Beyond Detection: Predicting Code-Switch Points in Multilingual Conversations

Anonymous ACL submission

Abstract

Code-switching, the practice of multilingual speakers switching between two or more languages within a conversation or sentence, is commonly observed in multilingual communities. It also poses unique challenges for natural language processing (NLP) system. While existing NLP models in the field can detect and process code-switched text, they do not predict switch points at token level. In this paper, we introduce a token-level prediction framework for identifying upcoming switch points in Chinese-English conversations. We present two approaches: a window-based model leveraging BERT embeddings and recurrent architectures, and a transformer-based model using mBERT and XLM-RoBERTa. Trained and evaluated on the ASCEND dataset, our best RNN-based model achieves an AUC of 0.91 for Chinese-to-English prediction, while our transformer-based model (mBERT) achieves an AUC of 0.98. These results show promise in the code-switch prediction task and offer potential to support mixed-language NLP applications such as conversational AI and machine translation.

1 Introduction

Code-switching is a practice when multilingual speakers alternate between two or more languages within a conversation or sentence. This is a common phenomenon in multilingual communities where speakers interchangeably navigate between languages based on context, topic, or social dynamics. Code-switching shows its linguistics richness, provides a way for multilingual speakers to express, grasp complex topics and foster cultural identities. However, it also introduces significant challenges for natural language processing (NLP) tasks that are typically designed for monolingual inputs (Aljoundi, 2013). This becomes particularly problematic in real-time applications like voice assistants, chatbots, predictive keyboards, and machine translation, where sudden language switches can disrupt downstream tasks like tokenization, tagging, or prediction.

Existing research in code-switching has focused primarily on detecting when a language switch has already happened. This includes tasks like identifying the language of each word, tagging parts of speech, or recognizing names in mixed-language text (Winata and et al., 2023). One early attempt by Solorio and Liu (Solorio and Liu, 2008) used simple machine learning models to predict switch points in Spanish-English speech, showing the task was possible. A few studies have explored detecting switch points directly in text, such as the work by Yirmibeolu and Eryiit (Yirmibeşoğlu and Eryiğit, 2018), which used character-level features and CRFs to identify Turkish-English switches. However, these approaches all act after the fact, offering little utility in real-time systems where we need to adapt before the switch happens. Our work in this paper attempts to fill that void. Our main contributions are as follows:

- We introduce a predictive framework for identifying upcoming code-switch points in real time as a token-level classification.
- We evaluate two modeling paradigms: (1) a window-based model using BERT embeddings with RNNs, and (2) a transformer-based approach using pretrained multilingual models (mBERT and XLM-RoBERTa).
- We conduct experiments on the ASCEND dataset, a spontaneous Chinese-English code-switching corpus, and analyze model

performance across multiple metrics. Our best results show that transformer-based and the best RNN-based models achieve 98.0% and 91.0% AUC, respectively.

To the best of our knowledge, this is the first study to investigate token-level code-switching prediction for Chinese-English using deep learning models. These results suggest that modern NLP systems can be trained not just to detect language switches after they occur, but to anticipate them opening the door to adaptive and responsive multilingual applications.

2 Related Work

The study of code-switching spans both linguistics and NLP. Linguists have long examined the structural, semantic, and social aspects of language switching (Sitaram and et al., 2020). These studies highlight that code-switching follows rules and constraints.

In NLP, early work on code-switching focused on language identification (Winata and et al., 2023), while recent efforts have introduced benchmarks covering a wider range of tasks in mixed-language text (Winata and et al., 2023). For example, (Aguilar et al., 2020) created the LINCE benchmark using social media data to evaluate tasks like LID, POS tagging, NER, and sentiment analysis across multiple language pairs. Building on this, (Khanuja et al., 2020) proposed GLUE-CoS, which included additional tasks, such as question answering and natural language inference. Both studies showed that multilingual models like mBERT benefit from fine-tuning on code-switched data, though challenges remain.

An early study by Solorio and Liu (Solorio and Liu, 2008) explored code-switch prediction using Naive Bayes models on a small Spanish-English dataset. While their results were encouraging, the approach was limited by dataset size and use of simple, hand-crafted features. Our work addresses the same prediction task in a new setting, Chinese-English, using a larger dataset and deep learning models to better capture switching behavior.

