Universiteit Leiden

Master Computer Science

Improving QA for low-resource languages with language alignment

Name:	Jie Wu
Student ID:	s2315211
Date:	12/03/2024
Specialisation:	Data Science
1st supervisor:	Dr. S. Verberne
2nd supervisor:	Dr. Z. Ren

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

To improve the performance of two low-resource languages, Amharic and Khmer, for the Question Answering (QA) task we post-trained the multilingual BERT (mBERT) model using Masked Language Modeling (MLM) and Translation Language Modeling (TLM) and fine-tuned the model using question-passage alignment. The alignment during TLM and fine-tuning is done with three different language pairs, 1) low-resource language with a related high-resource language, 2) low-resource language with a more distant language, and 3) low-resource language with English. With this, we want to investigate whether language alignment can improve the QA performance of these two low-resource languages that are not yet included in the mBERT model, and which of the language pairs is the most effective one. So, we want to know whether the model trained in a language pair with a related high-resource language is more effective than the model trained in a language pair with English (the language with the most data). Our results show that language alignments in general bring some improvements compared to the baseline, but the improvements are not substantial. Further, based on the overall results of models trained with language pairs involved with English, we find that MLM with TLM performs better than only having MLM, which shows that language alignment does have an effect. Then, the language pair with the best performance among all language pairs is the one with English. However, the results remain low for all language pairs. This means that it is difficult for the model to perform well for these two low-resource languages, and the transferability between the high-resource language and the low-resource language is less effective than expected. This is probably because Amharic and Khmer are not only low-resource but also high-distance, which means that they are very different (in their scripts) than the other languages existing in the mBERT model. This makes it extra difficult for the model to connect the high-resource language with these two low-resource languages.

Contents

1 Introduction

Most of the research done for open-domain Question Answering (QA) tasks mainly focuses on English QA. Also, for cross-lingual or multi-lingual QA tasks, the focus is on high-resource languages instead of low-resource ones due to the unavailability of QA datasets for low-resource languages. Since there is limited training data for low-resource QA tasks, the performance of QA retrieval models for those languages is often low.

The traditional methods for extractive open-domain QA tasks use TF-IDF or BM25 to retrieve documents, focusing on keyword matching between the questions and the documents/passages, where sparse vector representations are created [15]. However, to improve the match between questions and documents in the case of paraphrases or synonyms in documents that are closely related to the questions but contain very different tokens (and in the case that the documents are in a different language than the question), dense vector representations have been proposed since the rise of BERT models for ranking [13, 24]. With dense vector representations, words/sentences with similar meanings are closer to each other in their dense embeddings even though their token match is low. However, to obtain a good dense representation, the model requires a lot of training instances, therefore, dense retrievers (often) do not perform better than sparse retrievers [15, 36]. One of the few dense retrievers that show to outperform sparse retrievers is the Dense Passage Retriever (DPR) proposed by Karpukhin et al. [15], which is a dense embedding model that is trained on English question and passage pairs using a pre-trained BERT model with a dual-encoder architecture [15].

Since DPR is only trained for English QA, a multilingual Dense Passage Retriever (mDPR) is introduced for multilingual settings, which creates dense embeddings for multilingual QA and retrieves passages across several languages. In this research, we aim to utilize the mDPR presented by Asai et al. [4], which is based on the DPR model presented by Karpukhin et al. [15] but uses multilingual BERT (uncased) instead of the English BERT, thus, an extension of DPR to a multilingual setting.

To address the challenge of improving mDPR for low-resource languages, we aim to post-train the retrieval model using Masked Language Modeling (MLM) and Translation Language Modeling (TLM) [9]. TLM is a self-supervised learning method that can be used to map sentences of low-resource languages (Khmer and Amharic) to sentences of high-resource languages (Thai, Arabic, and English). Further, we also do a question-passage alignment between the language pairs during fine-tuning. So, we will have the questions in the low-resource language and passages in the high-resource language to improve the matching in their dense embeddings. In this way, we can improve the transferability between those low-resource languages such that only training data in high-resource languages (and limited training data in the low-resource languages) suffice for the QA task. For our research, we consider low-resource languages not included in the pre-training of the BERT model and high-resource languages included in the BERT model. In this way, we can evaluate the effectiveness of our method since the model does not have any information on those low-resource languages before training.

Our model is based on the CORA model presented by Asai et al. [4, 5], focusing on the mDPR part with multilingual BERT (cased) as the retriever model. So, we evaluate three models with different language alignments. The first one is the alignment between low-resource and

related high-resource languages (Khmer-Thai and Amharic-Arabic). The second one is the alignment between low-resource and non-related high-resource languages (Khmer-Arabic and Amharic-Thai). The last one is the alignment between low-resource languages and English. We decided to use these three high-resource languages in order to see which alignment is the most effective since the related high-resource language is closer to the corresponding low-resource language, but the used amount of training data to train the model is (much) less than English.

Therefore our research questions are as follows:

- RQ1: What is the impact on the performance of the mDPR model on the question-answering task for two low-resource languages, Amharic and Khmer, by using the mBERT (cased) model post-trained and fine-tuned on language alignment between high and low-resource languages compared to the original mBERT (cased) model.
- RQ2: What are the differences in model performance between 1) the alignment of low-resource and related high-resource languages (Amharic with Arabic and Khmer with Thai), 2) the alignment of low-resource and non-related high-resource languages (Amharic with Thai and Khmer with Arabic), and 3) the alignment of low-resource languages (Amharic and Khmer) and English.

In order to answer our research questions, we will post-train and fine-tune our models (multilingual BERT model with expanded vocabulary) with language alignment between different language pairs (sentence alignment during TLM and question-passage alignment during fine-tuning). According to our experimental results, we found that language alignment can indeed bring improvements above the baseline, and the best language pair is the one aligned with English.

So, our contributions are as follows:

- We provide datasets containing aligned texts (filtered) between several language pairs (Amharic with Arabic/Thai/English, and Khmer with Arabic/Thai/English).
- We post-trained and fine-tuned mBERT models on two low-resource languages not existing in the original mBERT model.
- We defined two alignment methods during post-training (using TLM) and during fine-tuning (questions in low-resource language and passages in high-resource language) to improve QA performance for two low-resource languages, Amharic and Khmer.
- We compared three alignments with different language pairs and presented which language alignment is the most effective one.

2 Background

2.1 DPR and CORA

Our research uses the CORA model presented by Asai et al. [4]. The CORA model consists of two parts, 1) mDPR (Multilingual Dense Passage Retriever) and 2) mGEN (Multilingual

Answer Generator). The mDPR uses the multilingual BERT model for producing dense representations for the passages and the questions and retrieves the top k passages for the given questions by applying the Maximum Inner Product Search (MIPS) algorithm. The mGEN uses the multilingual T5 model for generating answers to the questions based on the retrieved top k passages from the mDPR. The generated answers should be given in the language of the corresponding question [4]. The two components of CORA are illustrated in Figure 1.

The mDPR part of the CORA model is based on the DPR (Dense Passage Retrieval) model presented by Karpukhin et al. [15]. mDPR can be viewed as the multilingual version of DPR. DPR uses (English) BERT for producing dense question and passage embeddings, matching them using the MIPS algorithm for passage retrieval. The training is done by minimizing the negative log-likelihood of the positive passages. Each training instance consists of a question, a positive passage, and n negative passages. The authors aim to obtain a better dense passage retrieval model that is only trained on question-passage (or question-answer) pairs without additional pre-training [15]. Their results show that DPR substantially outperforms BM25, and largely improves the end-to-end open-domain question answering accuracy in comparison to ORQA [15, 38].

The initial training of mDPR is based on two QA benchmark datasets: 1) Natural Questions (NQ) [18], and 2) XOR-TyDi QA and TyDi QA [8]. Each training example includes a question, an answer, and a positive passage. The update of the mDPR parameters during training is done in the same way as for DPR, where the negative log-likelihood of the positive passages is minimized. Further, the mGEN is trained using the top k passages retrieved from the trained mDPR for the training data questions. To ensure that mGEN can generate answers in the target language disregarding the language of the given passages, language codes are added. Moreover, for the model to produce answers in languages not included in the original training data, the answers of the NQ training data are translated into different languages using Wikipedia language links. Then, the update of the mGEN parameters is done by minimizing the cross-entropy loss [4].

The authors of the CORA model also applied iterative training by mining new training data using 1) mDPR since mDPR can find new positive passages in other languages not included in the initial data, 2) Wikipedia language links for discovering potential positive passages in other languages, and 3) mGEN predictions to evaluate the passages found by mDPR and Wikipedia language links. The passages that can successfully lead to a correct answer by mGEN are evaluated as positive passages, and passages that do not lead to a correct answer are evaluated as negative passages. These newly discovered positive and negative passages are added to the training data for the next training iteration of the CORA model. This iterative training is repeated multiple times [4].

