	75.7 / 11.9	70.4 / 14.9	68.8 / 14.3
Downsampling BL	76.2 / 11.7	70.2 / 15.0	67.4 / 14.5
Experts BL	77.4 / 11.2	71.7 / 14.3	74.8 / 13.0
SILA	**78.2 / 10.8**	**73.1 / 13.2**	**77.1 / 12.8**

5. Results and Discussion

Evaluation was performed in the form of accuracy and Equal Error Rate(EER) following [18]. Since we did not have access to the ground truth labels for test set, we have compared the systems in terms of their performance on the development set. These results can be seen in the tables 3 and 4.

5.1. Discussion on system variants of SILA

The results evaluating different variants of SILA can be seen in table 3. The variant $\textbf{SILA}_{arff}^{G+en}$ achieves the best metrics in Gujarati, Tamil and is the second best in Telugu. From the scalar accuracy values the variants do not seem to be statistically different from each other. However, we noticed that the variants employing generative embeddings took longer to converge compared to the variants employing discriminative variants. Similarly we did not observe significant performance differences between the variants employing embeddings at the encoder and the decoder. We have also built a variant that uses a residual connection for speaker embeddings from encoder to the decoder. This system therefore can be seen as a combination of the systems $\textbf{SILA}^{en}$ and $\textbf{SILA}^{de}$(after removing the other terms from the system notation for brevity). We observed that this system obtains slightly degraded performance compared to the other variants in terms of accuracy. In addition, the latent units in this system were degenerate - they collapsed to a single latent unit. Since we are interested in discovering the latent units and exploiting them in our future works, we have ignored this system variant. Figure 2 depicts the entropy of individual articulatory groups during training. It can be seen that the entropy of some groups increases during the training. We attribute this to the model allocating additional capacity in the latent space and 'discovering' additional latent units. The behaviour seemed consistent across all the system variants that employed articulatory priors.

5.2. SILA vs Baselines

For each of the systems, we have only tabulated the best performing variant. For CBHG Baselines, Kaldi features with deltas and double deltas have significantly better performance

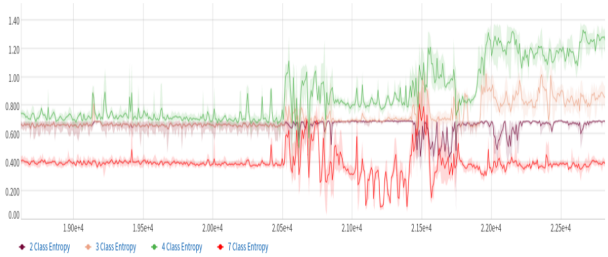

Figure 2: *Entropy of the individual latent articulatory groups. X axis denotes the number of gradient updates during training. Y axis denotes the scalar value of entropy. (Best viewed in color)*

compared to the other variants. We hypothesize that this might be due to the presence of information about deltas which is missing from Mel features. Similar observations have been made from the Downsampling Baselines. Best performance from Experts class of Baselines was obtained with 2 experts. This was contrary to our original hypothesis that having more number of experts might provide the model flexibility to encode more specific information. We plan to investigate this further as future work. Our proposed approach using discrete latent variables has shown to be significantly better than all of the baselines. This observation is in line with our hypothesis that the discrete latent units might be appropriate for detecting mixing in a speech utterance.

5.3. Discussion

To further validate our hypothesis about discrete units being different for different languages, we have also performed informal evaluation of the individual units being obtained given an utterance. We used data from Telugu for this as the primary author was proficient in that language. Manually inspecting the individual latent units did not yield units specific to monolingual or mixed lingual scenarios. We also found that the discrete units are not completely speaker and background independent. For instance, some of the utterances in the data were characterized by different room acoustics compared to the rest. While we would expect our model to normalize this effect and result in room acoustics independent discrete units, we have observed that this was not the case - rather some of the discrete units appear to have captured information about room acoustics. The authors acknowledge that that latent units do not correspond to exact articulatory features. In fact, when the model is trained without using reconstruction as an additional task, we observe that the latent units function as language embeddings rather than articulatory units. We intend to further investigate this behaviour in our future work.

6. Conclusion

In this work, we investigate approaches towards building systems capable of detecting code mixing from a speech utterance. We build a conditional variational encoder decoder where the latent space is characterized by discrete stochastic variables. We investigate two ways of engineering the prior space to incorporate articulatory constrains. We show that our approach outperforms the baseline methods.

7. References

[1] P. Muysken, *Bilingual speech: A typology of code-mixing*. Cambridge University Press, 2000, vol. 11.

[2] S. Gella, K. Bali, and M. Choudhury, "ye word kis lang ka hai bhai? testing the limits of word level language identification," 2014.

[3] J. Zhao, H. Shu, L. Zhang, X. Wang, Q. Gong, and P. Li, "Cortical competition during language discrimination," *NeuroImage*, vol. 43, no. 3, pp. 624–633, 2008.

[4] S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Jozefowicz, and S. Bengio, "Generating sentences from a continuous space," *arXiv preprint arXiv:1511.06349*, 2015.

[5] C. Zhou and G. Neubig, "Multi-space variational encoder-decoders for semi-supervised labeled sequence transduction," *arXiv preprint arXiv:1704.01691*, 2017.

[6] T. Zhao, R. Zhao, and M. Eskenazi, "Learning discourse-level diversity for neural dialog models using conditional variational autoencoders," *arXiv preprint arXiv:1703.10960*, 2017.

[7] A. van den Oord, O. Vinyals *et al.*, "Neural discrete representation learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 6306–6315.

[8] J. Chorowski, R. J. Weiss, S. Bengio, and A. v. d. Oord, "Unsupervised speech representation learning using wavenet autoencoders," *arXiv preprint arXiv:1901.08810*, 2019.

[9] Y. Jia, Y. Zhang, R. Weiss, Q. Wang, J. Shen, F. Ren, P. Nguyen, R. Pang, I. L. Moreno, Y. Wu *et al.*, "Transfer learning from speaker verification to multispeaker text-to-speech synthesis," in *Advances in neural information processing systems*, 2018, pp. 4480–4490.

[10] A. Razavi, A. van den Oord, and O. Vinyals, "Generating diverse high-fidelity images with vq-vae-2," in *Advances in Neural Information Processing Systems*, 2019, pp. 14 866–14 876.

[11] M. Huijbregts, M. McLaren, and D. Van Leeuwen, "Unsupervised acoustic sub-word unit detection for query-by-example spoken term detection," in *2011 IEEE international conference on Acoustics, speech and signal processing (ICASSP)*. IEEE, 2011, pp. 4436–4439.

[12] A. Jansen, S. Thomas, and H. Hermansky, "Weak top-down constraints for unsupervised acoustic model training," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 8091–8095.

[13] L. Badino, C. Canevari, L. Fadiga, and G. Metta, "An auto-encoder based approach to unsupervised learning of subword units," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 7634–7638.

[14] J. Ebbers, J. Heymann, L. Drude, T. Glarner, R. Haeb-Umbach, and B. Raj, "Hidden markov model variational autoencoder for acoustic unit discovery." in *INTERSPEECH*, 2017, pp. 488–492.

[15] W.-N. Hsu, Y. Zhang, and J. Glass, "Unsupervised learning of disentangled and interpretable representations from sequential data," in *Advances in neural information processing systems*, 2017, pp. 1878–1889.

[16] T.-Y. Hu, A. Shrivastava, O. Tuzel, and C. Dhir, "Unsupervised style and content separation by minimizing mutual information for speech synthesis," 2020. [Online]. Available: https://arxiv.org/pdf/2003.06227.pdf

[17] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework," 2016.

[18] M. R. India, "First workshop on speech technologies for code-switching in multilingual communities," 2020.

[19] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio *et al.*, "Tacotron: Towards end-to-end speech synthesis," *arXiv preprint arXiv:1703.10135*, 2017.

[20] Z. Zhao, L. Hong, L. Wei, J. Chen, A. Nath, S. Andrews, A. Kumthekar, M. Sathiamoorthy, X. Yi, and E. Chi, "Recommending what video to watch next: a multitask ranking system," in *Proceedings of the 13th ACM Conference on Recommender Systems*, 2019, pp. 43–51.

3942

Language Identification for Code-Mixed Indian Languages In The Wild

Parav Nagarsheth, Jehoshaph Akshay Chandran

AnnotateIt

parav@annotateit.com, akshay@annotateit.com

Abstract

In this paper, we identify challenges for language identification for code-mixed Indian languages with speech utterances captured in the wild. The study begins with building models for controlled datasets for Telugu, Tamil and Gujarati code mixed speech. This is part of the shared task of the Speech Technologies for Code-Switching in Multilingual Communities 2020 workshop. In the second half of the paper, we discuss an exercise to collect more diverse code-switched data for evaluating the language identification models. This data is derived from a wide range of YouTube videos in Gujarati and Telugu. The models get mixed results: the development set shows excellent performance, while the evaluations show degradation. The study identifies a need for more diverse, real-world code-mixed datasets for underserved languages.

Index Terms: multilingual speech recognition, code-mixed speech, language identification

1. Introduction

One of the challenges of building speech recognition models for modern Indian languages is the use of multiple languages in the same utterance. Unlike monolingual systems, multilingual or code-mixed systems need to be able to deal with utterances or words in different languages. Moreover, many publicly available datasets are imbalanced [1], with an over-representation of the primary language (or mother-tongue) of the speaker, or an over-reliance on written text to derive spoken utterances. This motivates our call for building higher quality datasets for code-mixed speech and NLP tasks.

To mitigate some of these challenges, some systems have explored conditioning the speech recognition models with a language vector [1, 2, 3, 4]. This method has shown to improve speech recognition performance for large multilingual speech recognition models [1]. In this paper, we study the task of language identification for code-mixed languages for use in downstream speech recognition tasks.

Our approach first builds monolingual speech models for the three languages (Telugu, Tamil, and Gujarati), which are subsequently used to train a binary classifier to predict between code-mixed and monolingual utterances. Our approach also takes into account the quantity and quality constraints of the available training data.

To build monolingual speech models, we use a modified Wav2Letter model [5] to train a simple speech recognition model. For the binary classification task, we use a linear SVM that we pass the character n-grams of the monolingual speech model.

In the second part of the paper, we present a framework for the collection of a large volume of high quality and well-annotated modern vernacular speech in realistic settings. We build a high-quality code-mixed dataset in two steps. Step one involves the collection of diverse content from YouTube consisting of interviews, educational videos, local film industry inspired by earlier work that used YouTube celebrity interviews to curate a vast dataset of English language speakers [14]. We use the SyncNet framework to extract audio clips from these videos. In the next step, to annotate this data, we build an annotation tool to annotate multilingual speech datasets. The tool uses several techniques to make annotating audio clips easy using tools inspired by the state of the art research in annotation of code-mixed speech.

Finally, we evaluate the performance of the language identification model on the shared task from the Code-Switching in Multilingual Communities 2020 Workshop and the YouTube dataset and analyze the underperformance of the model for the latter.

2. Related Work

Considerable effort has gone into building datasets and models for code-mixed and multilingual speech.

2.1. Speech Recognition and Language Identification Models

Anjuli Kannan et. al. [1] propose a low latency end-to-end system that works with real world data where they use a single multilingual model instead of separate models for each language. They propose sampling as one method to reduce the problem with imbalance in multilingual utterances. Additionally, they found the use of language vectors useful in improving the modified transliterated Word Error Rates (WER) [6]. In a review of the Shared Task Evaluation for Language Identification at VarDial, several top performing submissions were found to use linear discriminators for language identification [7]. Feature hashing has been found helpful to reduce the high-dimensionality of language identification datasets [8].

2.2. Data Annotation

Sanket Shah et. al. [9] describe an approach to building an annotation tool for collecting code switched data. Their annotation interface helps annotators transcribe code-switched speech faster and more accurately than a traditional input tool. In later sections, we use a similar interface for annotation, with some modifications to accommodate for input on mobile devices.

3. Shared Task

The Workshop on Speech Technologies for Code-switching in Multilingual Communities 2020 organized a shared task for Code-switched Spoken Language Identification (LID) for these language pairs:

1. **Gujarati-English** (GU-EN)
2. **Telugu-English** (TE-EN)
3. **Tamil-English** (TA-EN)

The shared task has two parts:

- **Part A**: Language identification at the utterance level
- **Part B**: Language identification at the frame level

For each of the sub-parts, datasets have speech data labeled for code-mixed and monolingual at the utterance and frame-level respectively.

The language identification models discussed in the following section solve the problem of utterance level language identification (Part A), which allows utterance level language vector conditioning for downstream speech recognition models.

3.1. Dataset Description

For Part A, the dataset for each language has been split into a training dataset and a development dataset. Each utterance is between 2 seconds to 24 seconds long, and the utterance is labeled as either monolingual or code-mixed speech.

The provenance of the data was not available from the organizers at the time of publication. However, after sampling a few audio files, it seems likely that the dataset is speech recorded by narrating target language text corpora. Due to constraints posed by recording in these conditions, the dataset may not cover certain real-world domains like spontaneous speech, colloquial speech and conversations.

Additionally, a much larger collection of monolingual data [1] was available for each target Indian language with utterance labels, that could be used to train the language identification model, with the condition that test data from this dataset should be excluded.

4. Language Identification Model

For language identification, we use a two-stage training and inference model. In the front-end, we train a monolingual speech recognition model using the monolingual speech corpus described in the previous section. In the second stage, we use the inference from the first stage to train a linear classifier for the binary language identification task: monolingual vs. code-mixed.

4.1. Monolingual Speech Recognition Model

Since CNNs are generally faster than LSTMs, we use a Wav2Letter model with CTC [10] loss to train a simple speech recognition model from the monolingual data.

The monolingual data has unicode sentence labels for Gujarati, Tamil and Telugu speech. Gujarati uses a variant of the Devanagri script, while Tamil and Telugu come from the family of Brahmic scripts. These languages have high grapheme-to-phoneme correspondence, hence grapheme based models should theoretically achieve similar performance to phoneme based models, even for relatively small amounts of training data. Moreover, there is very little phonetic data available for these languages.

End-to-end speech recognition models trained on character-based outputs like graphemes, byte-pair-encodings (BPEs), or word-pieces, jointly learn the acoustic model, pronunciation model and language model within a single neural-network [11, 12]. The latter two would be particularly useful during inference, since code-mixed speech with English words will be effectively treated as out-of-vocabulary (OOV) words.

[1] https://msropendata.com/datasets/7230b4b1-912d-400e-be58-f84e0512985e

We use 90-dimensional log-mel spectrum with a 20 millisecond window and a 10 millisecond hop length as frame level audio features. For the output, we tokenize the sentences using BPE with a vocabulary of size 1000.

The monolingual speech corpus is split into two parts: 80% of the corpus is used for fitting the model parameters using backpropagation and 20% of the corpus is used to keep track of the validation character error rate (CER). We also use audio augmentation (pitch shift and speed change) and spectral augmentation [13] to improve generalization performance.

To avoid overfitting, no additional hyperparameter search is performed. Additionally, to encourage better generalization performance, we use the same hyperparameters for training monolingual speech recognition models of all three languages.

The best performing speech models trained above are generally not suitable for practical speech recognition applications, however, the output is discernible enough to train downstream discriminators as described in the next section.