3 Problem Description

A core challenge in code-switching research is predicting future switches. Given a sequence of tokens up to a certain point in an utterance, predict the likelihood that the next token will be in a different language (i.e., will constitute a code-switch). Formally, the input (X) is a sequence of tokens $[w_1, w_2, ..., w_n]$, and the output (Y) is a sequence of labels or probabilities $[y_1, y_2, ..., y_n]$, where each y_i represents the likelihood (or a binary classification) of a code-switch occurring after the token w_i. Note that, the prediction is about the next token, which is not included in the input sequence. The code-switch prediction task is different from code-switch detection because it focuses on anticipating whether and where a language switch will occur in upcoming text, whereas detection focuses on identifying switches that have already taken place within a given sequence.

3.1 Dataset

For this study, we utilized the ASCEND (A Spontaneous Chinese-English Dataset) (Lovenia et al., 2022), a high-quality corpus of spontaneous, multi-turn conversational dialogue featuring Chinese-English code-switching. The dataset contains 10.62 hours of spontaneous speech comprising ∼12.3K utterances. Each utterance reflects natural bilingual interaction between speakers. The corpus is divided into training, validation, and test sets using an 8:1:1 ratio, with a balanced gender distribution maintained across all splits to ensure fair and representative evaluation. To process the data, we used NLTK, SpaCy, Jieba, etc. packages.

4 Methodology

In this paper, we present two experimental setups: (1) a window-based approach, both fixed and flexible, with separate models trained for Chinese-to-English and English-to-Chinese code-switching; and (2) a unified approach with a single multilingual transformer model (mBERT or XLM-RoBERTa), trained to handle both language directions simultaneously.

4.1 Window Based Approach

We apply two types of context windows:

- Fixed Size Window: A fixed number of tokens around the switch point.
- Flexible Size Window: A progressively expanding context that begins with the first token of the switch segment and adds one token at a time up to the full span of the switch-triggering portion.

For Chinese-to-English, we consider two tokenization methods: character-level and segment-level using Jieba (Sun, 2024). For English-to-Chinese, we tokenize at the word level. Each window is then labeled with a 1 or 0 depending on whether a switch happens immediately after. These labeled windows form the training input for our model.

Next, we implemented an RNN architecture using pre-trained BERT embeddings (Chinese and English separately). BERT token embeddings are fed into a unidirectional LSTM. While POS tagging for code-switched text traditionally presents challenges due to different tagsets across languages and ambiguity at switching boundaries (Vyas et al., 2014), our approach mitigates these issues by using a unified tagset and embedding space for both languages. The POS embeddings are jointly trained with the model, allowing it to learn language-specific nuances while maintaining cross-lingual consistency (Aguilar and Solorio, 2020). This integration enables the model to leverage syntactic information as an additional signal for predicting code-switch points without requiring pre-defined rules for handling cross-lingual POS transitions.

4.2 Transformer-based Approaches

Recent research has demonstrated the strong performance of transformer-based architectures for sequence labeling tasks across various NLP applications (Wu and Dredze, 2019). In particular, multilingual transformer models have shown remarkable cross-lingual transfer capabilities for tasks, like NER, POS tagging, and syntactic parsing. Their attention mechanisms are particularly well-suited for capturing long-range dependencies that could be crucial for identifying code-switching points, which often depend on both local syntactic patterns and broader discourse context. Therefore, we employed multilingual transformer models (mBERT and XLM-RoBERTa) that predict switch points directly at token-level (we use sequences of length ten). Detailed description of data pre-processing and labeling used for the future switch prediction task is provided in the Appendix B.

5 Experiments and Results

Window-based Approach We run each experiment on an A100 GPU for ∼4 hours. We experimented with hidden layer sizes of 32, 64, and 128 to determine the optimal model capacity over the following four configurations: 1. *Baseline (B)*: Pre–trained embedding layer followed by an RNN layer. 2. *B+POS*: Adds POS embeddings. 3. *B+POS+Dropout*: Adds dropout to reduce overfitting. 4. *B+POS+Dropout+LayerNorm*: Adds both dropout and layer normalization.

Table 1 presents the performance of a model on a code-switching prediction task, broken down by switch direction and context window strategy. For Chinese-to -English switches, the fixed window approach with the *64+POS* configuration achieves the best performance of an accuracy of 0.91, precision of 0.85, recall of 0.45, F1 of 0.59, and AUC of 0.91. In contrast, the English-to-Chinese direction exhibits lower overall performance. The flexible window approach with *128+POS+dropout+LN* performs best, with an accuracy of 0.81, precision of 0.62, recall of 0.61, F1 of 0.62, and AUC of 0.81. Overall, the model performs better at predicting Chinese-to-English switches than English-to-Chinese switches, and the window strategy and hidden layer configuration significantly impact performance.