2.2 Post-training

Post-training or post-pretraining is the step between the initial pre-training and fine-tuning, which can be applied when pre-training the model from scratch is too expensive. Pan et al. [25] proposed a method called PPA, which stands for Post-Pretraining Alignment where self-supervised word- and sentence-level alignment are applied as a post-pretraining step to improve cross-lingual transferability of the mBERT model using parallel data. In their method, Trans-

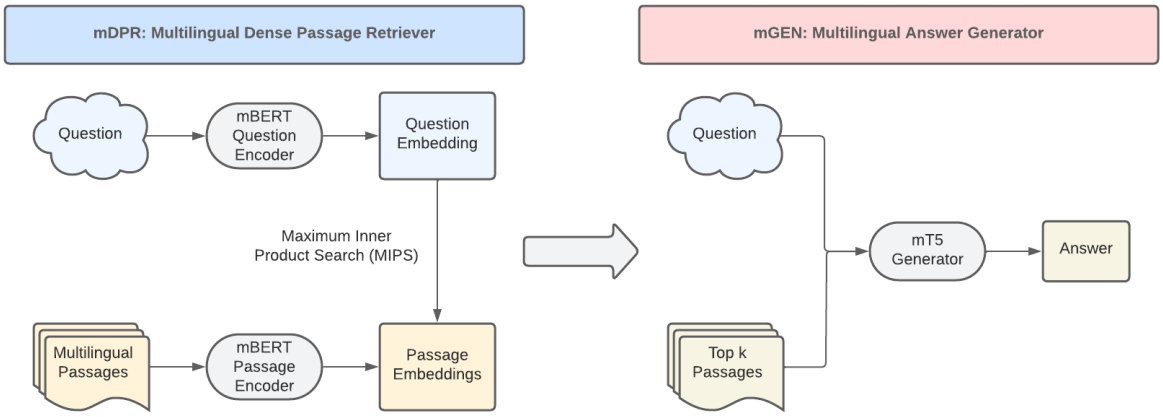

Figure 1: The components of the CORA model: 1) Multilingual Dense Passage Retriever (mDPR) and 2) Multilingual Answer Generator (mGEN). mDPR retrieves the top k passages based on the question and passage embeddings using the MIPS algorithm. mGEN generates an answer for the given question based on the retrieved top k passages from mDPR.

lation Language Modeling (TLM) is used to align multilingual contextual embeddings on the word level, and contrastive learning and random shuffling are used to align embeddings on the sentence level. Their results show that PPA efficiently improves the cross-lingual transferability of Language Models. Also, the proposed method largely improves the performance of mBERT on Natural Language Inference (NLI) and QA tasks [25].

Xu et al. [47] proposed a method of joint post-training where BERT is modified to contain domain and task knowledge before fine-tuning. This method uses unsupervised domain reviews to strengthen domain knowledge, and supervised Machine Reading Comprehension (MRC) data to enhance MRC task knowledge. Since this method is meant to be general-purpose, it also benefits Aspect Extraction and Aspect Sentiment Classification. Their results show that applying the proposed post-training method before fine-tuning is effective [47].

Other works focused on using post-training methods to improve model performances are, for example, the work done by Park et al. [27] to improve the question generation of KoBART using a novel post-training method, the work done by Lopes et al. [22] to improve the Aspect-Based Sentiment Analysis (ABSA) in Portuguese using a post-trained model with a Question-Answer approach, and the work done by Liu et al. [20] to improve the performance of low-resource languages by post-training models on high-resource languages and use the similarities between the low-resource and high-resource languages to mitigate the data scarcity of low-resource languages.

2.3 Question Answering for low-resource languages

Most of the research on multilingual or cross-lingual extractive QA focuses on high-resource languages instead of low-resource ones due to limited QA data in low-resource languages. In order to improve the QA model for low-resource languages, different techniques are proposed such as data augmentation [17, 30, 34, 37]. For example, to improve the QA performance in

two low-resource languages Hindi and Tamil, Kumar et al. [17] proposed a training pipeline containing three stages, namely 1) pre-train the mBERT model, 2) pre-train the QA head above the mBERT model on the SQuAD [31] dataset, and 3) fine-tune the mBERT model on the original ChAII[1] dataset and augmented examples in Tamil and Hindi. The data augmentation method utilizes translations and transliterations to increase the amount of data in ChAII. It is shown that the translation of the same language family increases the model performance, whereas transliteration does not increase the performance. Further, they propose a multilingual contrastive loss during fine-tuning between the translated pairs to enhance the closeness between the embeddings of different cross-family languages [17].

Other works focusing on low-resource QA include 1) the creation of datasets for low-resource languages such as Swahili languages [45], Telugu [41], Persian [10] and Vietnamese [19], 2) QA systems that are developed for a specific low-resource language, such as for Bengali [6, 12], Vietnamese [14, 29], Telugu [32], Persian [23] and Tamil [2], and 3) improving QA systems for low-resource languages such as using automatic data enrichment [43], unsupervised statistical methods [11], and syntactic graph [46].

In our work, we aim to improve the QA performance of two low-resource languages (Amharic and Khmer) by utilizing the TLM paradigm to map sentences of those low-resource languages to their corresponding sentences in high-resource languages during post-training of the mBERT (cased) model. The post-trained mBERT model is further fine-tuned utilizing question-passage alignment between the language pairs and evaluated on the QA datasets.

3 Replication of CORA

The authors of the CORA model updated their original CORA code[2] for the MIA 2022 Shared Task [5], therefore, for our research, we use the updated version[3]. Further, due to the computational complexity and time, we do not use the iterative training as was done for the original CORA model as our main focus is to measure the effectiveness of cross-lingual transferability between high and low-resource languages. In the MIA 2022 Shared Task, Asai et al. [5] mentioned two baselines, the first baseline is mDPR+mGEN, which is the CORA model without iterative training, and the second baseline is the CORA model with iterative training. As our research mainly focuses on the mDPR part, we only re-train the mDPR model and use the trained mGEN model from the baseline provided by Asai et al. [5] for our replication. The replication of the CORA baseline is described below.

3.1 mDPR training

According to the authors of the MIA 2022 Shared Task [5], the same hyperparameters were used as in DPR [15] for training the mDPR model. Further, they trained the model for 30 epochs and took the last checkpoint as their final mDPR model. However, the training script provided in the MIA repository[4] shows a different set of hyperparameters. To ensure the model

[1] https://www.kaggle.com/c/chaii-hindi-and-tamil-question-answering
[2] https://github.com/AkariAsai/CORA/tree/main
[3] https://github.com/mia-workshop/MIA-Shared-Task-2022/tree/main
[4] https://github.com/mia-workshop/MIA-Shared-Task-2022/tree/main/baseline/mDPR

is trained similarly to the one in the MIA task, we replicated the mDPR training two times, one using the hyperparameters values[5] from DPR [15] and trained it for 30 epochs (replication 1), and one according to the MIA training script[4] as provided in the MIA repository (replication 2).

For both variants, the mDPR is trained on the original training data (NQ training data and XOR-TyDi QA gold paragraph data), and validated on the XOR-TyDi development data[4] provided by Asai et al. [5]. After training, we took the last checkpoint as the final mDPR model for our replication. The mBERT model used for training is the multilingual BERT uncased (`bert-base-multilingual-uncased`). Further, the original mDPR model is trained on multiple GPUs, however, since we have limited resources, we only trained the model on a single GPU. This might affect the final performance of the model since mBERT is sensitive to this training configuration.

After we trained the model, we used the trained mDPR to generate dense embeddings for the pre-processed Wikipedia context passages[6] that are already split into 100-token length as provided for the MIA task. The original generation script[4] uses 8 GPUs, however, we only use 1 GPU. Further, to make this process reproducible, we modified the script slightly by adding a random seed and we fixed the random seed at 12345 at the beginning of the training process.

3.2 CORA evaluation

To evaluate our trained mDPR model and the mGEN model provided by Asai et al. [5], thus the CORA model without iterative training, on the XOR-TyDi QA and MKQA development data, we have to 1) run the trained mDPR to retrieve passages for the corresponding QA data, 2) convert the mDPR output to mGEN input data, 3) run the mGEN script[7] using the trained mGEN model and the corresponding mGEN input data for obtaining the mGEN results (mT5 generated answers for the input questions), and 4) run the evaluation script[8] for the corresponding QA data to evaluate the performance based on the mGEN results. The performance of the final model is measured by token-level F1, Exact Match (EM), and BLEU. We only report the F1 results.

Token-level F1 is computed by first normalizing the predicted and the ground truth answers. Then, the F1 score is computed based on token-level precision and token-level recall. In formula:

- $Precision = (1.0 \cdot N_{common})/N_{predicted}$
- $Recall = (1.0 \cdot N_{common})/N_{ground\_truth}$
- $F1 = (2 \cdot Precision \cdot Recall)/(Precision + Recall)$

N_{common} denotes the number of common tokens between the predicted and the ground truth answers. $N_{predicted}$ denotes the number of tokens in the predicted answer. N_{ground_truth} denotes the number of tokens in the ground truth answer.

[5] `https://github.com/facebookresearch/DPR/blob/main/conf/train/biencoder_nq.yaml`

[6] `https://nlp.cs.washington.edu/xorqa/cora/models/mia2022_shared_task_all_langs_w100.tsv`

[7] `https://github.com/mia-workshop/MIA-Shared-Task-2022/tree/main/baseline/mGEN`

[8] `https://github.com/mia-workshop/MIA-Shared-Task-2022/tree/main/eval_scripts`

The metric Exact Match is computed by comparing the normalized predicted answer to the normalized ground truth answer, and if they are equal, the EM score is 1, and if not, the EM score is 0. So, EM gives whether the normalized predicted answer is the same as the normalized ground truth answer. Further, the BLEU is computed in the same way as described in the BLEU paper by Papineni et al. [26]. The computation of BLEU is done via the NLTK toolkit.[9]

Besides the evaluation of the mDPR models trained by us, we also evaluated the provided mDPR and mGEN checkpoints by Asai et al. for the MIA task [5]. This is done for baseline 1 as well as baseline 2.

4 Methods

In this section, we describe our methods including 1) the creation of the datasets, 2) the post-training process, and 3) the finetuning process of the mDPR model, with the final evaluation of our models.