4.2. Language Identification Binary Task

From the tokenized output of the monolingual speech model, we extract character n-grams of length 1 to 5. For training a linear discriminator, these n-grams need to be categorized. However, because of the large space of possible n-grams, the dimensionality of the categorized features is very high.

Feature hashing provides an elegant way of reducing dimensionality without limiting the n-gram space. We use 1000 dimensional feature hashing to reduce the dimensionality of the Bag-of-Words (BoW) n-gram vectors for input to the classifier.

The features are used as input to a linear SVM that predicts between code-switched and monolingual utterances.

5. YouTube Data Collection and Annotation

The availability of high-quality resources for low-resource language is a significant constraint in building speech and natural language models. Each of India's top-10 native languages have more than 30 million speakers each. However, the amount of labeled speech data available is minuscule in comparison to other languages like English, Mandarin or Russian.

Since Indians are naturally multilingual, informal speech and text includes frequent code-mixing to a high-degree. On the other hand, formal text and speech, often used for training speech and language models, contains very little code-mixing.

With the proliferation of low-cost high-speed internet, there is a sizable amount of Indian language content available online on social media, video sharing platforms and streaming platforms. YouTube, in particular, hosts a large number of Indian content creators that cater diverse interests including celebrity interviews, news, educational videos, Bollywood film and music, fashion and beauty tips, agriculture, stand-up comedy, pranks etc.

Earlier work had used YouTube celebrity interviews to curate a vast dataset of English language speakers for speaker recognition applications [14]. Inspired from that work, we use a similar workflow to curate a small dataset of Gujarati and Telugu speech from YouTube videos for speech recognition.

5.1. Gujarati and Telugu Speech from YouTube Videos

We use the SyncNet framework [15] to extract speech from YouTube videos. The SyncNet model uses face tracking and lip tracking to sync speech with the speaker in a video frame. For the interested readers, we refer them to the original paper.

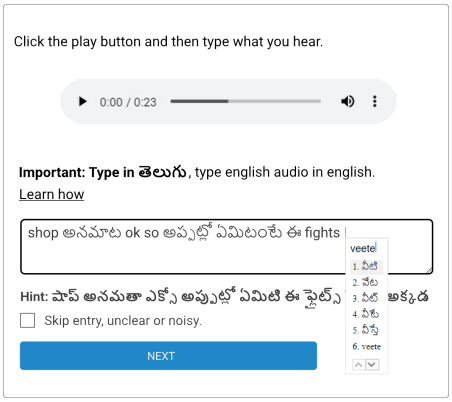

Figure 1: *Code-Mixed Speech Annotation Interface*

From the output of the model, audio clips are segmented into utterances of up to 30 seconds length to allow batch training of neural networks.

For the source Gujarati and Telugu videos, we choose topics that cover local celebrity interviews, official government bulletins, current affairs, food preparation videos and educational videos, where the speaker is visible in the frame most of the time. Videos with music are filtered out. The common theme among all of the videos is that they have a large amount of code-mixing with English.

5.2. Speech Annotation using Crowdsourcing

Crowdsourcing is commonly used to get speech annotations for low-resource languages. Crowdsourcing can be cheap, however quality can be a concern when annotators are untrained and focused on maximizing their throughput. For annotating the speech utterances derived from YouTube videos, we created a custom interface for our annotators, as shown in Figure 1. The annotators are able to playback the speech utterance on a desktop or mobile platform. Annotators can quickly switch between a standard English keyboard to type English words or a transliteration keyboard to type Gujarati/Telugu words. They are, optionally, given a transcription hint to increase annotation throughput.

For quality control we take two measures:

- For every annotation session, we benchmark annotator accuracy using a pool of expert-labeled speech, and reject utterances where the annotator accuracy falls below a calibrated quality threshold.
- For further reliability, we merge redundant annotations for the same utterance with a consensus algorithm similar to ROVER [16]. Each evaluation utterance has roughly 3-5 redundant annotations.

Table 1: *Number of utterances of code-mixed and monolingual speech in the YouTube dataset*

	Number of Utterances	
	Monolingual	**Code-Mixed**
Gujarati	766	517
Telugu	697	754

Table 2: *Results of Gujarati, Telugu and Tamil code-mixed identification (Part A of shared task evaluation)*

	Organizer Baseline		**Our Submission**			
	Dev		**Dev**		**Eval**	
	Acc.	**EER**	**Acc.**	**EER**	**Acc.**	**EER**
GU	76.8%	11.6%	81.2%	9.4%	66.1%	16.9%
TE	71.2%	14.4%	85.9%	7.1%	59.7%	20.1%
TA	74.0%	13.0%	85.6%	7.2%	61.8%	19.1%

To identify code-mixed speech, we sample the utterances for the presence of one or more words in the Latin script. This is not always accurate, for example in Gujarati, it is common to find English words suffixed with Gujarati morphemes to express the plural:

- Case becomes કેસો instead of cases
- Film becomes ફિલ્મો instead of films

Moreover, Indian languages use a lot of loaner words from English, which are interchangeably written in Latin or regional scripts. To minimize errors from these peculiarities, we reject utterances where there is annotator disagreement about specific words in the consensus building step.

In this exercise, we collected over 200 hours of annotated Gujarati and Telugu speech. However, for practical reasons, we chose a significantly smaller dataset for evaluation as shown in Table 1. The dataset is fairly balanced between code-mixed and monolingual utterances.

6. Results

6.1. Shared Task Evaluation

The results of the shared task evaluation are shown in Table 2. The language identification model outperforms the baseline on the development set for all three languages. This is encouraging, since the individual models do not have any language specific hyperparameter tuning.

There is a notable drop in accuracy and equal error rate (EER) for the testing set, as compared to the development set, yet the models perform comparably to the leaderboard available at the time of submission. The Gujarati language identification model, in particular, was ranked at third place in the leaderboard among five submissions hosted on the organizer's website. Due to a late submission of the evaluation, our results were not officially published on the leaderboard.

Labels for the test set were not available at the time of publishing for error analysis. However, we hypothesize that the development set for the code-mixed data and the training set for the monolingual corpus may have significant overlap for monolingual utterances. This may indicate that the model has not

Table 3: *Accuracy and EER for code-mixed language identification on YouTube dataset*

	YouTube	
	Accuracy	**EER**
Gujarati	42.1%	28.9%
Telugu	52.7%	23.6%

generalized to unseen monolingual speech data. Another hypothesis is that the provenance of the test dataset is significantly different from the training and development dataset.

6.2. Cross-Domain YouTube Dataset Evaluation

For the YouTube dataset, we generated ground truth labels from the annotated transcripts after data cleaning and quality control. The same Gujarati and Telugu models are used for evaluation as in the section 6.1, without any fine-tuning. Hence, this can classified as a cross-domain evaluation.

In Table 3, the performance of the language identification model trained on data from **Part A** of the Shared Task suffers a significant drop compared to the test results from the previous section. The model particularly underperforms for Gujarati, where we saw the best performance in the evaluation of the Shared Task among all three languages.

On further analysis, it was observed that monolingual utterances were frequently being erroneously classified as code-mixed. Since the binary classifier is presumably functioning as an anomaly detector, this may indicate that the front-end needs more in-domain monolingual data to improve generalization performance.

7. Conclusion and Future Work

In this submission we propose a two step approach to language identification for a code-switched utterance. We build a traditional monolingual speech recognition model using Wav2Letter. Next we use the n-grams of the monolingual model as input for a linear SVM to perform the binary classification task. From the data provided by the challenge, we show promising results on the dev set and peer-comparable results for the shared task evaluation. However, there is a notable degradation in accuracy and EER for the cross-domain YouTube dataset evaluation. The drop in accuracy could be ascribed to:

- Lack of large amounts of in-domain training data
- Lack of model generalization
- Noisy labels for monolingual and code-mixed speech

These results clearly demonstrate a pressing need for evaluating models on real-world and cross-domain datasets for speech recognition tasks in underserved languages. Datasets can be curated relatively inexpensively and scaled up quickly with the use of right annotation tools and quality control mechanisms.

This paper also recognizes the need for larger datasets for downstream applications in NLP for underserved languages. In the future, we anticipate participation in dataset annotation and collection exercises in text to speech, comprehension, intent detection and more for a wider range of Indian languages.

8. Acknowledgements

We are thankful to annotators who spent several hours annotating code-mixed Gujarati and Telugu speech for this project.

9. References

[1] A. Kannan, A. Datta, T. N. Sainath, E. Weinstein, B. Ramabhadran, Y. Wu, A. Bapna, Z. Chen, and S. Lee, "Large-scale multilingual speech recognition with a streaming end-to-end model," *arXiv preprint arXiv:1909.05330*, 2019.

[2] S. Toshniwal, T. N. Sainath, R. J. Weiss, B. Li, P. Moreno, E. Weinstein, and K. Rao, "Multilingual speech recognition with a single end-to-end model," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4904–4908.

[3] B. Li, T. N. Sainath, K. C. Sim, M. Bacchiani, E. Weinstein, P. Nguyen, Z. Chen, Y. Wu, and K. Rao, "Multi-dialect speech recognition with a single sequence-to-sequence model," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4749–4753.

[4] M. Grace, M. Bastani, and E. Weinstein, "Occams adaptation: A comparison of interpolation of bases adaptation methods for multi-dialect acoustic modeling with lstms," in *2018 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2018, pp. 174–181.

[5] R. Collobert, C. Puhrsch, and G. Synnaeve, "Wav2letter: an end-to-end convnet-based speech recognition system," *arXiv preprint arXiv:1609.03193*, 2016.

[6] J. Emond, B. Ramabhadran, B. Roark, P. Moreno, and M. Ma, "Transliteration based approaches to improve code-switched speech recognition performance," in *2018 IEEE Spoken Language Technology Workshop (SLT)*, 2018, pp. 448–455.

[7] M. Zampieri, S. Malmasi, Y. Scherrer, T. Samardzic, F. Tyers, M. Silfverberg, N. Klyueva, T.-L. Pan, C.-R. Huang, R. T. Ionescu *et al.*, "A report on the third vardial evaluation campaign," in *Proceedings of the Sixth Workshop on NLP for Similar Languages, Varieties and Dialects*, 2019, pp. 1–16.

[8] S. Malmasi and M. Dras, "Feature hashing for language and dialect identification," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2017, pp. 399–403.

[9] S. S. P. J. S. Santy and S. Sitaram, "Cossat: Code-switched speech annotation tool," *EMNLP 2019*, p. 48, 2019.

[10] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 369–376.

[11] K. Irie, R. Prabhavalkar, A. Kannan, A. Bruguier, D. Rybach, and P. Nguyen, "On the choice of modeling unit for sequence-to-sequence speech recognition," *Proc. Interspeech 2019*, pp. 3800–3804, 2019.

[12] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 4960–4964.

[13] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[14] A. Nagrani, J. S. Chung, W. Xie, and A. Zisserman, "Voxceleb: Large-scale speaker verification in the wild," *Computer Science and Language*, 2019.

[15] J. S. Chung and A. Zisserman, "Out of time: automated lip sync in the wild," in *Asian conference on computer vision*. Springer, 2016, pp. 251–263.

[16] J. G. Fiscus, "A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover)," in *1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings*. IEEE, 1997, pp. 347–354.

4347

Utterance-level Code-Switching Identification using Transformer Network

Krishna D N, Ankita Patil

HashCut Inc.

krishna@sizzle.gg, ankita@sizzle.gg

Abstract

A continuous change between two or more languages in a single utterance is known as code-switching. In this work, we develop a novel model to predict whether given audio is mono-lingual or contains code-switching. We propose to use a convolutional encoder in combination with transformer architecture for utterance-level code-switching detection. The convolution encoder processes the spectral features from audio with the help of 1D convolutional layers. The convolutional layer output features are processed by the Transformer network, consisting of a sequence of multi-head self-attention layers. The multi-head self-attention layers use attention mechanisms and help in selecting relevant information for better code-switching detection. Finally, our model aggregates frame-level features from the transformer network to create an utterance-level feature vector to predict the class label. We train and evaluate our model using the dataset provided by the Microsoft code-switching challenge. The training data contains three different languages, and each language contains both mono-lingual and code-switched utterances. Our model achieves significantly better accuracy compared to the baseline model. Our experiments show that, compared to the baseline, we obtain approximately 8%, 7%, and 3% improvement in accuracy for Telugu, Gujarati, and Tamil language, respectively.

Index Terms: code-switch detection,transformer, 1D-CNNs.

1. Introduction

Code-switching detection is about identifying whether an utterance contains multiple languages. Code-switching speech is defined as a speech that contains multiple languages within a single conversation [17]. Code-switching is known to occur in mostly low resource languages. In India, almost all of the spoken languages are multilingual in nature. Code-switching can sometimes cause problems to speech recognition systems, which are built using mono-lingual data. Many researchers in the speech community are trying to develop systems that can handle code-switching speech. Recently, robust acoustic-modeling techniques[8] are developed to handle code-switching. [5] proposes to map English phonemes to German phonemes to create a robust acoustic model for German speech recognition. [6] propose to use language-independent speech recognition and have shown to improve the system performance for Chinese speech recognition system. Recently, [7] built an end-to-end sequence-based neural network model for improving Chinese code-switching speech recognition accuracy by predicting language labels for every frame. Recently techniques [19,20] have shown to use language modeling in order to build robust speech recognition systems against code-switching. Language identification (LID) also seems to be an important technique to deal with code-switching [9,10,11,18]. Some of the interesting works[1,2,3,4] have shown different methods to detect the code-switching in single conversation/utterance. Multilingual training of DNN-based speech recognition systems has shown some improvements in the low resource languages[13,14,15,16].

Recent developments in the area of deep learning [27] have shown tremendous improvement in many areas of speech, including speech recognition [21], language identification [22], speaker identification [23], etc. Sequence to sequence [24] models have shown to be the best models for sequence prediction or sequence mapping problems like machine translation and speech recognition. Attention models [25] have become the dominating models in both NLP and speech problems due to their ability to focus on specific part of the signal to extract relevant information. Attention models have been the state of the art models for most of the sequence mapping problems like machine translation, speech recognition [21], text-to-speech, etc. Today state of the art speech recognition uses attention based sequence to sequence model due to its efficiency and capacity to align and predict. Transformers [26] is a special type of attention model that uses only attention technique without any other layers of convolution or RNNs, and they have been the best models for machine translation. Recently, attention models have also been extensively used in problems like emotion recognition [28], language identification and they are shown to be the best models for these problems.

Motivated by the work [26], this paper proposes to use transformer architecture in combination with a 1D convolutional neural network for code-switching identification. Our proposed model sequence of 257-dimensional spectral features as input and applies a sequence of 1D convolutional layers followed by a sequence of multi-head self-attention layers. We use the statistics pooling layer as an utterance-level aggregator to pool the frame-level features into an utterance-level feature vector. The utterance-level feature is used to predict if the input is mono-lingual or code-switched utterance. We conduct all our experiments on the Microsoft code-switching challenge dataset for three Indic languages Telugu, Tamil and Gujarati. Our experimental study shows that the transformer architecture can significantly improve the baseline for all three languages.

The organization of the paper is as follows. In section 2, we explain our proposed approach in detail. In section 3, we give a detailed analysis of the dataset, and in section 4, we explain our experimental setup in detail. Finally, in section 5, we describe our results.

