The Effectiveness of Intermediate-Task Training for Code-Switched Natural Language Understanding

Archiki Prasad†, Mohammad Ali Rehan*, Shreya Pathak*, Preethi Jyothi*
1UNC Chapel Hill 2IIT Bombay
archiki@cs.unc.edu
{alirehan, shreyapathak, pjyothi}@cse.iitb.ac.in

Abstract

While recent benchmarks have spurred a lot of new work on improving the generalization of pretrained multilingual language models on multilingual tasks, techniques to improve code-switched natural language understanding tasks have been far less explored. In this work, we propose the use of bilingual intermediate pretraining as a reliable technique to derive large and consistent performance gains using code-switched text on three different NLP tasks: Natural Language Inference (NLI), Question Answering (QA) and Sentiment Analysis (SA). We show consistent performance gains on four different code-switched language-pairs (Hindi-English, Spanish-English, Tamil-English and Malayalam-English) for SA and on Hindi-English for NLI and QA. We also present a code-switched masked language modeling (MLM) pretraining technique that consistently benefits SA compared to standard MLM pretraining using real code-switched text.

1 Introduction

Code-switching is a widely-occurring linguistic phenomenon in which multiple languages are used within the span of a single utterance or conversation. While large pretrained multilingual models like mBERT (Devlin et al., 2019) and XLM-R (Conneau et al., 2020) have been successfully used for low-resource languages and techniques to help these models generalize to code-switched text have not been sufficiently explored. Intermediate-task training (Phang et al., 2018, 2020) was recently proposed as an effective training strategy for transfer learning. This scheme involves fine-tuning a pretrained model on data from one or more intermediate tasks, followed by fine-tuning on the target task. The intermediate task could differ from the target task and it could also be in a different language. This technique was shown to help with both task-based and language-based transfer; it benefited target tasks in English (Vu et al., 2020) and helped improve zero-shot cross-lingual transfer (Phang et al., 2020).

In this work, we introduce bilingual intermediate-task training as a reliable training strategy to improve performance on three code-switched natural language understanding tasks: Natural Language Inference (NLI), factoid-based Question Answering (QA) and Sentiment Analysis (SA). Bilingual training for a language pair X-EN involves pretraining with an English intermediate task along with its translations in X. The NLI, QA and SA tasks require deeper linguistic reasoning (as opposed to sequence labeling tasks like part-of-speech tagging) and exhibit high potential for improvement via transfer learning. (The fact that NLI, QA and SA have more room for improvement compared to POS and NER tagging is evident from the leaderboard statistics in (Khanuja et al., 2020b).) We present SA results for four different language pairs: Hindi-English (Hi-EN), Spanish-English (Es-EN), Tamil-English (Ta-EN) and Malayalam-English (Ml-EN), and NLI/QA results for Hi-EN.† Our main findings can be summarized as follows:

- Bilingual intermediate-task training consistently yields significant performance improvements on NLI, QA and SA using two different pretrained multilingual models, mBERT and XLM-R. We also show the impact of translation and transliteration quality on this training scheme.
- Pretraining using a masked language modeling (MLM) objective on real code-switched text can be used, in conjunction with bilingual

---

* Equal contribution
† Work done at IIT Bombay

1These tasks present an additional challenge with the Indian languages written using transliterated/Romanized text.
training, for additional performance improvements on code-switched target tasks. We also present a code-switched MLM variant that yields larger improvements on SA compared to standard MLM.

2 Methodology

Intermediate-Task Training. This scheme starts with a publicly-available multilingual model that has been pretrained on large volumes of multilingual text using MLM-based training objectives. This model is subsequently fine-tuned using data from one or more intermediate tasks before finally fine-tuning on code-switched data from the target tasks.

Single Intermediate-Task Training makes use of existing monolingual NLI, SA and QA datasets as intermediate-tasks before fine-tuning on the respective code-switched target tasks. For a language pair X-EN, where $X \in \{\text{En, Hi, Ta, Ml}\}$, we explored the use of three different intermediate tasks:

1. Task-specific data in English (EN SING-TASK): In this setting, we carry out intermediate training using a (relatively) larger English corpus of the same task as our final down-stream task.

