Leveraging LLM for Augmenting Textual Data in Code-Switching ASR: Arabic as an Example

Sadeen Alharbi, Reem BinMuqbil, Ahmed Ali, Raghad AlOraini, Saiful Bari, Areeb Alowisheq, Yaser Alonaizan

National Center for Artificial Intelligence (NCAI), Saudi Data and Artificial Intelligence Authority (SDAIA), Riyadh, Saudi Arabia

Abstract

Intra-utterance code-switching (CS) is common in spoken language, posing significant challenges for automatic speech recognition (ASR) systems that need to handle mixed languages effectively. A primary obstacle in developing a CS-ASR system is the scarcity of suitable data. The complexity of CS grammatical structures further complicates the task, especially with Arabic, which has numerous dialects differing significantly in vocabulary, pronunciation, and syntax. To address CS, ASR systems are typically trained with available transcribed CS speech. This paper leverages advancements in large language models (LLMs) to enhance CS-ASR systems by generating Arabic-English code-switched textual data. Additionally, we introduce the Saudilang Code-switch Corpus (SCC), an evaluation dataset of Saudi CS with English. Our results show a relative reduction in perplexity by over 8% and a 5.5% average relative decrease in WER on two ecologically valid CS evaluation datasets. We plan to release the generated CS data and the new Arabic CS evaluation set to the research community.

1. Introduction

Code-switching (CS) is a linguistic phenomenon in which multiple languages are used within a single sentence or conversation. It naturally occurs in the speech of multilingual individuals. One of the primary challenges in creating models for conversational CS texts is the scarcity of conversational-style CS data. The computational processing of CS is inherently difficult due to the lack of available data. One solution is to apply linguistic knowledge, as suggested by various studies [1, 2].

Techniques such as Equivalence Constraint and Functional Head Constraint have been utilized to improve CS language models [3, 4]. Additionally, models incorporating syntactic and semantic features have been developed to leverage more information [5]. Given the plethora of monolingual data, separate language models for selected languages are trained and then aggregated by a probabilistic model to manage language switching [6].

Since CS is predominantly found in spoken language, the most effective way to generate data is by labeling CS speech. However, manual transcription is labor intensive and time consuming, creating a significant bottleneck in data collection. An alternative approach is to create CS data from existing monolingual text, although predicting code-switching points within a sentence is challenging due to individual variations in code-switching behavior. Efforts have been made to synthesize more CS text using models trained on data [7, 8]. Additionally, researchers are investigating the integration of linguistic theories into these models to enhance the naturalness and accuracy of the synthesized CS text. One approach involves modifying monolingual sentences into CS sentences, allowing the generator to use information from monolingual text by using generative adversarial networks (GAN) with reinforcement learning (RL) to automatically generate CS data from monolingual sentences [9]. Other techniques applied to neural machine translation have been explored to automate the generation process [10, 11, 12]. Recently, [13] assessed the translation ability of state-of-the-art LLMs for CS translation tasks, demonstrating high performance. Consequently, more studies have shown that generative models have been used to generate CS sentences [14]. However, these models learn to modify monolingual sentences into CS sentences by translating select words throughout the sentence. This approach allows the model to leverage information from monolingual sentences.

Our paper build on top of previous research and introduce leveraging LLMs to improve ASR performance by generating CS text. To mitigate bias toward a specific domain or dialect during evaluation, multi-dialectal Arabic-English CS training sets are generated, including Saudi, Egyptian, and Modern Standard Arabic (MSA). Subsequently, an interpolated language model is created using this data, which is then utilized to rescore the *N*-best list generated from the decoding process of the Whisper multilingual ASR system [15]. Our paper offers several notable contributions:

- **Leveraging LLMs to generate CS data** Elimination of the need for manual data labeling, which is typically required for training data generators in other methodologies.
- **Release of Textual Generated CS Sentences** The generated CS sentences in various dialects, including Saudi, Egyptian, and MSA, has been released.
- **Release of Saudilang Code-switch Corpus (SCC)** A new evaluation set for Arabic (Saudi) - English CS has been released.
- **CS *N*-best List Rescoring for** This approach explored incorporating the LLM into CS *N*-best List Rescoring. Experiments were conducted on two Arabic-English code-switching corpora, ESCWA[16] and SCC. These corpora exhibit different characteristics, demonstrating that the proposed approach generalizes well across different scenarios.