2. Proposed approach

In this section, we explain our proposed approach in detail. The detailed model architecture is shown in Figure 1. Our model consists of 3 main stages, 1) An Convolutional Encoder layer, which consists of series of 1D convolutional layers, 2) Transformer block, consisting of multiple self-attention layers to select important and relevant features for code-switching detection using attention weighting and 3)Utterance-level aggregation layer, which contains statistics pooling layer to obtain ut-

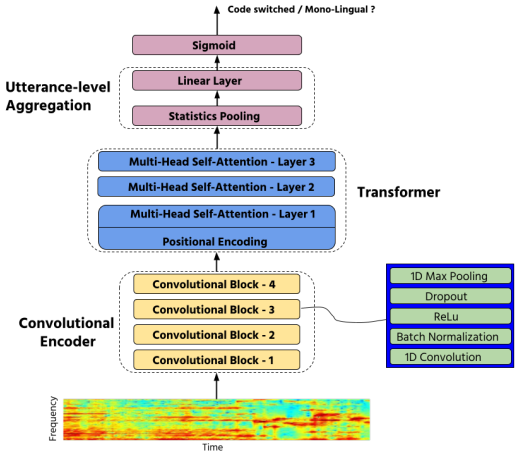

Figure 1: *Proposed model architecture.*

terance level feature vector for classification. The model takes a spectrogram feature matrix as input, and it is fed to the Convolutional Encoder as shown in Figure 1. The features from the convolutional encoder layer will go to the transformer block, which consists of a series of multi-head self-attention to extract relevant features from different parts of the input. The statistics pooling layer generates an utterance level feature vector by concatenating the mean and standard deviation vectors. The output of the statistics pooling layer gives us a single feature vector called the utterance level feature vector. This utterance level feature vector is fed into a projection layer followed by a Sigmoid layer to predict whether the input audio is monolingual or code-switched. We explain the details of each of these blocks in the following section.

2.1. Convolutional Encoder

Convolutional neural networks have been used a lot by computer vision community due to their nature of learning higher-level representation through the hierarchical structure. In speech, the audio waveform also contains both lower and higher-level abstractions. The higher-level abstractions like phonemes, syllables, or words are important in code-switching, whereas lower-level representations are may not of great importance. We use a convolutional neural network to learn these higher-level representations from lower-level spectral features. The Convolution encoder stage contains a series of 4 Convolution blocks, as shown in Figure 1. Each Convolutional block consits of 1D convolution operation, 1D Batch Normalization, Relu, Dropout and 1D max-pooling as depicted in Figure 1. Each of the convolution blocks operates at the same kernel size, but the number of filters varies between them. The convolution encoder takes a spectrogram feature matrix as input. The spectrogram contains a series of T spectral features with a feature dimension of 257. The input processed by the first convolution block *Convolution Block -1* consists of 64 1D convolution filters of kernel size 1x3, and the max-pooling operation is applied with a kernel size of 1x3 and stride 2. Similarly, the *Convolution Block -2*, *Convolution Block -3* and *Convolution Block -4* have 128,256,and 256 filters respectively. Each of the convolution blocks is operated with the same kernel size of 1x3. Also, each of the blocks is operated with a max-pooling kernel size of 1x3 and stride of 2. The convolution filter sizes and max-pooling filter sizes are chosen in such a way that the final output from the convolution encoder gives us a feature vector for every 200 ms. Let $\boldsymbol{X_n} = [\boldsymbol{x_1}, \boldsymbol{x_2}..\boldsymbol{x_n}, ...\boldsymbol{x_T}]$ be a spectrogram of an utterance consisting of T feature vectors. The input vector $\boldsymbol{x_n}$ is a spectral feature vector extracted from the raw audio.

$$\boldsymbol{H^C} = \text{Convolution-Encoder}(\boldsymbol{X_n}) \quad (1)$$

Where, Convolution-Encoder is a mapping function which consisting of series 1D Convolution blocks *Convolution Block -1*,*Convolution Block -2*, *Convolution Block -3*, and *Convolution Block -4*. After this operation, we obtain a feature sequence $\boldsymbol{H^C} = [\boldsymbol{h_1}, \boldsymbol{h_2}.....\boldsymbol{h^{T'}}]$ of length $\boldsymbol{T'}$ and $\boldsymbol{T'} << \boldsymbol{T}$. The Convolution-Encoder operation can be thought of as a feature learning neural network, and it also helps frame-rate reduction. The reason is as follows, typically the spectral features are extracted at a 10ms frame rate from the raw audio, but the code-switching can happen at a word or sub-word level, typically occurring for around 200ms. We design the network in such a way that we obtain a single feature vector approximately for every 200 ms so that the transformer model can look for variations in the features and learn to classify whether an utterance is mono-lingual or code-switched.

2.2. Transformer

Attention-based models have shown significant improvement in many speech problems these days. Specifically, the transformer [26] architecture is shown to the best neural network model for many sequence mapping problems like machine translation and speech recognition due to their ability to select important and relevant information during prediction. In this work, we propose to use a transformer for code-switching detection task. We expect the transformer model to learn to attend and select relevant features for better classification once we extract higher-level representations using convolutional layers. We assume that after training, the transformer will be able to focus and select the feature where code-switching occurs through an attention mechanism. Hence, we will be able to predict if the given utterance is monolingual or code-switched.

In this section, we describe the transformer block in detail. The transformer block contains three multi-head self-attention layers and an initial positional encoding layer [26], as shown in Figure 1. The positional encoding layer uses the sinusoidal positional embedding technique, as proposed in [26]. The multi-head self-attention layers in the transformer use scaled dot product attention mechanism to select the relevant features from the input feature sequence $\boldsymbol{H^C} = [\boldsymbol{h_1}, \boldsymbol{h_2}.....\boldsymbol{h^{T'}}]$ at multiple levels. The transformer helps attend to different parts of the input to detect if there is code-switching occurring in between monolingual words. The multi-head self-attention block is described, as shown in Figure 2. It consists of 3 different linear blocks, one for query $\boldsymbol{Q}$, one for key $\boldsymbol{K}$, and another for value $\boldsymbol{V}$. Each linear block consists of M independent linear layers, as shown in Figure 2, where M is the number of heads. The first multi-head attention block *Multi-Head Self-Attention -Layer 1* takes features $\boldsymbol{H^C} = [\boldsymbol{h_1}, \boldsymbol{h_2}.....\boldsymbol{h^{T'}}]$ from Convolution block after positional encoding operation and applies linear transformation to create $\boldsymbol{Q_i}$, $\boldsymbol{K_i}$ and $\boldsymbol{V_i}$ using $\boldsymbol{i}^{\text{th}}$ linear layers where, $\boldsymbol{i} = [\boldsymbol{1}, \boldsymbol{2}.....\boldsymbol{M}]$ and M is the total number of attention heads. The $\boldsymbol{Q_i}$, $\boldsymbol{K_i}$ and $\boldsymbol{V_i}$ are fed into scaled dot product attention layer. The scaled dot product attention $\boldsymbol{A_i}$ for $\boldsymbol{i}^{\text{th}}$ head is defined as follows.

$$\boldsymbol{A_i} = \text{Softmax}(\frac{\boldsymbol{Q_i K_i}}{\boldsymbol{d_q}})\boldsymbol{V_i} \quad (2)$$

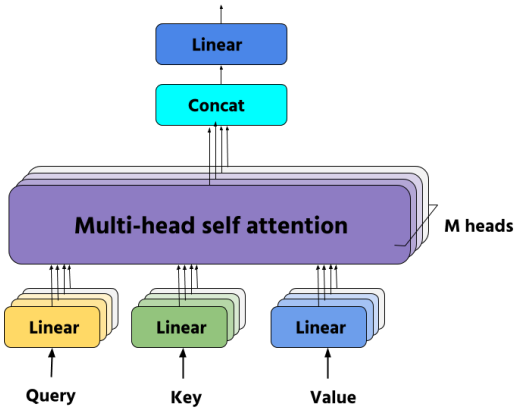

Figure 2: *multi-head self-attention*

Where d_q is the dimension of the query vector. We combine the attention output from all the heads using simple concatenation and feed into the feed-forward layer.

$$A = \text{Concat}(A_1, A_2, A_3 ... A_i ..... A_M)W_0 \quad (3)$$

Where, A_i is a d_qxT dimensional matrix. Since the Concat operation is applied to the feature dimension of all the matrices, the final output attention matrix A from multi-head attention block will have Md_qxT matrix dimensions. The output of the first layer A goes to the second layer *Multi-Head Self-Attention -Layer 2*, and this time the query, Key and Value vectors are generated from A. Similarly, the output from *Multi-Head Self-Attention -Layer 2* are fed *Multi-Head Self-Attention -Layer 3* to select more important features at a higher level. It can be seen that the output of the last layer of the transformer block will have the same temporal dimension as H^C because multi-head self-attention layers do not change the dimension. Overall, the transformer layer helps in finding regions that are more for detecting code-switch. The scaled dot product attention achieves this by giving more weighting to the features which are more relevant and less weighting to less relevant features. This process selects features from different parts of the input and helps in obtaining better classification performance due to the presence of multiple heads in the attention layer.

2.3. Statistics pooling

The idea of the statistics pooling layer is similar to max pooling. In the case of statistics pooling, we compute the mean and standard deviation from feature vectors generated by the transformer model. The mean and standard deviation features are concatenated in order to create the utterance level feature vector, as described in the equation below. Let $A = [a_1, a_2 ..... a_{T'}]$ is the output from the transformer block.

$$P = \text{Concat}(\text{mean}(A), \text{std}(A)) \quad (4)$$

Where, a_i is a feature vector of dimension $M * d_q$ and P is final pooled feature vector using statistics pooling layer. Since the dimension of the utterance level feature vector is P become bigger when M is large, we add a projection layer on top to the statistics pooling layer (Figure 1) in order to reduce the dimension of P. The project layer output will be passed to the Sigmoid layer in order to predict if the input utterance contains any code-switching.

Table 1: *Train and evaluation splits for different languages*

Datset	Train		Evaluation	
	Duration(Hrs)	Utterances	Duration(Hrs)	Utterances
Gujarati	31.59	16780	3.59	2091
Telugu	31.59	16991	4.0	2135
Tamil	30.24	10933	7.312	2642

3. Dataset

In this section, we briefly discuss the dataset. We obtain the dataset from Code-switching spoken language identification challenge[1]. The task of the challenge is to detect if an audio file is monolingual or code-switched. The shared task contains two subtasks, 1) Utterance-level identification of monolingual vs. code-switched utterances, and 2)Frame-level identification of language in a code-switched utterance. In this work, we chose to work on the first subtask. The dataset contains 3 Indic languages, Gujarati, Telugu, and Tamil. Each language contains audio files that are monolingual and code-switched along with their labels. The statistics of the dataset is given in Table 1. In India, most of the languages are code-switched with English languages. In this dataset also, each utterance can be completely monolingual, or it can be mixed with English. Each language pair contains training data and evaluation dataset. The dataset contains labels of every 200ms in the audio, but in our case, we convert all the frame-level labels utterance level labels. We train a separate model for each of the languages, and we report our results on the evaluation dataset. We also test our model on the blind dataset provided during the challenge.

4. Experiments

We conduct all our experiments on Microsoft's Code-switching spoken language identification challenge dataset. The dataset is described in the previous section in detail. We develop our model for an utterance-level code-switching identification task. Our model consists of a Convolutional encoder operating with 1D convolution kernels. The convolution encoder has 4 convolution block with [64,128,256,256] 1D convolution kernels of kernel size 1x3. Each layer also has a max-pooling layer operating with a kernel size of 1x3 with stride 2. The output of the Convolutional encoder goes to the transformer block, which consists of 3 multi-head self-attention blocks. Each self-attention block has eight attention heads. Finally, the statistics pooling layer computes the mean and standard deviation to pool frame-level features into utterance level feature. We use the sigmoid layer to predict the binary class label. In our case, we use 0-(class-1) corresponds to code-switched, and 1(class-1) corresponds to monolingual. The input to our model is a 2D magnitude spectrogram (we refer magnitude spectrogram as a spectrogram). We take the maximum length of the audio to be 25sec. We crop the audio if the duration of the audio is more than 25sec, and we pad zeros if the duration is less than 25sec. We compute a 257-dimensional spectral feature for every 25ms window with a frame-shift of 10ms. So, the spectrogram contains 2500

[1]https://www.microsoft.com/en-us/research/event/workshop-on-speech-technologies-for-code-switching-2020/

Table 2: *Acc (%) and EER(%) on Evaluation dataset: bold represents the best method*

Language	Baseline		Proposed	
	Acc (%)	EER(%)	Acc (%)	EER(%)
Te-En	71.2	14.4	**79.34**	**12.40**
Ta-En	74.0	13.0	**76.70**	**12.55**
Gu-En	76.8	11.6	**83.54**	**9.9**

frames, and each frame will be of 257 dimensions. We can think of the spectrogram as an image with a single channel. We also normalize the spectrogram by computing the mean and standard deviation. We use binary cross-entropy loss function to train our model. We use Adam [30] optimizer to train all our models with a learning rate of 0.0001 for up to 80 epochs. We use a batch size of 32 during training. We train all our models using Pytorch [29] toolkit. We compare our results with the baseline model, and we observe that our approach gives 8%, 7%, and 3% improvement in accuracy for Telugu, Gujarati, and Tamil language respectively over the baseline results. We use 3 RTX 2080Ti GPU cards for our experiments. We will open-source our codes and models soon.

5. Results

In this section, we describe the evaluation of different models and their performances. Our baseline model is a five layer LSTM model, each layer having a hidden dimension of 1024. The baseline model uses CTC loss function during training. The workshop organizers already provide the baseline model results for the evaluation dataset. We train our proposed model for three different languages Gujarati, Telugu, and Tamil. We compare both accuracy and equal error rate performance of our model with the baseline approach. The results are shown in Table 2. We represent the languages as Gu-En, Te-En, and Ta-En. Where Gu-En represents Gujarati code mixed with English, Te-En represents Telugu code mixed with English, and Ta-En represents Tamil code mixed with English. It can be seen that our approach obtains 8.74% absolute improvement in accuracy and 1.69% improvement in EER for Gu-En. Similarly, we obtain an 8.14% improvement in accuracy and 2.0% improvement in EER for Te-En. For Ta-En, we obtain 2.7% absolute improvement in accuracy and 0.5% improvement in EER.

We also test our models on the blind test data provided for the challenge, and the results are shown in Table 3. We can see that the model performance not as good as on the evaluation dataset, and this may be due to various reasons like speaker variability, perturbation like speed and volume, etc. We do not know the exact reason why there is a significant difference between the evaluation test set and the blind test set. We want to explore the reason and make improvements in the future.

6. Conclusions

In this work, we propose a new approach for utterance-level code-switching detection using transformer architecture. We use a convolutional encoder to extract higher-level abstractions at phoneme or syllable levels. We use a sequence of multi-head self-attention layers to find to select and learn relevant information to detect if the utterance is monolingual or contains code-switching. We also utterance-level aggregation module, which uses a statistics pooling approach to pool frame-level features into utterance-level features. Our experimental results show that

Table 3: *Accuracy and EER on blind test data*

Language	Proposed	
	Acc (%)	EER(%)
Te-En	69.23	15.38
Ta-En	69.08	15.45
Gu-En	48.46	25.76

a transformer-based model can outperform the LSTM baseline due to their ability to attend and select different parts of the input to detect code-switching. Our method consistently obtains huge improvement on Microsoft's Code-switching spoken language identification challenge dataset for Telugu, Tamil and Gujarati languages. We also see that the model suffers accuracy on the blind test data due to the speaker and other local variations. We would like to explore the reasons and make improvements in the future.

