TOWARDS CODE-SWITCHING ASR FOR END-TO-END CTC MODELS

Ke Li[1,2] *, Jinyu Li*[1]*, Guoli Ye*[1]*, Rui Zhao*[1]*, Yifan Gong*[1]

[1] Microsoft Speech and Language Group
[2] Center for Language and Speech Processing, Johns Hopkins University

ABSTRACT

Although great progress has been made on end-to-end (E2E) models for monolingual and multilingual automatic speech recognition (ASR), there is no successful study for E2E models on the challenging intra-sentential code-switching (CS) ASR task to our best knowledge. In this paper, we propose an approach for CS ASR using E2E connectionist temporal classification (CTC) models. We use a frame-level language identification model to linearly adjust the posteriors of an E2E CTC model. We evaluate the proposed method on Microsoft live Chinese Cortana data with 7000 hours Chinese and English monolingual data and 300 hours CS data as the training data. Trained with only monolingual data without observing any CS data, the proposed method can obtain up to 6.3% relative word error rate (WER) reduction. In the scenario of training with both monolingual and CS data, the proposed method can get up to 4.2% relative WER improvement. This approach can also maintain comparable performance on a Chinese test set compared with baseline models.

Index Terms— code-switching, ASR, end-to-end, CTC, language identification

1. INTRODUCTION

Traditionally, automatic speech recognition (ASR) systems contain components including acoustic, pronunciation, and language models that are separately trained, each with a different objective. End-to-end (E2E) ASR [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] is an emerging field because of its simplicity compared with conventional ASR. An E2E system directly maps an input sequence of acoustic features to an output sequence of characters, phonemes, or words. There are mainly three E2E ASR systems: (a) connectionist temporal classification (CTC) [11, 12], (b) attention based encoder-decoder networks [13, 14, 15, 16], and (c) recurrent neural network (RNN) transducer [17].

The above-mentioned E2E models have been successfully applied to large-scale monolingual [1, 3, 4, 5, 7, 9, 10, 18, 19, 20, 21, 22] as well as multilingual ASR tasks [23, 24, 25]. However, it is very challenging to be successful in code-switching (CS) ASR. CS refers to the phenomenon of mixed words or phrases from two or more distinct languages by a speaker. This phenomenon widely exists in multilingual communities such as Cantonese-English [26], Chinese-English [27], Spanish-English [28], Hindi-English [29], and Frisian-Dutch [30]. Depending on where the switching of languages happens, there are two types of CS phenomenon. One is intra-sentential CS where switches happen within an utterance. The other is inter-sentential CS where switches occur at the boundaries of utterances. The former one is more difficult as acoustical variations of mixed languages within utterances can be larger than across utterances.

To our best knowledge, there is no successful E2E work in the challenging intra-sentential CS scenario. For example, Kim et al. [23] utilized a language-specific gating mechanism to build a multilingual end-to-end ASR system, but it cannot deal with CS scenarios. Similarly, Google's encoder-decoder based E2E multilingual model [24] that is jointly trained on data from all languages with output as the union of language specific sets can not address the CS issue. Seki et al. [25] generated utterance-level synthetic CS text to improve an encoder-decoder model while it cannot handle intra-sentential CS.

One potential reason that the encoder-decoder based model cannot work well for CS scenarios is the output of the decoder depends on the previous outputs. When the previous steps keep emitting tokens from one language, it is very hard to immediately switch to tokens of another language due to this dependency. In contrast, the CTC model has the output independence assumption, which is in general inaccurate but may make it more desirable to handle CS scenarios as the current step output does not explicitly rely on previous outputs.

Note that most previous work on CS ASR studies how to improve the performance for CS utterances. However, in most scenarios, the utterances are mainly in one major monolingual language, and CS utterances contain both the major language and a secondary language. For example, for the Microsoft Cortana live data collected in China, most utterances mainly contain Chinese tokens. People only switch to English from time to time for words which cannot be easily spoken in Chinese (e.g., iPhone, Cortana etc.). Therefore, a better goal for CS ASR should maintain similar ASR accuracy as mononlingual systems for the major language, and improve the ASR performance of CS utterances at the same time. It is very hard to evaluate a method for CS if we do not know how much it hurts the recognition of the major language, which is missing in most CS ASR studies [26, 27, 28, 31, 29, 30, 32, 33, 34, 35].

