Does Transliteration Help Multilingual Language Modeling?

Ibraheem Muhammad Moosa
Pennsylvania State University
ibraheem.moosa@psu.edu

Mahmud Elahi Akhter and Ashfia Binte Habib
Dept. of Electrical and Computer Engineering,
North South University, Dhaka, Bangladesh
{mahmud.akhter01,ashfia.habib}@northsouth.edu

Abstract

Script diversity presents a challenge to Multilingual Language Models (MLLM) by reducing lexical overlap among closely related languages. Therefore, transliterating closely related languages that use different writing scripts to a common script may improve the downstream task performance of MLLMs. We empirically measure the effect of transliteration on MLLMs in this context. We specifically focus on the Indic languages, which have the highest script diversity in the world, and we evaluate our models on the IndicGLUE benchmark. We perform the Mann-Whitney U test to rigorously verify whether the effect of transliteration is significant or not. We find that transliteration benefits the low-resource languages without negatively affecting the comparatively high-resource languages. We also measure the cross-lingual representation similarity of the models using centered kernel alignment on parallel sentences from the FLORES-101 dataset. We find that for parallel sentences across different languages, the transliteration-based model learns sentence representations that are more similar.

1 Introduction

In the last few years, we have seen impressive advances in many NLP tasks. These advances have been primarily led by the availability of large representative corpora and improvement in the architecture of large language models. While improving model architectures, training methods, regularization techniques, etc., can help advance the state of NLP in general, the unavailability of large, diverse corpora is the bottleneck for most languages (Joshi et al., 2020). Thus to inclusively advance the state of NLP across languages, it is crucial to develop techniques for training MLLMs that can extract the most out of existing multilingual corpora. Here, we focus on the issue of diverse writing scripts used by closely related languages that may prevent MLLMs from learning good cross-lingual representations. Previous papers (Pfeiffer et al., 2021) have noted that low-resource languages that use unique scripts tend to have very few tokens representing them at the tokenizer. As a result, these languages tend to have more UNKnown tokens, and the words in these languages tend to be more split up by subword tokenizers. Often we can easily transliterate from one script to another using rule-based systems. For example, there are established standards that can be used to transliterate Greek (ISO 843), Cyrillic (ISO 9), Indic scripts (ISO 15919), and Thai (ISO 11940) to the Latin script.

In this paper, we focus on the Indic languages, which have the highest script diversity in the world. Many South Asian and Southeast Asian languages are intimately connected linguistically, historically, phonologically (Littell et al., 2017) and phylogenetically. However, due to different scripts, it is difficult for MLLMs to fully exploit this shared information. Among the Indic languages we considered in this study we encounter eleven different scripts. These are shown in Table 1. Nevertheless, these scripts have shared ancestry from the ancient Brahmic script (Hockett et al., 1997; Coningham et al., 1996) and have similar structures that we can easily use to transliterate them to a common script. Also, many of these languages heavily borrow from Sanskrit, and due to its influence, many words are shared among these languages. Therefore, due to their relatedness and highly diverse script barrier, the Indic languages presents a unique opportunity to analyze the effects of transliteration on MLLMs.

We empirically measure the effect of transliteration on the downstream performance of MLLMs. We pretrain ALBERT (Base, 11M Parameters) (Lan et al., 2020) and RemBERT (Base, 192M Parameters) (Chung et al., 2020) models from scratch.
on Indic languages. We pretrain two variants of each model, one with the original writing scripts and the other after transliterating to a common writing script. Henceforth, we will refer to the transliterated script model as uni-script model and the other as a multi-script model. We evaluate the models on downstream tasks from the IndicGLUE benchmark dataset (Kakwani et al., 2020). In order to rigorously compare the two models, we finetune using nine random seeds on all downstream tasks. Then we perform the Mann-Whitney U test (MWU) between the uni-script and multi-script models. Using the MWU test, we conclude that transliteration significantly benefits the low-resource languages without negatively affecting the comparatively high-resource languages.