Transformer-based Approach We run each experiment on an A100 GPU for 5 minutes. Table 2 presents the performance of two transformer-based multilingual models, mBERT and XLM-RoBERTa, over two training epochs on the code-switch prediction task. Both models achieve strong performance across all evaluation metrics, but mBERT consistently outperforms XLM-RoBERTa. Specifically, at Epoch 2, mBERT achieves a higher accuracy (0.9961 vs. 0.9945), F1-score (0.9737 vs. 0.9657), and AUC (0.9800 vs. 0.9702). Precision and recall also show favorable results for mBERT, suggesting that it is more effective in identifying code-switch points with

Table 1: Window Based Approach – Performance comparison across different window sizes and hidden layer configurations. RNN + Bert Embeddings were used here.

Switch Direction	Window Size	Hidden Layer Configurations	Accuracy	Precision	Recall	F1	AUC
Chinese → English	Fixed	64+POS	0.91	0.85	0.45	0.59	**0.91**
Chinese → English	Flex	64+POS+dropout+LN	0.91	0.70	0.29	0.41	0.87
English → Chinese	Fixed	64+POS+dropout	0.88	0.25	0.07	0.11	0.55
English → Chinese	Flex	128+POS+dropout+LN	0.81	0.62	0.61	0.62	0.81

Table 2: Transformer-based Approach – Results of mBERT and XLM-RoBERTa Models.

Model	Epoch	Training Loss	Validation Loss	Accuracy	Precision	Recall	F1	AUC
mBERT	1	0.0578	0.0128	0.9966	0.9977	0.9567	0.9768	0.9783
mBERT	2	0.0239	0.0128	0.9961	0.9867	0.9610	0.9737	**0.9800**
XLM-RoBERTa	1	0.1016	0.0254	0.9938	0.9943	0.9305	0.9613	0.9650
XLM-RoBERTa	2	0.0382	0.0222	0.9945	0.9915	0.9412	0.9657	0.9702

fewer false positives and false negatives. Notably, XLM-RoBERTa shows slightly higher precision (0.9915) at Epoch 2, but this comes at the cost of slightly lower recall and F1, indicating a tendency to be more conservative in its predictions. In conclusion, while both transformer models demonstrate strong capabilities for this task, mBERT shows slightly better and more balanced performance, making it a more suitable choice for code-switch prediction in this setting.

Discussion The results reveal a performance gap between Chinese-to-English and English-to-Chinese switch prediction. One likely explanation is the nature of the ASCEND dataset itself, which is more Chinese-dominant, containing a larger proportion of Chinese tokens and utterances. Additionally, English insertions in Chinese speech may follow more predictable discourse patterns or lexical cues, such as topic shifts or borrowed terminology. The fixed window strategy generally outperforms the flexible window in Chinese-to-English prediction. This could be because fixed windows offer more consistent, compact context, whereas the flexible window might introduce noisy or redundant information that dilutes predictive cues. While our labeling approach defines code-switches based on broad token categories, the high performance of mBERT and XLM-RoBERTa demonstrates their strong ability to identify fundamental language transitions at the token level. This success underscores the power of their multilingual pre-training providing a solid foundation for future work exploring the prediction of more linguistically nuanced code-switching phenomena using richer annotation schemes.

6 Conclusion and Future Direction

This study introduces a novel token-level approach for predicting code-switching points in bilingual Chinese-English conversations, investigating both window-based RNNs and powerful multilingual transformer models. Our findings robustly demonstrate that transformer-based architectures, particularly mBERT, excel at identifying upcoming language switches, achieving high scores across accuracy, precision, recall, and AUC. While mBERT consistently outperformed XLM-RoBERTa in our experiments on the ASCEND dataset, several promising avenues for future work remain. These include expanding our approach to handle multilingual contexts involving more than two languages and rigorously testing the generalizability of our models on diverse code-switching datasets with different language pairs and conversational domains. A significant next step involves enhancing the predictive capability to not only anticipate a switch but also to determine the target language of the upcoming code-switch, which holds substantial potential for advancing multilingual NLP applications. Another direction worth exploring is whether the optimal window size should differ by language. Since Mandarin characters might carry more information per token than English words, models might benefit from shorter context windows for Mandarin and longer ones for English. A window size sweep could help uncover language-specific patterns that improve prediction.

