Unsupervised Code-switched Text Generation from Parallel Text

Jie Chi[1†] *, Brian Lu*[2†]*, Jason Eisner*[2]*, Peter Bell*[1]*, Preethi Jyothi* [3]*, Ahmed M. Ali* [4]

[1] Centre for Speech Technology Research, University of Edinburgh
[2]Center for Language and Speech Processing, Johns Hopkins University
[3]Department of Computer Science, Indian Institute of Technology Bombay
[4]Qatar Computing Research Institute, HBKU, Qatar

jie.chi@ed.ac.uk, zlu39@jhu.edu

Abstract

There has been great interest in developing automatic speech recognition (ASR) systems that can handle code-switched (CS) speech to meet the needs of a growing bilingual population. However, existing datasets are limited in size. It is expensive and difficult to collect real transcribed spoken CS data due to the challenges of finding and identifying CS data in the wild. As a result, many attempts have been made to generate synthetic CS data. Existing methods either require the existence of CS data during training, or are driven by linguistic knowledge. We introduce a novel approach of forcing a multilingual MT system that was trained on non-CS data to generate CS translations. Comparing against two prior methods, we show that simply leveraging the shared representations of two languages (Mandarin and English) yields better CS text generation and, ultimately, better CS ASR.

Index Terms: code-switching, text generation, data augmentation, encoder-decoder, unsupervised learning

1. Introduction

Code-switching (CS) refers to a common phenomenon whereby speakers shift between languages during conversation, especially in multilingual areas, for example, in India and Singapore. Linguists categorize code-switched language according to where the switching occurs [1]: at sentence or clause boundaries (inter-sentential CS), within a sentence or clause (intra-sentential CS, also known as code-mixing), or by inserting a tag phrase (tag switching). There has been great interest in developing ASR systems for such settings to meet the demand of a growing bilingual population. However, the lack of CS data is a major hindrance to these efforts. This motivates techniques for generating synthetic code-switched sentences, which can be used to augment text [2] or speech [3] training data.

In contrast to existing methods of code-switched text generation that either rely on real code-switched training text or commit to specific linguistic theories [4], we propose a novel method. We pretrain a Transformer encoder-decoder model on parallel text without any real code-switched data, and then force the decoder to switch languages a given number of times. Our method simply leverages the shared representations induced by pretraining a multilingual translation model. Our experiments focus on intra-sentential CS and demonstrate improvements against two standard methods on the SEAME corpus [5].

† Equal contribution, part of this work was done during JSALT 2022 at JHU, with gift-funds from Amazon, Microsoft and Google. This work was also supported in part by the UKRI (EP/S022481/1). We thank Adithya Renduchintala for early discussions.

2. Related work

There have been many efforts to construct synthetic data to augment the small existing datasets of code-switched text. As parallel monolingual texts are far more plentiful (or can be created by machine translation), a popular research direction is to align parallel sentences and mix them under the guidance of linguistic theories of code-switching, such as Functional Head Constraints or Equivalence Constraints [6, 4, 7]. These rule-based methods can only extract and concatenate monolingual fragments from the parallel texts. Another line of work directly generates synthetic code-switched sentences from a language model or conditional language model that has been trained using a small amount of code-switched data. This work variously uses RNN models [8, 9, 10, 11], Pointer-Generator networks [2], GANs [12, 13, 14, 15], and VAEs [16].

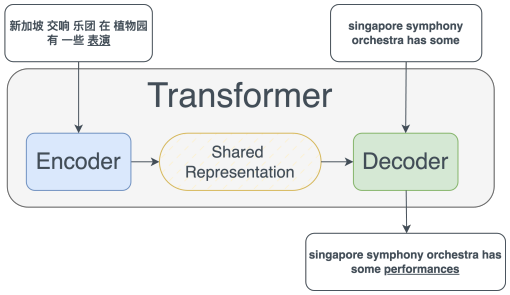
Similar to the linguistic rule-based approaches, our proposed method requires only parallel monolingual text. However, our method uses a sequence-to-sequence model, which enjoys the same flexibility as prior methods that take the conditional language modeling approach. When a small amount of code-switched text is available, we can use it to additionally fine-tune our model—which improves the quality of the generation to be on par with or better than other supervised methods.