We also measure the Cross-Lingual Representation Similarity (CLRS) to understand why the uni-script model performs better than the multi-script model. To measure the CLRS, we use the centered kernel alignment (CKA) (Kornblith et al., 2019) similarity score. We measure the CKA similarity score between the hidden representations of the models on the parallel sentences of the Indic languages from the FLORES-101 dataset (Goyal et al., 2022). We find that, compared to the multi-script models, the uni-script models achieve a higher CKA score, and it is more stable throughout the hidden layers of the models. Based on this, we conclude that the uni-script models learn better cross-lingual representation than the multi-script models. In summary, our contributions are primarily three-fold:

1. We find that transliteration significantly benefits the low-resource languages without negatively affecting the comparatively high-resource languages.

2. We establish this finding through rigorous experiments and show the statistical significance along with the effect size of transliteration using the Mann-Whitney U test.

3. Using CKA on the FLORES-101 dataset, we show that transliteration helps MLLMs learn better cross-lingual representation.

Our code is available at Github\(^1\) and our model weights can be downloaded from HF Hub\(^2\)\(^3\)\(^4\)\(^5\).

2 Motivation and Background

2.1 Motivation

In their study, Joshi et al. (2020) showed the resource disparity between low-resource and high-resource languages, and Ruder (2020) discussed the necessity of working with low-resource languages. A large body of work suggests that language-relatedness can help MLLMs achieve better performance on low-resource languages by leveraging related high-resource languages. For instance, Pires et al. (2019) found that lexical overlap improved mBERT’s multilingual representation capability even though it learned to capture multilingual representations with zero lexical overlaps. Dabre et al. (2017) showed that transfer learning in the same or linguistically similar language family gives the best performance for NMT. Lauscher et al. (2020) found that language relatedness is crucial for POS-tagging and dependency parsing tasks. Although, corpus size is much more important for NLI and Question Answering tasks. Wu and Dredze (2020) showed that bilingual BERT outperformed monolingual BERT on low-resource languages when the languages were linguistically closely related. Nevertheless, mBERT outperformed bilingual BERT on low-resource languages.

2.2 Script Barrier in Multilingual Language Models

One of the major challenges in leveraging transfer between high-resource and low-resource languages is overcoming the script barrier. Script barrier exists when multiple closely related languages use different scripts. Anastasopoulos and Neubig (2019) found that for morphological inflection, script barrier between closely related languages impedes cross-lingual learning, and language relatedness improved cross-lingual transfer. Transliteration and phoneme-based techniques have been proposed to solve this issue. For example, Murikinati et al. (2020) expanded upon Anastasopoulos and Neubig (2019) and showed that both transliter-
ation and grapheme to phoneme (g2p) conversion removes script barrier and improves cross-lingual morphological inflection and Rijhwani et al. (2019) showed that pivoting low-resource languages to their closely related high-resource languages results in better zero shot entity linking capacity and used phoneme-based pivoting to overcome the script barrier. Bharadwaj et al. (2016) showed that phoneme representation outperformed orthographic representations for NER. Chaudhary et al. (2018) also used phoneme representation to resolve script barriers and adapt word embeddings to low-resource languages.

2.3 Transliteration in Language Modeling

Different works have applied transliteration in different aspect for language models. For instance, Goyal et al. (2020) and Song et al. (2020) both utilized transliteration and showed that language relatedness was required for improving performance on NMT. Amrhein and Sennrich (2020) studied how transliteration improved NMT and came to the conclusion that transliteration offered significant improvement for low-resource languages with different scripts.