7 Limitations

We now discuss a few limitations of our work. First, we only tested our models on the ASCEND dataset. It's unclear how well the models would perform on other language pairs or on written code-switched text. Second, code-switches are relatively rare, creating data imbalance that makes models struggle to learn and predict switches, especially from English-to-Chinese. Lastly, our models rely solely on text-based features while prosodic, syntactic, or speaker-level information may carry important cues. The high performance of mBERT and XLM-RoBERTa is largely attributable to their ability to recognize transitions between broad token categories (defined by simple rules) that align closely with their subword tokenization. While effective at this simplified task, the models' ability to predict more nuanced, linguistically motivated code-switches in real-world scenarios remains to be established. Future research should investigate performance on datasets with more linguistically rich annotations.

8 Ethics Statement

We use only publicly available datasets and pre-trained models in this study, all of which are accessed and utilized strictly for research purposes. The use of these resources complies with their original licenses and terms of access. No personally identifiable or sensitive information is present in any of the data used.

Our code will be released under the MIT license to support transparency and reproducibility.

References

Gustavo Aguilar, Sudipta Kar, and Thamar Solorio. 2020. LinCE: A centralized benchmark for linguistic code-switching evaluation. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 1803–1813, Marseille, France. European Language Resources Association.

Gustavo Aguilar and Thamar Solorio. 2020. From English to code-switching: Transfer learning with strong morphological clues. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8033–8044, Online. Association for Computational Linguistics.

Entisar Khalifa Aljoundi. 2013. The strengths and weaknesses of code-switching and bilingualism in the language classroom. *ResearchGate*. Working Paper.

Simran Khanuja, Sandipan Dandapat, Anirudh Srinivasan, Sunayana Sitaram, and Monojit Choudhury. 2020. GLUECoS: An evaluation benchmark for code-switched NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3575–3585, Online. Association for Computational Linguistics.

Holy Lovenia, Samuel Cahyawijaya, Genta Indra Winata, Peng Xu, Xu Yan, Zihan Liu, Rita Frieske, Tiezheng Yu, Wenliang Dai, Elham J Barezi, and 1 others. 2022. Ascend: A spontaneous chinese-english dataset for code-switching in multi-turn conversation. In *Proceedings of the 13th Language Resources and Evaluation Conference (LREC)*.

Sunayana Sitaram and et al. 2020. A survey of code-switched speech and language processing. *arXiv preprint arXiv:1904.00784v3*.

Thamar Solorio and Yang Liu. 2008. Learning to predict code-switching points. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 973–981, Honolulu, Hawaii. Association for Computational Linguistics.

Junyi Sun. 2024. jieba: Chinese text segmentation. `https://github.com/fxsjy/jieba`. Accessed: 2024-04-20.

Yogarshi Vyas, Spandana Gella, Jatin Sharma, Kalika Bali, and Monojit Choudhury. 2014. POS tagging of English-Hindi code-mixed social media content. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 974–979, Doha, Qatar. Association for Computational Linguistics.

Genta Indra Winata and et al. 2023. The decades progress on code-switching research in nlp: A systematic survey on trends and challenges. *arXiv preprint*.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844, Hong Kong, China. Association for Computational Linguistics.

Zeynep Yirmibeşoğlu and Gülşen Eryiğit. 2018. Detecting code-switching between Turkish-English language pair. In *Proceedings of the 2018 EMNLP Workshop W-NUT: The 4th Workshop on Noisy User-generated Text*, pages

110–115, Brussels, Belgium. Association for Computational Linguistics.

A Input Processing and Labeling for Future Code-Switch Prediction

Here's an example of how the input is processed and labeled for the future code-switch prediction task, along with the expected output.

Example Input

Let's say our input sentence is:

`" 你好 hello world 再见"`

And the corresponding languages are:

`"zh en en zh"`

Where `"zh"` represents Chinese and `"en"` represents English.

1. Tokenization

The first step is tokenization, where the sentence is split into tokens. The tokenizer might split the sentence into the following tokens. For simplicity, let's assume the tokenizer doesn't further split these tokens:

`[" 你好", "hello", "world", " 再见"]`

2. Labeling for Future Switch Prediction

For the future switch prediction task, the goal is to predict whether the next word will be a code-switch. We create labels for each token (except the last one) to indicate this.