The experimental results demonstrate that LLMs can generate reasonable code-switching sentences in various Arabic dialects. These generated sentences were utilized to build a language model, which is employed in CS N-best List Rescoring after the decoding process of the ASR system.

2. Method

Although ASR training necessitates audio–text paired data, collecting a large text corpus is comparatively straightforward.

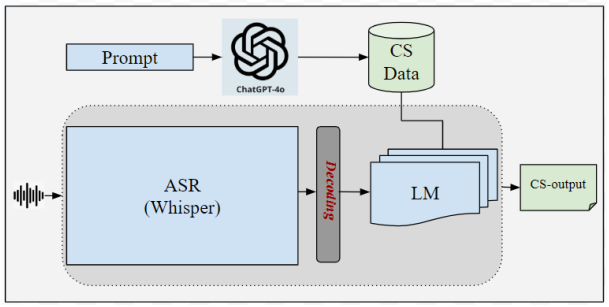

Figure 1: *The process of enhancing CS ASR through LLM-generated text. Initially, code-switching sentences in multiple Arabic dialects (Saudi, Egyptian) and English are generated using the GPT-4 LLM model API. Next, a LM is trained on these generated CS sentences. Finally, this LM is utilized to rescore the N-best list generated from the decoding process of the Whisper ASR system.*

Consequently, it is feasible to boost ASR performance using text data, often through the shallow fusion [17, 18] of external LMs. Employing LLMs to generate CS sentences serves as a method for gathering extensive textual data. The objective is to enhance ASR performance during the decoding process by integrating an externally trained LM in these contexts.

The process of enhancing CS ASR by leveraging text generation from the LLM is illustrated in Fig. 1. The first step involves generating code-switching sentences that incorporate various Arabic dialects. In this study, we focus on two Arabic dialects, Saudi and Egyptian dialects, alongside English and Modern Stndard Arabic, using the GPT-4 LLM model API [19]. This allows for the creation of a diverse and comprehensive dataset without the need for extensive manual labeling.

We train various *n*-gram LMs on these generated CS sentences. This training phase is crucial, as it enables the LM to learn the intricate patterns and structures inherent in code-switching language use. In the final step, the trained LM is employed to refine the decoding process in the Whisper ASR system. This approach helps improve transcription accuracy by incorporating language-specific knowledge, allowing the ASR system to more accurately interpret and transcribe spoken language that involves code-switching between Arabic dialects and English.

2.1. Decoding Strategy

An additional decoding mechanism was integrated into the Whisper model implementation during the inference stage. This mechanism was designed to enhance transcription accuracy by using a combination of beam search decoding and *N*-best list rescoring.

2.1.1. Beam Search Decoding

Beam search decoding was employed to generate multiple potential hypotheses for each input speech segment. During beam search, multiple potential tokens are evaluated simultaneously, considering a set number of options, or "beam width." Each token is assessed in the context of its neighbors, creating a composite score. The beam search process produces an *N*-best list of candidate transcriptions, each with a combined score based on acoustic model outputs and initial language model probabilities.

2.1.2. N-best List Rescoring

To further refine the hypotheses generated via beam search decoding, *N*-best list rescoring was applied using an *n*-gram language model. The steps involved in this process are as follows:

1. **Hypothesis Generation:** The Whisper model, utilizing beam search decoding, generates an *N*-best list of hypotheses. Each hypothesis in this list is a potential transcription of the input speech, ranked by its combined score from the initial decoding phase.
2. **Language Model Integration:** Each hypothesis from the output is then rescored using our *n*-gram language model. This model evaluates the linguistic coherence and probability of each hypothesis within its broader context.
3. **Perplexity Calculation:** During rescoring, the perplexity of each hypothesis is computed based on the *n*-gram language model. Perplexity measures the likelihood of a hypothesis given its context, with lower perplexity values indicating more probable and contextually appropriate hypotheses.
4. **Selection of Final Hypothesis:**The hypothesis with the lowest perplexity score is selected as the final transcription output. Mathematically, this can be represented as:

$$\text{Final Hypothesis} = \arg\min_{H_i}\{\text{LM}p(H_i)\}$$

where $\text{LM}p(H_i)$ denotes the perplexity produced by the language model for hypothesis H_i. This selection process ensures that the transcription is not only acoustically accurate but also linguistically coherent, adhering closely to natural language patterns.

The integration of the ASR system's robustness with language model insights produces contextually relevant transcriptions, improving performance in code-switching tasks by better handling mixed-language speech.

3. Data

3.1. Code-switching Text Data Generation

LLMs represent an advanced class of decoder architectures that learn autoregressively, predicting subsequent tokens based on preceding token sequences, in alignment with a language modeling objective. Distinguished from earlier language models by their vast parameter sizes—ranging from billions to trillions of weights—and their extensive training datasets comprising trillions of tokens, LLMs have set new benchmarks in the field. Moreover, through additional instruction fine-tuning, these models have demonstrated an exceptional ability to adhere to human-generated prompts with high fidelity, showcasing their potential for sophisticated language understanding and generation tasks. GPT-4 [20], as a language model, does not explicitly apply Equivalence Constraint Theory (EC) when generating code-switching sentences. However, it can produce plausible code-switching outputs based on its training data, which include many examples of natural language usage, including code-switching instances.

In this study, GPT-4 was employed to generate a comprehensive code-switching Arabic–English textual dataset. The methodology focused on utilizing specific prompts tailored to distinct Arabic dialects and MSA to ensure the authenticity and diversity of code-switching instances. Examples of these prompts include:

- **Modern Standard Arabic (MSA):** "Generate a code-switching sentence in Modern Standard Arabic (MSA) and

English. Display only the sentence, without any explanation or translation".

- **Saudi Arabic dialect:** "Generate a code-switching bilingual sentence in the Saudi Arabic Dialect and English. Display only the sentence, without any explanation or translation".
- **Egyptian Arabic dialect:** "Generate a code-switching bilingual sentence in the Egyptian Arabic Dialect and English. Display only the sentence, without any explanation or translation".

These prompts provided structured guidance to GPT-4, facilitating the generation of a varied dataset that captures nuanced language alternation patterns and contextual variations across different Arabic dialects and MSA. The instruction, "Display the sentence only without any explanation or translation," was included because GPT-4 occasionally provides translations or explanations alongside the generated sentences. By excluding these additional elements, the output retains the original sentences directly generated by the model. Table 1 presents samples from the generated dataset, demonstrating the outcomes of the structured prompts.

Table 1: *Sample code-switching sentences generated by GPT-4. MSA: Modern Standard Arabic, EG: Egyptian Arabic, SA: Saudi Arabic.*

Dialect	Sample Generated Sentence
MSA	بالأمس كنت at the park مع أصدقائي.
	(Yesterday I was) (With my friends).
SA	والله it was an amazing trip, ما تتخيلين how much fun!
	(I swear by God) (you can't imagine)
EG	النهاردة عندنا meeting ال مع team تخلص عشان project.
	(Today we have a) (with the) (to finish)

3.1.1. Processing of Generated Code-Switching Text Data

The processing of generated code-switching text data involves several crucial steps to ensure its usability and reliability for subsequent analyses and applications. These steps include removing duplicated sentences (exact match), punctuation, and Arabic diacritics to eliminate redundancy, standardize word forms, and simplify further processing. Additionally, sentences containing Arabizi[1] words or expressions are filtered out to maintain linguistic authenticity and purity. Other steps include normalizing Arabic letters to their standard forms, converting all English words to lowercase, and removing apostrophes from contractions to align written text with its spoken form during the annotation process. Upon completing these systematic processing stages, the data are refined into a meticulously prepared corpus, ready to perform effectively in code-switching NLP tasks.

