INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

Improving Under-Resourced Code-Switched Speech Recognition: Large Pre-trained Models or Architectural Interventions

Joshua Jansen van Vüren[1], *Thomas Niesler*[1]

[1]Stellenbosch University, Stellenbosch, South Africa

jjvanvueren@sun.ac.za, trn@sun.ac.za

Abstract

We present three approaches to improve language modelling of under-resourced code-switched speech. First, we challenge the practice of fine-tuning large pre-trained language models on small datasets. Secondly, we investigate the advantages of sub-word encodings for our multilingual code-switched speech. Thirdly, we propose an architectural innovation to the RNN language model that is specifically designed for code-switched text. We show a clear reduction in absolute word error rate of 0.17% for the adapted LSTM language model compared to M-BERT when employed in n-best rescoring experiments. Further, the LSTM models afford a seven-fold reduction in total number of parameters and reduces runtime during rescoring 100-fold. Contrary to recent research trends, our LSTM models do not outperform the word-level models when using sub-word vocabularies. Finally, the new architectural mechanism applied to the LSTM improves language prediction for a span of several words following a code-switch.

Index Terms: speech recognition, under-resourced, code-switched, n-best rescoring, African languages

1. Introduction

The adaptation of large language models trained in well-resourced settings to under-resourced settings has become a commonly applied strategy in research and industry [1, 2, 3]. However, it is not immediately apparent that this is the best strategy for severely under-resourced code-switched spontaneous speech, as the domain of the training data for the well-resourced models differs drastically from the target domain.

We therefore challenge this fine-tuning approach by designing an LSTM-based language model which explicitly models the language of the next token, inspired by [4]. We refer to this architecture as the code-predictive LSTM. We show that this model achieves a performance competitive with M-BERT [1] in n-best rescoring experiments, while being up to seven times smaller in number of total parameters, and affording a 100-fold reduction in execution time.

We then present an architectural adaption to the code-predictive LSTM (CP-LSTM) model which we show improves language prediction for several tokens following a code-switch, while reducing the computational complexity of the model. Finally, we investigate various adaptations to the token modelling strategy, which allows the inclusion of more training data from related languages.

Our experiments are conducted on a under-resourced corpus of spontaneous code-switched speech in five African languages (isiZulu, isiXhosa, Sesotho, Setswana, Sepedi) and English. The corpus is split into five bilingual sets, of which the least resourced contains only ≈10 minutes of training speech, whilst the largest contains ≈22 hours.

2. Related work

Transfer learning or fine-tuning large language models to under-resourced datasets has become a focus of much recent research [5, 6, 7, 8, 9]. In [2], transfer learning is applied to adapt a multilingual BERT to Afrikaans, using a sub-word encoded vocabulary trained on a corpus of Afrikaans text. BERT embeddings are utilised for the existing or overlapping tokens in the Afrikaans vocabulary, while all new tokens are randomly initialised. This approach was shown to outperform classical fine-tuning.

Although transfer learning is common, some research has found it not always to be the best training methodology. For several under-resourced African languages, a newly-initialised BERT architecture achieved a performance that is competitive with M-BERT utilising only 100 million tokens of training data [3] (M-BERT is trained on 13 billion tokens). Additionally, [10] found that heavily regularised LSTM models can outperform transformer models for several other African languages.

When speech contains code-switching, an alternative approach to alleviate data scarcity in under-resourced language modelling is the application of data-augmentation. This has been applied with varied success by several authors in English-Mandarin, English-Hindi, and Dutch-Frisian code-switching [11, 12, 13, 14, 15, 16, 17, 18]. Other work concerning code-switched language modelling has considered adaptations to the model architecture or to the objective function in order to include some linguistic information and thereby improve language modelling or speech recognition [19, 20, 21, 22]. Additionally, morphologically-premised algorithms have been utilised to adapt vocabularies and thereby improve language model performance in African languages [23].