4.1 Data

We use the CORA model presented by Asai et al. [4, 5] as the base for our model. Here, we modify the CORA model by post-training the mBERT cased model, used in the mDPR part, on our dataset, and measuring the performance of the resulting model. The CORA model consists of two parts, the first part is mDPR for dense passage retrieval, and the second part is the mGEN model for generating an answer to the query based on the retrieved top k passages. The main focus of our research is on the mDPR part.

The idea of post-training with language alignment is by encoding text pairs of two languages where the model can learn the interactions between the words in these texts. When getting the questions in one of these languages, the model can better map the question to the documents of the other language as well and therefore improve the multilingual retrieval performance, especially for low-resource languages, since we have limited training data and corpus available for these languages.

Our dataset consists of aligned sentences of low-resource and high-resource languages. So, there are three variants of language alignment. The first variant is alignment between low-resource and related high-resource languages (Amharic with Arabic and Khmer with Thai). The second variant is alignment between low-resource and non-related/more distant high-resource languages (Amharic with Thai and Khmer with Arabic). The third variant is alignment between low-resource languages (Amharic and Khmer) and English. The first variant is our proposed method for improving mBERT for low-resource languages. The second and third variants are used as experimental comparisons to investigate the effectiveness of our proposed method. In this way, we can see whether the relatedness between two language pairs is more or less useful than the amount of training data the model is trained on. Since English is a more high-resource language than Arabic and Thai, the mBERT model is also more trained in this language which makes the model perform better in this language. However, Arabic is more close to Amharic

[9] `https://www.nltk.org/api/nltk.translate.bleu`

and Thai is more close to Khmer than English, which might be better for learning the alignment.

As mentioned in [4], mDPR often retrieves passages from the language of the target question, from the languages that are typologically similar to the target language (for example, Spanish to Portuguese), or from English. Furthermore, Asai et al. [4] also did a controlled experiment to evaluate the retrieval performance for three different languages (Danish, Portuguese, and Malay) where their typologically similar languages (Spanish, Swedish, and Indonesian) were removed from the document embeddings. They found there was a drop in model performance for those languages. Thus, closer languages (with more resources) are helpful for cross-lingual retrieval [4]. We also believe that post-training the mBERT model on language alignment can improve the cross-lingual transferability between the language pairs, meaning that their dense embeddings should be closer to each other than before training. Especially, when the language pairs are related to each other, it should be easier for the model to learn the alignment/interaction. To verify this, we use the three variants of language alignment as mentioned earlier and compare them.

The reason that we choose Khmer and Amharic as low-resource languages for our research is that 1) we are able to find sentence-alignment data as these two languages are included in the CCAligned dataset [16], 2) we can find QA evaluation data, and 3) both languages are not included in mBERT. We also looked into other low-resource languages included in CCAligned but not in mBERT, that are of sufficient size. The list of these languages can be found in Table 1. However, we failed to find any QA dataset for those languages.

We choose Thai as the related high-resource language for Khmer because 1) the shared word vocabulary between the two languages is about 30%, 2) their syntax is almost the same, and 3) many idioms are shared between these two languages [39]. Further, the choice of Arabic as the related high-resource language for Amharic is because 1) we cannot find any other high-resource languages that are closer to Amharic than Arabic (none of the high-resource languages are linguistically similar to Amharic), and 2) they come from the same Afroasiatic language family (the Semitic branch), which means that there is some similarity between the two languages. The similarities between the two languages are for example the contribution of Amharic to Arabic in vocabulary, grammar, and syntax.[10] However, they do have a different writing system and script. Both high-resource languages, Thai and Arabic, are included in mBERT-cased (`bert-base-multilingual-cased`), which means that we use mBERT-cased as our retrieval model for mDPR. The original retrieval model used by CORA is mBERT-uncased, which does not include Thai in the pre-training of the model. Since there is no uppercase and lowercase in the languages Khmer and Amharic (same for Thai and Arabic), we do not expect that the cased version of mBERT will have a large effect on the performance of these languages.

4.1.1 Sentence alignment data

We use the CCAligned dataset created by El-Kishky et al. [16] to create training data for the sentence-alignment task. This dataset contains English-aligned document pairs in 137 diverse languages, created using language identification in URLs on web documents obtained from Common Crawl snapshots. With language identifiers, the authors were able to find doc-

[10] `http://www.kalmasoft.com/KLEX/dbamara.htm`

Language List		
Christmas Island (cx)	Maori (mi)	Somali (so)
Hausa (ha)	Nyanja (ny)	Xhosa (xh)
Igbo (ig)	Pashto (ps)	Zulu (zu)
Kurdish (ku)	Sinhala (si)	
Lao (lo)	Shona (sn)	

Table 1: List of considered low-resource languages, included in CCAligned and excluded from mBERT, for which no QA dataset is available or could be found.

uments containing parallel/translated or comparable content. For example, `en.aaa.com` and `nl.aaa.com` contain language identifiers `en` and `nl`, then, it is assumed that these two web documents contain comparable content, one in English and one in Dutch. The average precision of this cross-lingual document alignment is about 94.5% according to El-Kishky et al. [16].

Since each English document is aligned to multiple documents of different non-English languages, we can obtain non-English document pairs based on the English document. The authors also provide sentence pairs in different languages aligned with English sentences, which are extracted from the document pairs using LASER embeddings[11] similarity score. So, each document is decomposed into sentences, then, for each document pair, the sentences are aligned using the above-mentioned similarity method. Finally, for each language pair, the parallel sentences are aggregated across the corresponding document pairs [16]. Therefore, to build our dataset, we utilize the sentence pairs in the CCAligned[12] dataset, especially the Arabic-English, Amharic-English, Thai-English, and Khmer-English pairs. Similar to document pairs, we created non-English sentence pairs by joining them on the English sentences, in this way, we obtain Amharic-Arabic and Khmer-Thai sentence pairs, as well as Amharic-Thai and Khmer-Arabic sentence pairs.

In order to evaluate the final QA model performance on the two low-resource languages, Amharic and Khmer, we found two QA datasets, AmQA[13] presented by Abedissa et al. [1] and MKQA[14] presented by Longpre et al. [21]. AmQA is an Amharic QA dataset created by crowdsourcing 2,628 question-answer pairs across 378 Amharic articles on Wikipedia [1]. MKQA is a multilingual knowledge question and answer dataset that is created by converting 10k English questions of the NQ dataset into 26 other languages where human annotations are performed, which makes the dataset parallel aligned across those languages [21]. Khmer is one of those languages, therefore we can take the Khmer question-answer pairs from the MKQA dataset to form the Khmer evaluation set.

Further, we also looked into other multilingual corpora for sentence alignment to see if we could find any low-resource languages that meet the requirements for our research. In the ParaCrawl corpus[15] presented by Bañón et al. [7] we found four languages not included in mBERT (Khmer, Pashto, Sinhalese, and Somali), however, only for Khmer, we found a QA dataset as

[11] `https://github.com/facebookresearch/LASER`
[12] `https://www.statmt.org/cc-aligned/`
[13] `https://github.com/semantic-systems/amharic-qa`
[14] `https://github.com/apple/ml-mkqa/tree/main`
[15] `https://paracrawl.eu/index.php`

mentioned earlier. In the WikiMatrix dataset[16] presented by Schwenk et al. [35] we found two languages not included in mBERT (Esperanto and Sinhala), however, both languages do not have any QA dataset available. Therefore, the only two low-resource languages that meet the requirements for our research are Khmer and Amharic.

4.1.2 Data preprocessing

Although CCAligned provides English-aligned sentence pairs for the languages Arabic, Amharic, Thai, and Khmer, we still need to get aligned sentence pairs for Amharic-Arabic, Amharic-Thai, Khmer-Thai, and Khmer-Arabic. So, we obtained these sentence pairs by joining the original sentence pairs on the English sentences.

Creating the Amharic-Arabic dataset. The original English-Arabic dataset contains 25,309,750 sentence pairs, while the original English-Amharic dataset contains 346,517 sentence pairs, and by joining them, we obtained 8,817,926 sentence pairs for Amharic and Arabic. However, we found that some sentences in Arabic are not aligned with the corresponding sentences in Amharic. In order to filter out the noises, we decided to translate the original Amharic sentences into English (from the original English-Amharic dataset), do a cosine similarity check on the translated English sentences (from Amharic and Arabic) and the original English sentences, and keep those sentences above certain similarity score. This is done by looking at 50 randomly selected (translated) sentence pairs for each similarity threshold and the most suitable threshold is chosen where at least 80% of the selected (translated) sentence pairs are sufficiently similar.

So, firstly, we use the DL Translate tool[17] to translate all Amharic sentences into English. Secondly, we apply the sentence similarity tool[18] with `all-MiniLM-L6-v2` model (using sentence embedding) to get the similarity score between the translated English sentences (from Amharic) and the original English sentences. We choose the `all-MiniLM-L6-v2` model because this model is 5 times faster than the best model (`all-mpnet-base-v2`) and still gives a good performance according to Reimers et al. [33] in their documentation on sentence transformers[19]. After the similarity check, we obtained 2,208 sentences with a negative score, 344,291 sentences with a positive score, and 18 sentences with no score (NaN). We looked into the sentence pairs with no score and for two pairs, we found that the original English sentences are both 'None', which is considered a missing value (same as NaN) by Pandas. For the other 16 pairs, the Amharic sentences contain special symbols or non-Amharic characters that might be the reason why they are not effectively translated into English and therefore result in a NaN value after translation. Since at least one sentence in the sentence pair contains a NaN value, the resulting similarity score is also NaN. For the two English sentences with 'None', we recomputed the similarity scores, which are 0.328 and 0.309, respectively. In Figure 2 (left), we show the number of sentence pairs for different similarity thresholds between 0.0 and 1.0.