In this paper we study the effective way of doing intra-sentential CS ASR for E2E CTC models. It has been shown that E2E CTC models can produce comparable performance as traditional hybrid deep acoustic models given the training data size is reasonably large [3, 9]. We aim to improve the performance of E2E CTC systems for CS scenarios and at the same time to maintain reasonable performance on the major language. In this study, we propose a language identification (LID) based approach to deal with CS ASR for E2E CTC models. This approach separately trains an E2E CTC model and an LID model, and directly adjusts the posteriors of the CTC model with corresponding LID scores. We use greedy decoding without any language model or lexicon component. We focus on Chinese-English intra-sentential CS scenario.

The rest of the paper is organized as follows. In Sec. 2, we intro-

Ke Li performed the work while she was a research intern at Microsoft Speech and Language Group.

978-1-5386-4658-8/18/$31.00 ©2019 IEEE

duce related work on code-switched ASR systems. In Sec. 3, we propose the approach for incorporating a LID into an E2E CTC model to handle intra-sentential CSs. We evaluate the proposed method in Sec. 4, and present conclusions in Sec. 5.

2. RELATED WORK

Prior work of CS ASR is mainly in conventional hybrid systems [26, 27, 28, 31, 29, 30, 32, 33, 34, 35]. A traditional multi-pass approach is to first label speech frames with languages involving language boundary detection [36] and language identification (LID) [37, 38], and perform recognition on labeled speech segments with corresponding monolingual speech recognizers [34]. ASR performances of these piped approaches are restricted by the language boundary detection and LIDs at front-end. To alleviate this issue, a single-pass approach without language boundary detection and LID is proposed [31, 32, 30]. [39] adopted semi-supervised learning approaches of lexicons and acoustic models to improve CS ASR systems. [40] utilized a speech chain framework to enable ASR and text-to-speech synthesis systems to learn code-switching in a semi-supervised fashion. Besides, [41] proposed to jointly learn accent ID and acoustic models for better recognizing accented speech.

The above methods mainly focus on acoustic models. CS happens more in spoken than written scenarios. Thus, the CS text corpus is very limited. To solve this problem, semantic-based mappings for n-gram language models [42] as well as a word-class n-gram language model [43] are proposed to better model low-frequent and unseen CS n-grams. Sreeram et al. [33] proposed a monolingual class-based recurrent neural network language model (RNNLM) that incorporates a CS tagger predicting switches of languages. Yilmaz et al. [35] used RNNLM and neural machine translation based approaches to generate more CS text for better language modeling. Besides, transliteration based approaches are proposed to improve code-switched ASR performance [44].

3. METHODS

3.1. E2E CTC Models

We use bidirectional long short-term memory (LSTM) recurrent neural networks (RNNs) as model architecture and CTC loss as objective function [11, 12]. The ASR output symbols in an utterance are usually fewer than the input speech frames. Hence CTC paths that allow repeated labels and blank tokens are used to force the output to have the same length as the input speech frames. Let us denote $\mathbf{x}$ as the speech input sequence, $\mathbf{l}$ as the original label sequence, π as the CTC path, Θ as the network parameters, and $B^{-1}(\mathbf{l})$ as all possible CTC paths expanded from $\mathbf{l}$.

The CTC loss function is defined as the sum of negative log probabilities of correct labels as:

$$L_{CTC} = -\ln P_{\Theta}(\mathbf{l}|\mathbf{x}) = -\ln \sum_{\pi \in B^{-1}(\mathbf{l})} P_{\Theta}(\pi|\mathbf{x}) \qquad (1)$$

Based on the conditional independence assumption for output units, $P_{\Theta}(\pi|\mathbf{x})$ can be decomposed to a product of posteriors from each time step t as below:

$$P_{\Theta}(\pi|\mathbf{x}) = \prod_{t=1}^{T} P_{\Theta}(\pi_t|x_t), \qquad (2)$$

where x_t is the input speech at time t, π_t is the output unit at time t, and T is the sequence length in frames.