3. Dataset

The dataset utilised in this work comprises spontaneous code-switched speech in six languages[1] (isiZulu, isiXhosa, Sesotho, Setswana, Sepedi, and English). The corpus statistics are presented in Table 1 and is based on the corpus presented in [24][2], including some additional speech that primarily enlarges the test sets and adds a small amount of English-Sepedi data. As seen in the table, the dataset is split into five bilingual sub-corpora.

[1]isiZulu, isiXhosa, Sesotho, Sepedi, and Setswana all form part of the Bantu language family.

[2]Available from: https://repo.sadilar.org/handle/20.500.12185/545

10.21437/Interspeech.2023-1841

Table 1: *Soap opera dataset which is split into five bilingual pairs: English-isiZulu (EZ), English-isiXhosa (EX), English-Sesotho (ES), English-Setswana (ET), and English-Sepedi (EN).*

Pair	Partition	Speakers	Tokens	Types	Duration
English-isiZulu (EZ)	Train	147	254080	34809	21.75h
	Dev	12	3236	1318	0.26h
	Test	16	10760	3977	0.85h
English-isiXhosa (EX)	Train	35	15822	5738	1.51h
	Dev	12	3151	1594	0.26h
	Test	16	6761	3006	0.59h
English-Sesotho (ES)	Train	55	79472	7141	5.01h
	Dev	12	4589	1354	0.29h
	Test	16	13837	2302	0.83h
English-Setswana (ET)	Train	55	65047	6915	4.23h
	Dev	12	4041	1271	0.25h
	Test	16	14173	2403	0.84h
English-Sepedi (EN)	Train	9	2495	811	0.16h
	Dev	1	19	15	0.001h
	Test	7	4191	1194	0.26h
Pooled (ALL)	Train	245	468563	53173	36.41h
	Dev	19	15707	4603	1.1h
	Test	43	60649	12204	4.05h

4. Experimental setup

4.1. Baseline acoustic model

We utilise a CNN-TDNN-F Kaldi [25] acoustic model to generate the n-best hypotheses for rescoring. In heavily under-resourced settings such as ours, we have found this recipe remains competitive with state-of-the-art end-to-end models. However, the fusion of language models we propose here with end-to-end speech recognition systems is a subject of our ongoing work. The acoustic model is trained in two phases. Firstly, on the pooled set in Table 1, and then fine-tuning on each of the five sub-corpora, resulting in five language-pair dependent models [26]. Previous work has shown that language dependent phones as well as language dependent n-gram language models produce the best speech recognition performance.

4.2. Language models

All neural language models investigated in this work are applied to rescore 50-best lists by generating sequence level log-likelihoods. These scores are interpolated with the corresponding 50-best scores produced by 3-gram language models using unmodified Kneser-Ney smoothing and trained using the SRILM toolkit [27]. The interpolation weight between the scores of the n-gram language model and the LSTM language model is selected to optimise the development set word error rate. In order to validate that improvements in performance are consistent, we retrain each neural language model ten times with different random seeds and report the mean of the final speech recognition results.

We employ an LSTM model inspired by the architecture CP-LSTM-$\mathbb{E}$ proposed in [4], which we refer to in this paper as CP-LSTM. This model has been shown to outperform a classical LSTM in code-switched language modelling. The architecture comprises two monolingual LSTM language models which each produce likelihood vectors for the next word token ($\boldsymbol{o}_E^{(i)}$,$\boldsymbol{o}_B^{(i)}$), these likelihood vectors are interpolated using a language prediction produced by another LSTM model ($\hat{l}^{(i)}$). The model uses 128-dimensional embedding and 256-dimensional hidden state vectors. Additionally, L2 regularisation is applied to the parameters during training and layer normalisation [28] is applied to the LSTM hidden state. Because our dataset is heavily under-resourced, the language models quickly overfit on the training data, therefore training is halted when the lowest development set loss is achieved. We utilise the Adam [29] optimizer with a learning rate of $1 \cdot 10^{-3}$ and a batch size of 32.

4.3. M-BERT topline language model

We use a large language model (M-BERT) as a topline in our experiments. This model is finetuned for 10 epochs on the pooled set without any adaptation to the sub-word vocabulary. The rescoring process utilises a strategy similar to the masked language model (MLM) objective by individually masking each token in the input sequence to compute a score for the corresponding utterance in question, as described in [30].