3. Methodology

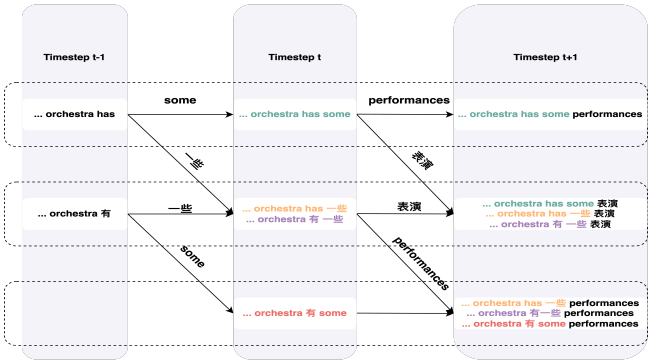
3.1. Parallel Text Pretraining

Word embeddings play an integral role in modern Speech and NLP models. Questions about the degree to which the embedding spaces of different languages share a similar structure have received much interest. Early work in the cross-lingual word embedding literature showed that separately learned, non-contextual word embeddings of different languages can be aligned via linear mappings [17, 18]. In the case of contextual word embeddings, similar alignment results (using more sophisticated mappings like centered kernel alignment) have been reported for separate monolingual BERTs as well as for a multilingual BERT [19, 20].

Would emergent alignments on word embeddings (contextual or not), learned simply from text prediction tasks on multiple languages, be enough to support code-switched generation among those languages? We study this question by pretraining a many-to-many machine translation system on monolingual inputs and outputs, and then forcing the decoder to translate monolingual inputs into code-switched outputs. Specifically, we train a Transformer encoder-decoder model [21] to translate between any pair of languages in {Mandarin, English}. We then translate monolingual sentences from either Mandarin or English to sentences that are forced to code-switch to varying degrees, via grid beam search [22]. Finally, we evaluate the



(a) *Transformer architecture, where for each parallel sentence pair* (S_{zh}, S_{en})*, we have four training examples* $(S_{zh}, S_{zh}), (S_{en}, S_{en}), (S_{zh}, S_{en}), (S_{en}, S_{zh})$*. Here we use* (S_{zh}, S_{en}) *for illustration*



(b) *Decoding process with grid beam search, where shaded boxes denote three subsets with 0, 1 and 2 code-switching points. Each box represents the top k hypotheses (not all hypotheses are shown in the figure) at each timestep in each subset. Colored text and edges show the expansion for each beam.*

Figure 1: *Model training and decoding*

utility of our synthetic CS sentences by using them to train a n-gram language model to use in a downstream ASR system.

3.2. Translation Model

We use a Transformer encoder-decoder architecture [23] (Figure 1a), with a vocabulary that is the disjoint union of the vocabularies of the two languages of interest.[1] These two vocabularies are harvested from the two sides of the parallel training corpus. We create 4 training examples for every pair of parallel sentences (x, y): each training example takes one of x or y as encoder input and one of x or y as decoder output. This is the same scheme used to train unified MT systems that translate between many language pairs using the same parameters [24], and indeed our method could be extended beyond 2 languages.

In a preliminary study, we found that it worked best to train the model with a modified version of softmax that normalizes over only words of the desired output language, as opposed to normalizing over the entire union vocabulary. The latter formulation penalizes assigning high logits to any word in the other language. That hinders the natural emergence of aligned word embedding spaces between the two languages, since it pushes away uniformly the embeddings of the other language.

3.3. Grid Beam Search

We follow [22] to constrain the decoding by the number of code-switching points. Different from regular beam search, the set of prefixes in the beam is partitioned into subsets according to the value of some feature (for us, the number of code-switching points). Each subset is separately pruned back to its top-k prefixes. Extending a prefix may change its feature value (for us, if the extension creates a new switching point), in which case the extension will fall into a different subset. At the end, the algorithm returns the top elements of each subset—in our case, the best outputs with 0, 1, 2, ... code-switching points. Figure 1b illustrates with a simplified example.