7. Acknowledgements

We want to thank Microsoft India for providing the dataset for this research work. We would also like to thank HashCut Inc. for supporting this work. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors(s) and do not necessarily reflect the views of HashCut Inc.

8. References

[1] E. Yılmaz, H. van den Heuvel and D. van Leeuwen, "Code-switching detection using multilingual DNNS," 2016 *IEEE Spoken Language Technology Workshop (SLT)*, San Diego, CA, 2016, pp. 610-616, doi: 10.1109/SLT.2016.7846326.

[2] C. Wu, H. Shen and C. Hsu, "Code-Switching Event Detection by Using a Latent Language Space Model and the Delta-Bayesian Information Criterion," in *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 11, pp. 1892-1903, Nov. 2015, doi: 10.1109/TASLP.2015.2456417.

[3] E. Yılmaz, H. Van den Heuvel, and D. A. Van Leeuwen, "Exploiting untranscribed broadcast data for improved code-switching detection," in *Proc. INTERSPEECH*, Aug. 2017, pp. 42–46.

[4] Emre Yılmaz, Henk van den Heuvel, and David van Leeuwen, "Code-switching detection with dataaugmented acoustic and language models," in the *Sixth International Workshop on Spoken Language Technology for Under-resourced Languages (SLTU)*. Procedia Computer Science, 2018, pp. 127–131.

[5] G. Stemmer, E. Noth, and H. Niemann, "Acoustic modeling of foreign words in a German speech recognition system," in *Proc.EUROSPEECH*, 2001, pp. 2745–2748

[6] D.-C. Lyu, R.-Y. Lyu, Y.-C. Chiang, and C.-N. Hsu, "Speech recognition on code-switching among the Chinese dialects", in *Proc. ICASSP*, vol. 1, May 2006, pp. 1105–1108

[7] N. Luo, D. Jiang, S. Zhao, C. Gong, W. Zou, and X. Li, "Towards end-to-end code-switching speech recognition," *arXiv preprint* arXiv:1810.13091, 2018.

[8] E. Yılmaz, H. Van den Heuvel, and D. A. Van Leeuwen, "Investigating bilingual deep neural networks for automatic speech recognition of code-switching Frisian speech," in *Proc. Workshop on Spoken Language Technology for Under-resourced Languages (SLTU)*, May 2016, pp. 159–166.

[9] J. Weiner, N. T. Vu, D. Telaar, F. Metze, T. Schultz, D.-C. Lyu, E.-S. Chng, and H. Li, "Integration of language identification into a recognition system for spoken conversations containing codeswitches," in *Proc. SLTU*, May 2012.

[10] K. R. Mabokela, M. J. Manamela, and M. Manaileng, "Modeling code-switching speech on under-resourced languages for language identification," in *Proc. SLTU*, 2014, pp. 225–230.

[11] D.-C. Lyu, E.-S. Chng, and H. Li, "Language diarization for codeswitch conversational speech," in *Proc. ICASSP*, May 2013, pp.7314–7318.

[12] T. I. Modipa, M. H. Davel, and F. De Wet, "Implications of Sepedi/English code switching for ASR systems," in *Pattern Recognition Association of South Africa*, 2015, pp. 112–117.

[13] S. Thomas, S. Ganapathy, and H. Hermansky, "Multilingual MLP features for low-resource LVCSR systems," in *Proc. ICASSP*, March 2012, pp. 4269–4272.

[14] Z. Tuske, J. Pinto, D. Willett, and R. Schluter, "Investigation on cross- and multilingual MLP features under matched and mismatched acoustical conditions," in *Proc. ICASSP*, May 2013, pp.7349–7353.

[15] N. T. Vu, F. Metze, and T. Schultz, "Multilingual bottle-neck features and its application for under-resourced languages," in *Proc.SLTU*, May 2012.

[16] K. Vesely, M. Karafiat, F. Grezl, M. Janda, and E. Egorova, "The language-independent bottleneck features," in *Proc. SLT*, Dec 2012, pp. 336–341.

[17] Peter Auer, "Code-switching in conversation: Language, interaction and identity", Routledge, 2013.

[18] K Bhuvanagiri and Sunil Kopparapu, "An approach to mixed language automatic speech recognition," *Oriental COCOSDA*, Kathmandu, Nepal, 2010.

[19] H. Adel, N. Vu, F. Kraus, T. Schlippe, H. Li, and T. Schultz, "Recurrent neural network language modeling for code switching conversational speech," in *Proc. ICASSP*, 2013, pp. 8411–8415.

[20] Heike Adel, Katrin Kirchhoff, Dominic Telaar, Ngoc Thang Vu, Tim Schlippe, and Tanja Schultz, "Features for factored language models for code-switching speech," in *Spoken Language Technologies for Under-Resourced Languages*, 2014.

[21] W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals, "Listen, Attend and Spell: A Neural Network for Large Vocabulary Conversational Speech Recognition" in *ICASSP*, 2016

[22] E. Variani, X. Lei, E. McDermott, I. Lopez-Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Florence, Italy, May 2014.

[23] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur,"X-vectors: Robust dnn embeddings for speaker recognition," in 2018 *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE, 2018

[24] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[25] D. Bahdanau, K. Cho, and Y. Bengio. "Neural machine translation by jointly learning to align and translate", *arXiv preprint arXiv:1409.0473*, 2014

[26] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp.5998–6008.

[27] Schmidhuber, J. (2015). "Deep learning in neural networks: An overview". *Neural Networks*, 61 ,85–117.

[28] M. Sarma, P. Ghahremani, D. Povey, N. K. Goel, K. K. Sarma, and N. Dehak, "Emotion identication from raw speech signals using dnns", *Proc. Interspeech* 2018, pp. 3097–3101, 2018.

[29] Paszke, Adam and Gross, Sam and Chintala, Soumith and Chanan, Gregory and Yang, Edward and DeVito, Zachary and Lin, Zeming and Desmaison, Alban and Antiga, Luca and Lerer, Adam "Automatic differentiation in PyTorch" in *NIPS*, 2017

[30] K. Miyazaki, T. Komatsu, T. Hayashi, S. Watanabe, T. Toda and K. Takeda, "Weakly-Supervised Sound Event Detection with Self-Attention," *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain*, 2020, pp. 66-70, doi: 10.1109/ICASSP40776.2020.9053609.

[31] B. Kim and S. Ghaffarzadegan, "Self-supervised Attention Model for Weakly Labeled Audio Event Classification," *27th European Signal Processing Conference (EUSIPCO), A Coruna, Spain*, 2019, pp. 1-5, doi: 10.23919/EUSIPCO.2019.8902567

4852

Learning not to Discriminate: Task Agnostic Learning for Improving Monolingual and Code-switched Speech Recognition

Gurunath Reddy M[+], *Sanket Shah*[+], *Basil Abraham*[*],
Vikas Joshi[*], *Sunayana Sitaram*[+]

[+]Microsoft Research India,
[*]Microsoft Corporation
{t-gumadh, t-sansha, basil.abraham, vikas.joshi, sunayana.sitaram}@microsoft.com

Abstract

Recognizing code-switched speech is challenging for Automatic Speech Recognition (ASR) for a variety of reasons, including the lack of code-switched training data. Recently, we showed that monolingual ASR systems fine-tuned on code-switched data deteriorate in performance on monolingual speech recognition, which is not desirable as ASR systems deployed in multilingual scenarios should recognize both monolingual and code-switched speech with high accuracy. Our experiments indicated that this loss in performance could be mitigated by using certain strategies for fine-tuning and regularization, leading to improvements in both monolingual and code-switched ASR. In this work, we present further improvements over our previous work by using domain adversarial learning to train task agnostic models. We evaluate the classification accuracy of an adversarial discriminator and show that it can learn shared layer parameters that are task agnostic. We train end-to-end ASR systems starting with a pooled model that uses monolingual and code-switched data along with the adversarial discriminator. Our proposed technique leads to reductions in Word Error Rates (WER) in monolingual and code-switched test sets across three language pairs.

Index Terms: speech recognition, code-switching, adversarial learning, transfer learning

1. Introduction

Recognizing code-switched speech is challenging for Automatic Speech Recognition (ASR) systems due to the lack of large amounts of labeled code-switched speech and text data for training Acoustic and Language Models. Recently, we showed that even if there is sufficient code-switched speech data to train models, there is a loss in performance on monolingual test sets when monolingual models are trained or fine-tuned with code-switched data [1]. Since code-switched and monolingual speech co-occur, it is imperative that models perform well on code-switched speech while not deteriorating on monolingual speech.

With this goal in mind, in [1] we proposed strategies for learning how to recognize code-switched speech while not forgetting monolingual speech recognition in the following scenarios:

Case 1: If monolingual and code-switched data are both available and a model can be trained from scratch, regularization strategies and fine-tuning a pooled model that uses all data leads to best results across data sets.

Case 2: If only a monolingual model is available and a new model cannot be trained from scratch, the Learning Without Forgetting [2] framework can be used to improve performance on all test sets compared to a monolingual model fine-tuned on code-switched data [1].

In this work, we build upon our findings for Case 1, in which we have access to both monolingual and code-switched data and can train a model from scratch. When we train a joint model to learn both monolingual and code-switched speech recognition tasks with task specific and shared layer parameters, the model tends to drift towards one particular task. This drift is because shared layers try to learn task specific features which is not ideal for a joint model that needs to perform well on both tasks. Hence, we need to learn task invariant or agnostic shared layer parameters which lead to task agnostic features at shared layers and discriminant features at task specific layers.

In this work, we learn task agnostic shared layer parameters by adversarial discriminative learning. We show that it is possible to improve performance by using adversarial learning over our previously proposed techniques of fine-tuning and regularization on monolingual and code-switched test sets that span three language pairs - Tamil-English, Telugu-English and Gujarati-English. In this work, we assume that there exists a classifier that will classify code-switched and monolingual utterances prior to recognition by our model, however, our technique can also be used if this assumption does not hold.

The rest of the paper is organized as follows. Section 2 relates our work to prior work. Section 3 describes our experimental setup and results. Section 4 concludes.

2. Relation to Prior Work

In this paper, we learn task agnostic shared layer parameters by adversarial learning inspired by the domain-adversarial training of neural networks [3]. Originally, domain-adversarial learning was proposed to adapt models trained on labeled data to new unlabeled data by adversarial discrimination. Adversarial training has been adopted recently for many tasks: [3] and [4] use adversarial learning for domain adaptation for image classification. [5] utilized adversarial strategies for word boundary segmentation of the Chinese heterogeneous data. [6] and [7] utilized adversarial learning for environment and speaker adaptation for robust speech recognition. Recently, [8] explored adversarial learning for transferring knowledge from source language to target language for low-resource ASR models.

The following approaches have been explored for end-to-end code-switched speech recognition. A hybrid attention based architecture is described in [9] for Mandrian-English code-switched ASR. Multiple fine-tuning approaches have been studied to improve code-switched speech recognition in [10] and [11]. Multi-task learning strategies have also been proposed for improving code-switched speech recognition in [12] and [13]. Recently, we proposed approaches to learn code-

switched speech recognition without forgetting monolingual speech recognition [1] using various regularization and fine-tuning strategies, as well as the Learning Without Forgetting [2] framework.

3. Experimental Setup

3.1. Data

We use the same data and baselines as described in [1], which we mention in brief in this section. We carried out experiments for three languages - Tamil (TA), Telugu (TE) and Gujarati (GU) and their code-switched counterparts with English - Tamil-English, Telugu-English and Gujarati-English. Although all three languages were mixed with English, the type and extent of mixing was different. We used two types of speech data for training - conversational data as well as phrasal data, which is similar to read speech, while for testing, only phrasal data was used. We test our models on monolingual and code-switched data sets separately to ensure that models perform well on both. Hence, we have six test sets that we evaluate our models on. Table 1 describes the dataset size in hours.

Table 1: *Training and test data statistics*

	Train + Dev (MONO)	***Train + Dev (CS)***	***Test (MONO)***	***Test (CS)***
TA	212 hrs	177 hrs (CMI: 22.08)	24 hrs	19 hrs (CMI: 17.07)
TE	170 hrs	243 hrs (CMI: 23.85)	19 hrs	28 hrs (CMI: 21.62)
GU	241 hrs	186 hrs (CMI: 18.91)	26 hrs	18 hrs (CMI: 16.32)

The Code Mixing Index (CMI) [14] measures the amount of code-switching in a corpus by using word frequencies. We measure the CMI of our code-switched train and test sets and report them in parentheses in Table 1. The CMI of Telugu-English is the highest, while Gujarati-English is the lowest suggesting that Telugu-English is the most code-switched while Gujarati-English is the least code-switched among the languages under consideration.

Table 2: *Baseline Word Error Rates (WER)*

Test Set	***Exp1***	***Exp2***	***Exp3***
TA-MONO	45.81	66.11	**44.42**
TA-CS	62.73	58.63	**50.92**
TE-MONO	41.40	52.00	**39.67**
TE-CS	52.70	37.21	**33.70**
GU-MONO	39.20	51.70	**37.82**
GU-CS	46.50	42.50	**40.81**

3.2. Baseline experiments

We denote our training monolingual datasets (X_1^M, Y_1^M),...,(X_n^M, Y_n^M) where $M \in$ {TE/TA/GU}, code-switched datasets (X_1^{CS}, Y_1^{CS}),...,(X_n^{CS}, Y_n^{CS}) where $CS \in$ {TE-EN/TA-EN/GU-EN}. The labels Y are graphemes and the character set includes the union of English and the respective language's characters. Further, we denote $T = \{M, CS\}$ where T is the monolingual or code-switched speech recognition task.

Our baseline model consists of two Convolution Neural Network (CNN) layers followed by five bidirectional long-short term (BLSTM) layers of 1024 dimension. These parameters are shared between the monolingual and code-switched task and are denoted by θ_s. Further, the frame-wise posterior distribution is conditioned on the input frame X_i^T, is calculated by a forward pass through the shared layers, θ_s and through a fully-connected layer, θ_T followed by softmax computation over labels as shown in Fig. 1(a). We maximize the conditional posterior distribution by minimizing the Connectionist Temporal Classification (CTC) [15] criterion represented by $L_T(\theta_s, \theta_T)$. The model parameters are trained using stochastic gradient descent (SGD) optimizer. The learning rate (λ) is initialized with 3e-4. The model is trained for 40 epochs, with mini-batch size equal to 64 per GPU. The model parameters are updated using the back propagation algorithm.

We evaluated our proposed approach against three baselines, which we refer to as Exp1, Exp2 and Exp3.

- **Exp1**: Monolingual-only baseline, consisting of models trained only on monolingual data
- **Exp2**: Code-switched-only baseline, consisting of models trained only on code-switched data
- **Exp3**: Pooled model, consisting of models trained using all the data from Exp1 and Exp2

An n-gram Language Model (LM), trained using transcriptions from the training data is used during decoding. Table 2 shows Word Error Rates (WER) of all three baselines on both monolingual and code-switched test sets. Exp3, which is the pooled model consisting of monolingual and code-switched data performs best on all test sets. Exp1 performs better on monolingual test sets than Exp2, and the reverse is true for code-switched test sets, as expected.