4.4. Training set pooling

Our initial experiments begin by attempting to leverage the data from all of the bilingual sub-corpora (Table 1) rather than using only a single bilingual set. The CP-LSTM language model described in Section 4.2 was originally developed to use a word level language-dependent vocabulary. In order to effectively leverage training data from similar languages within the same language group[3] we require a language independent vocabulary, which does not distinguish between identical word tokens across languages.

Two different pooling strategies are investigated. Firstly, two training pools are created by including the training data within the two language families present in our data (Nguni, Sotho-Tswana). Secondly, we include all available training data (Pooled), as shown in Table 1. This training pool contains additional monolingual speech as well as code-switched speech in two, three or even four languages, which is not present in the bilingual sets.

4.5. Sub-word encoding

Most modern large language models utilise sub-word encoded vocabularies [1, 31]. Therefore, we investigate whether a Byte-Pair encoded [32] vocabulary is able to improve language model performance since it mitigates out-of-vocabulary (OOV) instances. OOV tokens are a common occurrence in our under-resourced dataset, 8.63% and 9.41% on the pooled development and test sets respectively. Sub-word vocabulary sizes are defined in logarithmic steps: 2000, 4000, 8000, 16000, and 32000. We note that the pooled training set contains 53k words types (Table 1).

4.6. Improved code-predictive LSTM Architecture

The primary contribution of this work is the extension of the CP-LSTM architecture presented in [4] by applying two adaptations. The adapted architecture is illustrated in Figure 1, and is denoted as CP-LSTM-2. For each training batch, the code-predictive LSTM is trained in two steps. First, gradient descent is performed after backpropagating the gradients associated with the logit vector and ground truth target vector $\mathcal{L}(\boldsymbol{o}^{(i)}, \boldsymbol{v}^{(i)})$. Second, gradient descent is performed after back-

[3]Specifically in our case isiZulu and isiXhosa belong to the Nguni language family, while Sesotho, Setswana, and Sepedi belong to the Sotho-Tswana language family.

Table 2: *Test set speech recognition (WER) for the five bilingual language pairs. FAM: Training sets pooled in two family groups (Nguni and Sotho-Tswana). ALL: Pooled training set.*

Model	EZ	EX	ES	ET	EN	**Overall**
Baseline	47.2	56.4	43.9	46.6	59.2	50.6
CP-LSTM	46.6	56.8	43.1	45.4	61.1	50.6
CP-LSTM-FAM	46.8	55.8	43.0	44.8	60.0	50.1
CP-LSTM-ALL	46.5	**55.6**	42.8	44.7	59.2	49.8
CP-LSTM-2	**46.2**	**55.6**	42.8	44.5	**58.8**	**49.6**
M-BERT	46.4	55.8	**42.7**	**44.2**	59.6	49.8

propagating the gradients associated with the language prediction loss $\mathcal{L}(\hat{l}^{(i)}, l^{(i)})$. Hence both language modelling and language prediction are explicitly trained.

The first adaptation extends the aforementioned CP-LSTM by removing the LSTM associated with the generation of the language prediction scalar $\hat{l}^{(i)}$, and instead includes a dense layer at the output of each language-specific LSTM (LSTM_E and LSTM_B) to generate a confidence score for the language of the next token - $\hat{l}_\text{E}^{(i)}$ and $\hat{l}_\text{B}^{(i)}$ respectively.

As a consequence of this adaptation, there is no longer a single language prediction signal $\hat{l}^{(i)}$, which was previously used to select the LSTM (LSTM_E or LSTM_B) whose updated hidden and cell state vectors ($\boldsymbol{h}_\text{E}^{(i)}$,$\boldsymbol{c}_\text{E}^{(i)}$ or $\boldsymbol{h}_\text{B}^{(i)}$,$\boldsymbol{c}_\text{B}^{(i)}$) would be passed to the next timestep. The second adaptation is to therefore maintain separate state vectors for each LSTM ($\boldsymbol{h}_\text{E}^{(i)}$,$\boldsymbol{c}_\text{E}^{(i)}$ and $\boldsymbol{h}_\text{B}^{(i)}$,$\boldsymbol{c}_\text{B}^{(i)}$) throughout the utterance, thereby avoiding the hard decision previously made using $\hat{l}^{(i)}$. Thus, the language predictions no longer influence the state vectors but are instead only used to mask tokens from the logit vector of a specific language ($\boldsymbol{o}_E^{(i)}$ and $\boldsymbol{o}_B^{(i)}$).