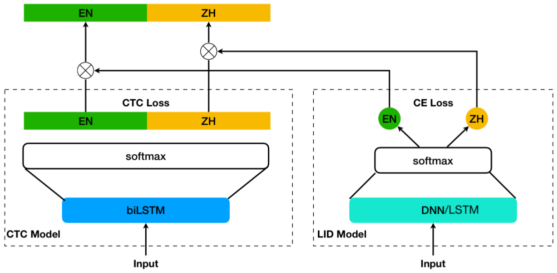

Fig. 1: A explicitly combined CTC model and a frame-level LID model for recognizing CS utterances. Chinese (ZH) and English (EN) are languages in this example.

The output labels of E2E CTC systems can be either letters or words. CTC outputs are usually dominated by blank labels. The outputs corresponding to the non-blank labels usually occur with spikes in their posteriors. Thus, an easy way to generate ASR outputs using CTC is to concatenate the non-blank labels corresponding to the posterior spikes and collapse those labels into word outputs if needed. This is known as greedy decoding. It is a very attractive feature for E2E modeling as there is neither any LM nor any complex decoding involved. We thus use greedy decoding in this study.

3.2. CTC with Language Identification

In this study, we propose to use LIDs to improve the E2E CTC's performance on CS utterances. As CS ASR is a challenging task of recognizing large amount of output units from both the major and secondary languages, it is relatively easier to recognize which language the segments come from. If we can accurately predict the language at each time step, especially for the switching cases, we can use this information to better guide the original ASR model to handle CS scenarios. Hence, we can improve the CS ASR performance by switching from a challenging CS ASR problem to an easier LID task.

Various methods can be applied [37, 45] to LID. Usually LIDs are on utterance or phrase level. In this study, we aim to build a frame-level LID for CS ASR so that we can combine LID outputs with CTC outputs at the frame level. We use feed-forward deep neural networks (DNNs) and LSTMs to build frame-level LIDs for predicting both the major and secondary languages (as well as silence).

We thus propose an explicit combination of a frame-level LID and an E2E CTC model. In Figure 1, we give an example how to train a CS model with the help of LID for Chinese-English CS utterances. The detailed process is as follows.

- Step 1: Initialize the CS CTC model from a major language CTC model by keeping the network hidden weights and replacing the output targets of the major language with the union of units from the major and secondary languages.
- Step 2: Train this CS CTC model by updating all the parameters with data from both languages.
- Step 3: Train an LID model on the same training data with three outputs, predicting the major language, the secondary language, and silence frame-by-frame.
- Step 4: During decoding, if the blank symbol dominates current frame, emit the blank token as the label of this frame. Otherwise, multiply the posteriors of the major and second

language outputs from the CTC model with the LID probabilities of the corresponding languages. And then emit the label with the maximum posterior among the units of these two languages.

- Step 5: The decoding hypothesis is generated by collapsing the above output tokens with greedy decoding.

Based on the experience that CTC models are very sensitive to initialization [10], initializing them from the CTC model of the major language at step 1 is critical to guarantee that the CS model can still work reasonably on the major language. This is also different from the common multilingual setup which usually trains the shared hidden layer by treating all languages equally [46].

Both DNN and LSTM models can be used to predict frame-level LID at step 3. However, it is impossible to accurately predict which language a frame is from given a single frame. Therefore, for DNN-based LID prediction, we use a relatively large context window so that sufficient information is provided to predict current frame's language. For LSTM-based LID prediction, if we use a bi-directional model, we should be able to predict well because the bi-directional processing has the access to the whole utterance.