2. Task-specific data in X (X SING-TASK): Here, we carry out intermediate training using a corpus of the same task in the matrix language (i.e., not English) present in our code-switched corpus. This corpus can be constructed by translating a monolingual English corpus into the target language, and then further transliterating it to be consistent with the Romanized forms present in the target tasks.\(^3\)

3. Task-specific data in both English and X that we refer to as bilingual intermediate-task training (X-EN SING-TASK): This intermediate-task pretraining method involves creating training batches with an equal number of examples from both languages. We conjecture that interleaving training instances from both languages within a batch encourages the model to simultaneously perform well on both languages, and could subsequently translate to improved performance on code-switched text in these specific language pairs. This claim is borne out in our experimental results detailed in Section 4. (We also show the importance of mixing instances from both languages rather than adopting a sequential training strategy on instances from both languages in Section 4.2.)

Multi Intermediate-Task Training involves two intermediate-tasks ($T_1$ and $T_2$) simultaneously. This training is done using two different task heads (one per task) with the pretrained models. Each batch is randomly populated with instances from tasks $T_1$ or $T_2$. We follow Raffel et al. (2020) to sample batches from task $T_1$ with probability $P_{T_1} = \frac{\min(e_{T_1}, K)}{\min(e_{T_1}, K) + \min(e_{T_2}, K)}$ where $e_{T_1}$ and $e_{T_2}$ are the number of training examples in task $T_1$ and $T_2$, respectively; $P_{T_2}$ is similarly computed. The constant $K = 2^{16}$ is used to prevent over-sampling. We experiment with NLI and QA as the two intermediate-tasks $T_1$ or $T_2$ and refer to this system as Hi-EN/NLI-QA MULTI-TASK. We use the merged EN and Hi datasets from Hi-EN SING-TASK for each task. We also explored MLM training on real code-switched text as one of the tasks, in addition to the merged X-EN task-specific intermediate-tasks (referred to as X-EN/MLM MULTI-TASK).

Code-Switched MLM. A common approach to training models for code-switched tasks is to perform additional MLM on real (or synthetic) code-switched text. However, randomly masking from the pool of all tokens in a sentence may not be the most effective use of real code-switched text and differentiating it from monolingual text, especially if one has access to word-level language tags. Given word-level language labels for each token in the code-switched sentences, we aim to emphasize switching via the MLM training objective by masking tokens from words that lie on the switching boundaries. We refer to this training strategy as code-switched MLM. For example, consider the following sentence where tokens that can be masked are enclosed within boxes for both the standard MLM and code-switched MLM strategies, respectively:

\[
\begin{array}{cccccccccccc}
\text{Yeh} & \text{files} & \text{EN} & \text{ko} & \text{HI} & \text{desk} & \text{EN} & \text{pe} & \text{HI} & \text{rakh} & \text{HI} & \text{do} & \text{HI} \\
\text{Yeh} & \text{files} & \text{EN} & \text{ko} & \text{HI} & \text{desk} & \text{EN} & \text{pe} & \text{HI} & \text{rakh} & \text{HI} & \text{do} & \text{HI} \\
\end{array}
\]

(EN Translation: Put these files on the desk.)

\(^3\)Code-switched data for some Indic languages in our target corpora were only available in the Romanized form. Therefore, we only work with Romanized text in all our experiments including intermediate tasks.
In the first sentence, tokens from all the words can be masked, as in standard MLM pretraining. In the second sentence that uses code-switched MLM pretraining, only tokens from words at the boundary of a language switch can be masked. To implement this, we need access to annotated language tags for each sentence or a highly accurate language identity detection system. (Neither of these were available for Tamil or Malayalam datasets; hence our results for code-switched MLM are restricted to Hindi and Spanish.) An analysis of the MLM data showed that 45% of all tokens belonged to words on a switching boundary, therefore, the MLM masking probability of these tokens was increased from 0.15 to 0.3 to roughly balance the number of tokens that are masked on average.