These two architectural changes remove the need for the LSTM that explicitly modeled language prediction in the original CP-LSTM. This leads to a reduction on the total number of weights in the language model, and an associated reduction in computational complexity. Because parallel state vectors for both languages are maintained, rather than a single state vector that changes at language switches, we hypothesize that this model may be better able to recover from language prediction errors. Additionally, because the language prediction is only used to mask the output probabilities for each respective model, in cases where both languages are likely, this model may produce more favourable predictions.

5. Results

5.1. Language independence and training set pooling

Table 2 presents speech recognition results for the training set pooling experiments described in Section 4.4. In a preliminary investigation, not reported in the table, it was found that allowing language independence does not substantially deteriorate speech recognition performance. On average, the absolute word error rate for the development set increased by 0.13% when moving from a language-dependent to a language-independent model. This small regression is far outweighed by the benefit of being able to include data from the the other bilingual sets, which then improves the performance of the CP-LSTM-ALL model by 0.87% absolute compared to the baseline on average over the five language pairs.

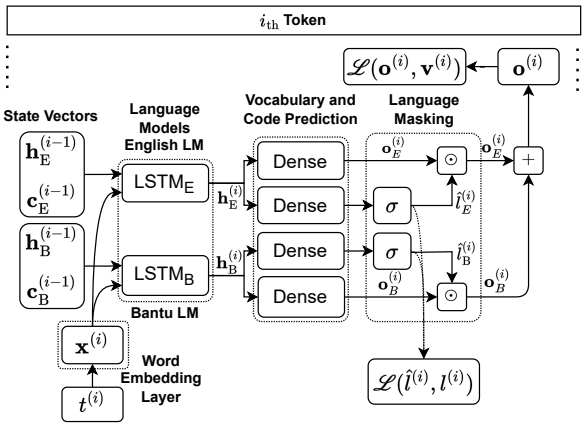

Figure 1: *The adapted code-predictive language model architecture as described in Section 4.6. Dense embedding vectors ($\boldsymbol{x}^{(i)}$) of word tokens ($t^{(i)}$) are fed as input to two LSTMs. Separate hidden and cell state vectors ($\boldsymbol{h}^{(i)}$,$\boldsymbol{c}^{(i)}$) are maintained for each LSTM language model (LSTM_E and LSTM_B). The output hidden states from each language model ($\boldsymbol{h}_E^{(i)}$,$\boldsymbol{h}_B^{(i)}$) are passed to a dense layer to generate scores $\hat{l}_E^{(i)}$ and $\hat{l}_B^{(i)}$ for the language of the next token where ($0 \leq \hat{l}^{(i)} \leq 1$). These scalars are utilised to mask token probabilities from vectors in the same language ($\boldsymbol{o}_E^{(i)}$,$\boldsymbol{o}_B^{(i)}$) from the two language models. $\odot$ denotes an element-wise multiplication, $l^{(i)}$ is the ground truth language, and $\boldsymbol{v}^{(i)}$ is the ground truth target vector.*

Interestingly, performance is better for all language pairs when including the data in all five languages (ALL) rather than only including data from the same language family (FAM). This result is encouraging for under-resourced speech recognition as it shows that performance can be improved by leveraging text data from more than only closely related languages. Additionally, this training strategy has the benefit of leading to a single unified model for all five bilingual pairs, and not five different models.