A more formal expression of the proposed combination at step 4 is as below. Let us denote o_t as the predicted language at time t, the probability of each language given a speech frame x_t at time t is $P(o_t|x_t)$. And the adjusted posterior of CTC $P_\Theta^{adj}(\pi_t|x_t)$ at time step t can be written as:

$$P_\Theta^{adj}(\pi_t|x_t) = P_\Theta(\pi_t|x_t) \times P(o_t|x_t)^\alpha \tag{3}$$

where α is the hyper-parameter that controls the influence of the LID on the CTC posteriors.

It should be noted that the combination of CTC and LID posteriors only makes sense when the CTC model is a bi-directional model. This is because the uni-directional CTC model has the notorious output delay issue [47] which means the position of CTC output is not aligned well. In contrast, the bi-directional CTC can give reliable output alignment given it has the information from both directions.

4. EXPERIMENTS

4.1. Datasets and Setups

We evaluate the proposed method on the live utterances collected from Microsoft live Cortana application in China. In this work, we only focus on CS scenarios of Chinese (ZH) and English (EN). While our approach can also be applied to other languages and situations with more than two languages. In the Chinese market, the major language is Chinese and the secondary language is English. The goal is to improve Chinese-English CS performance without sacrificing too much the ASR accuracy for Chinese only utterances.

We have two test sets. One is a CS test set, containing 30k Chinese characters and English words (accounts for 33.8%). The other is a Chinese only test set with 50k Chinese characters. We treat every Chinese character as a word, and thus we can report word error rate (WER) for the evaluation. For training, we have both monolingual and CS data, with around 4000 hours Chinese Cortana data, 3400 hours US-English Cortana data, and 300 hours code-switch data.

We use CNTK to train all the CTC models. All the E2E CTC models contain 6 bi-directional LSTM layers, with hidden dimension 512 in each direction. We derived 80-dimensional log Mel filter-bank energies at 10-ms intervals and stacked 3 contiguous frames to form 240-dimensional features for CTC [18]. We apply greedy decoding to generate word sequences based on posterior spikes of the CTC models.

Table 1: WERs (%) of baseline CTC models trained from Chinese (ZH) data or the combination with English (EN) or CS data(CS).

Test sets	ZH	ZH+EN	ZH+EN+CS	CS
CS set	58.06	59.08	30.81	25.48
ZH set	11.03	11.04	11.01	21.92

4.2. Baseline CTC Models

We first train a baseline CTC model with monolingual 4000hr Chinese data. It uses around 7k Chinese characters together with blank as the outputs. This model obtains 11.03% WER on the Chinese only set and 58.06% WER on the CS test set, as shown in Table 1. All the CS CTC models are initialized from this baseline Chinese CTC model as described at step 1 in Section 3.2.

In the following, we will build several CS CTC models which have both Chinese and English output units. Given we have around 7k Chinese characters, we select around 8k letter trigrams as the English outputs. In this way, we can balance the Chinese and English outputs. We merge the Chinese characters and English letter trigrams to form 15k output units used for all the CS models.

We build the first CS CTC model by merging the 4000 hours Chinese only and 3400 hours English only data to investigate whether we can build a CS model from monolingual data. Unfortunately, this model does not improve the accuracy for CS test set, but does not degrade performance on the Chinese test set neither. This is possibly because we initialize the model from the Chinese CTC model, which makes the final model still be biased towards the Chinese outputs. We can have better performance on the CS test set if we do not have such initialization but degrade the accuracy on Chinese test set, deviating from our goal.

Next, we build the second CS CTC model by merging all the monolingual data with 300 hours CS data so that this model can observe the CS phenomenon. Without surprise, it significantly improves the performance on the CS test set with 30.81% WER and has similar performance on the Chinese test set with 11.01% WER.

Finally, we build the third CS CTC model with only 300 hours CS data. It performs the best on the CS test set with 25.48% WER, but works the worst on the Chinese test set with 21.92% WER. Clearly, this behavior is not desirable.

Since our goal is to improve CS performance and keep comparable performance on the Chinese test set, we will only present how to improve the first and second CS models in the following sections.