3.4. Other Approaches

To compare to prior methods for generating CS text [4, 2], we also implement models based on Equivalence Constraint Theory (ECT) and on Pointer-Generator Networks (PGN). Both of these models depend on parallel data. The PGN additionally requires CS training data, so it serves as a supervised baseline against which to compare our unsupervised method.

Equivalence Constraint Theory claims that code-switching can only happen at boundaries where both languages have the same surface structure. Following the pipeline in [4], we first use *fast_align* to obtain the word alignment between parallel sentences and then generate the parse tree for English text with the Berkeley neural parser [25]. In contrast to the EC baseline in [2], where they used a simplified linear version of EC that determines the acceptability of a substitution solely by checking whether there are crossing alignments, here we follow [4] and use the alignment together with the constituency parses to determine if a substitution is acceptable.

Pointer-Generator Networks require supervised training. The input is the *concatenation* of the parallel sentences x and y and the output is the desired code-switched sentence. We re-implement the model introduced by [2], except that we do not use part-of-speech tags as additional input features.

4. Experimental setup

4.1. ASR Framework

We begin by pretraining an acoustic model on the union of the training portions of **TED-LIUM 3** [26], an English speech corpus collected from TED talks, and **AISHELL-1** [27], a Mandarin speech corpus consisting of over 170 hours of speech. We then fine-tune these acoustic models on the *monolingual* utterances from **SEAME** [5], a Mandarin-English code-switching speech corpus collected in conversations and interviews from Malaysian and Singapore bilingual speakers. SEAME labels each utterance as English, Mandarin, or CS. As our acoustic model is not trained on any of the CS utterances, it is typical of those used in existing multilingual ASR systems.

We combine this hybrid acoustic model with a language model trained on CS text, and experiment with different ways of obtaining CS text. We compare the performance of the resulting ASR systems on the held-out portion of SEAME.[2]

4.2. Real CS Text

SEAME contains 50K CS utterances for training. We use their transcripts to train a LM on real data. Our goal in the next sec-

[1] For simplicity, strings that exist in both vocabularies, such as numbers, are given two separate embeddings, one for each language. This also facilitates the softmax modification described later in this section.

[2] Both the `dev_man` and `dev_sge` subsets are from `https://github.com/zengzp0912/SEAME-dev-set.git`. In total they contain 5384 monolingual and 6468 code-switched utterances.

tions is to synthesize ersatz data that works almost as well.

4.3. Parallel Non-CS Text

For each CS utterance z from SEAME, we also ask Google Translate to translate it to both English (x) and Mandarin (y), in each case treating the source sentence z as if it were in the other language.[3] This yields 50K parallel utterances (x, y, z).

4.4. Synthetic CS Data

For a controlled comparison, we take care to have all of our methods generate synthetic datasets of the same size.

For our proposed *unsupervised* Constrained Translation (**CT**) approach, we train a unified MT model (§3.2) on the 50K (x, y) pairs and then use grid beam search (§3.3) to decode 3 CS translations of each x and each y. Specifically, for each $c \in [1, 3]$, the final beam holds up to 5 prefixes with exactly c switching points, and we return the top 1 of those. That yields 6 sentences per pair, which we then randomly subsample to 3.

As our unsupervised baseline, we run **ECT** (§3.4) on the (x, y) pairs. Hybridizing each pair in all legal ways yields about 12 CS utterances on average, which we then subsample to 3.

To make use of SEAME *supervised* z data, we start with the unified MT model above, and fine-tune it to translate each of x and y to each of x, y, and z. That is, each SEAME utterance now yields 6 training examples instead of 4.[4] We refer to this model as **CST** (code-switched translation) and use it to retranslate each x and each y to 2 new CS utterances (totaling 4), which we then subsample to 3.

As our supervised baseline, we train **PGN** (§3.4) to translate from the concatenated input xy to the code-switched z, and use it to retranslate each xy pair to 3 new CS utterances.