[16] `https://metatext.io/datasets/wikimatrix`
[17] `https://github.com/xhluca/dl-translate`
[18] `https://github.com/Susheel-1999/Sentence_Similarity`
[19] `https://www.sbert.net/docs/pretrained_models.html#sentence-embedding-models`

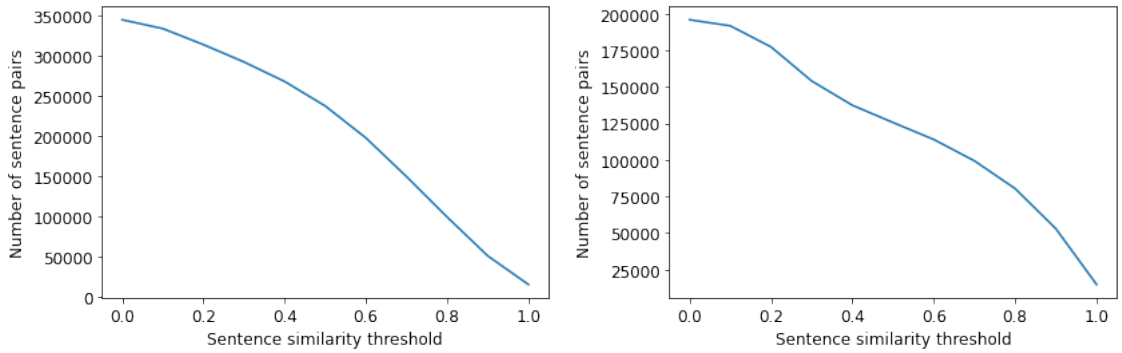

Figure 2: The number of sentence pairs at different sentence similarity thresholds between 0.0 and 1.0. The similarity check is done on the translated English sentences (left: from Amharic and right: from Arabic) and the original English sentences. This check is done for creating the Amharic-Arabic dataset.

In order to choose an appropriate threshold, we took 50 random pairs (English and translated English sentences) for each threshold (from 0.0 to 0.9) and checked whether the translated English sentences (from Amharic) were sufficiently similar to their English counterparts. After that, we choose the threshold where at least 80% of the sentence pairs are sufficiently similar to each other. Since the translation using the DL Translate tool is not always accurate in the sense that the sentence pairs might be similar in their meaning, but due to the mistranslation of some words, it is difficult to decide whether the sentences are similar or not, we manually re-translated the 50 Amharic sentences for each threshold using Google Translate[20] to verify our decision about the similarity between the English and translated English sentences. For example, we have the following English sentence:

> *13:5 For, by the greatness of the creation and its beauty, the creator of these will be able to be seen discernibly.*

The corresponding DL translation of the same Amharic sentence into English is:

> *13:5 For, due to the greatness of speed and weakness, these speed can be discernibly seen.*

The corresponding Google translation is:

> *13:5 For, by the greatness of creation and beauty, these creators can be seen discernibly.*

As we can see, the DL translated sentence is different than the original English sentence, however, they do share the same structure and some of the words/parts such as '13:5', 'greatness', and 'discernibly'. Also, according to the cosine similarity check, the DL translation shares a similarity score of 0.707 with the original English sentence. Then, if we translate the same Amharic sentence using Google Translate, we can see the corresponding translation is highly similar to the original English sentence. In this case, there is a mistranslation of certain words in the sentence by DL translation (maybe the corresponding embeddings of these words are relatively close to each other). However, with the double-check using Google Translate, we can consider

[20] `https://translate.google.com/?hl=nl`

this sentence pair as similar. So, using this method, we ended up with a chosen threshold of 0.7 and a filtered Amharic-English dataset containing 149,573 Amharic-English sentence pairs.

To reduce the time of translating all Arabic sentences, we only translate these sentences that can be mapped to the filtered Amharic dataset by joining them based on the English sentences. We obtained 211,316 English-Arabic sentence pairs after the mapping. Similar to the Amharic sentences, we translated the Arabic sentences using the DL Translate tool and computed the similarity score using the sentence similarity tool. After the similarity check, we obtained 494 sentences with a negative score, 210,816 with a positive score, and 6 with no score (NaN). Those NaN values are due to the translation error where the Arabic sentences are not effectively translated into English, therefore resulting in NaN values. We looked into these Arabic sentences with NaN values. We found them to contain special symbols or non-Arabic characters, which might be the reason for the mentioned translation error. Then, in Figure 2 (right) we show the number of sentence pairs for each similarity threshold. Further, we checked all 50 randomly selected sentence pairs and double-checked using Google Translate (in the same way as was done for Amharic). Here, we obtained a threshold of 0.8 and a filtered Arabic-English dataset containing 87,970 Arabic-English sentence pairs.

Finally, we merged the filtered Amharic-English dataset with the filtered Arabic-English dataset and obtained a joined dataset with English, Arabic, and Amharic sentences, containing 107,667 instances. This dataset is used to get the Amharic-Arabic sentence pairs. After removing all duplicated sentence pairs, we obtained a total number of 102,994 sentence pairs for the Amharic-Arabic dataset, which will be used for the sentence alignment task. For the Amharic-English sentence alignment task, we use the filtered Amharic-English dataset containing 149,573 sentence pairs. However, to make the data size comparable to the Amharic-Arabic dataset, we only selected the Amharic-English sentence pairs with a similarity score equal to or higher than 0.8 to form our Amharic-English sentence alignment dataset, which gives us 99,403 sentence pairs.

Creating the Khmer-Thai dataset. The same process is applied to Khmer and Thai. The original English-Khmer dataset contains 412,381 sentence pairs. The original English-Thai dataset contains 10,746,367 sentence pairs. The initial join on English sentences results in 5,510,283 sentence pairs. So, to filter out the noises, we first translate Khmer sentences into English and get the corresponding similarity score with the original English sentences. After this, we obtained 3,439 sentences with a negative score, 408,855 with a positive score, and 87 with no score (NaN). Then, we looked into the sentence pairs with no score and found that for 6 pairs, the original English sentences are 'None', which is considered a missing value (same as NaN) by Pandas. For the other 81 pairs, the Khmer sentences contain special symbols or non-Khmer characters which might be the reason why they are not effectively translated into English and therefore resulting in a NaN value after the translation. Since at least one sentence in the sentence pair contains a NaN value, the resulting similarity score is also NaN. For the 6 English sentences with 'None', we recomputed the similarity scores, which are between 0.2 and 0.3. Further, in Figure 3 (left) we show the number of sentence pairs for each similarity threshold. Then, using the same method mentioned earlier, we ended up with a similarity threshold of 0.8 for Khmer and obtained a filtered Khmer-English dataset containing 119,757 sentence pairs.

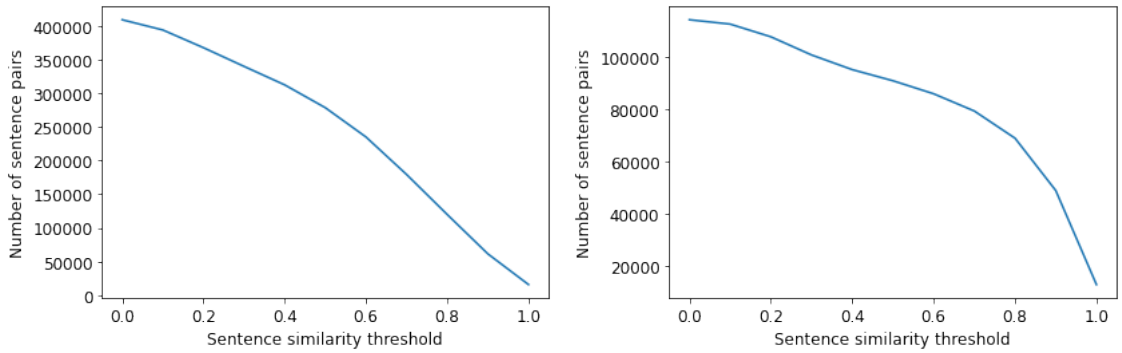

Figure 3: The number of sentence pairs at different sentence similarity thresholds between 0.0 and 1.0. The similarity check is done on the translated English sentences (left: from Khmer and right: from Thai) and the original English sentences. This check is done for creating the Khmer-Thai dataset.

Similar to Arabic and Amharic, we only translate the Thai sentences that can be mapped to the filtered Khmer dataset by joining them based on the English sentences to reduce the translation time. By doing so, we obtained 114,394 Thai-English sentence pairs after the mapping. Then, we translated these Thai sentences and computed the corresponding similarity score. Here, we obtained 269 sentences with a negative score, 114,123 sentences with a positive score, and 2 sentences with no score (NaN). We checked the 2 sentences with no score and found that both Thai sentences contain special symbols which might be the reason for the translation error resulting in NaN values. Further, in Figure 3 (right) we show the number of sentence pairs for each similarity threshold. Here, we choose a threshold of 0.8, which is decided using the same method as mentioned earlier, and we obtained a filtered Thai-English dataset containing 68,833 sentence pairs.

Finally, we merged the filtered Khmer-English dataset with the filtered Thai-English dataset and obtained a joined dataset with English, Thai, and Khmer sentences, containing 84,725 instances. This dataset is used to obtain the Khmer-Thai sentence pairs. So, after removing the duplicated sentences, we obtained a Khmer-Thai dataset containing 80,935 instances which will be used for the Khmer-Thai sentence alignment task. Further, we used the filtered Khmer-English dataset containing 119,757 instances for the Khmer-English sentence alignment task.