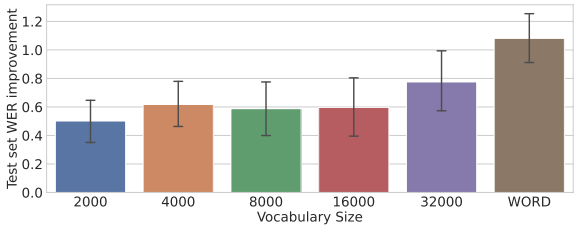

Figure 2: *Average absolute test set speech recognition improvement compared to baseline for the CP-LSTM-ALL model over four larger bilingual sub-corpora.*

5.2. Sub-word encoding

Figure 2 presents the average absolute test set speech recognition performance improvement (compared to the baseline) achieved after training and applying the CP-LSTM-ALL model in n-best rescoring with several sub-word encoded vocabularies of different sizes. From the figure, it is clear that sub-word encoded vocabularies do not improve the performance of the CP-LSTM-ALL model compared to the word level vocabulary.

This is an interesting and unexpected result, since other researchers have found sub-word encoded vocabularies to offer superior performance compared to word based methods [10, 23]. We leave more investigation into the optimisation of sub-word encodings for future work.

5.3. Improved code-predictive LSTM

Table 2, shows that, compared to the CP-LSTM-ALL baseline, CP-LSTM-2 improves absolute speech recognition for English-isiZulu, English-Setswana and English-Sepedi and matches performance for the remaining two language pairs. CP-LSTM-2 also outperforms the much larger BERT model by 0.85% and 0.17% absolute on the development and test sets respectively on average over the five bilingual sets. In fact, for English-Sepedi, a heavily under-represented partition, BERT led to a regression in speech recognition accuracy (0.4% absolute) while CP-LSTM-2 improved on all other candidates by 0.4% absolute.

Therefore, we conclude, that although the BERT model has been trained on substantially more data from over 100 languages, and can leverage context from the entire hypothesized sequence during rescoring, it is possible to train competitive models from scratch using only the under-resourced dataset. In addition, the LSTM architectures are substantially smaller (≈37M parameters compared to ≈249M).

5.4. Analysis of language prediction

In this section we show that the adapted code-predictive LSTM model improves the language prediction for several tokens following a code-switch. We hypothesised that, for the CP-LSTM architecture, the practice of replacing the hidden states at each timestep can hinder performance due to the possibility of an incorrect state being passed forward after a language prediction error. This is rectified in our adapted code-predicitive model (CP-LSTM-2).

In Figure 3, we show the language probabilities for an English-isiZulu code-switched utterance from the test set. The blue line indicates the ground truth language ($l^{(i)}$), while the green, red, and orange lines indicate the language predictions $\hat{l}^{(i)}$, $\hat{l}_E^{(i)}$, and $(1 - \hat{l}_B^{(i)})$ respectively. From the figure, it is clear that language predictions made by CP-LSTM-2 are better aligned with the ground truth compared to the original architecture. The improvement is especially apparent for the span of tokens after each code-switch occurs.

To determine whether this is true on average, we calculate the binary cross-entropy for the language predictions by CP-LSTM-ALL and CP-LSTM-2 as shown in Equation 1, where N is the total number of tokens. In Table 3, we present these cross-entropies specifically for one, two, three, and four tokens following a code-switch. From the table we can see that the updated architecture improves the language prediction specifically for the tokens following a code-switch (1,2,3,4).

$$\text{BCE} = \frac{1}{2N}\sum_{i=1}^{N}\Big\{l^{(i)} \cdot \Big(\log(\hat{l}_E^{(i)}) + \log(1 - \hat{l}_B^{(i)})\Big) + (1 - l^{(i)}) \cdot \Big(\log(1 - \hat{l}_E^{(i)}) + \log(\hat{l}_B^{(i)})\Big)\Big\} \quad (1)$$

6. Conclusions

In this work we have investigated the impact of different interventions to an LSTM based language model applied in n-best rescoring. By utilising language agnostic vocabularies and pooling the training data of our under-resourced corpus, performance of an LSTM model was improved by 0.87% absolute on average over five different bilingual test sets compared to a baseline speech recognition system. We found that sub-word encodings did not improve speech recognition accuracy compared to models with word level vocabularies.