Khemchandani et al. (2021) showed on Indo-Aryan languages that language relatedness could be exploited through transliteration along with bilingual lexicon-based pseudo-translation and aligned loss to incorporate low-resource languages into pretrained mBERT. Muller et al. (2021) showed that for unseen languages, the script barrier hindered transfer between low-resource and high-resource languages for MLLMs and transliteration removed this barrier. They showed that transliterating Uyghur, Buryat, Erzya, Sorani, Meadow Mari, and Mingrelian to Latin script and finetuning mBERT on the respective corpus with masked language modeling objective improved their downstream POS performance significantly. In contrast, K et al. (2020) and Artetxe et al. (2020) proposes that mBERT can learn cross-lingual representations without any lexical overlap, a shared vocabulary, or joint training. However, these works focus on zero-shot cross-lingual transfer learning only. From the literature, it can be seen that many in the community believe transliteration to be a potential solution for script barriers. However, most of the work shows the benefits of transliteration for NMT. Nevertheless, there is no solid empirical analysis of the effects of transliteration for MLLMs apart from Dhamecha et al. (2021); Muller et al. (2021). Hence, the motivation behind this paper is to provide a solid empirical analysis of the effect of transliteration for MLLMs with statistical analysis and determine whether or not it helps models learn better cross-lingual representation.

It should also be noted that, even though our idea seems to be similar to Muller et al. (2021) and Dhamecha et al. (2021), there are major differences. For instance, Muller et al. (2021) adapted existing pretrained model to very low-resource languages. Whereas, we focus on training the models with transliteration from scratch. We also train our models on 20 languages and evaluate on more than 50 tasks. Unlike Dhamecha et al. (2021), we also include Dravidian Languages in our analysis. Furthermore, we focused on the issue of script barrier while Dhamecha et al. (2021) focused on multilingual fine-tuning. Whereas, we adopt multilingual fine-tuning on all our models. Thus the improvement we see comes only from circumventing the script barrier. Moreover, we have provided statistical testing to show the significance of transliteration instead of just showing better metrics. We also performed cross-lingual representation similarity analysis to show the benefits of transliteration.

2.4 Cross Lingual Similarity Learning in Language Modeling

Several techniques have recently been used to study the hidden representations of multilingual language models. Kudugunta et al. (2019) study CLRS of NMT models using SVCCA (Raghu et al., 2017). Singh et al. (2019) used PWCCA (Morcos et al., 2018) to study the CLRS of mBERT and found that it drastically fell with depth. (Conneau et al., 2020) have used CKA to study the CLRS of bilingual BERT models. They found that similarity is highest in the first few layers and drops moderately with depth. Müller et al. (2021) used CKA to study CLRS of mBERT before and after finetuning on downstream tasks. They found in all cases that CLRS increases steadily in the first five layers, then it decreases in the later layers. From this, they concluded that mBERT learns multilingual alignment in the early layers and preserves it throughout finetuning. Del and Fishel (2021) applied various similarity measures to understand CLRS of various multilingual masked language models. Their results also show that CLRS increases in the first half of the models, while in the later layers, this
similarity steadily falls.

3 Experiment and Results

3.1 Mann–Whitney U test

We perform Mann–Whitney U test (MWU) (Mann and Whitney, 1947; Wilcoxon, 1945) to determine if the performance differences between the multi-script and the uni-script models are significant. In short, it tells us the effect of transliteration on model performance. MWU is a non-parametric hypothesis test between two groups/populations. MWU is chosen because it has weak assumptions. The only assumptions of MWU are that the samples of the two groups are independent of each other, and the samples are ordinal. Under the MWU, our null hypothesis or $h_0$ is that the performances of the uni-script (group 1) and the multi-script (group 2) models are similar, and the alternative hypothesis or $h_a$ is that the performances (groups) are different. We set our confidence interval $\alpha$ at 0.05 and reject the $h_0$ for the p-values $< \alpha$. We also report three test statistics as the p-value only gives statistical significance, which can be misleading at times (Sullivan and Feinn, 2012).