3 Experimental Setup

3.1 Code-switched Target Datasets

The Hi-EN NLI dataset is from a recent code-switched benchmark GLUECoS (Khanuja et al., 2020a) comprising 1.8K/447 training/test examples, respectively. The Hi-EN factoid-based QA dataset (Chandu et al., 2018a) is also from GLUECoS, consisting of 259/54 training/test question-answer pairs (along with corresponding context), respectively. While code-switched NLI and QA tasks were only available in Hi-EN, we show SA results for four language pairs. The Es-EN SA dataset (Vilares et al., 2016) in GLUECoS consists of 2.1K/211/211 examples in train/dev/test sets, respectively. The Hi-EN SA dataset (Patwa et al., 2020) comprises 15K/1.5K/3K code-switched tweets in train/dev/test sets, respectively. The train/dev/test sets in the TA-EN SA dataset (Chakravarthi et al., 2020b) and ML-EN SA dataset (Chakravarthi et al., 2020a) comprise 9.6K/1K/2.7K and 3.9K/436/1.1K code-switched YouTube comments, respectively. As the evaluation metric, we use accuracies for NLI and SA over two (entailment/contradiction) and three labels (positive/negative/neutral), respectively, and F1 scores for the QA task.

3.2 Intermediate Task Datasets

As intermediate tasks for NLI and QA, we used EN and Hi versions of the MultiNLI dataset (Williams et al., 2018) with 250/10K examples in the train/dev sets and the SQuAD dataset (Rajpurkar et al., 2016) consisting of 82K/5K question-answer pairs in its train/dev sets, respectively. The Hi translations for SQuAD (in Devanagari) are available in the XTREME (Hu et al., 2020) benchmark. We used indic-trans (Bhat et al., 2014) to transliterate the Hi translations, since NLI and QA in GLUECoS use Romanized Hi text. For sentiment analysis in Es-EN and Hi-EN, we used the TweetEval (Barbieri et al., 2020) dataset (63K sentences in total) and its translations in Es and Hi generated via MarianMT3 (Junczys-Dowmunt et al., 2018) and Indic-Trans MT (Ramesh et al., 2021), respectively, for intermediate-task training. For TA-EN and ML-EN, we used the positive, negative and neutral labelled sentences from the SST dataset (Socher et al., 2013) (100K instances) as the intermediate task. The TA and ML translations were also generated using the IndicTrans MT system. The translations were further transliterated using Bhat et al. (2014) for Hi and the Bing Translator API4 for TA and ML.

3.3 Masked Language Modelling Datasets

We use a corpus of 64K real code-switched sentences by pooling together data from prior work (Singh et al., 2018; Swami et al., 2018; Chandu et al., 2018b); we will call this corpus GENCS. We supplant this text corpus with an additional 28K code-switched sentences mined from movie scripts (referred to as MOVIE-CS in Tarunesh et al. (2021b)), which is more similar in domain to GLUECoS NLI. We further used code-switched text from Patwa et al. (2020), Bhat et al. (2017), and Patro et al. (2017) resulting in a total of 185K Hi-EN sentences. For Es-EN, 66K real code-switched sentences were accumulated from prior work (Patwa et al., 2020; Solorio et al., 2014; AlGhamdi et al., 2016; Aguilar et al., 2018; Vilares et al., 2016). For TA-EN and ML-EN (Chakravarthi et al., 2020b, 2021; Banerjee et al., 2018; Mandl et al., 2020; Chakravarthi et al., 2020a), we used roughly 130K and 40K real code-switched sentences, respectively.

3.4 XTREME Translation-Transliteration

As mentioned previously, for intermediate-task training, we use the MultiNLI and SQuAD v1.1 data from the translate-train sets of the XTREME benchmark5.6. The Romanized version of

3Implementation used: http://bit.ly/MarianMT
6SQuAD available at: https://storage.cloud.google.com/xtreme_translations/SQuAD/
these datasets are generated using the indic-trans tool (Bhat et al., 2014) starting from their Devanagari counterparts. For NLI, we directly transliterated the premise and hypothesis. For QA, the context, question and answer were transliterated and the answer span was corrected. This was done by calculating the start and stop indices of the span, followed by a piece-wise transliteration. We finally checked if the context-span matched the answer text. All instances passed this check. To benefit future work in this direction, we provide these transliterated datasets.

3.5 Model Details

mBERT is a transformer model (Vaswani et al., 2017) pretrained using MLM on the Wikipedia corpus of 104 languages. XLM-R uses a similar training objective as mBERT but is trained on orders of magnitude more data from the CommonCrawl corpus spanning 100 languages and yields competitive results on low-resource languages (Conneau et al., 2020). We use the bert-base-multilingual-cased and xlm-roberta-base models from the Transformers library (Wolf et al., 2019). We refer readers to Appendix A and Appendix B for more implementation details.