Other datasets. The same process is repeated to create the Amharic-Thai and Khmer-Arabic datasets. The number of sentence pairs for each similarity threshold for the corresponding Thai (left) and Arabic (right) datasets can be found in Figure 4. For Amharic-Thai, we obtained a dataset containing 77,413 instances and for Khmer-Arabic, we obtained a dataset containing 106,148 instances. A summarization of these created datasets about their threshold and data size is given in Table 2.

4.2 mBERT post-training

Since mBERT is not trained in Amharic and Khmer, the corresponding vocabulary also does not contain Amharic and Khmer tokens as both scripts are different from the languages existing in the original mBERT model. However, as mBERT is trained on a large corpus and 104

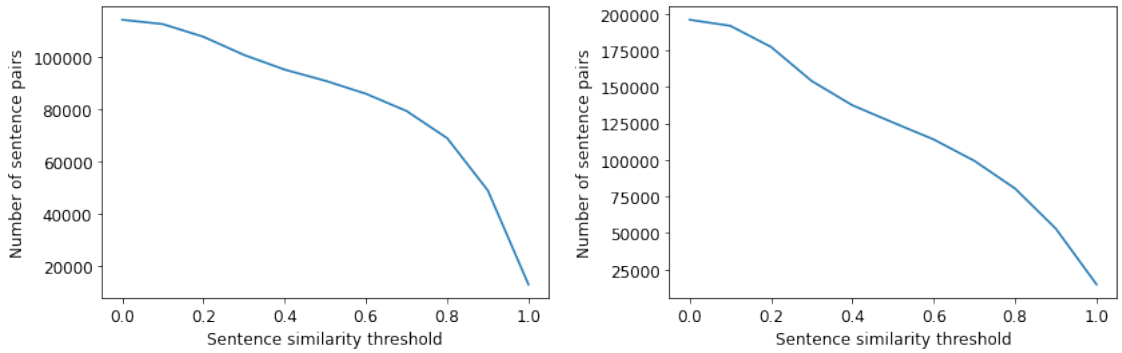

Figure 4: The number of sentence pairs at different sentence similarity thresholds between 0.0 and 1.0. The similarity check is done on the translated English sentences (left: from Thai and right: from Arabic) and the original English sentences. This check is done for creating the Amharic-Thai and Khmer-Arabic datasets.

Dataset	**Threshold L1**	**Size L1**	**Threshold L2**	**Size L2**	**Final data size**
Amharic-Arabic	0.7	149,573	0.8	87,970	102,994
Amharic-Thai	0.7	149,573	0.8	69,892	77,413
Amharic-English	0.8	99,403	–	–	99,403
Khmer-Thai	0.8	119,757	0.8	68,833	80,935
Khmer-Arabic	0.8	119,757	0.8	80,303	106,148
Khmer-English	0.8	119,757	–	–	119,757

Table 2: A summarization of the created datasets. Threshold L1 indicates the chosen threshold for the first language in the dataset, e.g., in Amharic-Arabic, the first language is Amharic, and 0.8 is the corresponding chosen similarity threshold above which the sentences are kept in the dataset. Size L1 indicates the number of instances (rows) in the (filtered) dataset for the first language before merging. Threshold L2 indicates the chosen threshold for the second language. Size L2 indicates the data size of the second language before merging. Final data size indicates the number of instances in the final language-pair dataset after the merge and duplicate removal, e.g., the final Amharic-Arabic dataset after joining on English and duplicate removal.

different languages using Masked Language Modeling (MLM) and Next Sentence Prediction (NSP), it might still be able to learn these languages (such as their structures) through sentence alignment. But, due to a large number of unknown tokens, the learning might not be effective. The problem with unknown tokens is that they all are mapped to the token '[UNK]' and the embeddings of these tokens are the same, therefore, possibly leading to performance loss [40]. So, one solution is to add a certain amount of unique Amharic and Khmer tokens to the mBERT vocabulary such that the embeddings of these newly added tokens can be learned through the MLM and sentence alignment (TLM) tasks. Wang et al. [44] also proposed a similar method where they extend the vocabulary of mBERT to low-resource languages outside of the mBERT model and further post-train the model with the extended vocabulary on the corresponding low-resource languages. They found the performance of the extended mBERT model to be better than the bilingual BERT model and the original mBERT model on the NER task for the corresponding low-resource languages [44]. Therefore, for our research, we extend the mBERT's vocabulary with the tokens of the considered low-resource languages.

The model will learn the embeddings of these newly added tokens through MLM and TLM tasks.

4.2.1 Vocabulary expansion

In order to effectively train the mBERT model (initialized from the BERT multilingual base model (cased)) to map these low-resource languages to the (relatively) high-resource languages (existing in mBERT), we extracted the words from the Amharic and Khmer sentences in order to extend the vocabulary of mBERT with these unique Amharic or Khmer tokens. By doing so, we obtained two extended mBERT models, one extended with Amharic tokens and one extended with Khmer tokens.

For Amharic, we first get all Amharic sentences from the Amharic-Arabic and Amharic-Thai datasets. Since there is already a trained mBERT[21] model that is fine-tuned on Amharic texts (where the original mBERT vocabulary is replaced by an Amharic vocabulary), we use the tokenizer of this Amharic-BERT model to tokenize all Amharic sentences to get the corresponding Amharic tokens. From the obtained list of Amharic tokens, we only keep those full Amharic words that are unknown by the original mBERT model. This is done by retokenizing each Amharic token using the original mBERT model (and rechecking with the add_tokens() method). By doing so, we obtained a total of 37,852 unique Amharic tokens. After adding these Amharic tokens, we can directly use the extended mBERT tokenizer to encode the Amharic sentences since Amharic words are separated by whitespace. We will refer to our extended mBERT model with Amharic as the amBERT model.

Below, we show an example of how an Amharic sentence is tokenized using the original mBERT (`bert-base-multilingual-cased`) tokenizer and using our Amharic extended mBERT tokenizer (amBERT). So, we have the following Amharic sentence:

```
'ጥሩ ጥራት ጋር ጥሩ ዋጋ. አንተ ንጽጽር በኋላ ያውቃሉ.'
```

The corresponding tokenized Amharic sentence using the original mBERT model:

```
['[UNK]', '[UNK]', '[UNK]', '[UNK]', '[UNK]', '.', '[UNK]', '[UNK]', '[UNK]', '[UNK]', '.']
```

The corresponding tokenized Amharic sentence using our amBERT model:

```
['ጥሩ', 'ጥራት', 'ጋር', 'ጥሩ', 'ዋጋ', '.', 'አንተ', 'ንጽጽር', 'በኋላ', 'ያውቃሉ', '.']
```

As we can see, the original mBERT tokenizer does not have Amharic tokens (not even the subtokens) in the corresponding vocabulary, therefore resulting in unknown tokens ('[UNK]') for all Amharic words. After we extended the vocabulary with Amharic tokens, the amBERT tokenizer was able to tokenize the Amharic sentence successfully.

For Khmer, similarly, we get all Khmer sentences from the Khmer-Thai and Khmer-Arabic datasets. Further, as mentioned earlier, the Amharic words are separated by whitespace which is easier to tokenize, but this is not the case with Khmer, therefore we have to preprocess the Khmer sentences before applying our extended mBERT tokenizer such that the sentences

[21] `https://huggingface.co/Davlan/bert-base-multilingual-cased-finetuned-amharic`

can be successfully encoded. So, we decided to use the word tokenizer from khmer-nltk[22] to tokenize all Khmer sentences. The obtained unique Khmer tokens are considered to be added to the model's vocabulary. In order to extend our mBERT vocabulary with only Khmer tokens that do not yet exist in the original mBERT vocabulary, we retokenize the Khmer tokens again with the original mBERT tokenizer (and rechecked with add_tokens() method), and only add tokens to our mBERT vocabulary that are unknown by the original mBERT model. In this way, we obtained 22,164 unique Khmer tokens. When our mBERT vocabulary is extended with these Khmer tokens, we can use the corresponding tokenizer to tokenize Khmer sentences that are already separated by whitespace. We will refer to our extended mBERT model with Khmer as the kmBERT model.