Note that we used only the CS portion of SEAME, not the monolingual portion, to generate our synthetic CS utterances. This ensured a (unrealistically good) match with the topics and lengths of the held-out CS utterances.[5]

4.5. Model Architectures and Training

4.5.1. Translation Models

We use 8-layer, 12-head Transformer encoder-decoders with dimension size 768 for our **CT** and **CST** systems. For the **PGN** baseline, we implemented our own Pointer-Generator Network following [2] using a one-layer Bi-LSTM encoder and a one-layer LSTM decoder with hidden dimension 256.[6]

We tokenize all Mandarin parts of the data using JieBa[7] for pretraining our translation models, while we use character tokenization for our implementation of pointer-generator networks and the ASR language models. For English parts, we always tokenize at whitespace and punctuation.

4.5.2. Language Models

For each dataset described in §4.2, §4.3, and §4.4, we use the SRILM toolkit to train a trigram model with Kneser-Ney smoothing [29]. For each *generated* dataset in §4.3, and §4.4, we additionally train a trigram model by *combining* it with real code-switched data (§4.2) via LM interpolation. This leads to improvements in WER and PPL discussed in §5. Interpolation weights are optimized on the monolingual SEAME data, disjoint from both SEAME code-switched training and test set.

4.5.3. Acoustic Model

We use the Kaldi toolkit to train a hybrid acoustic model. AISHELL and TED-LIUM datasets are combined to train a standard speaker adaptive GMM-HMM model at first, then we use it to produce alignments to train a CNN-TDNN model with lattice-free maximum mutual information (LF-MMI) criterion, which consists of 6 CNN layers and 12 TDNN layers. We pretrain the acoustic model on AISHELL and TED-LIUM and then fine-tune it on SEAME monolingual. Although we have never used CS speech during acoustic training, by pretrainging on additional two monolingual speech corpora, the obtained acoustic model has already achieved a competitive result compared with models trained on the entire SEAME corpus in [10, 30].

The lexicon is obtained by combining the pronunciations of English words from CMU dictionary and Mandarin characters from AISHELL dictionary. We use different phoneme units for each language. Pronunciations for OOV words in the training data are generated by Phonetisaurus [31]. In the end, we have 180K entries in the lexicon and any uncovered words are treated as UNK.[8] Compared with training two monolingual models using the same architecture, the performance of the obtained bilingual ASR model only drops by 0.5 absolute word error rate.

5. Results and Discussion

5.1. ASR Results

Table 1 presents word error rate (WER) of our ASR system on SEAME test set. The ASR system uses the pretrained acoustic model with trigram LMs trained on synthetic CS text. We break down the test set further by whether there is any code-switching contained. The unsupervised (**CT**) and supervised (**CST**) versions of our proposed model respectively achieves better overall WER than the **ECT** and **PGN** baselines, regardless of LM interpolation with **RealCS**. Among unsupervised methods, **CT** consistently gets lower WER than **ECT** on both CS and monolingual utterances. Among supervised methods, **CST**'s superior performance compared to **PGN** on the monolingual subset and worse performance on the CS subset could be attributed to the multi-task regularization as well as its lack of the dual-language input and a copying mechanism which can make learning the alignment between the two languages easier.

5.2. Language Modeling Results

We also evaluate the various LMs on the text transcripts of SEAME test set, ignoring the audio. The perplexity (PPL) results are in Table 1, with a cross-entropy breakdown in Figure 2 (Left). As the plot shows, both **ZH** and **EN** tokens cause higher surprisal when the previous token is in the other language, but models that use more real CS data are less surprised.

Figure 2 (Mid) shows the percentage of the CS bi/trigrams contained in the test set that appear in the synthetic texts.[9] Fig-

[3] Google Translate may not be optimized to deal with code-switched inputs like z. As a result, its supposedly monolingual translations sometimes contain some code-switched content from z.

[4] This can be seen as multi-task regularization. Our actual goal is to learn to translate $x \mapsto z$ and $y \mapsto z$, but fine-tuning on those pairs alone would lead to low diversity in the beam search, which generates duplicate n-grams and prevents us from training a KN-discounted n-gram LM. Thus, we also include the other 4 pairs when fine-tuning.

[5] SEAME CS sentences differ significantly in length from monolingual ones. In general, longer sentences code-switch more likely [28].

[6] We also explored adding more parameters in a preliminary study but did not observe significant improvements in downstream ASR.

[7] `https://github.com/fxsjy/jieba.git`

[8] There are 117 OOV out of 151146 total word tokens on test data.