Table 1: Our main results for sentiment analysis. Best results for each model are underlined and the overall best results are in bold. All scores are weighted averages and are further averaged over five runs with random seeds. The last row gives the F1, precision and recall value for the method with the maximum F1 across all five seeds.

4 Results and Analysis

4.1 Results on Sentiment Analysis

Table 1 shows our main results for SA on Es-EN, Hi-EN, Ta-EN and Mr-EN. We observe that bilingual intermediate-task training, X-EN SING-TASK, outperforms EN SING-TASK and X SING-TASK with both mBERT and XLM-R. The relative improvements of X-EN SING-TASK over the baseline vary across language pairs reaching up to 9.33% for Es. For all language pairs except Hi-EN, X-EN/MLM MULTI-TASK is the best-performing system. This demonstrates the benefits of MLM training in conjunction with intermediate-task training. A notable advantage of our bilingual training is that we outperform (or match) previous state-of-the-art with an order of magnitude less data. Our best Es-EN system yields an F1 of 71.7 compared to Pratapa et al., 2018) with an F1 of 64.6. For Hi-EN, our best F1 of 72.6 matches the 2nd-ranked system (Srinivasan, 2020) on SentiMix 2020 (Patwa et al., 2020). For Ta-EN and Mr-EN, our best systems match the score of the best TweetEval model in Gupta et al., 2021). While prior work required roughly 17M sentences in Es-EN, 2.09M sentences in Hi-EN and 60M tweets to train TweetEval for Ta and Ma, we use 192K, 180K, 330K and 240K sentences for the four respective language pairs. While MLM training (i.e., +MLM in Table 1) consistently improves over the baseline, we observe that code-switched MLM (i.e., CODE-SWITCHED MLM

\footnote{We also explored a multilingual model IndicBERT (Kakwani et al., 2020) trained exclusively on Indian languages. However, preliminary experiments using this model did not yield satisfactory performance, so we did not pursue it further. In future work, we will aim to use other recently released pretrained models such as MuRIL (Khanuja et al., 2021).}

\footnote{We hypothesize the drop in performance for Hi-EN could be attributed to domain differences between the SA and MLM corpora.}
Table 2: Our main results for NLI and QA from intermediate-task training. All scores are averaged over five runs with random seeds. Max and mean accuracies (for NLI) and F1-scores (for QA) over these runs are listed. Best results for each model are underlined and the overall best results are in bold. Due to dataset changes, we cannot directly cite the results from the paper and report the numbers from the leaderboard after consulting the authors of GLUECos.

<table>
<thead>
<tr>
<th>Method</th>
<th>mBERT GLUECos NLI (acc.)</th>
<th>GLUECos QA (F1)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Max</td>
<td>Mean</td>
</tr>
<tr>
<td>Baseline</td>
<td>61.07</td>
<td>57.51</td>
</tr>
<tr>
<td>+MLM</td>
<td>59.94</td>
<td>58.75</td>
</tr>
<tr>
<td>+EN SING-TASK</td>
<td>62.40</td>
<td>60.73</td>
</tr>
<tr>
<td>+HI SING-TASK</td>
<td>63.73</td>
<td>62.09</td>
</tr>
<tr>
<td>+HI-EN SING-TASK</td>
<td>65.55</td>
<td>64.1</td>
</tr>
<tr>
<td>+HI-EN/NLI-QA MULTI-TASK</td>
<td>66.74</td>
<td>63.3</td>
</tr>
<tr>
<td>+HI-EN/MLM MULTI-TASK</td>
<td>66.66</td>
<td>65.61</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Intercept-Task Paradigm</th>
<th>GLUECos NLI (acc.)</th>
<th>GLUECos QA (F1)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Max</td>
<td>Mean</td>
</tr>
<tr>
<td>EN SING-TASK</td>
<td>62.40</td>
<td>60.73</td>
</tr>
<tr>
<td>HI SING-TASK</td>
<td>63.73</td>
<td>62.09</td>
</tr>
<tr>
<td>HI-EN SING-TASK</td>
<td>65.55</td>
<td>64.1</td>
</tr>
<tr>
<td>Sequential Training: EN \rightarrow HI</td>
<td>62.02</td>
<td>59.94</td>
</tr>
</tbody>
</table>

Table 3: Sequential bilingual training with mBERT yields poor performance on both NLI and QA. Scores correspond to five random runs with random seeds.