The test statistics are three different effect sizes that convey three different information. These test statistics are absolute effect size ($\delta$), common language effect size ($\rho$), and standardized effect size ($r$). The absolute effect size $\delta$ is the difference between the mean of the models’ performance metric, which is given as,

$$\delta = \mu_{uni-script} - \mu_{multi-script}$$

for any given task and language. When the $h_0$ is rejected for any given task, a positive $\delta$ indicates the uni-script model is better, and a negative $\delta$ indicates the multi-script model is better. The details and results of common language effect size ($\rho$), and standardized effect size ($r$) are presented in appendix D.

3.2 Dataset

The ALBERT models were pretrained on a subset of the OSCAR corpus containing Indo-Aryan languages. We use the unshuffled deduplicated version of OSCAR corpus (Ortiz Su’arez et al., 2019) available via Huggingface datasets library (Lhoest et al., 2021). We pretrain on Panjabi, Hindi, Bengali, Oriya, Assamese, Gujarati, Marathi, Sinhala, Nepali, Sanskrit, Goan Konkani, Maithili, Bihari, and Bishnupriya portion of the OSCAR corpus.

The RemBERT models were trained on a significantly larger pretraining corpus with additional languages. We pretrained the RemBERT models on a combination of Wikipedia (Foundation), mC4 (Raffel et al., 2019), OSCAR2109 (Abadji et al., 2021) and OSCAR corpus. These datasets are also available via the Huggingface datasets library. In addition to the languages in the ALBERT pretraining corpus, we consider English, four Dravidian languages Kannada, Telugu, Malayalam, and Tamil, and an Indo-Aryan language Dhivehi. We evalu-
peration of the Aksharamukha library to be better. Thus we use this library for transliterating the inputs to the ALBERT uni-script model. However, the Aksharamukha implementation is very slow compared to the PyICU implementation. As we significantly expanded our pretraining corpus for the RemBERT model, we switched to PyICU for the RemBERT uni-script model.

3.4 Downstream Finetuning

We finetune the models on each downstream task independently. The specific hyperparameters used for each task are reported in the appendix B. On all tasks, we finetune with nine random seeds and report the average and standard deviation of the metrics. In Table 2 and Table 4, we report the performances on IndicGLUE benchmark tasks and in Table 3 on other publicly available datasets. Here, we discuss the results on each of the the models on each of the tasks. Furthermore, in appendix D, we show the test statistics for all the datasets.

Wikipedia Section Title Prediction: For both RemBERT and ALBERT models, the uni-script model performed better on all languages except Malayalam (ml). We noticed that a letter of Malayalam script is not properly transliterated by the PyICU library. This introduced some artifacts in the form of unnecessary splitting of words by the subword tokenizer.

News Category Classification: It is interesting that on this task the uni-script models performed better for Panjabi (pa) and Oriya (or) languages. It is clear from Table 1 that these two languages are low-resource compared to Bengali (bn) and Marathi (mr). On Bengali and Marathi we see slight performance degradation which is not statistically significant. This shows the validity of our first finding.

Named Entity Recognition: On this task we see that the uni-script model performs much better for Assamese (as), Oriya(or), Panjabi (pa) and Gujarati (gu). These languages are low-resource and here again the uni-script model shines. The large performance improvement on this task can be explained by the fact that Named Entities usually have the same spelling after transliteration for Indian languages. Thus the uni-script model has better chances for learning various named-entities during pre-training.

Article Genre, Sentiment & Discourse Mode Classification: We evaluate the models on various other sequence classification datasets that are part of the IndicGLUE benchmark. Here again the uni-script model usually performs better than the multi-script model. However for two tasks in Malayalam (ml) and Tamil (ta) we see better performance for the multi-script model. We already mentioned that there is some tokenization issue for Malayalam which can explain the results for Malayalam. The results for Tamil suggests that it may be a good idea to try both uni-script and multi-script model if they are available to see which performs best on a particular task. However this is the only instance of a task where we see the multi-script model perform better.

3.5 Zero Shot Capability Testing

We use the CSQA task to test the zero-shot capability of the models as we can use the models without finetuning. This task is designed to test whether language models can be used as knowledge bases (Petroni et al., 2019). In Table 4 we report the results. We note that the RemBERT models perform much better than the ALBERT models on this task. This is expected as the ALBERT models’ memorization capability is hampered by weight sharing.