[9] A truly Non CS corpus would contain no code-switching bigrams and trigrams. However they exist in our Non CS dataset because our Non CS dataset is generated by (Google) translating code-switched sentences into monolingual ones, and Google Translate sometimes fails to produce a purely monolingual mandarin output, especially for interjec-

Table 1: *ASR WER and LM perplexity evaluations on SEAME dev sets. In each row, the overall best system is bolded and best systems within categories are underlined. When combining with RealCS, an optimal weight is selected following §4.5.2*

			Supervised		Unsupervised		
		RealCS (50K)	CST (150K)	PGN(150K)	CT(150K)	ECT(150K)	Non CS(100K)
Individual Datasets	WER_{all}	**31.32**	31.44	31.80	33.52	34.68	33.97
	WER_{cs}	**29.82**	30.58	30.50	32.80	33.73	33.44
	WER_{mono}	34.94	**33.78**	35.11	35.53	37.03	35.45
	PPL	**123.43**	128.86	136.60	161.21	183.77	153.57
Combined with RealCS	WER_{all}	/	**30.80**	30.90	30.92	31.06	31.18
	WER_{cs}	/	29.73	**29.48**	29.62	29.69	30.09
	WER_{mono}	/	**33.63**	34.38	34.21	34.46	34.07
	PPL	/	119.42	134.08	133.15	130.70	129.87

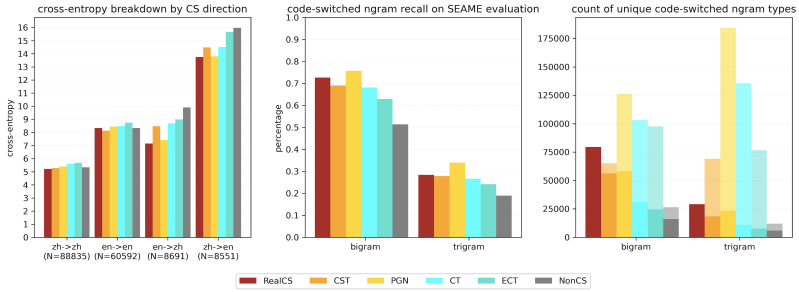


Figure 2: *Cross-entropy breakdown and n-gram coverage. Left: Cross-entropy breakdown by the transition of language IDs. Middle: Token-level code-switched bigram and trigram recall on the SEAME evaluation set. Right: Type-level code-switched bigram and trigram counts. Darker bars count the number of shared n-gram types between a particular dataset and the SEAME training data.*

ure 2 (Right) shows the number of unique CS bi/trigrams in the synthetic texts (lighter bars) as well as the number of those bi/trigrams that also appear in the real CS training data (darker bars).[10] Unsurprisingly, **RealCS** and supervised methods generate data with much better n-gram coverage. Under unsupervised training, **CT** can generate more diverse language transitions compared with **ECT**, as **ECT** is constrained to output segments that are in the original sentence pair. Under supervised settings, **PGN** generates more diverse n-grams and has better coverage of the test set.

5.3. Qualitative Properties of Synthetic CS Text

The code-switched sentences generated by our **CT** model are not always perfect translations of the input, but are they reasonable CS text? Most of the code-switching consists of lexical substitutions (e.g. "you go to take 营销(marketing) loh you are the 最好的(best)"). We find that the resulting sentences are mostly understandable, but errors occur (e.g. "it s fun to 火车(train, the noun) with them"), and they don't always code-switch in the same places that a bilingual speaker would (e.g. "my dad is the one who 给(give) him the 工作(job)"). Some sentences could code-switch *too often* because **CT** required them to do so.

This matches our intuition that our model should prefer switching at words that have good translations in the other language. Why? Compared to switching at words without good translations in the other language, switching at words *with* good translations creates sentences that are, in the embedding space, close to sentences those completely in English/Mandarin, which our model has been trained to generate during training.

5.4. Limitations

We only experimented with two languages in this work, but the framework could be generalized to generate text that code-switches among any number of languages. Although only hybrid ASR systems have been used in this paper, which generally perform better when limited data is available [30], it is interesting to investigate if the same finding still holds in an End-to-End framework. We plan to use the synthetic CS text for LM rescoring in an End-to-End framework as in [32] or LM decoding for RNN-T [33] in the future.