4.2 Results on NLI and QA

NLI/QA Single Task Results. Table 2 shows our main results for the NLI and QA tasks in Hi-EN. (Code-switched benchmarks in other language pairs are not available for NLI and QA.) Among the Single-Task systems, Hi-EN SING-


does not converge and hence we do not report these scores in Table 2.

In Table 1) provides additional performance gains for Es-EN and Hi-EN. We do not report code-switched MLM results for Ta-EN and ML-EN since we do not have access to language labels or a trained language identification system for either language. The Es-EN MLM dataset contains several sentences with no switching which are discarded for both standard and code-switched MLM. In Table 1, we compare +MLM and Code-switched MLM only using sentences that contain code-switching.10

With access to translation and transliteration tools for a target language, we show superior results on four different language pairs for the sentiment analysis task. Even in resource-constrained settings like Ta-EN and ML-EN, we obtain state-of-the-art performance using our proposed techniques. In Section 4.3, we will examine the influence of translation and transliteration quality on performance.

10On using the complete En-Es MLM corpus for +MLM, we obtained an F1 of 62.57 using mBERT and 67.6 using XLM-R on the SA test set of Es-EN.

11Like Chakravarthy et al. (2020), we also find that XLM-R baseline/+MLM on GLUECos NLI does not converge and hence we do not report these scores in Table 2.
To assess the impact of both translation and transliteration quality on NLI and QA performance, we use two small datasets XNLI (Conneau et al., 2018) and XQuAD (Artetxe et al., 2020) for which we have manual Hi (Devanagari) translations. We combined the test and dev sets of XNLI to get the data for intermediate-task training. We discarded all examples labelled neutral and instances where the crowdsourced annotations did not match the designated labels. After this, we were left with roughly 4.2K/0.5K instances in the train/dev sets, respectively (the dev set is used for early stopping during intermediate-task training). For XNLI, the premises and hypotheses were directly translated and for XQuAD we adopted the same translation procedure listed in Hu et al. (2020).

Table 4, 5: Effect of translation and transliteration quality on intermediate-task training, using Hi SING-TASK for NLI and QA. Scores correspond to five random runs with random seeds.

To verify the impact of domain similarity on NLI and QA performance, we use two small datasets XNLI (Conneau et al., 2018) and XQuAD (Artetxe et al., 2020) for which we have manual Hi (Devanagari) translations. We combined the test and dev sets of XNLI to get the data for intermediate-task training. We discarded all examples labelled neutral and instances where the crowdsourced annotations did not match the designated labels. After this, we were left with roughly 4.2K/0.5K instances in the train/dev sets, respectively (the dev set is used for early stopping during intermediate-task training). For XNLI, the premises and hypotheses were directly translated and for XQuAD we adopted the same translation procedure listed in Hu et al. (2020).

Table 4, 5: Effect of translation and transliteration quality on intermediate-task training, using Hi SING-TASK for NLI and QA. Scores correspond to five random runs with random seeds.

4.3 Influence of Translation and Transliteration Quality

Transliteration and translation are the two key pre-processing steps employed for bilingual pretraining. Since we make use of existing translation and transliteration tools that are not error-free, it is useful to understand the impact of such translation and transliteration tools on final downstream task performance.
Table 6: Effect of transliteration quality of intermediate-tasks on SA results. Scores are weighted averages further averaged over 5 random runs.

Table 6 shows the impact of transliteration on sentiment analysis of TA-EN and ML-EN. Again, we see that using an improved transliteration tool led to improved performance across both Tamil and Malayalam.

Figure 1 illustrates different MA transliterations. From the figure, we notice that indic-trans tends to retain some Malayalam characters in its native script (possibly due to incomplete Unicode support) and also does not produce very accurate transliterations. Transliterations from the Bing API are more phonetically accurate. Table 7 shows an example from the Hi-EN NLI dataset, that is translated and transliterated using Google Translate and indic-trans. The color-coded transliterations indicate that indic-trans often uses existing English words as transliterations. While this is helpful for some specific (uncommon) words, in most cases it leads to ambiguity in sentence meaning (shown in blue). Further, these ambiguous words are far more common in the Hi language, and thus have a greater impact on model performance.

In summary, developing more accurate tools for translation and transliteration would be very beneficial for downstream code-switched tasks.

4.4 MLM and Intermediate-Task Training

How does MLM pretraining in conjunction with intermediate-task training impact performance? What is the influence of changing the MLM corpus (and hence its domain) on final task performance?