4.3. Language IDs

We trained both DNN and LSTM based LIDs. The DNN LID model has 6 hidden layers with 2048 nodes at each layer. We choose a context window with 41 frames for the DNN model. The bi-directional LSTM LID model has 6 hidden layers with 512 LSTM units in each direction at each layer. Based on the model type and whether CS data is used for training, we have 6 LID models: two types for scenarios with only monolingual data (ZH+EN), merged monolingual and CS data (ZH+EN+CS), and only CS data (CS).

We first evaluate the performance of the six LID models. The frame level accuracies on the CS test set are given in Table 2. The overall accuracy (acc.) is evaluated for all the switches (between ZH, EN, and silence) while the switching acc. is only evaluated at frames where a switch between ZH and EN happens in Table 2. Results in Table 2 show that with CS training data, both DNN and LSTM based

Table 2: Frame-level accuracy (%) of various LIDs on the CS test set. Overall acc. is evaluated for all switches (between ZH, EN, and silence) while the switching acc. is only evaluated at frames where a switch between ZH and EN happens.

ID	Model	Training data	Overall acc.	Switching acc.
LID1	DNN	ZH+EN	68.9	40.1
LID2	LSTM	ZH+EN	64.6	37.5
LID3	DNN	ZH+EN+CS	76.7	67.0
LID4	LSTM	ZH+EN+CS	86.2	68.5
LID5	DNN	CS	88.5	78.7
LID6	LSTM	CS	94.4	84.0

Table 3: WERs (%) of a CTC model adjusted by LIDs with $\alpha = 1$. Both CTC and LID models are trained with monolingual data only without observing CS data.

Testset	no LID	ZH+EN	
		+LID1	+LID2
CS set	59.08	**55.38**	58.11
ZH set	11.04	11.26	**11.11**

LIDs perform better than those trained without CS data. We can also observe that for scenarios without CS training data, DNN based LIDs perform better than LSTM based ones. Without seeing any CS data during training, the LSTM model cannot easily switch for CS utterances because of its memory learning doesn't see such patterns. In contrast, the frame-by-frame processing of the DNN model make it easier to switch between languages. While for scenarios with CS training data, LSTM based LIDs outperform DNN based ones in general. This is because the powerful LSTM model now observed CS patterns during training.

4.4. Results of CTC and LID models trained with monolingual data

We then evaluate the performance of adjusting the baseline CTC model via different LIDs. Table 3 shows the WERs of the CTC model adjusted by two LIDs on the CS and Chinese test sets. Both CTC and LIDs are trained with monolingual ZH and EN data. "LID1" is a DNN model and "LID2" is a LSTM model. α in Eq. 3 is set to default value 1.

When combining with the LID1 model, we can reduce the WER on the CS test set from 59.08% to 55.38%, which stands for 6.3% relative WER reduction, while the degradation on the Chinese test is controlled within 2% relative. Combining with the LID2 model, there is almost no degradation on the Chinese test with 1.6% relative WER reduction on the CS test set. The improvement on the CS test set without seeing any CS data during training shows the effectiveness of the proposed approach.

4.5. Results of CTC and LID models trained with monolingual and CS data

Table 4 shows the WERs of a CTC model (trained with both monolingual and CS data) adjusted by four LIDs on the CS and Chinese test sets with $\alpha = 1$. "LID3" and "LID4" are DNN and LSTM based LIDs trained with monolingual plus CS data. "LID5" and "LID6" are DNN and LSTM based LIDs trained with only CS data. Combining with the LID3 model reduces the WER on the CS test set from 30.81% to 29.52%, which stands for 4.2% relative WER reduction, but degrades the Chinese test set with 3.7% relative. Combining with the LID4 model can control the loss on the Chinese set within 1%, with also around 1% relative improvement on the CS test set. Given the LID6 model is trained with only CS data, it has the best LID accuracy on the CS test set, resulting in the best combination WER as 28.06%. However, this also brings the WER on the Chinese set up to 12.23%. Hence, simply improving the performance on the CS test sets without caring about the major language is not desirable.