The ALBERT uni-script model is better on all languages compared to the ALBERT multi-script model. This shows the potential of a uni-script model in a restricted low parameter situation. For the RemBERT models, the results are mixed. However, on average the uni-script model performs better than the multi-script model. The worst results are for Malayalam (ml) which as we mentioned before has some tokenization issues.

4 Cross-lingual Representation Similarity

In this section, we analyze why the uni-script model performs better than the multi-script model from the perspective of Cross-Lingual Representation Similarity. Following (Müller et al., 2021), (Conneau et al., 2020) and (Del and Fishel, 2021) we apply CKA to measure CLRS. We use the CKA implementation from the Eccolo library (Alammar, 2021). We use parallel sentences on thirteen languages from the FLORES-101 (Goyal et al., 2022) dataset. For the ALBERT models, which are trained on only the Indo-Aryan languages, we only consider Panjabi, Hindi, Bengali, Oriya, Assamese, Gujarati, Marathi, and Nepali sentences. For the RemBERT models, we additionally consider Kannada, Telugu, Malayalam, Tamil, and English sentences.
First, we calculate the sentence embeddings of these parallel sentences from the models. Sentence embedding is calculated by averaging the hidden state representations of the tokens. Then, we calculate the CKA similarity score between the sentence embeddings for each language pair. For each language, we average its CKA similarity scores. In Figure 1 we plot this average CKA similarity for each layer of the models.

We see that CLRS score drops significantly at the last layer for all models. However, the uni-script models retain high CLRS score until the eleventh layer, whereas the multi-script models have low CLRS score from the ninth layer. Overall the CLRS score of the uni-script models are more stable. This indicates that the uni-script models have learned
better cross-lingual representations.

5 Tokenizer Quality Analysis

In terms of performance, we expect the transliteration model to exploit better tokenization across the languages. Following (Ács, 2019) and (Rust et al., 2021), we measure the subword fertility (average number of tokens per word) and the ratio of words unbroken by the tokenizer. From figure 2, we can see that transliteration reduces the splitting of words. This indicates that many words that were represented by different tokens in the multi-script model are represented by a single token in the transliteration model. On average, the ALBERT uni-script tokenizer has a lower subword fertility score of 1.55 compared to the multi-script tokenizer’s 1.825. The uni-script tokenizer also has a lower proportion of continued word score of 0.36 while the multi-script tokenizer has a score of 0.45.

6 Conclusion and Future Work

In this paper, we show that transliterating closely related languages to a common script improves multilingual language model performance and leads to better cross-lingual representations. We conducted rigorous statistical analysis to quantify the significance and effect size of transliteration on downstream task performance. We found that transliteration especially improves performance on comparatively low-resource languages and did not hurt the performance on high-resource languages. This findings are in agreement with (Dhamecha et al., 2021; Muller et al., 2021). Our results indicate that in other scenarios where closely related languages use different scripts, transliteration can be used to improve the performance of language models. For example, Slavic and Turkic languages present similar scenarios. We would like to extend our study to models at different scales and more languages in the future. Also, another interesting future direction would be to just use the transliteration for pretraining signal but give the model the ability to deal with the original scripts.

Limitations

A limitation of our work is that it introduces a transliteration step into the model pipeline. Thus we need a stable implementation of the transliteration scheme. Thus the model can become tied to a specific version of the transliteration library. Also
the transliteration scheme is not perfect as we saw for Malayalam, it introduced some artifacts. Finally given our limited computational budget, we could not run experiments with a lot of models at different scales. Thus the impact of transliteration over different model scales has not been explored. Even though our work has these limitations, it clearly shows transliteration as an important tool for training better multilingual models.