The processed dataset comprises of 330k sentences, totaling 3M tokens, with 1.6M tokens in Arabic and 1.3M in English. This distribution results in English tokens constituting approximately 44% of the dataset. Table 2 presents detailed statistics for the corpus. Data split for training and testing as shown in Table 3.

[1] Arabizi is defined as an encoding system that uses the Latin script and Arabic numbers instead of Arabic letters. Each English letter represents an Arabic phoneme that matches it in pronunciation, whereas the Arabic numerals compensate for Arabic phonemes that are nonexistent in the English language, but resemble Arabic letters and their shapes. [21]

Table 2: *Statistics on the generated code-switching dataset. CMI stands for Code Mixing Index*

Statistics	Value
Total # of Sentences	330k
Total # of Tokens	3M
Total # of Arabic Tokens	1.6M
Total # of English Tokens	1.3M
Ratio of English to Arabic Tokens	1:1.25
Avg. Words per Sentence	9.27
Avg. English Words per Sentence	4.11
Avg. CMI of all Sentence	14%
#Tokens per Dialect	
MSA Tokens	1.2M
SA Tokens	1.2M
EG Tokens	1.2M

Table 3: *Generated code-switching dataset distribution*

Dataset Partition	#Sentences	Split Percentage (%)
Train Set	300k	(90%)
Evaluation Set	30k	(10%)

3.2. Saudilang Code-switch Corpus (SCC)

3.2.1. Data Overview

We introduce Saudilang Code-switch Corpus, an evaluation dataset noted as the first cultural Saudi dialect dataset. It consists of conversational audio from the YouTube podcast 'Thmanyah', featuring three episodes covering different domains: investment, establishing restaurants, and entrepreneurship. It includes 5 hours of labeled audio spoken in Arabic with occasional English words, featuring Saudi dialect, thus providing a linguistic and cultural context for studying code-switching phenomena. The corpus comprises recordings from 4 unique speakers across 3 files, including segments with overlapping speakers. Table 4 shows a detailed statistics. Note: The dataset will be released with more details on camera ready version.

Table 4: *Statistics on the code-switching evaluation datasets*

Statistics	SCC	ESCWA.CS
Total # of Sentences	3296	841
Total # of Tokens	39.2K	16.3K
Total # of Arabic Tokens	34K	12K
Total # of English Tokens	6.7K	4.1K
Ratio of English to Arabic Tokens	1:5	1:3
Avg. Words per Sentence	11.9	19.38
Avg. Words per Second	2.4	2.0
Avg. English Words per Sentence	2.04	4.9
Avg. CMI of all Sentence	4%	8%
Avg. Segment Duration	4.9 (s)	11.9 (s)
Minimum Segment Duration	0.5 (s)	3.2
Maximum Segment Duration	18.6 (s)	23.7

3.2.2. Data Pre-Processing

In preparing the data, we followed the steps mentioned in section 3.1.1, with the addition of a crucial step involving inserting spaces between Arabic and English words to handle code-switching nuances. For instance, this ensures separation to prevent the concatenation of English words with the Arabic definite article' ال' (al) or ' لل' (for) that some speakers use before switching to English words. This step is crucial to maintain the clarity and accuracy of code-switched speech.

4. Experiments

Objective evaluations were conducted to assess the quality of the generated CS text. The quality of the generated text and its effectiveness in managing CS in *n*-gram language models were evaluated using perplexity measurements. A standard trigram LM was developed using Kneser-Ney smoothing with the SRILM toolkit [22]. The impact of this language modeling on speech recognition was measured by reporting word error rate (WER).

4.1. Experimental setup

4.1.1. Training Language Models

We developed two distinct trigram LMs:

- Baseline LM: Trained on four distinct monolingual datasets, namely SADA [23][2], QASR [24][3](excluding code-switched part), Arabic Common Voice[4], and English Common Voice.[5] Details regarding the token counts for each dataset are provided in Table 5
- CS LM: Trained on the augmented generated code-switching dataset using LLM[6].