Below, we show an example of how a Khmer sentence is tokenized using the original mBERT (`bert-base-multilingual-cased`) tokenizer and using our Khmer extended mBERT tokenizer (kmBERT). So, we have the following Khmer sentence (not separated by whitespace):

```
'ពួកគេមិនគួរស្រដៀងទៅនឹងសម្លៀកបំពាក់ឬសម្លៀកបំពាក់របស់ disbelievers នេះបុរស។'
```

And the same Khmer sentence separated by whitespace:

```
'ពួកគេ មិន គួរ ស្រដៀង ទៅនឹង សម្លៀកបំពាក់ ឬ សម្លៀកបំពាក់ របស់ disbelievers នេះ បុរស ។'
```

The corresponding tokenized Khmer sentence using original mBERT where the Khmer sentence is not yet separated by whitespace:

```
['[UNK]', 'dis', '##beli', '##ever', '##s', '[UNK]', '[UNK]']
```

The corresponding tokenized Khmer sentence using original mBERT where the Khmer sentence is separated by whitespace:

```
['[UNK]', '[UNK]', '[UNK]', '[UNK]', '[UNK]', '[UNK]', '[UNK]', '[UNK]', '[UNK]', 'dis', '##beli', '##ever', '##s',
'[UNK]', '[UNK]', '[UNK]']
```

The corresponding tokenized Khmer sentence using kmBERT where the Khmer sentence is not yet separated by whitespace:

```
['[UNK]', 'dis', '##beli', '##ever', '##s', '[UNK]', '។']
```

The corresponding tokenized Khmer sentence using kmBERT where the Khmer sentence is separated by whitespace:

```
['ពួកគេ', 'មិន', 'គួរ', 'ស្រដៀង', 'ទៅនឹង', 'សម្លៀកបំពាក់', 'ឬ', 'សម្លៀកបំពាក់', 'របស់', 'dis', '##beli', '##ever', '##s',
'នេះ', 'បុរស', '។']
```

As we can see, the original mBERT tokenizer does not have any Khmer tokens (not even in the subtokens' form) in the corresponding vocabulary, therefore resulting in unknown tokens ('[UNK]') for all Khmer words/parts. After we extended the vocabulary with Khmer tokens, the kmBERT tokenizer is able to tokenize the Khmer sentence successfully when all words in the Khmer sentence are already separated by whitespace. This is because the kmBERT tokenizer is not trained in Khmer, and by default, the BERT tokenizer splits the sentence into tokens by whitespace. Since Khmer does not use whitespace between words, the kmBERT

[22] `https://pypi.org/project/khmer-nltk/#description`

tokenizer was not able to split the Khmer sentence into tokens, even though we extended the vocabulary with Khmer tokens. This results in one unknown token for the Khmer sentence part that is not separated by whitespace. When we separate the Khmer sentence by whitespace (using the khmer-nltk tool) and then feed the separated sentence into the kmBERT tokenizer, it successfully tokenizes the Khmer sentence.

After we obtained our amBERT tokenizer and kmBERT tokenizer, the embedding size of the corresponding mBERT model should be resized such that it is equal to the length of the corresponding vocabulary size. The embedding of these newly added tokens will be randomly initialized and learned through the post-training process.

4.2.2 Data preparation

MLM. We use the documents of low-resource languages (Amharic and Khmer) obtained from the CCAligned dataset [16] as input data for the MLM task. For Amharic, we have 46,066 documents, where 43,762 documents are used as training data, 921 documents as validation data, and 1,383 documents as test data. For Khmer, we have 71,994 documents, where 68,394 are used as training data, 1,439 as validation data, and 2,161 as test data. The MLM task is used to learn the representations of the newly added Amharic/Khmer words. Specifically, we randomly masked 15% of the input data where the model has to learn to predict the masked words depending on the surrounding context/words. In this way, it can learn the representation of these (Amharic/Khmer) words in the input data. As mentioned on the GitHub page[23] of the original BERT model [13], training using a batch with longer sequences is much more expensive than training using a batch with shorter sequences. A suggestion given by the authors on this GitHub page is to first train the model with sequences of 128 tokens for, e.g., 90,000 steps, and then additionally train the model with sequences of 512 tokens for, e.g., 10,000 steps. According to the authors, the very long sequences (of 512 tokens) are mainly required for the model to learn the positional embeddings. These positional embeddings can be learned quickly in a relatively small amount of steps. Therefore, we decided to split the documents into chunks of 128 tokens and chunks of 512 tokens, separately, before applying the masking (the remainder of each document is discarded). So, the model is first trained with a sequence length of 128 tokens and then further trained with a sequence length of 512 tokens.

Further, we also noticed that using AutoTokenizer with Transformers' recent version, our kmBERT tokenizer is able to tokenize the Khmer text without the text being separated by whitespace. However, since during finetuning, we use the same Transformers version as given in the MIA repository (v3.0.2), we would not be able to utilize this functionality, therefore, we have to separate the text using khmer-nltk. To make both the alignments during post-training and during fine-tuning the same, we applied both alignments the khmer-nltk to separate the Khmer text. Whereas, for MLM, as we use much larger document texts than the sentences used in the TLM task, we noticed that it was not efficient to use khmer-nltk to separate the text (it takes time). Since we used for post-training the most recent version (v4.33.3 at the time) of Transformers, we were able to use the kmBERT tokenizer directly to tokenize the Khmer text, and the resulting tokens are similar to that of the khmer-nltk, therefore, we decided to use the kmBERT tokenizer to tokenize the MLM's input texts including the separation (thus khmer-nltk is not used here).

[23] `https://github.com/google-research/bert/tree/master`

Dataset	Train data size	Validation data size	Test data size
Amharic-Arabic	195,688	4,118	6,182
Amharic-English	188,864	3,976	5,966
Amharic-Thai	147,084	3,096	4,646
Khmer-Arabic	201,680	4,244	6,372
Khmer-English	227,538	4,790	7,186
Khmer-Thai	153,776	3,236	4,858

Table 3: The training, validation, and test size of the input data for each language pair for the sentence-alignment post-training task (TLM).

TLM. In the data section, we already created datasets for the sentence alignment task for each language pair. So, for the TLM task, we combined the sentences from the low-resource language with the corresponding sentences from the high-resource language to form one single sentence. Here, we will have two variants of each sentence pair, one with the sentence from the low-resource language followed by the high-resource language, and one with the sentence from the high-resource language followed by the low-resource language. This will give us more data to train on and variation between the position of the low- and high-resource languages. Also here, we masked 15% of the input data. Since we have sentence pairs, we cannot split them into chunks of a certain length, but we have to limit the length of the input data due to ineffective learning and GPU memory. After experimenting, we decided a maximum length of 256 to be most appropriate. This means that we apply truncation with a maximum length of 256, where each sentence pair that is longer than 256 tokens will be truncated to the maximum length, and sentence pairs shorter than 256 tokens will be padded to the maximum length. More details about the training/validation/test data size for each language pair can be found in Table 3.

Using the mBERT tokenizer, we can encode the sentence pairs to make them ready as input data for our mBERT model. The tokenizer will give us the `input_ids`, `token_type_ids`, and `attention_masks`. The `input_ids` are the ids of the corresponding tokens in the vocabulary. The `token_type_ids` gives whether the tokens belong to the first or second sentence. The `attention_masks` gives whether the corresponding token is masked or not. The original MLM task uses chunks of text and therefore we do not have multiple sentences to which the token should belong. The `token_type_ids` is mostly useful for the NSP task. However, in our case, for the TLM task, we do have two different sentences, therefore we apply the `token_type_ids` such that the model understands that the tokens belong to different sentences. In this way, we hope the model can learn the alignment between the words in the sentence pair (of different languages).

In the case of an extended vocabulary and when the low-resource language is Khmer, we also need to preprocess the Khmer sentences before applying the mBERT tokenizer since the mBERT tokenizer separates the words by whitespace (when using an older version of AutoTokenizer). Therefore, we must ensure that the Khmer words are separated by whitespace by first tokenizing the sentences into words using khmer-nltk and rejoining the tokens/words using whitespace. After that, the Khmer sentences can be processed by the kmBERT tokenizer.

Hyperparameter	Value
Training batch size	32 / 8 / 16
Evaluation batch size	8
Learning rate	2×10^{-5}
Weight decay	0.01
Training steps	500,000 / 100,000 / 300,000
Scheduler type	Linear
Warmup steps	10,000

Table 4: The hyperparameter settings for the post-training process with Masked Language Modeling (MLM) and Translation Language Modeling (TLM).

4.2.3 Post-training

MLM. For post-training our extended mBERT model(s) with MLM using a sequence length of 128 for the two low-resource languages, we use a training batch size of 32, evaluation batch size of 8, and Adam as our optimizer with a learning rate of 2×10^{-5}. Further, a linear learning schedule with warmup is used where the learning rate linearly decreases from the initial learning rate to 0 after a warmup period. Here, we used a weight decay of 0.01 and warmup steps of 10,000. At the start of the post-training process, the model for each language pair is initialized with the weights of the original mBERT model and post-trained for 500,000 steps. To reproduce the training process, we set the random seed to 12345. Then, we additionally post-trained the resulting mBERT model(s) with MLM using a sequence length of 512. For this, we used a training batch size of 8 and we trained the model for another 100,000 steps. The other hyperparameter values remain the same as mentioned earlier. The hyperparameter settings are also shown in Table 4.

TLM. After MLM, we further trained our extended mBERT model(s) with TLM. Here, we use a sequence length of 256 and a training batch size of 16. The model(s) is trained for 300,000 steps. The other hyper-parameter values are the same as described earlier in the MLM paragraph.

After the post-training process, we obtained six models with extended vocabularies for the following language pairs: 1) Amharic-Arabic, 2) Amharic-Thai, 3) Amharic-English, 4) Khmer-Thai, 5) Khmer-Arabic, and 6) Khmer-English.

4.3 mDPR finetuning (QA task)

4.3.1 Data

Training and development data. The original training data provided by Asai et al. [5] for the MIA 2022 shared task includes the following languages: Arabic, Bengali, English, Finnish, Japanese, Korean, Russian, and Telegu. For our task, we need training data including the following high-resource languages: English, Arabic, and Thai. Therefore, we use the extended version of the MIA training data. The extended version includes Thai QA data as well. The Thai data are obtained from the XQuAD[24] dataset provided by Artetxe et al. [3] and the

[24] https://github.com/google-deepmind/xquad

iapp-wiki-qa-squad[25] dataset made by iApp Technology [42]. The XQuAD dataset consists of parallel data in 11 languages including Thai, we only took the Thai dataset containing 1,190 question-answer pairs. While the iapp-wiki-qa-squad dataset contains 7,242 question-answer pairs in Thai. We combined both datasets and obtained a combined Thai dataset containing 8,432 examples. From this dataset, we randomly took 675 (8%) examples as development data, and the remaining 7,757 examples as training data. Note that both the obtained training and development data are converted into the corresponding training and development data format as the one used for the MIA task.