**Ethics Statement**

In their study, Joshi et al. (2020) showed the resource disparity between low-resource and high-resource languages, and (Ruder, 2020) also highlighted the necessity of working with low-resource languages. However, creating representative and inclusive corpora is a difficult task and an ongoing process and is not always possible for many low-resource languages. Thus to inclusively advance the state of NLP across languages, it is crucial to develop techniques for training MLLMs that can extract the most out of existing multilingual corpora.

Hence, we believe our analysis might help MLLMS with low-resource languages in real-world applications. However, there is one ethical issue that we want to state explicitly. Even though we pre-train on a comparatively large multilingual corpus, the model may exhibit harmful gender, ethnic and political bias. If the model is fine-tuned on a task where these issues are important, it is necessary to take special consideration when relying on the model’s decisions.

**References**


Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu,
A Cloze Style QA Evaluation Method

Since a word can be tokenized to multiple tokens by the subword tokenizer, correctly evaluating the model on this task requires special care. Specifically, we have to use the same number of mask tokens as the number of subword tokens that a word gets split into. Then we calculate the probability for the word by multiplying the probability of the subword tokens predicted by the masked language model.

B Pretraining Details

Corpus Preparation: Since the OSCAR corpus contains raw text from the Web, we apply a few filtering and normalization. First, we discard entries where the dominant script does not match the language tag provided by the OSCAR corpus. Then we use the IndicNLP normalizer (Kunchukuttan, 2020) to normalize the raw text. For the uniscript model, we then transliterate all the text to ISO-15919 format using the Aksharamukha (Rajan, 2015) library.

For the RemBERT models we do not perform any of the filtering mentioned above since our pretraining corpus is comparatively very large. In this case, we use the PyICU library (PyICU) for transliterating to ISO-15919 format.

Tokenizer Training: For the ALBERT models, we train two SentencePiece tokenizers (Kudo and Richardson, 2018) on the transliterated and the non-transliterated corpus with a vocabulary size of 50,000. For the RemBERT models we train Unigram tokenizers from the Tokenizers library (Wolf et al., 2020) with a vocabulary size of 65,536.

ALBERT Model Training: We first pretrained an ALBERT base model from scratch on the non-transliterated corpus as our baseline. Afterward, we pretrained another ALBERT base from scratch on the transliterated corpus. We chose the base model due to computing constraints. We trained the models on a single TPUv3 VM. Both models were trained using the same hyperparameters. We followed the hyperparameters used in (Lan et al., 2020) except for batch size and learning rate. The pretraining objective is also the same as (Lan et al., 2020). We used a batch size of 256, which is the highest that fits into TPU memory, whereas the ALBERT paper used a batch size of 4096. As our batch size is 1/16th of the ALBERT paper, we use a learning rate of 1e-3/8, which is approximately 1/16th of the learning rate used in the ALBERT paper (1.76e-2). Additionally, we use the Adam optimizer (Kingma and Ba, 2015) instead of the LAMB optimizer. The rest of the hyperparameters were the same as the ALBERT paper. Specifically, we use a sequence length of 512 with absolute positional encoding, weight decay of 1e-2, warmup steps of 5000, max gradient norm of 1.0, and Adam epsilon of 1e-6. The models were trained for 1M steps. Each model took about 7.5 days to train. We use the ALBERT implementation from the Huggingface Transformers Library (Wolf et al., 2020).

RemBERT Model Training: We pretrained an RemBERT base models similar to the ALBERT models. We trained the models on a single TPUv3 VM. Both models were trained using the same hyperparameters. We followed the hyperparameters used in (Chung et al., 2020) except for batch size and learning rate. The pretraining objective is also the same as (Chung et al., 2020). We used a batch size of 256, which is the highest that fits into TPU memory, whereas the RemBERT paper used a batch size of 2048. As our batch size is 1/8 of the RemBERT paper, we use a learning rate of 2e-4/8, which is 1/8 of the learning rate used in the RemBERT paper. Similar to the ALBERT model, we use the Adam optimizer (Kingma and Ba, 2015). The rest of the hyperparameters were the same as the RemBERT paper. Specifically, we use a sequence length of 512 with absolute positional encoding, weight decay of 1e-2, warmup steps of 15000, max gradient norm of 1.0, and Adam epsilon of 1e-6. The models were trained for 1M steps. Each model took about 7.5 days to train. We use the RemBERT implementation from the Huggingface Transformers Library (Wolf et al., 2020).