- `" 你好"` (`zh`): The next word is `"hello"` (`en`), so this is a switch. Label: `1`
- `"hello"` (`en`): The next word is `"world"` (`en`), so this is not a switch. Label: `0`
- `"world"` (`en`): The next word is `" 再见"` (`zh`), so this is a switch. Label: `1`
- `" 再见"` (`zh`): There is no next word. Label: `-100` (This special label tells the model to ignore this position in the loss calculation).

So, the labels are: `[1, 0, 1, -100]`

3. Input to the Model

The input to the model will be:

- **Tokens:** The tokenized sequence, converted to numerical IDs by the tokenizer.
- **Attention Mask:** A mask indicating which tokens are "real" and which are padding (used to handle variable-length sequences).
- **Labels:** The sequence `[1, 0, 1, -100]`.

4. Model Output

The model will output a probability distribution for each token (except the padded ones). For each token, the model predicts the probability of the *next* token being a code-switch (`0` or `1`).

Example:

For the token `" 你好"`, the model might output `[0.1, 0.9]`. This means:

- Probability of the next word not being a switch: `0.1`

The model made correct predictions in this example. Metrics like accuracy, precision, recall, and F1-score are used to quantify the model's performance. Below are the model training parameters used for both XLM-RoBERTa (base-sized model) and Multilingual BERT (M-BERT).

```
training_args = TrainingArguments(
    output_dir="./results",
    per_device_train_batch_size=16,
    per_device_eval_batch_size=16,
    num_train_epochs=2,
    learning_rate=2e-5,
    eval_strategy="epoch",
    save_strategy="epoch",
    load_best_model_at_end=True,
    metric_for_best_model="f1",
    greater_is_better=True,
    weight_decay=0.01,
    report_to="none",
    max_seq_length=10,
)
```

B Data Description

Ascend Dataset includes the following information:

- `id`
- `path`

- audio
- transcription
- duration
- language
- original_speaker_id
- session_id
- topic

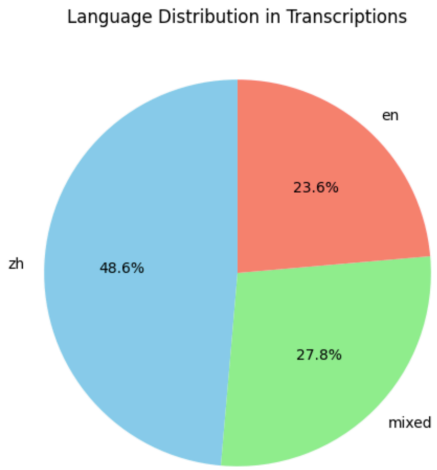

Figure 1: Language Distribution in Transcriptions. Here Mixed indicates the transcription includes both English and Chinese words, Zh indicates only Chinese and En indicates only English.

Topic Distribution in Mixed Transcriptions. Among the transcriptions labeled as code-mixed, topics related to Technology and Education appear to exhibit the most frequent instances of language switching.

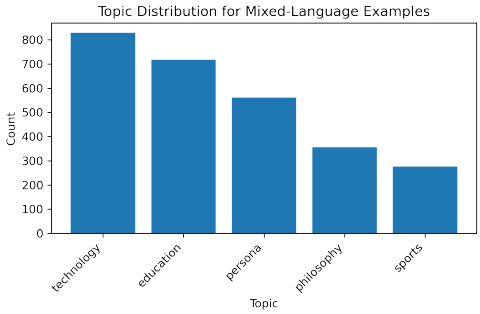

Figure 2: Topic Distribution in Transcriptions labeled as Mixed. Technologies and Education seem to be the topic where people switch between languages frequently.

C Character and Segment Level Tokenization

	windowed_text_char	language	switch_flag
0	我刚刚	mixed	0
0	刚刚开	mixed	0
0	刚开始	mixed	1
1	嗯	mixed	1
1	我的名	mixed	0

Figure 3: Tokenized Input for Character-level

	windowed_text_segment	language	switch_flag
0	我刚刚开始	mixed	1
1	嗯	mixed	1
1	我的名字	mixed	0
1	的名字叫	mixed	0
1	名字叫徐妍	mixed	0

Figure 4: Tokenized Input for Segment-level

D Full Results for Window Based Approach

The full experiments result of the Window Based Approach is shown in Table 3 and Table 4.