We address these questions in this section by focusing on NLI and QA for Hi-EN using mBERT.

Table 8 provides a summary of our experiments on intermediate-task training of mBERT using only English (EN) and both English and Hindi (HI-EN) in conjunction with MLM in the MULTI-TASK setting described in Section 2.

From Table 8, we observe that intermediate training using MLM on code-switched data alone (i.e., the first row for each task) is not as effective as using both MLM and intermediate-task pretraining. NLI benefits from MLM in a multi-task setup in both monolingual and bilingual settings. Further, we note that adding in-domain MOVIE-CS data yields additional improvements for NLI. This shows that sufficient amount of in-domain data is needed for performance gains, and augmenting out-of-domain with in-domain code-switched text can be effective. In the case of QA, MLM does not improve performance in the monolingual setting, although the mean scores are statistically close. In the bilingual setting, we see a clear improvement using GEN-CS for MLM training. However, using both GEN-CS and MOVIE-CS for MLM results in significant degradation of performance. We believe that this is due to the domain of the passages in GLUECO5 QA being similar to the HI-EN blog data present in GEN-CS. However, the MOVIE-CS dataset comes from a significantly different domain and thus hurts performance. This indicates that in addition to the amount of unlabelled real code-switched text, when using MLM training, the domain of the text is very influential in determining the performance on downstream tasks (Gururangan et al., 2020).

For both NLI and QA, we observe the following common trend: Adding code-switched data from the training set of GLUECO5 tasks (referred to as GLUECO5 NLI CS and GLUECO5 QA CS) degrades performance. This could be due to the quality of training data in the GLUECO5 tasks. Each dialogue in the NLI data does not have a lot of content and is highly conversational in nature. In addition to this, the dataset is also very noisy. For example, a word ‘humko’ is split into its characters ‘h u m k o’. Thus,
MLM Data
Max  Mean  Std.
1.93  79.63  78.27  1.46
-65.61  80.38
G 1.68  0.86
77.62  65.22  64.19  1.22
81.05  79.11  1.40
62.84  62.40  60.73  1.78
75.77  83.03
G 65.55  66.17  65.21  0.96
59.94  58.75  0.93
59.26  57.84  1.29
66.67  81.61  79.97  1.29

Intermediate-task training of mBERT. We underline Winata et al. (2021) show that embeddings from pretrained multilingual models are not very effective for code-switched tasks and more work is needed to effectively adapt them.

Table 7: NLI examples from some of our datasets. †: obtained by translation of the second row using Google Translate API, ⋄: transliterated using Google Translate API, ⋆: transliterated using indic-trans (Bhat et al., 2014). In blue, we show some of the words with ambiguous transliterations by indic-trans. In purple, we show some words that are better transliterated by indic-trans. Best viewed in color.

Table 8: Performance on different variations of MLM + intermediate-task training of mBERT. We underline the relatively best model and bold-face the model with the highest performance for each task.

MLM on such data may not be very effective and could hurt performance. For QA, passages in significant portions of the train set are obtained using DrQA - Document Retriever module15 (Chen et al., 2017). These passages are monolingual in nature and thus potentially not useful for MLM training with code-switched text.

5 Related Work

While pretrained multilingual models are being increasingly used for cross-lingual natural language understanding tasks, their effectiveness for code-switched tasks has not been thoroughly explored. Winata et al. (2021) show that embeddings from pretrained multilingual models are not very ef-

15https://github.com/facebookresearch/DrQA

Table 8: Performance on different variations of MLM + intermediate-task training of mBERT. We underline the relatively best model and bold-face the model with the highest performance for each task.
requiring significantly smaller amounts of data. Instead of using mBERT and XLM-R that are very broad in their coverage of languages, it would be interesting to examine whether our observed trends hold when using pretrained models specifically trained for the chosen target languages. We could consider using very recent models like IndicBERT (Kakwani et al., 2020) and MuRIL (Khanuja et al., 2021) that are trained exclusively on Indian languages and have been shown to outperform mBERT on cross-lingual tasks (e.g., XTREME) and tasks like IndicGLUE, respectively. We leave this investigation for future work.