We then presented a new LSTM model architecture adaptation that explicitly models the code of the next token and reduces model complexity. We found that this adaption improved speech recognition performance on average over all five bilingual test sets compared to the original architecture, and achieved the best performance for the heavily under-resourced English-Sepedi bilingual pair outperforming all other architectures by 0.4% absolute on the test set. When compared with a fine-tuned multilingual M-BERT architecture, the code predictive LSTM produced an improvement in terms of absolute test set word error rate of 0.17%. In addition to this, our LSTM models are substantially smaller, affording an up to a seven-fold reduction in total number of parameters, and reducing runtime during rescoring 100-fold.

Finally, we investigated the language prediction performance for the adapted code-predictive LSTM model and showed that average language prediction improves for a span of several tokens following a code-switch.

Table 3: *Balanced language binary cross-entropy (Equation 1) for one, two , three, and four tokens following a code-switch. Results are calculated on the pooled development set from all five bilingual pairs and averaged over the ten re-runs for each architecture.*

Words since code-switch	1	2	3	4
CP-LSTM-ALL	0.351	0.338	0.327	0.327
CP-LSTM-2	**0.350**	**0.333**	**0.324**	**0.324**

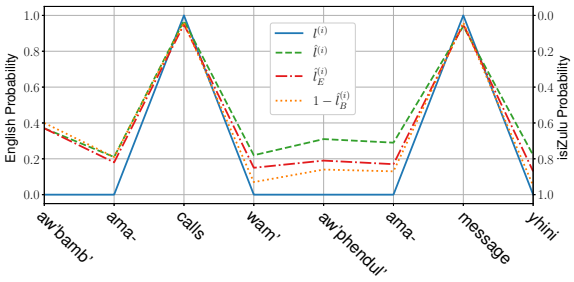

Figure 3: *Per token language probability for an English-isiZulu code-switched utterance from the test set. Blue: ground truth language ($l^{(i)}$), green: CP-LSTM-ALL ($\hat{l}^{(i)}$), red: CP-LSTM-2 English score ($\hat{l}_E^{(i)}$), and orange: CP-LSTM-2 isiZulu score ($1 - \hat{l}_B^{(i)}$). Note that the isiZulu and the English probabilities sum to one.*

7. Acknowledgements

This research was supported by the Department of Sports, Arts and Culture of the Republic of South Africa. We thank NVIDIA for their donation of GPU resources. We gratefully acknowledge the support of Telkom South Africa.

8. References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Minneapolis, USA, 2019.

[2] S. Ralethe, "Adaptation of deep bidirectional transformers for Afrikaans language," in *Proceedings of the 12th Language Resources and Evaluation Conference (LREC)*, Marseille, France, 2020.

[3] K. Ogueji, Y. Zhu, and J. Lin, "Small data? No problem! Exploring the viability of pretrained multilingual language models for low-resourced languages," in *Proceedings of the 1st Workshop on Multilingual Representation Learning*, Punta Cana, Dominican Republic, 2021.

[4] J. Jansen van Vüren and T. Niesler, "Code-switched language modelling using a code predictive LSTM in under-resourced South African languages," in *2022 IEEE Spoken Language Technology Workshop (SLT)*, Doha, Qatar, 2023.

[5] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," in *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., Vancouver, Canada, 2019.

[6] A. Radford and K. Narasimhan. (2018) Improving language understanding by generative pre-training. Accessed on: 08 March 2023. [Online]. Available: https://bit.ly/3kXBXp5

[7] A. F. Akyürek, L. Guo, R. Elanwar, P. Ishwar, M. Betke, and D. T. Wijaya, "Multi-label and multilingual news framing analysis," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online, 2020.

[8] X. Qiu, T. Sun, Y. Xu, Y. Shao, N. Dai, and X. Huang, "Pre-trained models for natural language processing: A survey," *Science China Technological Sciences*, vol. 63, no. 10, 2020.