3.3. Adversarial task agnostic pooled model

From the baseline experiments we observe that the pooled model performs better than the monolingual or code-switched model for all test sets. Even though the pooled model performs significantly better than code-switched only baseline, the improvements are only marginally better on monolingual test sets compared to the monolingual only baseline.

We hypothesize that this is because shared layer parameters learn unwanted task specific features which drifts the performance of the monolingual recognition task towards the code-switched task. In order to alleviate this performance drift, we propose learning task invariant shared layer parameters in the pooled model by adversarial training as shown in Fig. 1(b). The adversarial pooled model consists of task independent shared (θ_s) layers, task dependent (θ_T) layers and adversarial task discriminator consisting of a fully connected (FC) layer, gradient reversal layer (GRL) and sigmoid activation. The parameters of the adversarial discriminator are denoted by θ_a.

The gradient reversal layer of the adversarial task discriminator ensures that features from the shared layers are as in-discriminant as possible for the given task so that the shared layers learn a generalized representation. The GRL contains no trainable parameters and acts as a identity transformer during forward pass. However, during back-propagation the GRL reverses the gradients of the previous layers i.e., multiplies the gradients by -1 and passes it to the next layers which helps in making the shared layer features in-discriminant to specific task. For each utterance u, the adversarial task discriminator

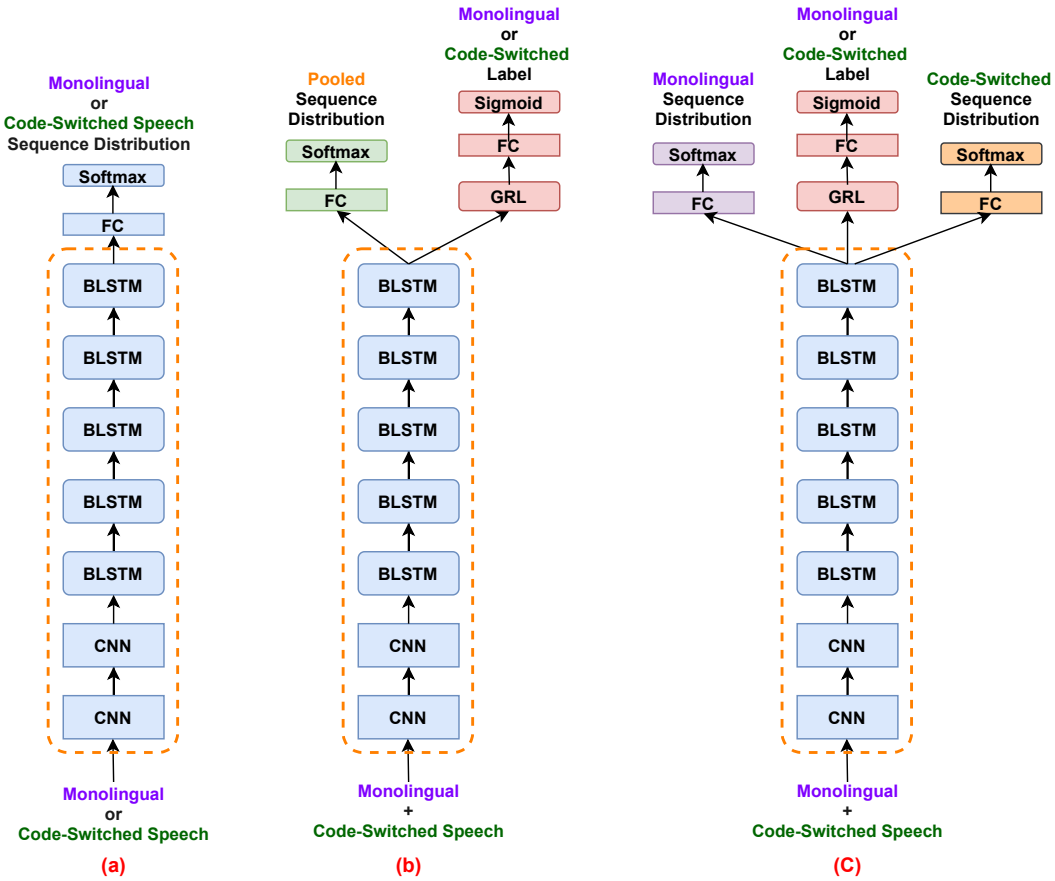

Figure 1: *Illustration of the proposed adversarial learning of task agnostic shared layer parameters for monolingual and code-switched speech recognition. (a) Baseline model trained with either monolingual or code-switched speech utterances. (b) Pooled model trained with both monolingual and code-switched speech utterances along with adversarial task (monolingual or code-switched) discriminator to learn task agnostic shared layer parameters. (c) Multi-task adversarial discriminator trained to recognize monolingual and code-switched speech recognition independently. CNN and BLSTM are the shared layers shown inside the dotted rectangular box. Set of FC and Softmax are the task specific layers. GRL, FC and Sigmoid form the adversarial task discriminator layers.*

is trained to discriminate speech utterances into either monolingual or code-switched ($T = \{M, CS\}$).

$$L_A(\theta_s, \theta_a) = -\sum_{u=1}^{N} \log P(T_u | X_u^T; \theta_s, \theta_a) \quad (1)$$

where X_u^T and T_u represents u_{th} input utterance and corresponding label and N represents the total numbers of utterances in the dataset. Even though the discriminator is trained to minimize the classification loss, the gradients of the discriminator is negative so that the shared layers are trained to be task independent. The parameters of the adversarial task discriminator are updated as

$$\theta_s \longleftarrow \theta_s + \lambda \frac{\partial L_A}{\partial \theta_s} \quad (2)$$

$$\theta_a \longleftarrow \theta_a - \lambda \frac{\partial L_A}{\partial \theta_a} \quad (3)$$

In our previous work [1], we found that speech recognition performance can be improved by initializing the parameters of the model from a pretrained model. Hence, shared layer parameters of the adversarial task agnostic pooled model is initialized from the baseline pooled model (Exp3) and trained with the loss function

$$L_{AP}(\theta_s, \theta_T, \theta_a) = L_T(\theta_s, \theta_T) + L_A(\theta_s, \theta_a) \quad (4)$$

The parameters of the model are updated as

$$\theta_s \longleftarrow \theta_s - \lambda \left(\frac{\partial L_T}{\partial \theta_s} - \frac{\partial L_A}{\partial \theta_s} \right) \quad (5)$$

$$\theta_T \longleftarrow \theta_T - \lambda \frac{\partial L_T}{\partial \theta_T} \quad (6)$$

$$\theta_a \longleftarrow \theta_a - \lambda \frac{\partial L_A}{\partial \theta_a} \quad (7)$$

The performance of the adversarial task agnostic pooled model (Exp5) compared with the pooled model (Exp3) is shown in Table 3. We can see that Exp5 performs better than Exp3 for all test sets. This indicates that the model benefits from adversarial task-discriminative training for improving both monolingual and code-switched speech recognition.

3.4. Multi-task adversarial speech recognition model

In our previous work [1], we observed that a multi-task model trained with separate monolingual and code-switched task specific layers yields better performance than the pooled model.

Hence, we trained a joint monolingual and code-switched multi-task model as shown in Fig. 1(c). The multi-task adversarial speech recognition model consists of shared layers (θ_s), task specific monolingual (θ_m) and code-switched (θ_c) layers, and the adversarial task discriminator (θ_a) as shown in Fig 1(c). The shared layer parameters of the multi-task model are initialized from the pooled model as before and trained jointly end-to-end along with the adversarial task discriminator.

$$L_{MA}(\theta_s, \theta_m, \theta_c, \theta_a) = L_M(\theta_s, \theta_m) + L_{CS}(\theta_s, \theta_c) + L_A(\theta_s, \theta_a) \quad (8)$$

where $L_M(\theta_s, \theta_m)$, and $L_{CS}(\theta_s, \theta_c)$ are the individual monolingual and code-switched loss functions. Similar to Exp5, the utterance level adversarial loss $L_A(\theta_s, \theta_a)$ makes the shared layer features as in-discriminant as possible to monolingual and code-switched speech utterances while learning discriminant features at task specific monolingual and code-switched private layers by updating the parameters

$$\theta_s \longleftarrow \theta_s - \lambda \left(\frac{\partial L_M}{\partial \theta_s} - \frac{\partial L_{CS}}{\partial \theta_s} - \frac{\partial L_A}{\partial \theta_s} \right) \quad (9)$$

$$\theta_m \longleftarrow \theta_m - \lambda \frac{\partial L_M}{\partial \theta_m} \quad (10)$$

$$\theta_c \longleftarrow \theta_c - \lambda \frac{\partial L_{CS}}{\partial \theta_c} \quad (11)$$

$$\theta_a \longleftarrow \theta_a - \lambda \frac{\partial L_A}{\partial \theta_a} \quad (12)$$

The performance comparison of Multi-task adversarial model (Exp6), adversarial task agnostic pooled model (Exp5), the best fine-tuned pooled model [1] (Exp4), and the pooled model (Exp3) are shown in Table 3. We can see that the Multi-task adversarial model (Exp6) outperforms all other models on all monolingual and code-switched test sets. The improved WER can be attributed to the fact that adversarial training helps the shared layer parameters to learn task invariant monolingual and code-switched features, and having task specific layers further help in improving accuracy for individual tasks.

An important caveat to note here is that task specific layers require knowledge of whether an utterance is code-switched or not. This can be achieved either by using a classifier to classify an utterance as monolingual or code-switched, or the output from both task-specific layers can be averaged to make the final prediction. In future work, we plan to compare the results of both these techniques to the proposed model that uses ground-truth knowledge of monolingual and code-switched utterances.

Table 3: *WER[%] of pooled (Exp3), fine-tuned pooled model [1] (Exp4), adversarial task agnostic pooled model (Exp5) and multi-task adversarial speech recognition model (Exp6)*

Test Set	***Exp3***	***Exp4***	***Exp5***	Exp6
TA-MONO	44.42	46.62	45.52	**44.40**
TA-CS	50.92	50.55	50.33	**49.00**
TE-MONO	39.67	41.40	41.60	**40.40**
TE-CS	33.70	33.04	33.80	**32.80**
GU-MONO	37.82	35.80	38.00	**36.10**
GU-CS	40.81	37.50	40.00	**35.70**

3.5. Classification experiments

In order to test whether the shared layers indeed learn task-invariant parameters, we perform classification experiments. We trained a model with shared layers (θ_s) and speech utterance classifier layers which classifies the utterances into monolingual or code-switched with GRL (adversarial discriminator) and without GRL (vanilla classifier) to observe the effect of GRL on the shared layers.

The validation accuracy of both models is shown in Fig. 2. We observe that the validation accuracy remains constant for the adversarial discriminator, while it keeps increasing for the vanilla classifier. This indicates that the shared layers below the classification layers learn task agnostic features in case of the adversarial discriminator. In contrast, the shared layers learn discriminative features for the vanilla classifier.

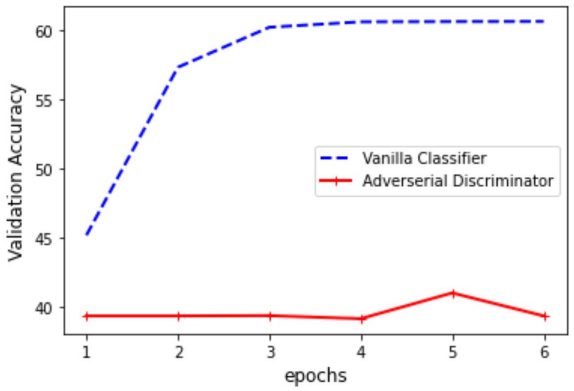

Figure 2: *Validation accuracy of the vanilla classifier and the adversarial discriminator.*

4. Summary and Conclusions

Although monolingual and code-switched speech recognition tasks are similar, we see that trying to improve performance on one hampers the performance on the other. Specifically, training a single model with pooled data containing both monolingual as well as code-switched speech performs better than individual models trained on task-specific data. However, gains on monolingual speech recognition are much lower compared to code-switched speech recognition due to the fact that shared layers learn some task-specific features.

In this paper, we show that learning task invariant shared layer parameters in a pooled model using adversarial training outperforms a pooled model on both monolingual as well as code-switched test sets across three language pairs. We further experiment with adding task specific layers to this model to allow the model to learn some task specific parameters and show improvements on all test sets. Thus, we show that to improve performance on both monolingual as well code-switched speech recognition task, having task invariant shared layers as well as task specific layers are necessary. In future work, we plan to explore techniques to incorporate a Language Identification (LID) system to classify utterances into code-switched or monolingual, as well as explore techniques to use outputs from task specific output layers in the absence of an LID system.

5. References

[1] S. Shah, B. Abraham, G. Reddy, S. Sitaram, and V. Joshi, "Learning to recognize code-switched speech without forgetting monolingual speech recognition," *arXiv preprint arXiv:2006.00782*, 2020.

[2] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, pp. 2935–2947, 2018.

[3] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2096–2030, 2016.

[4] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7167–7176.

[5] X. Chen, Z. Shi, X. Qiu, and X. Huang, "Adversarial multi-criteria learning for chinese word segmentation," *arXiv preprint arXiv:1704.07556*, 2017.

[6] Y. Shinohara, "Adversarial multi-task learning of deep neural networks for robust speech recognition." in *Interspeech*. San Francisco, CA, USA, 2016, pp. 2369–2372.

[7] G. Saon, G. Kurata, T. Sercu, K. Audhkhasi, S. Thomas, D. Dimitriadis, X. Cui, B. Ramabhadran, M. Picheny, L.-L. Lim *et al.*, "English conversational telephone speech recognition by humans and machines," *arXiv preprint arXiv:1703.02136*, 2017.

[8] J. Yi, J. Tao, Z. Wen, and Y. Bai, "Language-adversarial transfer learning for low-resource speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 3, pp. 621–630, 2018.

[9] N. Luo, D. Jiang, S. Zhao, C. Gong, W. Zou, and X. Li, "Towards end-to-end code-switching speech recognition," *ArXiv*, vol. abs/1810.13091, 2018.

[10] G. I. Winata, S. Cahyawijaya, Z. Lin, Z. Liu, P. Xu, and P. Fung, "Meta-transfer learning for code-switched speech recognition," *ArXiv*, vol. abs/2004.14228, 2020.

[11] 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 *Proceedings of the 14th International Conference on Natural Language Processing (ICON-2017)*. Kolkata, India: NLP Association of India, Dec. 2017, pp. 65–74. [Online]. Available: https://www.aclweb.org/anthology/W17-7509

[12] C. Shan, C. Weng, G. Wang, D. Su, M. Luo, D. Yu, and L. Xie, "Investigating end-to-end speech recognition for mandarin-english code-switching," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 6056–6060.

[13] X. Song, Y. Liu, D. Yang, and Y. Zou, "A multi-task learning approach for mandarin-english code-switching conversational speech recognition," 2017.

[14] B. Gambäck and A. Das, "On measuring the complexity of code-mixing," in *Proceedings of the 11th International Conference on Natural Language Processing, Goa, India*, 2014, pp. 1–7.