Table 3: Results of RNN + BERT (Chinese-to-English Flow)
– is missing value.

Window Size	Hidden Layer	Configurations	Accuracy	Precision	Recall	F1	AUC
Fixed	32	Baseline	**0.92**	0.90	0.43	0.59	0.88
Fixed	32	POS	0.91	0.92	0.35	0.51	0.90
Fixed	32	POS+dropout	0.86	–	–	–	0.78
Fixed	32	POS+dropout+LN	**0.92**	0.88	0.45	0.59	0.88
Fixed	64	Baseline	**0.92**	0.88	**0.46**	**0.60**	0.88
Fixed	64	POS	0.91	0.85	0.45	0.59	**0.91**
Fixed	64	POS+dropout	**0.92**	0.89	0.44	0.59	0.90
Fixed	64	POS+dropout+LN	0.86	–	–	–	0.60
Fixed	128	Baseline	**0.92**	0.87	0.45	**0.60**	0.90
Fixed	128	POS	**0.92**	0.89	0.43	0.58	0.90
Fixed	128	POS+dropout	**0.92**	0.87	**0.46**	**0.60**	0.89
Fixed	128	POS+dropout+LN	**0.92**	0.90	0.44	0.59	0.89
Flex	32	Baseline	0.89	–	–	–	0.43
Flex	32	POS	0.90	0.62	0.31	0.41	0.63
Flex	32	POS+dropout	0.89	–	–	–	0.52
Flex	32	POS+dropout+LN	0.89	–	–	–	0.60
Flex	64	Baseline	0.91	**1.00**	0.13	0.23	0.84
Flex	64	POS	0.91	**1.00**	0.12	0.22	0.74
Flex	64	POS+dropout	0.91	**1.00**	0.13	0.23	0.86
Flex	64	POS+dropout+LN	0.91	0.70	0.29	0.41	0.87
Flex	128	Baseline	0.91	0.74	0.24	0.36	0.86
Flex	128	POS	0.90	0.82	0.05	0.10	0.81
Flex	128	POS+dropout	0.91	0.71	0.27	0.39	0.85
Flex	128	POS+dropout+LN	0.91	0.72	0.27	0.39	0.86

Table 4: Results of RNN + BERT (English-to-Chinese Flow)
– is missing value.

Window Size	Hidden Layer	Configurations	Accuracy	Precision	Recall	F1	AUC
Fixed	32	Baseline	**0.90**	0.57	0.09	0.16	0.53
Fixed	32	POS	0.89	0.33	0.05	0.08	0.48
Fixed	32	POS+dropout	0.89	0.33	0.05	0.08	0.50
Fixed	32	POS+dropout+LN	**0.90**	**0.67**	0.05	0.09	0.49
Fixed	64	Baseline	0.89	0.33	0.02	0.04	0.52
Fixed	64	POS	0.87	0.21	0.09	0.13	0.53
Fixed	64	POS+dropout	0.88	0.25	0.07	0.11	0.55
Fixed	64	POS+dropout+LN	0.88	0.25	0.05	0.08	0.54
Fixed	128	Baseline	0.89	–	–	–	0.63
Fixed	128	POS	0.89	0.33	0.07	0.12	0.49
Fixed	128	POS+dropout	0.89	–	–	–	0.52
Fixed	128	POS+dropout+LN	0.88	0.30	0.07	0.11	0.48
Flex	32	Baseline	0.80	0.60	0.56	0.58	0.79
Flex	32	POS	0.79	0.56	**0.63**	0.59	0.80
Flex	32	POS+dropout	0.79	0.56	0.62	0.59	0.79
Flex	32	POS+dropout+LN	0.81	0.62	0.62	**0.62**	0.80
Flex	64	Baseline	0.80	0.60	0.60	0.60	0.80
Flex	64	POS	0.80	0.60	0.54	0.57	0.78
Flex	64	POS+dropout	0.79	0.57	0.58	0.57	0.80
Flex	64	POS+dropout+LN	0.80	0.58	**0.63**	0.61	0.80
Flex	128	Baseline	0.80	0.59	0.60	0.60	0.80
Flex	128	POS	0.80	0.59	0.59	0.59	0.80
Flex	128	POS+dropout	0.79	0.58	0.60	0.59	0.79
Flex	128	POS+dropout+LN	0.81	0.62	0.61	**0.62**	**0.81**