Compared with **ECT**, whose performance is constrained by the effectiveness of the tools for natural language processing, **CT** is fully data-driven, relying on learning shared representations of the language pair from translation pretraining. Hence, it is less sensitive to the formality of the data. SEAME contains mostly conversational speech and transcriptions, which is harder to parse and align, but on more formal domains where linguistic knowledge may help, **ECT** may become more competitive since it has a better inductive bias. **CT** is able to freely generate CS text, including content (such as paraphrases) not contained in either of the parallel sentences, which could improve diversity. On the other hand, the popular **ECT** approach is constrained not to do this, which prevents it from generating wildly incorrect outputs.

6. Conclusions

We presented a simple yet effective idea: to leverage the emergence of shared representations in pretrained encoder-decoder models to generate synthetic code-switched data without using any prior knowledge about code-switching. Although the data it generates does not outperform methods that use real CS as supervision, it performs slightly better than other unsupervised methods such as ECT, and without needing a parser or specialized knowledge about code-switching.

Showing the possibility of a fully data-driven, learning approach to unsupervised CS generation opens up opportunities for more research in the design of the model architectures and training objectives. While we explored a simple instantiation with Transformer encoder-decoders and just the translation objective, more specialized architectures could lead to better representation sharing and in turn better CS generation.

tion words such as *lor*, *ah* and *er*.

[10] Recall that the training data of the various synthetic generations methods were derived from SEAME CS training set.

7. References

[1] *The Cambridge Handbook of Linguistic Code-switching*, ser. Cambridge Handbooks in Language and Linguistics. Cambridge University Press, 2009.

[2] G. I. Winata, A. Madotto, C. Wu, and P. Fung, "Code-switched language models using neural based synthetic data from parallel sentences," *CoRR*, vol. abs/1909.08582, 2019.

[3] C. Du, H. Li, Y. Lu, L. Wang, and Y. Qian, "Data augmentation for end-to-end code-switching speech recognition," in *2021 IEEE Spoken Language Technology Workshop (SLT)*, 2021, pp. 194–200.

[4] A. Pratapa, G. Bhat, M. Choudhury, S. Sitaram, S. Dandapat, and K. Bali, "Language modeling for code-mixing: The role of linguistic theory based synthetic data," in *Proceedings of ACL*, Jul. 2018, pp. 1543–1553.

[5] D.-C. Lyu, T. P. Tan, C. E. Siong, and H. Li, "Seame: A Mandarin-English code-switching speech corpus in South-East Asia," in *INTERSPEECH*, 2010.

[6] Y. Li and P. Fung, "Code switch language modeling with functional head constraint," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 4913–4917.

[7] A. Hussein, S. Absar Chowdhury, A. Abdelali, N. Dehak, and A. Ali, "Code-switching text augmentation for multilingual speech processing," 2022.

[8] H. Adel, N. T. Vu, and T. Schultz, "Combination of recurrent neural networks and factored language models for code-switching language modeling," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Aug. 2013, pp. 206–211. [Online]. Available: https://aclanthology.org/P13-2037

[9] E. Yılmaz, H. van den Heuvel, and D. van Leeuwen, "Acoustic and Textual Data Augmentation for Improved ASR of Code-Switching Speech," in *Proc. Interspeech 2018*, 2018, pp. 1933–1937.

[10] G. I. Winata, A. Madotto, C.-S. Wu, and P. Fung, "Code-switching language modeling using syntax-aware multi-task learning," in *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*, Jul. 2018, pp. 62–67. [Online]. Available: https://aclanthology.org/W18-3207

[11] K. Chandu, T. Manzini, S. Singh, and A. W. Black, "Language informed modeling of code-switched text," in *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*, Jul. 2018, pp. 92–97. [Online]. Available: https://aclanthology.org/W18-3211

[12] C.-T. Chang, S.-P. Chuang, and H. yi Lee, "Code-switching sentence generation by generative adversarial networks and its application to data augmentation," in *Interspeech*, 2018.