Table 4: WERs (%) of a CTC model (trained with CS data) adjusted by LIDs with $\alpha = 1$.

Testset	no LID	with CS		only CS	
		+LID3	+LID4	+LID5	+LID6
CS set	30.81	29.52	30.53	30.08	**28.06**
ZH set	11.01	11.42	**11.14**	13.90	12.23

5. CONCLUSION AND FUTURE WORK

In this study we propose a LID based approach for CS ASR in E2E CTC models. We intend to improve the performance by switching from the challenging problem of recognizing large amount of ASR units from both languages to an easier task of recognizing languages. This approach separately trains a E2E CTC model and a LID and directly adjusts the posteriors of the E2E CTC model with the posteriors of the LID. We applied this method to improve the CS performance on the Microsoft Chinese Cortana data by using totally 7000 hours monolingual Chinese and English data and 300 hours CS data. Using only monolingual data without observing any CS data during training, the proposed method can get up to 6.3% relative WER reduction. If CS data is observed during training, the proposed method still can get up to 4.2% relative improvement. For both scenarios, the performance on the Chinese test set is controlled to be similar to the baseline performance.

In the future, we plan to further improve the LID with weighted decisions among frames. We also plan to investigate joint training approach and apply the proposed method to other E2E models.

6. REFERENCES

[1] Y. Miao and et al., "EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding," in *ASRU*, 2015.

[2] W. Chan and et al., "Listen, attend and spell," *CoRR*, vol. abs/1508.01211, 2015.

[3] H. Soltau and et al., "Neural speech recognizer: Acoustic-to-word LSTM model for large vocabulary speech recognition," 2017.

[4] R. Prabhavalkar and et al., "A comparison of sequence-to-sequence models for speech recognition," in *Interspeech*, 2017.

[5] E. Battenberg and et al., "Exploring neural transducers for end-to-end speech recognition," in *ASRU*, 2017.

[6] Hasim Sak and et al., “Recurrent neural aligner: An encoder-decoder neural network model for sequence to sequence mapping,” in *Interspeech*, 2017.

[7] Chung-Cheng Chiu and et al., “State-of-the-art speech recognition with sequence-to-sequence models,” in *ICASSP*, 2018.

[8] Tara N Sainath and et al., “Improving the performance of online neural transducer models,” in *ICASSP*, 2018.

[9] J. Li and et al., “Advancing acoustic-to-word CTC model,” in *ICASSP*, 2018.

[10] Kartik Audhkhasi and et al., “Building competitive direct acoustics-to-word models for english conversational speech recognition,” in *ICASSP*, 2018.

[11] A. Graves and et al., “Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks,” in *ICML*, 2006, pp. 369–376.

[12] A. Graves and et al., “Towards end-to-end speech recognition with recurrent neural networks,” in *PMLR*, 2014, pp. 1764–1772.

[13] K. Cho and et al., “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” in *EMNLP*, 2014.

[14] D. Bahdanau and et al., “Neural machine translation by jointly learning to align and translate,” in *ICLR*, 2015.

[15] D. Bahdanau and et al., “End-to-end attention-based large vocabulary speech recognition,” *CoRR*, vol. abs/1508.04395, 2015.

[16] J. Chorowski and et al., “Attention-based models for speech recognition,” in *NIPS*, 2015.

[17] A. Graves, “Sequence transduction with recurrent neural networks,” *CoRR*, vol. abs/1211.3711, 2012.

[18] H. Sak and et al., “Fast and accurate recurrent neural network acoustic models for speech recognition,” in *Interspeech*, 2015.

[19] J. Li and et al., “Acoustic-to-word model without OOV,” in *ASRU*, 2017.

[20] A. Das and et al., “Advancing connectionist temporal classification with attention modeling,” in *ICASSP*, 2018.

[21] Kanishka Rao and et al., “Exploring architectures, data and units for streaming end-to-end speech recognition with RNN-transducer,” in *ASRU*, 2017.