[15] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd International Conference on Machine Learning*, ser. ICML '06. New York, NY, USA: Association for Computing Machinery, 2006, p. 369–376. [Online]. Available: https://doi.org/10.1145/1143844.1143891

5357

Multilingual Bottleneck Features for Improving ASR Performance of Code-Switched Speech in Under-Resourced Languages

Trideba Padhi, Astik Biswas, Febe de Wet, Ewald van der Westhuizen & Thomas Niesler

Department of Electrical and Electronic Engineering, Stellenbosch University
Stellenbosch, South Africa

{tpadhi, abiswas, fdw, ewaldvdw & trn}@sun.ac.za

Abstract

In this work, we explore the benefits of using multilingual bottleneck features (mBNF) in acoustic modelling for the automatic speech recognition of code-switched (CS) speech in African languages. The unavailability of annotated corpora in the languages of interest has always been a primary challenge when developing speech recognition systems for this severely under-resourced type of speech. Hence, it is worthwhile to investigate the potential of using speech corpora available for other better-resourced languages to improve speech recognition performance. To achieve this, we train a mBNF extractor using nine Southern Bantu languages that form part of the freely-available multilingual NCHLT corpus. We append these mBNFs to the existing MFCCs, pitch features and i-vectors to train acoustic models for automatic speech recognition (ASR) in the target code-switched languages. Our results show that the inclusion of the mBNF features leads to clear performance improvements over a baseline trained without the mBNFs for code-switched English-isiZulu, English-isiXhosa, English-Sesotho and English-Setswana speech. This represents a step forward in the use of out-of-domain data to improve the automatic recognition of code-switched speech in under-resourced South African languages.

Index Terms: Multilingual bottleneck features, acoustic modelling, code-switching.

1. Introduction

With recent rapid advances in the field of artificial intelligence, the ease with which humans can interact with machines has become a yardstick with which the sophistication of a system is assessed [1]. This has had a particularly strong effect in stimulating research interest in ASR. However almost all current speech interfaces assume monolingual input, while most of the world's population is conversant in more than one language. Hence, there had recently also been a surge in interest in the automatic recognition of code-switched speech.

The population of South Africa is highly multilingual and this has recently motivated the development of code-switching ASR systems for African languages [2, 3]. In South Africa, most code-switching occurs between English and one or more South African languages. However, annotated speech corpora that include such mixed-language speech are extremely scarce and those that are available are small. Several approaches have been proposed to address the limitations posed by this lack of annotated speech data. One major drive considers the incorporation of speech in other better-resourced languages to leverage improved ASR performance in the target languages. In [4], the authors show that the overall performance of a multilayer perceptron acoustic model increases substantially when the system is initialized using bottleneck features (BNFs). This acoustic modelling strategy was coupled with a new language modelling strategy called "open target language" which trains more flexible models for language adaptation and with which improvements in performance were reported for under-resourced languages. In [5], improvements in the region of 45% over baseline features were reported when incorporating BNFs for ASR on DARPA RATS data. Two deep bottleneck neural networks were trained on English and Mandarin and the resulting features fused in [6], yielding improvements of between 2% and 7% in equal error rate for longer and shorter segments respectively on the NIST language recognition evaluation 2009 (LRE09) dataset. The authors of [7] analysed the practical aspects of training bottleneck networks as well as their integration in ASR. They also compared monolingual and multilingual training for ASR by evaluating different systems on the LRE09 dataset.

A BNF extractor that was specifically designed for subword modelling and was trained on the GlobalPhone database was proposed in [8]. From 16 of the languages in the Globalphone corpus, 10 high resource languages were used for training the extractor and the remaining 6 for ASR performance evaluation. It was shown that an ASR system trained on a single language using this BNF extractor outperformed a baseline whose features were computed using a correspondence autoencoder and vocal tract length normalization. It was also found that using two or more languages in the BNF extractor training pool resulted in better performance than using a training data set of the same size from only one language.

In previous work we have explored the effectiveness of using out-of-domain monolingual South African speech to improve the performance of code-switched ASR [9]. We found that better-resourced monolingual speech helped to enhance code-switched ASR performance, but only by a small margin considering the amount of out-of-domain data and the computational resources that were required to incorporate the additional data during acoustic model training. Much more out-of-domain data than in-domain data was required to improve code-switched speech recognition accuracy, and thus the most effective way of improving performance has remained the extension of the in-domain training set.

However, given the severe scarcity of resources in the target languages, we have also actively explored other ways to exploit available sources of out-of-domain data. This study represents our first use of BNF extractors to leverage out-of-domain data to improve the accuracy of code-switched speech in five South African languages. Although BNF extractors are generally well established in ASR and other speech processing tasks, to the best of our knowledge, this is the first attempt to train BNF extractors using South African Bantu languages.

We investigate the benefit of training a BNF feature extrac-

tion network on related but out out-of-domain data and then using the extracted BNF features in combination with baseline features (MFCC, pitch and i-vectors) to train acoustic models for South African code-switch ASR. To achieve this, a multilingual BNF (mBNF) extractor is developed using nine South African Bantu languages from the freely-available multilingual NCHLT corpora [10]. Two mBNF extractors with different bottleneck dimensions are trained and used to extract BNFs from the target code-switched speech. We observe that the incorporation of the mBNFs improves the code-switched speech recognition accuracy relative to the system trained using the baseline features.

2. Data

This section introduces two data sets: a set of monolingual corpora in South Africa's 11 official languages that was used to train our mBNF extractor and a corpus of code-switched (CS) South African speech that was used to train acoustic models for ASR purposes.

2.1. Monolingual Speech Data

The NCHLT speech corpora contain monolingual wide-band prompted speech in each of the eleven official languages of South Africa [10]. A greedy algorithm was used to select the prompts from a body of text during the compilation of each corpus [11]. Trigram or five-gram prompts were derived from the text data, depending on the orthographic conventions of each language. This approach resulted in prompts that vary in length from single word utterances to short phrases of up to 10 words. The South African English and Afrikaans corpora were not included in our current investigation, only data from the nine remaining languages that all belong to the Bantu language family were used.

Table 1: *Statistics of the training sets of the NCHLT Bantu speech corpora.*

Language	Speakers	Duration (hours)	Word types	Word tokens
IsiNdebele (nbl)	132	47.3	14 679	132 529
Sepedi (nso)	194	50.6	11 056	266 859
Sesotho (sot)	194	50.7	10 424	250 125
SiSwati (ssw)	181	48.4	11 925	115 611
Setswana (tsn)	194	50.6	5 495	254 274
Xitsonga (tso)	182	49.6	5 934	208 684
Tshivenda (ven)	192	49.3	7 579	218 820
IsiXhosa (xho)	193	49.1	27 856	122 236
IsiZulu (zul)	194	48.3	23 912	116 319
Total	**1 656**	**443.9**	**118 860**	**1 685 457**

We used the predefined NCHLT training sets[1] summarised in Table 1 to train the feature extraction network introduced in Section 3. The NCHLT development and test sets were not used.

2.2. Code-switched Speech Data

For building code-switched (CS) ASR systems, a dataset of multilingual speech was compiled from South African soap opera episodes [12]. The data contains examples of code-switching between four language pairs: English-isiZulu (EZ), English-isiXhosa (EX), English-Setswana (ET) and English-Sesotho (ES). IsiZulu and isiXhosa belong to the Nguni language family whereas Setswana and Sesotho belong to the Sotho-Tswana family. Both these belong to the larger Southern Bantu language family. The available training data consists of three subsets: (1) manually segmented and transcribed data; (2) manually segmented but automatically transcribed data; and (3) automatically segmented and transcribed data. The subsets are described in more detail below. When combined, the the three subsets contain 78.1 hours of speech. The development and test sets were taken from the manually segmented and transcribed data.

2.2.1. Manually segmented and transcribed data

In most of our experiments concerning code-switched speech, we have used a 23-hour set of of annotated speech. This set was partitioned into a training set of 21.1 hours and development and test sets of 48.3 and 78 minutes respectively. The training set includes a language balanced subset as well as additional data that, although skewing the data towards English, was found to enhance ASR performance [13]. Table 2 gives an overview of the manually transcribed component of the training data used in this study.

Table 2: *Duration in minutes (m) and hours (h) as well as word type and token counts for the unbalanced manually segmented and transcribed training set.*

Language	Mono (m)	CS (m)	Total (h)	Total (%)	Word tokens	Word types
English	755.0	121.8	14.6	69.3	194 426	7 908
isiZulu	92.8	57.4	2.5	11.9	24 412	6 789
isiXhosa	65.1	23.8	1.5	7.0	13 825	5 630
Setswana	36.9	34.5	1.2	5.6	21 409	1 525
Sesotho	44.7	34.0	1.3	6.2	22 226	2 321
Total	994.5	271.5	21.1	100.0	276 290	24 170

A similar overview of the development and test sets is given in Table 3. It is noteworthy that the test sets present a strict evaluation as these utterances are never monolingual but always contain code-switching.

Table 3: *Duration in minutes of English, isiZulu, isiXhosa, Sesotho and Setswana monolingual (mdur) and code-switched (cdur) segments in the development and test sets.*

			English-isiZulu		
	emdur	zmdur	ecdur	zcdur	**Total**
Dev	0.0	0.0	4.0	4.0	8.0
Test	0.0	0.0	12.8	17.9	30.4
			English-isiXhosa		
	emdur	xmdur	ecdur	xcdur	**Total**
Dev	2.9	6.5	2.2	2.1	13.7
Test	0.0	0.0	5.6	8.8	14.3
			English-Setswana		
	emdur	tmdur	ecdur	tcdur	**Total**
Dev	0.8	4.3	4.5	4.3	13.8
Test	0.0	0.0	8.9	9.0	17.8
			English-Sesotho		
	emdur	smdur	ecdur	scdur	**Total**
Dev	1.1	5.1	3.0	3.6	12.8
Test	0.0	0.0	7.8	7.7	15.5

[1]The definitions of the predefined NCHLT training, development and test sets are available at `https://sites.google.com/site/nchltspeechcorpus/`

2.2.2. *Manually segmented and automatically transcribed data*

During the compilation of the corpus described in the previous section, some data was manually segmented but not transcribed. In a previous investigation, these segments were transcribed using a semi-supervised procedure, resulting in an additional 11 hours of training data [13].

2.2.3. *Automatically segmented and transcribed data*

The semi-supervised approach applied in the previous section was subsequently extended to include a CNN-GMM-HMM based VAD system (without speaker diarization). This system was used to segment the raw audio of additional soap opera episodes and the resulting segments were transcribed in a semi-supervised manner [14]. Using this procedure a further 45.6 hours of training data was generated. Table 4 provides a summary of the language tags assigned to the data by the semi-supervised procedures.

Table 4: *Number of segments assigned to each language by the semi-supervised transcription systems.*

Language	Eng	Zul	Xho	Sot	Tsn	CS
Man Seg	2 780	3 113	657	3 370	32	13 338
Auto Seg	4 754	2 122	236	719	2 196	17 911

3. Multilingual Bottleneck Feature Extraction

Multilingual bottleneck features (mBNF) have been shown to outperform traditional spectral features as well as monolingual bottleneck features in a variety of speech processing tasks [7, 8, 15]. Table 5 provides an overview of two mBNF extractors that were evaluated in this study. Details on each configuration are provided in subsequent sections.

Table 5: *BNF extractor configurations.*

Extractor	Training data	BNF dimension
$mBNF_1$	NCHLT	39
$mBNF_2$	NCHLT	80

To the best of our knowledge, this is the first attempt to develop a mBNF extractor trained on Bantu languages. Our primary objective is to use the mBNF features to enhance the performance of code-switched ASR. The extractor was developed using the monolingual NCHLT data introduced in Section 2.1 according to the Babel multilang recipe in the Kaldi ASR toolkit [16]. Due to time and computational constraints, fine-tuning of the hyperparameters could not be performed.

As a starting point, a context-dependent Gaussian mixture model-hidden Markov model (GMM-HMM) system is trained for each language to obtain the alignments required for training the feature extraction network. The features used in this step comprise 39-dimensional MFCCs (including Δ and $\Delta\Delta$) calculated over a 25ms window size with 10ms overlap between windows. Three-dimensional pitch features were also included since the Bantu languages are tonal.

A block diagram of the time-delay neural network (TDNN) architecture we used to train the feature extractors is shown in Figure 1. The network is based on the well-established block softmax approach described in [15] and [7]. It consists of six

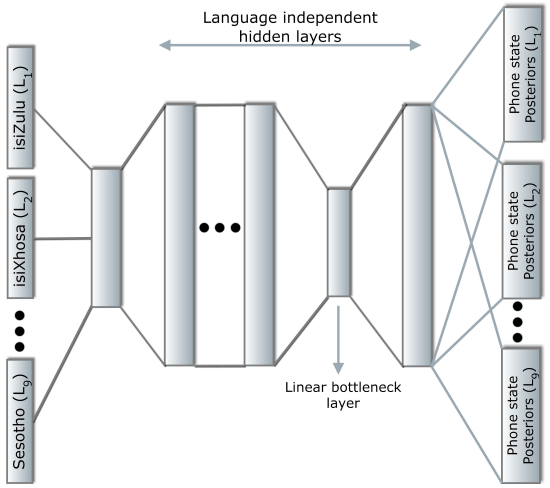

Figure 1: *Multilingual bottleneck feature extractor trained on nine South African Bantu languages with block softmax.*

1024-dimensional hidden layers followed by a 39-dimensional ($mBNF_1$) and 80-dimensional ($mBNF_2$) linear bottleneck layer and terminates in a block softmax output layer. The hidden layers are shared across languages while the block softmax output layer separates the phone state posterior training targets per language. The number of output phone state units varies for each block with a minimum of 4520 for Sesotho and a maximum of 4920 for isiZulu. The input features comprise high resolution 40-dimensional MFCCs (no derivatives), 3-dimensional pitch features and 100-dimensional i-vectors for speaker adaptation. The bottleneck layer is used for mBNF extraction.

4. ASR for code-switched Speech

4.1. Acoustic Model

All acoustic models were trained using the Kaldi ASR toolkit [16] and the training data described in Section 2.2. Three-fold data augmentation was applied prior to feature extraction [17]. The feature set included standard 40-dimensional MFCCs (no derivatives), 3-dimensional pitch and 100-dimensional i-vectors. For the mBNF experiments, combination features were created by appending the mBNFs to these features.

The models were trained with lattice-free maximum mutual information objective [18] using the standard Kaldi CNN-TDNN-F [19] Librispeech recipe (6 CNN layers and 10 time-delay layers followed by a rank reduction layer) and the default hyperparameters. All acoustic models have a single shared softmax layer for all languages as, in general, there is more than one target language in a segment.

No phone merging was performed between languages and the acoustic models were all language dependent. For the bilingual experiments, the multilingual acoustic models were adapted to each of the four target language pairs.