Then, we added the Thai training data to the original MIA training data and the Thai development data to the original MIA (XORQA) development data. The final training data consists of 120,880 examples and the final development data consists of 5,203 examples. Both the training data and development data are used for finetuning our mDPR models. Each training example consists (at least) of a question, the answers, the positive contexts, a question ID, and the language. Each validation example (from the development data) has similar entries as the training data, only the language is left out.

Since we have limited QA data in Amharic or Khmer, we took 5,000 training examples from the above-mentioned training data in the corresponding high-resource language of the considered language pair. Then, we translated the questions in the training examples into the corresponding low-resource language using Google Translate through deep-translator.[26] For example, for the language pair Amharic and Arabic, the model is already post-trained on this language pair using TLM. Then, we took 5,000 examples from the Arabic training data and translated those Arabic questions into Amharic (the passages are kept in Arabic). The resulting training data with translated questions from Arabic to Amharic formed our fine-tuning data for Amharic. This process is repeated for each language pair. Finally, we add these training examples to the above-mentioned training data. Note that we only add the training examples that correspond to the specific language pair, thus we only add training examples with translated questions from Arabic to Amharic if the current language pair is Amharic and Arabic. Here, we use another translation tool than before because dl-translate is based on deep neural networks, which requires GPU resources, and as we have limited resources, we decided to use Google Translate instead. Another reason why we use Google Translate here is our texts for fine-tuning are shorter than the texts used in the post-training, and Google Translate cannot translate long texts. This is why we can use it here and not for the post-training texts.

Evaluation data. For evaluation, Asai et al. [5] provided the XORQA development data and MKQA development data. Since these development data are used for evaluation, it only consists of the following entries: a question with the corresponding question ID, answers, and the language. We also add the abovementioned Thai development data to this XORQA development data. Further, the MKQA development data provided by Asai et al. [5] consists of the following languages: Arabic, English, Spanish, Finnish, Japanese, Khmer, Korean, Malay, Russian, Swedish, Turkish, and Simplified Chinese. Since the original MKQA dataset provided by Longpre et al. [21] also includes Thai, we added the corresponding Thai examples to the MIA version of the MKQA dataset. As we are not interested in all languages provided by the MKQA dataset, we only use the evaluation data of the following languages for evaluating our

[25] https://github.com/iapp-technology/iapp-wiki-qa-dataset
[26] https://pypi.org/project/deep-translator/

Language	Training data	Evaluation		
		XOR dev data	MKQA dev data	Test data
Amharic (am)	-	-	-	2,622*
Arabic (ar)	18,430	1,378	1,758	-
Bengali (bn)	5,010	490	-	-
English (en)	51,568	-	1,758	-
Finnish (fi)	9,775	974	–	–
Japanese (ja)	7,870	693	–	–
Khmer (km)	-	-	1,758	5,000
Korean (ko)	4,398	473	–	–
Russian (ru)	9,302	1,018	–	–
Telegu (te)	6,770	564	-	-
Thai (th)	7,757*	675*	1,758*	-

Table 5: The data size for each language in the corresponding training data and evaluation data (including our extended MIA XORQA development data, modified/extended MIA MKQA development data, and our test data). * are the extended examples (originally not included in the MIA training/development data or the original MKQA dataset). - means the dataset does not include examples in the corresponding language. – means the original dataset does include examples in the corresponding language, but we do not include that language for evaluation.

fine-tuned models: Arabic, English, Khmer, and Thai. The mentioned development data are used to evaluate the performance of the mDPR+mGEN models compared to the baseline to see whether the performance of these languages is comparable to the baseline.

However, our main purpose is to evaluate the performance of Amharic and Khmer QA data. Therefore, our test data only consists of these two languages. The Khmer test data are taken from the original MKQA dataset (leaving out the used development data), and it contains 5,000 examples. The Amharic test data are taken from the Amharic Question Answering Dataset (AmQA) provided by Abedissa et al. [1], and it contains 2,622 examples. Each test data example has the same entries as the development data (used for evaluation) described above.

The data size for each language in the corresponding training and evaluation datasets (including development and test data) are summarized in Table 5.

4.3.2 Training

For training our mDPR models, we use the mDPR training script[4] provided in the MIA repository with the training parameter values as shown in Table 6. The parameter values are chosen based on the results of our mDPR replications. Since replication 1 (which uses the hyperparameter values of DPR [15]) gives better results than replication 2 (which uses the hyperparameter values given in the MIA repository), we used the parameter values of replication 1 for training our mDPR models. Further, due to limited GPU resources, we only used one single GPU to train the models and applied a half-precision floating point format (FP16) instead of a single-precision floating point format (FP32) during training. Half precision uses 16 bits instead of

Parameters	Value
Max grad norm	2.0
Random seed	12345
Sequence length	256
Warmup steps	1237
Training batch size	16
Learning rate	2×10^{-5}
Number of training epochs	30
Development batch size	64
Validation average rank start epoch	20

Table 6: Training parameters of our mDPR models using the post-trained mBERT retrievers with extended vocabularies.

32 bits which reduces the memory usage of neural networks and also makes the data transfer faster.[27] Further, we used the training and development data as described in Section 4.3.1 for training our models. Finally, we took the last trained checkpoints as our final models.

After training, we used our trained mDPR models to generate dense embeddings for the Wikipedia context passages. The Wikipedia context passages consist of the processed 100-token length passages[6] as provided in the MIA repository. However, the passages provided in the MIA repository do not contain passages in Thai and Amharic, therefore, we added the passages in Thai and Amharic as well to our context passages. The Thai passages are obtained from the Wikipedia context passages[28] provided in the CORA repository[2]. The Amharic passages are obtained from the Wikipedia dumps in Amharic[29] (processed as described in the MIA repository).

4.3.3 Evaluation

Development data evaluation. The evaluation process for evaluating the development data to see the performance compared to the baseline is the same as described in Section 3.2. This evaluation process is mainly done to see whether the final performance of the mDPR+mGEN models is still comparable to the original models provided in the MIA repository. This is not the evaluation of the alignment between low- and high-resource languages, but only a check to see that there is no large performance drop in other languages. So, firstly, we obtain the top 20 passages from our trained mDPR model using the evaluation data described in Section 4.3.1. Secondly, we convert the mDPR output to mGEN input data and obtain the mGEN results. Lastly, we run the evaluation script to evaluate the mGEN results. We used the following evaluation metrics to measure the final performance of our development data: token-level F1, Exact Match (EM), and BLEU.

Test data evaluation. During the retrieval process using mDPR, it is also checked whether the answer(s) to the questions can be found in the top 20 passages based on either the string

[27] https://docs.nvidia.com/deeplearning/performance/mixed-precision-training/index.html

[28] https://nlp.cs.washington.edu/xorqa/cora/models/all_w100.tsv

[29] https://archive.org/download/amwiki-20210801

match or the regex match. The string match is done by normalizing the context and answer(s) and then checking whether the sequence of answer tokens appears in the context. The regex match is done using the pattern search method to check whether the (normalized) answer(s) appears in the context. Since our main focus is on the mDPR model performance, we leave out the mGEN step in the evaluation of the test data to minimize the impact of the mGEN model. Therefore, to evaluate the performance of the mDPR model on the two low-resource languages, we use the following evaluation metrics based on regex match: answer-level Recall@10, answer-level Recall@20, Top10 ROUGE-1, and Top20 ROUGE-1.

Answer-level Recall@10 is computed by looking at whether a normalized answer can be found in the top 10 documents retrieved by the mDPR model for each question among all the questions (through regex match). For example, if we have 100 questions, and for 20 questions we can find an answer in the top 10 documents (no matter how many documents contain an answer for a single question), the Recall@10 score is 20%. Recall@20 is computed in the same way, but we looked at the top 20 documents instead of the top 10.

Besides the Recall evaluation metric, we also applied the ROUGE-1 metric as we assume that some passages might not contain the whole answer, but a part of the answer. Using the ROUGE-1 method we can detect the fraction of the answer that appears in the retrieved passages. The ROUGE-1 result is computed as the fraction of the answer (total amount of words in the answer) that appears in the retrieved passage. For example, if the answer consists of two words, and only one word appears in the passage, then the ROUGE-1 metric gives 0.5 as a result. This is done for each retrieved passage separately and we only take the highest fraction among all retrieved passages as the final result for that single question.

Taking into account that the retrieved documents can be in a different language than the question and answers, we looked at different evaluation methods. The first one is looking at the match between the original answers and the retrieved documents. The second one is by translating the answers and retrieved documents into English and then performing the matching process. The last one is by translating the answers into the language of the retrieved documents before the matching process. We will report the results of all three methods to ensure our final performance is more reliable since such evaluation methods can be influenced by the translation API. The translation is done using Google Translate.

4.3.4 Baseline

In our research, we have one main baseline, which is the mDPR model using the original mBERT-cased (`bert-base-multilingual-cased`) model without any modification or post-training (combined with the mGEN model provided by Asai et al. [5] in the MIA repository). The baseline model is fine-tuned and evaluated in the same way as described in the previous sections. Note that for the baseline, we did not apply any form of language alignment. We chose this model as our baseline because this will be similar to the original mDPR model used by Asai et al. [5], but only we used the mBERT-cased model since this included Thai in the pre-training, instead of the mBERT-uncased model. Our models will be an extension of this baseline model trained on language alignment to see the effect of language alignment compared to the baseline model.