[22] Amit Das and et al., “Advancing Acoustic-to-Word CTC Model with Attention and Mixed-Units,” *arXiv e-prints*, p. arXiv:1812.11928, Dec 2018.

[23] Suyoun Kim and et al., “Towards language-universal end-to-end speech recognition,” in *ICASSP*, 2018.

[24] Shubham Toshniwal and et al., “Multilingual speech recognition with a single end-to-end model,” in *ICASSP*, 2018.

[25] Hiroshi Seki and et al., “An end-to-end language-tracking speech recognizer for mixed-language speech,” in *ICASSP*, 2018.

[26] David CS Li, “Cantonese-english code-switching research in hong kong: A y2k review,” *World Englishes*, vol. 19, no. 3, pp. 305–322, 2000.

[27] Dau-Cheng Lyu and et al., “An analysis of a mandarin-english code-switching speech corpus: Seame,” *Age*, vol. 21, pp. 25–8, 2010.

[28] Alfredo Ardila, “Spanglish: an anglicized spanish dialect,” *Hispanic Journal of Behavioral Sciences*, vol. 27, no. 1, pp. 60–81, 2005.

[29] Anik Dey and et al., “A hindi-english code-switching corpus.,” in *LREC*, 2014, pp. 2410–2413.

[30] Emre Yilmaz and et al., “Investigating bilingual deep neural networks for automatic speech recognition of code-switching frisian speech,” in *SLTU*, 2016.

[31] Dau-Cheng Lyu and et al., “Speech recognition on code-switching among the chinese dialects,” in *ICASSP*, 2006.

[32] Tetyana Lyudovyk and et al., “Code-switching speech recognition for closely related languages,” in *SLTU*, 2014.

[33] Ganji Sreeram and et al., “A novel approach for effective recognition of the code-switched data on monolingual language model,” in *Interspeech*, 2018.

[34] Jochen Weiner and et al., “Integration of language identification into a recognition system for spoken conversations containing code-switches,” in *SLTU*, 2012.

[35] Emre Yılmaz and et al., “Acoustic and textual data augmentation for improved asr of code-switching speech,” in *Interspeech*, 2018.

[36] Joyce YC Chan and et al., “Detection of language boundary in code-switching utterances by bi-phone probabilities.,” in *ISCSLP*, 2004, pp. 293–296.

[37] Marc A Zissman, “Comparison of four approaches to automatic language identification of telephone speech,” *IEEE Transactions on speech and audio processing*, vol. 4, no. 1, pp. 31, 1996.

[38] Koena Ronny Mabokela and et al., “Modeling code-switching speech on under-resourced languages for language identification,” in *SLTU*, 2014.

[39] Pengcheng Guo and et al., “Study of semi-supervised approaches to improving english-mandarin code-switching speech recognition,” in *Interspeech*, 2018.

[40] Sahoko Nakayama and et al., “Speech chain for semi-supervised learning of japanese-english code-switching asr and tts,” in *SLT*, 2018.

[41] Xuesong Yang and et al., “Joint modeling of accents and acoustics for multi-accent speech recognition,” in *ICASSP*, 2018.

[42] Houwei Cao and et al., “Semantics-based language modeling for cantonese-english code-mixing speech recognition,” in *ISCSLP*. IEEE, 2010, pp. 246–250.

[43] Zhiping Zeng and et al., “Improving n-gram language modeling for code-switching speech recognition,” in *APSIPA ASC*. IEEE, 2017, pp. 1596–1601.

[44] Jesse Emond and et al., “Transliteration based approaches to improve code-switched speech recognition performance,” in *SLT*, 2018.

[45] Haizhou Li and et al., “Spoken language recognition: from fundamentals to practice,” *Proceedings of the IEEE*, vol. 101, no. 5, pp. 1136–1159, 2013.

[46] Jui-Ting Huang and et al., “Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers,” in *ICASSP*, 2013.

[47] Andrew Senior and et al., “Acoustic modelling with CD-CTC-SMBR LSTM RNNs,” in *ASRU*, 2015.