4.2. Language Model

The EZ, EX, ES, ET vocabularies respectively contain 11 292, 8 805, 4 233, 4 957 word types and were closed with respect to the train, development and test sets. The SRILM toolkit was used to train and evaluate all trigram language models [20]. The EZ, EX, ES and ET development set perplexities are 425.8, 352.9, 151.5, and 213.3 respectively. The corresponding values

Table 6: *Word error rate performance of the four bilingual code-switch ASR systems with and without mBNF features.*

System	Feature extractor	Avg		EZ		EX		ES		ET	
		Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
A	Baseline MFCC [14]	39.5	42.1	33.3	38.9	34.7	42.3	49.1	47.9	40.8	39.3
B	$mBNF_1$	38.9	41.5	32.5	38.8	34.1	41.5	49.3	46.5	39.8	39.0
C	$mBNF_2$	39.4	41.0	34.0	38.0	34.0	41.1	48.8	46.7	40.8	38.1

Table 7: *Language specific WER (%) (lowest is best) for English (**E**), isiZulu (**Z**), isiXhosa (**X**), Sesotho (**S**), Setswana (**T**) and code-switched (CS) bigram correct ($\mathbf{Bi_{CS}}$) (%) (highest is best) for the test set.*

System	English-isiZulu			English-isiXhosa			English-Sesotho			English-Setswana		
	E	Z	$\mathbf{Bi_{CS}}$	E	X	$\mathbf{Bi_{CS}}$	E	S	$\mathbf{Bi_{CS}}$	E	T	$\mathbf{Bi_{CS}}$
A (baseline)	32.4	43.9	38.6	35.1	47.9	32.4	34.7	57.0	34.0	27.4	45.4	42.5
B	32.2	43.5	38.7	34.9	45.3	34.2	34.8	54.2	34.9	26.9	45.5	42.8
C	31.7	43.0	39.2	34.6	44.3	34.5	35.0	53.9	34.9	26.7	44.9	43.1

for the test set are 601.7, 788.8, 180.5, 224.5.[2]

5. Results and Discussion

ASR performance was evaluated by measuring word error rate (WER) on the EZ, EX, ES and ET development and test sets described in Table 3. In Tables 6 and 7, System A is the baseline MFCC-based system without BNF features, while Systems B and C use features produced by $mBNF_1$ and $mBNF_2$, respectively.

5.1. BNF Extractor Performance

As can be observed in Table 6, for almost all the cases in the four bilingual data sets, the ASR performance of Systems B and C improve over the baseline when the mBNFs are included in the feature set. Hence, it can be concluded that including mBNFs in the acoustic model training improves ASR performance for code-switched speech. Furthermore, using 80-dimensional BNFs (System C) offers improved test set performance over using 39-dimensional mBNFs (System B) for three of the four language pairs as well as on average. However, training acoustic models with the higher dimensional mBNF features requires more computational resources.

5.2. Language Specific WER Analysis

For code-switched ASR, the recognition performance at code-switch points is of particular interest. Language specific WERs and code-switched bigram correct ($\mathbf{Bi_{CS}}$) values for the different systems are presented in Table 7. Code-switch bigram correct is defined as the percentage of words correctly recognised immediately after a language switch. All values are percentages.

It is interesting to note that mBNFs contributed significantly to reduce the Bantu WER, especially for isiXhosa and Sesotho. Modest reductions in WER were also obtained for English in most of the language pairs. This may be because the extractor was only trained on Bantu languages. However, ultimately the aim would be to have a feature extractor that has generalised well over all data and could be used to extract features for any language equally accurately. Further, the accuracy at the code-switch points is also substantially higher for Systems B and C compared to the baseline (System A). Hence, adding mBNF features enhanced system performance at code-switch points.

[2] A more detailed description of the development of our code-switched language models is provided in [13].

Although current improvements are modest, we would like to point out that initial experiments on our code-switching data that used a BNF extractor trained on a proprietary data set containing other languages yielded similar improvements.

6. Conclusions

We studied the potential benefit of using bottleneck features for acoustic modelling of under-resourced code-switched speech in four South African language pairs. A new bottleneck feature extractor was developed using the Bantu languages in the freely-available NCHLT Speech corpus. Two bottleneck feature extractors producing mBNFs with different dimensionalities were included in the investigation. Recognition results have shown that including the mBNFs in the acoustic modelling not only improved the overall ASR performance for mixed-language speech, but also contributed to improving performance specifically at code-switch points. Future work will include the addition of English and other languages to the pool of languages for mBNF extractor training, optimization of network hyperparameters and investigating the trade-off between performance and the bottleneck dimension.

The source code for training the multilingual bottleneck feature extractor is available at `https://github.com/ewaldvdw/kaldi/tree/mbnf_cs2020/egs/nchlt_multi_bnfs/s5`.

7. Acknowledgements

We would like to thank the former Department of Arts & Culture (DAC) of the South African government for funding this research. We are grateful to e.tv and Yula Quinn at Rhythm City, as well as the SABC and Human Stark at Generations: The Legacy, for assistance with data compilation. We also gratefully acknowledge the support of NVIDIA corporation with the donation GPU equipment used during the course of this research, as well as the support of Council for Scientific and Industrial Research (CSIR), Department of Science and Technology, South Africa for provisioning us the Lengau CHPC cluster for seamlessly conducting our experiments.

8. References

[1] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath et al., "Deep neural networks for acoustic modeling in speech recognition: The

shared views of four research groups," IEEE Signal processing magazine, vol. 29, no. 6, pp. 82–97, 2012.

[2] A. Biswas, F. de Wet, E. van der Westhuizen, E. Yılmaz, and T. Niesler, "Multilingual neural network acoustic modelling for ASR of under-resourced English-isiZulu code-switched speech," in Proc. Interspeech, 2018, pp. 2603–2607.

[3] E. Yılmaz, A. Biswas, E. van der Westhuizen, F. de Wet, and T. Niesler, "Building a unified code-switching ASR system for South African languages," in Proc. Interspeech, 2018, pp. 2603–2607.

[4] N. T. Vu, F. Metze, and T. Schultz, "Multilingual bottle-neck features and its application for under-resourced languages," in Proc. SLTU, 2012.

[5] P. Matejka, L. Zhang, T. Ng, O. Glembek, J. Z. Ma, B. Zhang, and S. H. Mallidi, "Neural network bottleneck features for language identification," in Proc Odyssey, 2014.

[6] B. Jiang, Y. Song, S. Wei, J.-H. Liu, I. V. McLoughlin, and L.-R. Dai, "Deep bottleneck features for spoken language identification," PloS one, vol. 9, no. 7, p. e100795, 2014.

[7] R. Fer, P. Matějka, F. Grézl, O. Plchot, K. Veselỳ, and J. H. Černockỳ, "Multilingually trained bottleneck features in spoken language recognition," Computer Speech & Language, vol. 46, pp. 252–267, 2017.

[8] E. Hermann and S. Goldwater, "Multilingual bottleneck features for subword modeling in zero-resource languages," in Proc. Interspeech, Hyderabad, India, 2018.

[9] A. Biswas, E. van der Westhuizen, T. R. Niesler, and F. de Wet, "Improving ASR for code-switched speech in under-resourced languages using out-of-domain data," in Proc. SLTU, Gurugram, India, 2018.

[10] E. Barnard, M. H. Davel, C. van Heerden, F. De Wet, and J. Badenhorst, "The NCHLT speech corpus of the South African languages," in Proc. SLTU, St.Petersburg, Russia, 2014.

[11] R. Eiselen and M. J. Puttkammer, "Developing text resources for ten South African languages." in Proc LREC, 2014, pp. 3698–3703.

[12] E. van der Westhuizen and T. R. Niesler, "A first South African corpus of multilingual code-switched soap opera speech," in Proc. LREC, Miyazaki, Japan, 2018.

[13] A. Biswas, F. de Wet, E. van der Westhuizen, and T. R. Niesler, "Semi-supervised acoustic model training for under-resourced English-isiZulu code-switched speech," in Proc. Interspeech, Graz, Austria, 2019.

[14] N. Wilkinson, A. Biswas, E. Yılmaz, F. de Wet, E. van der Westhuizen, and T. R. Niesler, "Semi-supervised acoustic modelling for five-lingual code-switched ASR using automatically-segmented soap opera speech," in Proc. SLTU, Marseille, France, 2020.

[15] K. Veselỳ, M. Karafiát, F. Grézl, M. Janda, and E. Egorova, "The language-independent bottleneck features," in Proc. SLT. IEEE, 2012, pp. 336–341.

[16] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz et al., "The Kaldi speech recognition toolkit," in Proc. ASRU, no. CONF. IEEE Signal Processing Society, 2011.

[17] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in Proc. Interspeech, Dresden, Germany, 2015.

[18] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely sequence-trained neural networks for ASR based on lattice-free MMI," in Proc. Interspeech, San Francisco, USA, 2016.

[19] D. Povey, G. Cheng, Y. Wang, K. Li, H. Xu, M. Yarmohammadi, and S. Khudanpur, "Semi-orthogonal low-rank matrix factorization for deep neural networks," in Proc. Interspeech, Hyderabad, India, 2018.

[20] A. Stolcke, "SRILM – An extensible language modeling toolkit," in Proc. ICSLP, Denver, USA, 2002.

5862

The ASRU 2019 Mandarin-English Code-Switching Speech Recognition Challenge: Open Datasets, Tracks, Methods and Results

Xian Shi[1]*, Qiangze Feng*[2]*, Lei Xie*[1]

[1]Audio, Speech and Language Processing Group (ASLP@NPU), School of Computer Science, Northwestern Polytechnical University, Xi'an, China
[2]Datatang (Beijing) Technology Co., LTD, Beijing, China
xshi@npu-aslp.org, xiaoqiang@datatang.com, lxie@nwpu.edu.cn

Abstract

Code-switching (CS) is a common phenomenon and recognizing CS speech is challenging. But CS speech data is scarce and there's no common testbed in relevant research. This paper describes the design and main outcomes of the ASRU 2019 Mandarin-English code-switching speech recognition challenge, which aims to improve the ASR performance in Mandarin-English code-switching situation. 500 hours Mandarin speech data and 240 hours Mandarin-English intra-sentencial CS data are released to the participants. Three tracks were set for advancing the AM and LM part in traditional DNN-HMM ASR system, as well as exploring the E2E models' performance. The paper then presents an overview of the results and system performance in the three tracks. It turns out that traditional ASR system benefits from pronunciation lexicon, CS text generating and data augmentation. In E2E track, however, the results highlight the importance of using language identification, building-up a rational set of modeling units and spec-augment. The other details in model training and method comparsion are discussed.

Index Terms: automatic speech recognition, code-switching, end-to-end ASR

1. Introduction

Code-Switching (CS), the alternating use of more than one languages inside a single utterance [1], is a special and complicated language phenomenon, which has become an important field of both linguistics and ASR research. For instance, the Interspeech 2020 workshop on speech technologies for code switching is a recent platform particularly focusing on CS related research.[1] Code-switching has many varieties, and a classification method based on mixed position is often used, which classifies CS into two primary categories: *inter-sentencial* (switch happens at the sentence boundaries) and *intra-sentencial* (switch happens in the middle of a sentence).

An ASR system usually contains the ability of modeling linguistic information and acoustic information at the same time. In CS situation however, language switching happens at unpredictable positions makes it difficult to train a multilingual language model (LM) while the varied accent of non-native speakers and mixing of phonemes from different language bring difficulty to acoustic model (AM) training.

Previous work has made continous progress in Code-switching area, and a variety of modeling methods are proposed, which can be roughly divided into three categories. The first kind of optimization aims at modeling units, which turns out that both phone merging methods and new modeling units building methods [2,3] are helpful to code-switching ASR. The second kind of methods focus on the nerual network structure, making deep neural networks-hidden Markov model (DNN-HMM) based ASR system more competent for CS tasks by optimizing the neural network and training strategy [4,5]. The third kind of efforts is explored on End-to-End (E2E) speech recognition [6–9]. E2E ASR framework enables lexicon-free recognition, which is an important advantage over traditional hybrid system, especially for CS tasks. An encoder-decoder based CS ASR system was built by Hiroshi et al. [10]. Zhang et al. [11] built a bilingual Mandarin-English acoustic model by putting two separately pre-trained DFSMN-CTC-sMBR together. Li et al. [12] added a frame-level language identification (LID) loss to bilingual CTC model, assisting CTC to distinguish the language ID of frames. Although those efforts have improved the performance of CS ASR, robust ASR system that supports arbitrary switching of languages still remains a challenging goal.

This paper describes the design and outcomes of the ASRU 2019 Mandarin-English Code-Switching Speech Recognition Challenge, a special event of IEEE Automatic Speech Recognition and Understanding Workshop (ASRU 2019).Actually, the difficulties discussed above also reveal one of the bottelnecks in CS ASR research: *data insufficiency*. Code-switching speech data is always scarce, only SEAME [13], a small set of 30 hours Mandarin-English speech data collected in Singapore and Malaysia is released to the public. Besides, in Mandarin-English CS ASR, there is no common testbed and open datasets for method validation and model comparison, especially in the area of fast development of data-hungry deep learning approaches. This challenge is especially designed for these reasons. Three speech datasets are released to participants, 740 hours in total, and 240 hours of them are Mandarin-English CS data[2]. A well-trained 3-gram CS language model in ARPA format is also provided. The participants are supposed to use the permitted data only to build CS ASR systems in three tracks: i) Traditional ASR system with identical official N-gram LM; ii) Traditional ASR system without LM limitation; iii) End-to-End ASR system. Totally 72 teams participated in the challenge. Participants have around 50 days to finish their system building and submit the recognition results.

The rest of this paper is organized as below: Section 2 describes detailed information of the datasets. In Section 3, rules, data using limitation of each track and results evaluation method are explained. Section 4 describes a overview of results submitted and advancing system building methods in all the tracks. Summary of the main findings in the challenge is in Section 5.

[1]*www.microsoft.com/en-us/research/event/workshop-on-speech-technologies-for-code-switching-2020*

[2]Exploring *www.datatang.com/competition* for more detail about datasets and the challenge.

2. Open Source Datasets

Code-switching speech data is always scarce, which hinders the research of CS ASR seriously. For this challenge, DataTang released 3 ASR datasets to participants. The basic information of the datasets is as below.

Table 1: *Basic information of the 3 released datasets*

Dataset	Transcripts Type	Dur/hours
$Train_{Man}$	Mandarin only	500
$Train_{CS}$	Intra-sen Mandarin-English CS	200
Dev_{CS}	Intra-sen Mandarin-English CS	40

All the data are collected by smart phones in quiet rooms from various Android phones and iPhones. The speakers were from 30 provinces in China. 70% of the speakers were under 30 years old, with no significant difference in the number of male and female.

The transcripts of data cover many common fields including entertainment, travel, daily life and social interaction. In $Train_{Man}$, each sentence has 10 Chinese characters in average. As for $Train_{CS}$ and Dev_{CS}, each sentence has 8.6 Chinese characters and 1.6 English words in average. Most English words are nouns, personal names, song names and some adjectives. Besides, there are 6 kinds of symbols and tags for noise and English abbreviation in $Train_{CS}$ and Dev_{CS} transcripts. Several examples from the dataset are shown in Table .

Table 2: *Examples of CS transcription from the dataset*

1	CS transcription: EN translation:	“我今天要去买一个iPhone.” “I’ll buy an iPhone Today.”
2	CS transcription: EN translation:	“Jeff是一个很sensitive 的学生.” “Jeff is such a sensitive student.”