C Downstream Hyperparameters

Hyperparameters for downstream tasks are presented in Table 5 and Table 6.
For the ALBERT models batch size was chosen to be the maximum that fits in memory. This was done so that each batch contains approximately the same number of tokens. Otherwise the hyperparameters were chosen following the recommendations of (Mosbach et al., 2021). On the highly skewed IITP Movie Review, IITP Product Review and MIDAS Discourse we found that this default setting resulted in worse performance compared to the independent baselines. So we finetuned the learning rate and classifier dropout on the validation set of these tasks.

For the RemBERT models learning rate, weight decay, dropout, steps and label smoothing were chosen based on grid search with a few values.

### D Test Statistics Results

$\rho$ gives us the probability of one group being better than the other group. That is the probability that a random performance sample of the uni-script model is greater than a random performance sample of the multi-script model. The last test statistic is $r$ which indicates the magnitude of difference between the performance values of the uni-script model (group 1) and the multi-script model (group 2). $r$ shows us how realistically significant the performance differences are between models even if the performance difference is statistically significant. It gives us a value between 0 to 1 and its ranges are: small effect ($0 \leq r \leq 0.3$), medium effect ($0.3 < r \leq 0.5$) and large effect ($0.5 < r$). We performed MWU on all downstream tasks except CSQA. On CSQA, we only report the $\delta$. The MWU is performed using the SciPy library (Virtanen et al., 2020), and the results are further validated using R (Lüdecke, 2020). These statistic are reported in Table 7 for the IndicGLUE classification tasks and in Table 8 for the public dataset classification tasks.

### E Cross-lingual Similarity of ALBERT Models on All Language Pairs
<table>
<thead>
<tr>
<th>Model</th>
<th>pa</th>
<th>hi</th>
<th>bn</th>
<th>or</th>
<th>as</th>
<th>gu</th>
<th>mr</th>
<th>kn</th>
<th>te</th>
<th>ml</th>
<th>ta</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Wikipedia Section Title Prediction</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RemBERT$_p$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.91</td>
<td>1</td>
<td>1</td>
<td>0.67</td>
<td>0.99</td>
<td></td>
</tr>
<tr>
<td>RemBERT$_r$</td>
<td>0.83</td>
<td>0.83</td>
<td>0.83</td>
<td>0.84</td>
<td>0.83</td>
<td>0.84</td>
<td>0.69</td>
<td>0.83</td>
<td>0.83</td>
<td>0.27</td>
<td>0.81</td>
</tr>
<tr>
<td>ALBERT$_p$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ALBERT$_r$</td>
<td>0.83</td>
<td>0.83</td>
<td>0.83</td>
<td>0.83</td>
<td>0.83</td>
<td>0.83</td>
<td>0.83</td>
<td>0.83</td>
<td>0.83</td>
<td>0.83</td>
<td>0.83</td>
</tr>
<tr>
<td><strong>News Category Classification</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RemBERT$_p$</td>
<td>1</td>
<td>-</td>
<td>0.27</td>
<td>1</td>
<td>-</td>
<td>0.87</td>
<td>0.46</td>
<td>0.27</td>
<td>0.45</td>
<td>0.94</td>
<td>0.75</td>
</tr>
<tr>
<td>RemBERT$_r$</td>
<td>0.85</td>
<td>-</td>
<td>0.39</td>
<td>0.84</td>
<td>-</td>
<td>0.68</td>
<td>0.07</td>
<td>0.39</td>
<td>0.07</td>
<td>0.75</td>
<td>0.41</td>
</tr>
<tr>
<td>ALBERT$_p$</td>
<td>1</td>
<td>-</td>
<td>0.31</td>
<td>1</td>
<td>-</td>
<td>0.80</td>
<td>0.31</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ALBERT$_r$</td>
<td>0.86</td>
<td>-</td>
<td>0.32</td>
<td>0.83</td>
<td>-</td>
<td>0.51</td>
<td>0.32</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>Named Entity Recognition</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RemBERT$_p$</td>
<td>1.00</td>
<td>0.95</td>
<td>0.99</td>
<td>0.70</td>
<td>0.91</td>
<td>1.00</td>
<td>0.57</td>
<td>1.00</td>
<td>1.00</td>
<td>0.56</td>
<td>0.65</td>
</tr>
<tr>
<td>RemBERT$_r$</td>
<td>0.83</td>
<td>0.75</td>
<td>0.81</td>
<td>0.33</td>
<td>0.69</td>
<td>0.83</td>
<td>0.10</td>
<td>0.83</td>
<td>0.83</td>
<td>0.08</td>
<td>0.25</td>
</tr>
<tr>
<td>ALBERT$_p$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ALBERT$_r$</td>
<td>0.83</td>
<td>0.83</td>
<td>0.83</td>
<td>0.83</td>
<td>0.83</td>
<td>0.83</td>
<td>0.83</td>
<td>0.83</td>
<td>0.83</td>
<td>0.83</td>
<td>0.83</td>
</tr>
</tbody>
</table>