Table 5: *Baseline LM training datasets #tokens.*

Dataset	#Tokens
SADA	3,3M
QASR	13,3M
Arabic Common Voice	37.8k
English Common Voice	1M

4.1.2. Interpolation of Language Models

To enhance the performance of the language model, we employed an interpolation approach that combines the probabilities from both the baseline LM and the code-switching LM using designated weights. This method allows us to leverage the strengths of both models. Various weight combinations were systematically explored to determine the optimal interpolation settings, aiming to identify the best-performing model and maximize overall accuracy and robustness in processing code-switched text. We used LM perplexity to tune weightes and the best number combination was used for Whisper model evaluation.

[2]https://www.kaggle.com/datasets/sdaiancai/sada2022

[3]https://arabicspeech.org/resources/qasr

[4]https://commonvoice.mozilla.org/ar/datasets

[5]https://commonvoice.mozilla.org/en/datasets

[6]https://www.kaggle.com/datasets/sdaiancai/arabic-english-code-switching-textual-dataset

4.2. Results and Discussion

Our experiments involved evaluating the performance of interpolated language models using different weights on the ESCWA-CS [16][7] and SCC[8] datasets. The analysis reveals that increasing the weight of the CS LM generally leads to a reduction in perplexity. Specifically, models with a higher proportion of CS LM demonstrated lower perplexity values, indicating better performance in handling code-switching data. Table 6 reports perplexity (the lower the better) for various model interpolations weights.

Table 6: *Interpolated Language Models with Different Weights and Perplexity*

			Perplexity	
Baseline LM	**CS LM**	**CS Dataset**	**ESCWA-CS**	**SCC**
1	0	1,373	4,210	4,116
0.6	0.4	55	4,238	4,534
0.7	0.3	67	4,002	4,234
0.8	0.2	88	3,871	4,040
0.9	0.1	135	**3,861**	**3,955**

Table 7: *Overall WER% on two code-switching evaluation sets.*

ASR Configuration	**ESCWA-CS**	**SCC**
Whisper	47	42
Whisper + CS LM N-best Rescoring	43	41

Table 7 shows results using Whisper model on the ESCWA-CS and SCC datasets. The results indicate a relative improvement in WER of 8.5% on the ESCWA-CS dataset upon integrating the LM for *N*-best rescoring. On the SCC dataset, a relative improvement of 2.4% was observed. The smaller improvement on the SCC dataset can be attributed to the linguistic diversity of the Saudi dialects, which introduces additional complexity. Nonetheless, the integration of the LM enhances ASR performance in processing code-switched text, resulting in more accurate and reliable transcriptions.

5. Conclusion

This study utilized LLMs to improve ASR for Arabic-English CS. We generated a comprehensive CS dataset using GPT-4 and introduced the Saudilang Code-switch Corpus (SCC), addressing data scarcity. Key preprocessing steps included removing duplicates, punctuation, and Arabic diacritics.

Experiments showed significant improvements. Interpolated language models combining baseline and CS-specific models reduced perplexity and WER. Incorporating the LM for *N*-best rescoring with the Whisper model on two datasets led to a 5.5% average relative improvement in WER.

This work demonstrates the potential of LLM-generated CS data to enhance ASR performance, providing valuable resources for future research in multilingual speech processing.

For future, we plan to further train LLM to generate CS data for specific dialect and potentially explore text to speech to generate audio data as well.

[7]https://arabicspeech.org/resources/escwacs

[8]https://huggingface.co/datasets/SDAIANCAI/Saudilang-Code-Switch-Corpus

6. References

[1] R. M. Bhatt, "Code-switching and the functional head constraint," in *Janet Fuller et al. Proceedings of the Eleventh Eastern States Conference on Linguistics. Ithaca, NY: Department of Modern Languages and Linguistics*, 1995, pp. 1–12.