6 Conclusion

This is the first work to demonstrate the effectiveness of intermediate-task training for code-switched NLI, QA and SA on different language pairs, and present code-switched MLM that consistently benefits SA more than standard MLM. We also carry out ablations of transliteration systems and compare their performance across the same corpora translated using different techniques. We observe that high-quality translations and transliterations are important to derive performance improvements on downstream tasks. For future work, we plan to continue exploring pretraining strategies, based on more informed masking objectives and task-adaptive techniques. One key limitation of the newly introduced code-switched MLM approach is the requirement of LID systems for the languages under consideration. Future work can focus on mitigating this requirement.

7 Acknowledgements

The authors would like to thank the anonymous reviewers for their useful and constructive comments that helped improve the draft.
References


Appendix

A Implementation Details

The mBERT model comprises 179M parameters with the MLM head comprising 712K parameters. The XLM-R model comprises 270M parameters with an MLM head with 842k parameters. For both models, the NLI (sequence classification) and QA heads comprise 1536 parameters each. For SA (sequence classification) the head comprises of 2304 parameters.

B Hyperparameter Tuning

In all experiments, we have used the AdamW algorithm (Loshchilov and Hutter, 2019) and a linear scheduler with warm up for the learning rate. These experiments were run on a single NVIDIA GeForce GTX 1080 Ti GPU. Some crucial fixed hyperparameters are: learning_rate = 5e-5, adam_epsilon = 1e-8, max_gradient_norm = 1, and gradient_accumulation_steps = 10.

B.1 Intermediate-Task Training

The training for all the main intermediate-task experiments was carried out for 4 epochs and the model with the highest performance metric on the task dev set was considered (all the metrics stagnated after a certain point in training). For NLI + QA tasks, two separate models were stored depending on the performance metric on the respective dev set. No hyperparameter search was conducted at this stage. During bilingual training, the batches were interspersed—equal number of examples from English and Romanized H1 within each batch. In the single-task systems, we used batch_size = 8 and max_sequence_length = 128 for NLI, batch_size = 8 and max_sequence_length = 256 for SA, batch_size = 4 and max_sequence_length = 512 for QA. During multi-task training, the max_sequence_length was set to the maximum of the aforementioned numbers and the respective batch-sizes. Any multi-task training technique requires at least 14-15 hours for validation accuracy to stagnate. Single task intermediate training requires 4-5 hours for monolingual versions and 8-9 hours for the bilingual version. SA data being smaller in size requires 8-9 hours for multitask, 4-5 hours for bilingual intermediate task and 1-2 hours for monolingual intermediate task. The logging_steps are set to approximately 10% of the total steps in an epoch.

B.2 Fine-tuning on GLUECoS NLI & QA Tasks

The base fine-tuning files have been taken from the GLUECoS repository. Given that there no dev sets in GLUECoS, and that the tasks are low-resource, we use train accuracy in NLI and train loss in QA as an indication to stop fine-tuning. Manual search is performed over a range of epochs to obtain the best test performance. For NLI, we stopped fine-tuning when training accuracy is in the range of 70-80% (which meant fine-tuning for 1-4 epochs depending upon the model and technique used). For QA, we stopped when training loss reached ~ 0.1. Thus, we explored 3-5 epochs for mBERT and 4-8 epochs for XLM-R. We present the statistics over the best results on 5 different seeds. We used batch_size = 8 and max_sequence_length = 256 for GLUECoS NLI and batch_size = 4 and max_sequence_length = 512 for GLUECoS QA. All our fine-tuning runs on GLUECoS take an average of 1 minute per epoch.

B.3 Fine-tuning on downstream SA tasks

The dev set, being available for all language pairs was used to find the checkpoint with best F1 score, and this model was used for evaluation on the test set. The mean values were presented after carrying out the above procedure for 6 different seeds. The logging_steps are set to approximately 10% of the total steps in an epoch. Each epoch takes around 1 minute for TA, MA and ES, 2 minutes for H1 (SemEval).

C Example Outputs

In Table 9, we show some instances from the H1-EN QA dataset. The color-coded transliterations indicate that indic-trans often uses existing English words as transliterations. While for some specific (uncommon) words that is helpful, in most cases it leads to ambiguity in the sentence meaning (shown in blue). Further, these ambiguous words (in blue) are far more common in the H1 language, and thus, have a greater impact on model performance. We also note that transliterations of these common words in the GLUECoS dataset matches closely with the transliterations produced using the
Table 10: Sentiment analysis examples from our datasets.