Table 7: Test Statistics on Classification Tasks from IndicGLUE Benchmark

<table>
<thead>
<tr>
<th>Language</th>
<th>Dataset</th>
<th>RemBERT$_p$</th>
<th>RemBERT$_r$</th>
<th>ALBERT$_p$</th>
<th>ALBERT$_r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Article Genre Classification</td>
<td>hi BBC News</td>
<td>0.78</td>
<td>0.47</td>
<td>0.87</td>
<td>0.62</td>
</tr>
<tr>
<td>hi Soham News Article Classification</td>
<td>1</td>
<td>0.84</td>
<td>0.87</td>
<td>0.62</td>
<td></td>
</tr>
<tr>
<td>hi INLTK Headlines</td>
<td>0.98</td>
<td>0.79</td>
<td>0.36</td>
<td>0.22</td>
<td></td>
</tr>
<tr>
<td>ml INLTK Headlines</td>
<td>0.08</td>
<td>0.70</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ml INLTK Headlines</td>
<td>0.15</td>
<td>0.59</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>te ACTSA</td>
<td>0.31</td>
<td>0.30</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

Sentiment Analysis

<table>
<thead>
<tr>
<th>Language</th>
<th>Dataset</th>
<th>RemBERT$_p$</th>
<th>RemBERT$_r$</th>
<th>ALBERT$_p$</th>
<th>ALBERT$_r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>hi IITP Product Reviews</td>
<td>0.51</td>
<td>0.01</td>
<td>0.79</td>
<td>0.48</td>
<td></td>
</tr>
<tr>
<td>hi IITP Movie Reviews</td>
<td>0.93</td>
<td>0.72</td>
<td>0.52</td>
<td>0.03</td>
<td></td>
</tr>
<tr>
<td>te ACTSA</td>
<td>0.31</td>
<td>0.30</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

Discourse Mode Classification

<table>
<thead>
<tr>
<th>Language</th>
<th>Dataset</th>
<th>RemBERT$_p$</th>
<th>RemBERT$_r$</th>
<th>ALBERT$_p$</th>
<th>ALBERT$_r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>hi MIDAS Discourse</td>
<td>0.79</td>
<td>0.48</td>
<td>0.45</td>
<td>0.07</td>
<td></td>
</tr>
</tbody>
</table>

Table 8: Test Statistics on Public Datasets
Figure 3: CKA of multi-script and uni-script on all language pairs for pa, hi, bn and or
Figure 4: CKA of multi-script and uni-script on all language pairs for AS, GU, MR and NE