[13] Y. Gao, J. Feng, Y. Liu, L. Hou, X. Pan, and Y. Ma, "Code-Switching Sentence Generation by Bert and Generative Adversarial Networks," in *Proc. Interspeech 2019*, 2019, pp. 3525–3529.

[14] B. Samanta, S. Reddy, H. Jagirdar, N. Ganguly, and S. Chakrabarti, "A Deep Generative Model for Code-Switched Text," Jun. 2019.

[15] K. Raghavi Chandu and A. W. Black, "Style Variation as a Vantage Point for Code-Switching," May 2020.

[16] B. Samanta, S. Reddy, H. Jagirdar, N. Ganguly, and S. Chakrabarti, "A deep generative model for code-switched text," *arXiv e-prints*, Jun. 2019.

[17] T. Mikolov, Q. V. Le, and I. Sutskever, "Exploiting Similarities among Languages for Machine Translation," *arXiv e-prints*, Sep. 2013.

[18] S. Ruder, I. Vulić, and A. Søgaard, "A Survey Of Cross-lingual Word Embedding Models," *arXiv e-prints*, Jun. 2017.

[19] A. Conneau, S. Wu, H. Li, L. Zettlemoyer, and V. Stoyanov, "Emerging cross-lingual structure in pretrained language models," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Jul. 2020, pp. 6022–6034. [Online]. Available: https://aclanthology.org/2020.acl-main.536

[20] B. Muller, Y. Elazar, B. Sagot, and D. Seddah, "First align, then predict: Understanding the cross-lingual ability of multilingual BERT," *arXiv e-prints*, Jan. 2021.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention Is All You Need," *arXiv e-prints*, p. arXiv:1706.03762, Jun. 2017.

[22] C. Hokamp and Q. Liu, "Lexically constrained decoding for sequence generation using grid beam search," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Jul. 2017, pp. 1535–1546. [Online]. Available: https://aclanthology.org/P17-1141

[23] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[24] M. Johnson, M. Schuster, Q. V. Le, M. Krikun, Y. Wu, Z. Chen, N. Thorat, F. Viégas, M. Wattenberg, G. Corrado, M. Hughes, and J. Dean, "Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 339–351, 10 2017.

[25] N. Kitaev and D. Klein, "Constituency parsing with a self-attentive encoder," *CoRR*, vol. abs/1805.01052, 2018.

[26] F. Hernandez, V. Nguyen, S. Ghannay, N. Tomashenko, and Y. Estève, "TED-LIUM 3: Twice as much data and corpus repartition for experiments on speaker adaptation," in *Speech and Computer*. Springer International Publishing, 2018, pp. 198–208. [Online]. Available: https://doi.org/10.1007%2F978-3-319-99579-3_21

[27] H. Bu, J. Du, X. Na, B. Wu, and H. Zheng, "AISHELL-1: An open-source Mandarin speech corpus and a speech recognition baseline," in *Oriental COCOSDA 2017*, 2017, submitted.

[28] J. Calvillo, L. Fang, J. Cole, and D. Reitter, "Surprisal predicts code-switching in Chinese-English bilingual text," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Nov. 2020.

[29] H. Ney, U. Essen, and R. Kneser, "On structuring probabilistic dependences in stochastic language modelling," *Comput. Speech Lang.*, vol. 8, pp. 1–38, 1994.

[30] Z. Zeng, Y. Khassanov, V. T. Pham, H. Xu, E. S. Chng, and H. Li, "On the End-to-End Solution to Mandarin-English Code-Switching Speech Recognition," in *Proc. Interspeech 2019*, 2019, pp. 2165–2169. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2019-1429

[31] J. R. NOVAK, N. MINEMATSU, and K. HIROSE, "Phonetisaurus: Exploring grapheme-to-phoneme conversion with joint n-gram models in the wfst framework," *Natural Language Engineering*, vol. 22, no. 6, p. 907–938, 2016.

[32] S. A. Chowdhury, A. Hussein, A. Abdelali, and A. Ali, "Towards one model to rule all: Multilingual strategy for dialectal code-switching arabic asr," in *Interspeech*, 2021.

[33] S. Thomas, B. Kingsbury, G. Saon, and H.-K. J. Kuo, "Integrating text inputs for training and adapting rnn transducer asr models," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 8127–8131.