[2] C. W. Pfaff, "Constraints on language mixing: Intrasentential code-switching and borrowing in spanish/english," *Language*, pp. 291–318, 1979.

[3] Y. Li and P. Fung, "Code-switch language model with inversion constraints for mixed language speech recognition," in *Proceedings of COLING 2012*, 2012, pp. 1671–1680.

[4] ——, "Language modeling with functional head constraint for code switching speech recognition," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 907–916.

[5] C.-F. Yeh and L.-S. Lee, "An improved framework for recognizing highly imbalanced bilingual code-switched lectures with cross-language acoustic modeling and frame-level language identification," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 7, pp. 1144–1159, 2015.

[6] S. Garg, T. Parekh, and P. Jyothi, "Dual language models for code switched speech recognition," *arXiv preprint arXiv:1711.01048*, 2017.

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

[8] S. Garg, T. Parekh, and P. Jyothi, "Code-switched language models using dual rnns and same-source pretraining," *arXiv preprint arXiv:1809.01962*, 2018.

[9] C.-T. Chang, S.-P. Chuang, and H.-Y. Lee, "Code-switching sentence generation by generative adversarial networks and its application to data augmentation," *arXiv preprint arXiv:1811.02356*, 2018.

[10] I. Tarunesh, S. Kumar, and P. Jyothi, "From machine translation to code-switching: Generating high-quality code-switched text," *arXiv preprint arXiv:2107.06483*, 2021.

[11] A. Hussein, S. A. Chowdhury, A. Abdelali, N. Dehak, A. Ali, and S. Khudanpur, "Textual data augmentation for arabic-english code-switching speech recognition," in *2022 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2023, pp. 777–784.

[12] A. Hussein, D. Zeinali, O. Klejch, M. Wiesner, B. Yan, S. Chowdhury, A. Ali, S. Watanabe, and S. Khudanpur, "Speech collage: code-switched audio generation by collaging monolingual corpora," in *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2024, pp. 12 006–12 010.

[13] M. Huzaifah, W. Zheng, N. Chanpaisit, and K. Wu, "Evaluating code-switching translation with large language models," in *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, 2024, pp. 6381–6394.

[14] A. Heakl, Y. Zaghloul, M. Ali, R. Hossam, and W. Gomaa, "Arzen-llm: Code-switched egyptian arabic-english translation and speech recognition using llms," *arXiv preprint arXiv:2406.18120*, 2024.

[15] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in *International conference on machine learning*. PMLR, 2023, pp. 28 492–28 518.

[16] A. Ali, S. A. Chowdhury, A. Hussein, and Y. Hifny, "Arabic Code-Switching Speech Recognition Using Monolingual Data," in *Proc. Interspeech 2021*, 2021, pp. 3475–3479.

[17] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates *et al.*, "Deep speech: Scaling up end-to-end speech recognition," *arXiv preprint arXiv:1412.5567*, 2014.

[18] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," *Advances in neural information processing systems*, vol. 28, 2015.

[19] OpenAI, "Gpt-4 technical report," *OpenAI*, 2023. [Online]. Available: https://cdn.openai.com/papers/gpt-4.pdf

[20] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[21] W. H. Allehaiby, "Arabizi: An analysis of the romanization of the arabic script from a sociolinguistic perspective," *Arab World English Journal*, vol. 4, no. 3, 2013.

[22] A. Stolcke *et al.*, "Srilm-an extensible language modeling toolkit." in *Interspeech*, vol. 2002, 2002, p. 2002.

[23] S. Alharbi, A. Alowisheq, Z. Tüske, K. Darwish, A. Alrajeh, A. Alrowithi, A. B. Tamran, A. Ibrahim, R. Aloraini, R. Alnajim *et al.*, "Sada: Saudi audio dataset for arabic," in *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2024, pp. 10 286–10 290.

[24] H. Mubarak, A. Hussein, S. A. Chowdhury, and A. Ali, "Qasr: Qcri aljazeera speech resource–a large scale annotated arabic speech corpus," *arXiv preprint arXiv:2106.13000*, 2021.