3. Tracks Setting and Rules

Traditional forced alignment based ASR system consists of two separately trained components: acoustic model and language model. The hybrid MMI-Chain model build by Kaldi [14] is regarded as one of the state-of-the-art ASR systems. The first two tracks aim at making the traditional ASR system able to recognize Mandarin-English code-switching speech. E2E ASR is drawing increasing attention in recent years, and it seems to have more potential and possibility to solve CS question. Track 3 was built for E2E systems to compare with each other.

A series of instructions are designed to ensure an equitable comparison. In Section 3.4, the evaluation of code-switching recognition results is explained.

3.1. Track1: Traditional ASR with identical N-gram LM

Acoustic model in traditional ASR system is used to bind a speech frame to a certain unit through computing acoustic likelihood. This track focus on AM behavior only.

A Mandarin language model and a code-switching language model are trained separately by KenLM toolkit [15] and merged with SRILM toolkit [16]. The triple-gram in Mandarin training data occurs less than 10 times are pruned while no pruning was applied to uni-gram and bi-gram. The final size of the merged arpa is 2G.

Rules that participants should follow in track 1 is as below:

1. The acoustic model should be a frame-level force-alignment model, and CTC model is prohibited.
2. Data used for AM training is limited to $Train_{Man}$, $Train_{CS}$ and Librispeech [17] 960 hours English speech data. Data augmentation methods are allowed.
3. Multi-system fusion techniques including recognizer output voting error reduction (ROVER) [18] are prohibited.
4. Decoding graph should be complied with G.fst generated by the given arpa. Any kind of lattice rescoring is prohibited.

3.2. Track2: Traditional ASR without LM limitation

Language model also plays an important role in traditional ASR. It estimates the grammatical rationality of character or word sequences. In this track, any training data for LM and all kinds of techniques including but not limited to RNN (Recurrent Neural Network) LM , large scale LM rescoring are allowed, but AM still should be trained under the rule 1-3 of track 1.

3.3. Track3: End-to-End ASR

End-to-End ASR here refers to systems without frame-level forced-alignment, always modeling acoustic information and language information jointly. It is becoming an increasingly topical field and various of E2E ASR systems are proposed. Encoder-Decoder based system LAS [7] and transformer [6] use global attention and multi-head self-attention to generate implicit alignment. RNN-transducer combines two RNNs into a sequence transduction system [19, 20].

Including the models above, any E2E ASR system is allowed in track 3, and CTC model is also regarded as an E2E model. Rule 2-3 in track 1 are effective in track 3. Besides, as for systems need to model acoustic information and language information jointly, the text training data is limited to transcripts of permitted speech data.

3.4. Results Submission and Evaluation Plan

Competitors are supposed to submit their recognition results and system descriptions of each track they participated in. Recognition accuracy is the only target considered in the evaluation. Mixture error rate (MER) considers Mandarin characters and English words as the tokens in the edit distance calculation. Errors of Chinese and English will be counted separately according to the language of the reference token.

The error rate of the Chinese part and the English part in the final publicity result is only for reference, ranking is based on MER only.

4. Results and Discussion on Methods

4.1. Track1

35 teams submit their results of track 1, the top 10 best systems is listed in Table 3, along with the their key features. The following part introduces the main outcomes in three aspects.

4.1.1. Phone Sets

Building a traditional ASR system starts with building a phone set. Among the 20 teams that introduced their phone sets building methods, 11 teams use totally separate phone sets for Mandarin and English, 6 teams bind partial phones according to

Table 3: *Top 10 of 35 submitted systems in track 1. The columns in the middle summarize the key features of systems, CER(%) for Chinese part error rate, WER(%) for English part error rate, and MER(%) for mixture. The right side of the table describe the error rate of Mandarin part, English part and total MER. Phone merging includes partial combining and totally binding of Chinese phones and English phones.*

Team	NN Structure	*Phoneme Merge*	*G2P for OOV*	*Data Augment*	*i-vector extract*	*Spec Augment*	CH ER(%)	EN ER(%)	**MER(%)**
MobvoiASR	CNN-TDNN	✔	✔	✔	✔	✔	4.04	12.33	**4.94**
Qdreamer	CNN-LSTM-TDNN	✔	✔	✔		✔	3.85	14.88	**5.05**
XNXYZ	LSTM-TDNN		✔	✔		✔	4.05	15.43	**5.28**
SZSXW	TDNN	✔	✔				4.61	14.44	**5.66**
JRYY	TDNN			✔	✔		4.60	15.06	**5.74**
VIVO ASR	TDNN		✔	✔	✔		4.50	16.63	**5.81**
I2R	CNN-TDNN	✔			✔		4.95	14.32	**5.97**
Paopao	TDNN	✔	✔	✔	✔	✔	5.22	14.14	**6.19**
SCUT-ASR	CNN-TDNN	✔	✔				5.43	15.99	**6.57**
Royalflush	CNN-TDNN		✔	✔		✔	5.18	18.37	**6.61**

phonetics. 2 teams map all of the English phones to Chinese phones, above that, one team marked partial English words which appear frequently with Chinese phones.

Concatenating a Chinese lexicon and an English lexicon is the most simple and commonly used method, and the merged phone sets cat be extracted from the lexicon directly. Several teams with higher ranking proved that binding partial Chinese and an English phonemes works to acoustic modeling benefit. The last method 'map partial English words' refers to marking the high frequency English words with Chinese phonemes, with the rest part of the lexicon still using the first method. However, there is no detailed contrastive experiment results about phone sets reported.

4.1.2. Feature Extraction and Data Augmentation

Concatenating i-vectors feature to MFCC (Mel Frequency Cepstral Coefficents) or Filter-bank feature brings 5%-7% relative improvement. Librispeech data is abandoned by most teams as it raised error rate, even when using only a small part of it. This may because of the mismatch of native English speaker and Chinese speaker. Speed augmentation can enhance the robustness of the model modestly, while volume augmentation and reverberation simulation help little. This may because the training data and test data are collected in the environment with similar acoustic conditions. Spec-augment is a data augmentation method proposed by Google [21]. Several teams gain about 2% relative improvement using spec-augment layer in Kaldi Nnet3.

4.1.3. Network Structure

Kaldi chain model with lattice-free maximum mutual information (LF-MMI) [22] is used by all the teams, there are seldom differences among different systems. CNN or LSTM are used to combine with time-delay neural network (TDNN). The 1st team MobvoiASR used max-likelihood path to fix the original loss function and gain 3% relative improvement. The 2nd place team Qdreamer use LF-MMI-SMBR (State-level Minimum Bayes Risk) and gain 9% relative MER reduction comparing to original LF-MMI.

Table 4: *The top 10 teams in track 2, along with their key method used, MER(%) stands for mixture error rate. MERR are calculated based on their results in track 1. A negative MERR indicates that participants achieved a better LM than the 3-gram released in track 1.*

Team	*Weight Tuning*	*Text Generation*	*Lattice Rescoring*	**MER(%)**	**MERR(%)**
MobvoiASR	✔		✔	4.72	− 4.5
Qdreamer	✔			5.64	+ 11.7
JingRong	✔			5.80	+ 1.0
Royalflush	✔	✔	✔	5.88	− 11.0
VIVO ASR	✔	✔	✔	5.91	− 1.7
I2R	✔			6.25	+ 4.7
Aisg-xju	✔		✔	6.65	− 6.0
Xmuspeech	✔		✔	7.01	+ 2.5
LKDMM	✔	✔		8.57	− 11.2
MiniSpeech				8.95	− 1.1

4.2. Track2

The recognition results of the top 10 systems are shown in Table 4. The efforts teams made for track 2 are mainly about CS text generation, balancing Chinese and English text proportion, and RNN-LM rescoring.

Spontaneous code-switching text data for LM training is always scarce because of the randomness and casualness of CS

Table 5: *Top 10 of submitted systems in track 3.*

Team	Model	*Char + BPE*	*Transfer Learning*	*Data Augment*	*Language Model*	*Spec Augment*	*LID Multitask*	CH ER(%)	EN ER(%)	**MER(%)**
WYHZ	Transformer	✔		✔		✔	✔	4.33	18.95	**5.91**
SJTU SL	Transformer	✔	✔	✔		✔	✔	6.93	24.35	**8.82**
Royalflush	Transformer	✔		✔		✔	✔	7.49	21.40	**9.00**
Code-switcher	LAS	✔	✔		✔	✔		7.38	25.69	**9.37**
ZFZ	Transformer	✔		✔		✔		8.49	24.56	**10.24**
Qdreamer	CLDNN+CTC		✔	✔	✔	✔		8.23	33.32	**10.90**
VIVO ASR	Transformer			✔				9.05	32.21	**11.57**
UVoice	BLSTM+CTC	✔			✔	✔		8.94	41.46	**12.48**
xmuspeech	Joint-CTC Attention			✔				9.59	37.20	**12.59**
Aisg-xju	Transformer	✔			✔	✔		10.44	31.60	**12.74**

phenomenon. Therefore, it's necessary to expand the text data. The parallel language pair in machine translation is widely used in ASR to generate CS transcripts. Besides, text generation based on pointer generator [23] is used by team Royalflush, but this method is limited by the scale of CS text and only a small amount of available data is generated. As reported, a well-trained RNN-LM using external CS data can yield 2%-4% recognition improvement.

4.3. Track3

In E2E track, 29 teams submit their results, the top 10 teams' results and key features are described in Table 5. The outcomes of this track are mainly about modeling units, network structure, as well as taking language identification into consideration.

4.3.1. Modeling Units

The E2E ASR systems use sequence-to-sequence model to map the speech frames to the character sequence. In Chinese ASR, character is commonly used to be the modeling units directly as the amount of Chinese characters is around 6k. But modeling words in English directly is difficult because of the large amount and the sparsity of low frequency words. So in the challenge, Chinese character and English word piece [24] is mostly used. Its advantages mainly come from two aspects: balancing the granularity of Chinese and English modeling units and solving OOV problem with limited English training data. The number of English word pieces that teams used varied from 1k to 3k. Except for char + bpe, there are also teams using syllable for Chinese and letter for English.

4.3.2. Network and Language Modeling

The winner of track 3 went to a Transformer model [10] trained by ESPnet [25], using multi-task learning to guide the decoder to distinguish Chinese and English characters (as reported, the language distinguishing CE loss optimized at decoder outperformed it at encoder). Label smoothing, averaging checkpoints and spec-augment all yield recognition improvements. Data augmentation performs almost same as in track 1. Transfer learning in the table refers to all kinds of different languages pre-training and fine-tuning strategies.

4.3.3. Language Modeling

As for language information modeling, 4 of the 10 teams in Table 5 use language model for rescoring or fusion with AM. Aisg-xju's and Royalflush's language models are RNN-LMs used for shallow fusion. UVoive's language model is a 4-gram LM used in CTC prefix beam search. Qdreamer uses a 3-gram LM as first pass and an RNN-LM for rescoring.

5. Conclusions

In the ASRU 2019 code-switching automatic speech recognition challenge, participants used 500 hours Mandarin speech data and 200 hours intra-sentencial CS data to build ASR systems with recognition ability for Mandarin and English within a single utterance. Most teams achieve 5% Chinese part error rate and English error rate under 20% with DNN-HMM based models. The E2E models haven't outperformed the traditional model yet. It is clear that the systems tend to have higher recognition accuracy for Chinese part in the utterance, the reason may come from the imbalance of the data in two languages, which brings difficulty for LM training. The grammar of skipping between English words is completely invalid. According to the results of the three tracks aforementioned, traditional ASR trained by Kaldi chain model outperformed the E2E models, but the gap is quickly narrowing. The result has highlighted that the detail of pronunciation lexicon and neural network effect a lot. In track 2, text generation is proved to be the most effective way to augment the language model, both word substitution according to grammatical rules and generative neural network help in data expansion. It is worth noting that RNN-LM did not replace N-gram LM but complemented it. As to E2E models, it turns out that attention based models performed more competitive and language identification help the model distinguish languages. Besides, spec-augment is proved a robust method of data augmentation with obvious performance gain.

In this challenge, however, only recognition accuracy is considered in the evaluation. In the future, higher and more comprehensive requirements will be put forward, like streaming ASR system and ASR under complex acoustic environments.

6. References

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

[2] B. Li, Y. Zhang, T. Sainath, Y. Wu, and W. Chan, "Bytes are all you need: End-to-end multilingual speech recognition and synthesis with bytes," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5621–5625.

[3] H. Lin, L. Deng, D. Yu, Y.-f. Gong, A. Acero, and C.-H. Lee, "A study on multilingual acoustic modeling for large vocabulary asr," in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2009, pp. 4333–4336.

[4] S. Tong, P. N. Garner, and H. Bourlard, "An investigation of deep neural networks for multilingual speech recognition training and adaptation," in *Proc. of INTERSPEECH*, 2017.

[5] A. Ghoshal, P. Swietojanski, and S. Renals, "Multilingual training of deep neural networks," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 7319–7323.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[7] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 4960–4964.

[8] T. Hori, S. Watanabe, Y. Zhang, and W. Chan, "Advances in joint ctc-attention based end-to-end speech recognition with a deep cnn encoder and rnn-lm," *arXiv preprint arXiv:1706.02737*, 2017.

[9] S. Watanabe, T. Hori, S. Kim, J. R. Hershey, and T. Hayashi, "Hybrid ctc/attention architecture for end-to-end speech recognition," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 8, pp. 1240–1253, 2017.

[10] H. Seki, S. Watanabe, T. Hori, J. Le Roux, and J. R. Hershey, "An end-to-end language-tracking speech recognizer for mixed-language speech," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4919–4923.

[11] S. Zhang, Y. Liu, M. Lei, B. Ma, and L. Xie, "Towards language-universal mandarin-english speech recognition," *Proc. Interspeech 2019*, pp. 2170–2174, 2019.

[12] K. Li, J. Li, G. Ye, R. Zhao, and Y. Gong, "Towards code-switching asr for end-to-end ctc models," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6076–6080.

[13] D.-C. Lyu, T.-P. Tan, E. S. Chng, and H. Li, "Seame: a mandarin-english code-switching speech corpus in south-east asia," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[14] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, no. CONF. IEEE Signal Processing Society, 2011.

[15] K. Heafield, "Kenlm: Faster and smaller language model queries," in *Proceedings of the sixth workshop on statistical machine translation*. Association for Computational Linguistics, 2011, pp. 187–197.

[16] A. Stolcke, "Srilm-an extensible language modeling toolkit," in *Seventh international conference on spoken language processing*, 2002.

[17] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.

[18] J. G. Fiscus, "A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover)," in *1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings*. IEEE, 1997, pp. 347–354.

[19] A. Graves, "Sequence transduction with recurrent neural networks," *arXiv preprint arXiv:1211.3711*, 2012.

[20] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, "High-dimensional sequence transduction," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 3178–3182.

[21] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[22] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, and S. Khudanpur, "Purely sequence-trained neural networks for asr based on lattice-free mmi," in *Interspeech 2016*, 2016.

[23] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," *arXiv preprint arXiv:1704.04368*, 2017.

[24] C.-C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, E. Gonina *et al.*, "State-of-the-art speech recognition with sequence-to-sequence models," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4774–4778.

[25] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. E. Y. Soplin, J. Heymann, M. Wiesner, N. Chen *et al.*, "Espnet: End-to-end speech processing toolkit," *arXiv preprint arXiv:1804.00015*, 2018.