[9] V. Snæbjarnarson, H. B. Símonarson, P. O. Ragnarsson, S. L. Ingólfsdóttir, H. Jónsson, V. Thorsteinsson, and H. Einarsson, "A warm start and a clean crawled corpus - a recipe for good language models," in *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, Marseille, France, 2022.

[10] S. Mesham, L. Hayward, J. Shapiro, and J. Buys, "Low-resource language modelling of South African languages," in *Proceedings of 2nd Workshop on African Natural Language Processing, 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, Virtual, 2021.

[11] Y. Gao, J. Feng, Y. Liu, L. Hou, X. Pan, and Y. Ma, "Code-switching sentence generation by BERT and generative adversarial networks," in *Proceedings of Interspeech*, Graz, Austria, 2019.

[12] G. Lee, X. Yue, and H. Li, "Linguistically motivated parallel data augmentation for code-switch language modeling," in *Proceedings of Interspeech*, Graz, Austria, 2019.

[13] X. Hu, Q. Zhang, L. Yang, B. Gu, and X. Xu, "Data augmentation for code-switch language modeling by fusing multiple text generation methods," in *Proceedings of Interspeech*, Shanghai, China, 2020.

[14] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," in *Proceedings of 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, Vancouver, Canada, 2017.

[15] I. Tarunesh, S. Kumar, and P. Jyothi, "From machine translation to code-switching: Generating high-quality code-switched text," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, Online, 2021.

[16] J. Jansen van Vueren and T. Niesler, "Optimised code-switched language model data augmentation in four under-resourced South African languages," in *Proc. SPECOM*, St. Petersburg, Russia, 2021.

[17] A. Gupta, A. Vavre, and S. Sarawagi, "Training data augmentation for code-mixed translation," in *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online, 2021.

[18] E. Yilmaz, H. van den Heuvel, and D. van Leeuwen, "Acoustic and textual data augmentation for improved ASR of code-switching speech," in *Proceedings of Interspeech*, Hyderabad, India, 2018.

[19] S. Garg, T. Parekh, and P. Jyothi, "Code-switched language models using dual RNNs and same-source pretraining," in *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Brussels, Belgium, 2018.

[20] H. Adel, N. T. Vu, F. Kraus, T. Schlippe, H. Li, and T. Schultz, "Recurrent neural network language modeling for code switching conversational speech," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vancouver, Canada, 2013.

[21] G. I. Winata, A. Madotto, C.-S. Wu, and P. Fung, "Code-switching language modeling using syntax-aware multi-task learning," in *Proceedings of 3rd Workshop on Computational Approaches to Linguistic Code-Switching*, Melbourne, Australia, 2018.

[22] 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 *Proceedings of Interspeech*, Graz, Austria, 2019.

[23] S. Wills, P. Uys, C. van Heerden, and E. Barnard, "Language modeling for speech analytics in under-resourced languages," in *Proceedings of Interspeech*, Shanghai, China, 2020.

[24] E. van der Westhuizen and T. Niesler, "A first South African corpus of multilingual code-switched soap opera speech," in *Proceedings of Eleventh International Conference on Language Resources and Evaluation (LREC)*, Miyazaki, Japan, 2018.

[25] 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 *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Hawaii, USA, 2011.

[26] A. Biswas, E. Yilmaz, F. de Wet, E. van der Westhuizen, and T. Niesler, "Semi-supervised acoustic model training for five-lingual code-switched ASR," in *Proc. Interspeech*, Graz, Austria, 2019.

[27] A. Stolcke, "SRILM-an extensible language modeling toolkit," in *Proceedings of 7th International Conference on Spoken Language Processing (ICSLP)*, Colorado, USA, 2002.

[28] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," in *Proceedings of NIPS Deep Learning Symposium*, Barcelona, Spain, 2016.

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of 3rd International Conference on Learning Representations (ICLR)*, San Diego, USA, 2015.

[30] J. Jansen van Vüren and T. Niesler, "Improving n-best rescoring in under-resourced code-switched speech recognition using pre-training and data augmentation," *Languages*, vol. 7, no. 3, 2022.

[31] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, Virtual, 2020.

[32] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Proceedings of 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Berlin, Germany, 2016.