<table>
<thead>
<tr>
<th>Language</th>
<th>QA Context</th>
<th>Label</th>
<th>Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>EN</td>
<td>It’s definitely Christmas season! My social media news feeds have been all about Hatchimals since midnight! Good luck parents!</td>
<td>positive</td>
<td>TweetEval</td>
</tr>
<tr>
<td>Hi</td>
<td>ये निश्चित रूप से नवंबर का मौसम है! मेरी सोशल मीडिया न्यूज़ फीड केवल Hatchimals की लोकप्रियता पर लग रही है। अच्छी झुकाव! बच्चों के लिए बेहद फायदेमय</td>
<td>positive</td>
<td>Translation‡</td>
</tr>
<tr>
<td>ES</td>
<td>¡Es definitivamente la temporada de Navidad! Mis noticias en las redes sociales han sido todo acerca de Hatchimals desde medianoche! ¡Buena suerte padres!</td>
<td>positive</td>
<td>Translation‡</td>
</tr>
<tr>
<td>ML</td>
<td>इस तरह के मज़कूर नए सोर्स नये मेवाई, क्या नए सोशल मीडिया न्यूज़ फीड आदि रात से हच्चिमिल के बारे में हैं। अच्छी सफाई! बच्चों के लिए बहुत फायदेमय</td>
<td>positive</td>
<td>Translation‡</td>
</tr>
<tr>
<td>TA</td>
<td>இது தோன்றும் கையல் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிரம்ஸ் சிர</td>
<td>positive</td>
<td>Translation‡</td>
</tr>
<tr>
<td>EN</td>
<td>The story and the friendship proceeds in such a way that you’re watching a soap opera rather than a chronicle of the ups and downs that accompany lifelong friendships.</td>
<td>negative</td>
<td>SST</td>
</tr>
<tr>
<td>Hi</td>
<td>कहानी और दोस्ती के साथ इस तरह के हैं कि आप एक सोपा ऑपरा के बजाए एक चरित्र का दृष्टिकोण देख रहे हैं।</td>
<td>negative</td>
<td>Translation‡</td>
</tr>
<tr>
<td>ES</td>
<td>La historia y la amistad proceden de tal manera que estás viendo una telenovela en lugar de una crónica de los altibajos que acompañan a las amistades de toda la vida.</td>
<td>negative</td>
<td>Translation‡</td>
</tr>
<tr>
<td>ML</td>
<td>இந்த கதையும் முற்பகுதியில் கதையும் முற்பகுதியில் கதையும் முற்பகுதியில் கதையும் முற்பகுதியில் கதையும் முற்பகுதியில் கதையும் முற்பகுதியில் கதையும் முற்பகுதியில் கதையும் முற்பகுதியில் கதையும் முற்பகுதியில் கதையும் முற்பகுதியில் கதையும் முற்ப</td>
<td>negative</td>
<td>Translation‡</td>
</tr>
<tr>
<td>TA</td>
<td>கதையும் நடுக்கும் முழுவாக்க முழுவாக்க முழுவாக்க முழுவாக்க முழுவாக்க முழுவாக்க முழுவாக்க முழுவாக்க முழுவாக்க முழுவாக்க முழுவாக்க முழுவாக்க முழுவாக்க முழுவாக்க முழுவாக்க முழுவாக்க முழுவாக்க முழு</td>
<td>negative</td>
<td>Translation‡</td>
</tr>
</tbody>
</table>

Table 10: Sentiment analysis examples from our datasets.

<table>
<thead>
<tr>
<th>Language</th>
<th>Sentence</th>
<th>Label</th>
<th>Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>ES</td>
<td>Mitashi ha llevado un Android TV a la venta. Precio de 51,990 Rs. Ha sido lanzado Android TV Mitashi Company hace Android KitKat OS en televisores inteligentes de 50 pulgadas. Tiene resolución Full HD y soporta 27 formatos de video. Tiene entrada de video HDMI, PC, Wi-Fi y Ethernet. Un mejor procesamiento con procesador dual-core con 512 MB de RAM. Es el primer Android Gaming Device en la India.</td>
<td>Positive</td>
<td>GLUECOSQA</td>
</tr>
</tbody>
</table>

Google Translate API. Further, there is not a lot of difference between the machine and human translations, which might be due to translation bias. Table 10 shows examples from the sentiment analysis datasets in Hi-EN, Es-EN, Ta-EN and Ml-EN.