5 Results

5.1 CORA replication

The obtained results (macro F1) from the replication of the mDPR (and mGEN) models with different settings on the XOR-TyDi QA and MKQA development data can be found in Table 7. The corresponding macro F1 is computed by taking the average over all languages. Also, the re-evaluated results of the MIA baselines using the provided checkpoints in the MIA repository are shown in the corresponding table. Further, in the table, we also show the reported results by the authors of the MIA 2022 Shared Task in the MIA paper [5] (on the test data) as well as in the MIA repository[3] (on the development data). In replication 1, we obtained the results from our last trained mDPR checkpoint, using the DPR [15] hyperparameters, and the mGEN checkpoint from MIA baseline 1. In replication 2, the results are obtained from our last trained mDPR checkpoint, using the MIA training script hyperparameters as shown in the MIA repository[4], and the mGEN checkpoint from MIA baseline 1. The re-evaluated results for baselines 1 and 2 are obtained from the evaluation (by us) using the mDPR and mGEN checkpoints provided by the authors in the MIA repository for the corresponding baseline.

As we can see, the replicated results are lower than that of the MIA baselines (the reported and the re-evaluated ones) based on macro F1. This might be due to the training process since we have limited resources to train the model, therefore, our training process is not the same as the training process of the MIA baselines. The MIA baselines are trained using 24GB RAM * 8 GPUs, whereas our replicated model is trained on a single GPU with 24GB RAM. As mDPR is sensitive to this training configuration, it might cause a drop in performance if not the exact configuration is used as the MIA baseline. Further, the fluctuations in the results can come from the random seed. In the original script, no random seed is set for generating the embeddings, whereas, in our replication, we fixed the random seed at 12345 for reproducibility. This might cause a difference in the performance of mDPR in our replication since the generated embeddings differ from the MIA baseline models.

However, the overall trend of our replication is similar to the MIA baselines as we can see in Table 8 and Table 9. So, in Table 8, we can see the F1 results per language for the XOR-TyDi development data. Overall, for all settings, Arabic is the best-performing language, and Bengali and Korean are the worst-performing languages based on token-level F1. Further, in Table 9, we can see the F1 results per language for the MKQA development data. For all settings, English is the best-performing language, and Khmer is the worst-performing language based on token-level F1. Thus, the overall trend is similar across the different settings. One big difference that we can see between the replicated (as well as re-evaluated) results and the reported results for baseline 1 is Khmer. The Khmer results are lower than the reported results for baseline 1, even though, we used the provided checkpoints by the authors[30]. To check whether the problem comes from the mDPR or mGEN (or both) in baseline 1, we evaluated the mDPR checkpoint from baseline 1 with the mGEN checkpoint from baseline 2, and the other way around, so the mDPR checkpoint from baseline 2 with the mGEN checkpoint from baseline 1. We observed that no matter which mDPR checkpoint is used, as long as it is combined with the mGEN checkpoint from baseline 1, the Khmer results are lower (F1 is around 2). Whereas, using the mGEN checkpoint from baseline 2, the Khmer results are better (F1 is around 6) for both the

[30]We contacted the authors about this issue, but they also cannot provide a clarification for this.

Setting	Macro F1		
	XOR-TyDi	MKQA	Total
Re-evaluated results			
(a) MIA baseline 1 (dev)	39.85	18.53	29.19
(b) MIA baseline 2 (dev)	40.55	17.38	28.97
Replicated results			
(c) Replication 1 - baseline 1 (dev)	36.52	15.61	26.07
(d) Replication 2 - baseline 1 (dev)	35.42	15.08	25.25
Reported results			
(e) MIA paper - baseline 2 (test)	37.95	17.14	27.55
(f) MIA repository - baseline 1 (dev)	38.90	18.10	28.50
(g) MIA repository - baseline 2 (dev)	39.80	17.40	28.60

Table 7: Macro F1 on the XOR-TyDi and MKQA development (dev) data (and test data). Settings (a) and (b) show the re-evaluated results by us using the baselines 1 and 2 models provided in the MIA repository. Setting (c) shows the results obtained using the mDPR model trained by us, with the DPR hyperparameters, and the provided mGEN model from MIA baseline 1. Setting (d) shows the results obtained using the mDPR model trained by us, with the hyperparameters from the training script in the MIA repository, and the provided mGEN checkpoint from MIA baseline 1. Setting (e) shows the results of baseline 2 reported by the authors in the MIA paper on the test data. Settings (f) and (g) show the results reported by the authors in the MIA repository for baselines 1 and 2.

mDPR checkpoints from baselines 1 and 2.

5.2 Post-training

To compare the performance of different setups in batch size and sequence length and whether the additional post-training is useful, we carried out a sequence of experiments for Amharic. Firstly, we experimented with two different sequence lengths (128 and 256) with the corresponding batch size (32 and 16), without additional post-training on the long sequences of 512 tokens. Besides that, we also train the model utilizing the additional post-training on the long sequences with 512 tokens and a batch size of 8. The MLM results are shown in Table 10. As we can see, the loss of the model on the test data and the corresponding perplexity are the lowest for amBERT$_{128-512}$, which is expected. This model is also further used for the TLM task.

Further, we have different TLM results for the different language pairs. The results are shown in Table 11. According to the test loss and perplexity, the best-performing language pair on the TLM task is kmBERT$_{th}$ (Khmer and Thai), which is expected since we assume that this language pair is the closest among all language pairs. However, for the Amharic model, the best-performing high-resource language is also Thai. This is not expected as we assume that Thai is a more distant language for Amharic than Arabic and less high-resourced than English. However, this might imply that according to the mBERT model, both low-resource languages are closer to Thai and therefore easier for the model to learn the corresponding embeddings. We will see whether this is still applicable during the QA task.

Setting	XOR-TyDi Language F1 (dev)						
	ar	bn	fi	ja	ko	ru	te
Re-evaluated results							
(a) MIA baseline 1	50.11	29.66	44.42	41.94	31.08	41.04	40.68
(b) MIA baseline 2	51.57	27.56	45.72	43.42	29.90	42.23	43.43
Replicated results							
(c) Replication 1 - baseline 1	44.73	27.43	40.25	38.86	27.91	37.27	39.21
(d) Replication 2 - baseline 1	45.33	23.67	40.96	38.71	24.66	37.41	37.17
Reported results							
(f) MIA repository - baseline 1	49.70	29.20	42.70	41.20	30.60	40.20	38.60
(g) MIA repository - baseline 2	51.30	28.70	44.40	43.20	29.80	40.70	40.20

Table 8: The evaluation of the CORA model on the XOR-TyDi development data. Settings (a) and (b) show the re-evaluation of the mDPR and mGEN models provided for baselines 1 and 2 in the MIA 2022 Shared Task. Setting (c) shows the evaluation of the mDPR model trained by us, using the DPR hyperparameters, and the mGEN model provided for baseline 1 in the MIA repository. Setting (d) shows the evaluation of the mDPR trained by us, using the hyperparameters from the training script in the MIA repository, and the mGEN model provided for baseline 1 in the MIA repository. Settings (f) and (g) show the results reported by the authors in the MIA repository for baselines 1 and 2.

5.3 Question Answering

5.3.1 Main evaluation

Model check. We evaluated the different models for each language pair according to the evaluation pipeline of the MIA task [5]. This evaluation is only done to see whether the models are still performing well for the original set of languages in the development data. As we can see in Table 12 and Table 13, for almost all languages, the performance shown by our models is worse than the baseline. The difference in the overall F1 between our models and the baseline is around 3-5% points. This is not a substantial difference, but still, we can see that language alignment between the language pairs can bring some confusion to the models. This might be because when we align two languages, the embeddings of the high-resource languages are affected by the low-resource languages, therefore diminishing the performance of the high-resource languages. This is also known as the curse of multilinguality [28].

mDPR evaluation. For evaluating our mDPR models on the two low-resource languages Amharic and Khmer, we used the answer-level Recall@10 and answer-level Recall@20 metrics. In Table 14, we can see the evaluation results for Amharic and Khmer. We have different evaluation settings for the models: NoTL, NoTL_tr, TL_EN, and TL_ret. In NoTL, we trained the mDPR model on the original training data provided by Asai et al. [5] for the MIA task without any modification or translation. Further, during evaluation, we directly evaluate the results given by the models without translating the retrieved passages or the answers. This setting is used to see the effect of post-training (MLM with TLM). In NoTL_tr, we trained the models on the extended training data with translated data for the low-resource language (question-passage alignment). The evaluation is still done without any modification or translation of the results given by our mDPR models.

Setting	MKQA Language F1 (dev)											
	ar	en	es	fi	ko	ms	ja	km	ru	sv	tr	zh
Re-evaluated results												
(a)	10.14	33.89	25.86	22.46	7.89	25.15	17.05	1.97	17.28	26.98	21.06	12.67
(b)	8.77	27.86	24.92	23.25	8.28	22.64	15.18	5.73	14.00	24.13	20.60	13.14
Replicated results												
(c)	8.70	30.61	24.84	18.00	5.97	19.12	14.29	1.83	13.75	21.65	19.91	8.59
(d)	8.55	30.39	24.53	17.05	5.79	16.71	13.54	1.48	13.95	21.04	19.13	8.73
Reported results												
(f)	8.90	33.90	25.10	21.10	6.70	24.60	15.30	6.00	15.60	25.50	20.40	13.70
(g)	8.80	27.90	24.90	23.30	8.30	22.60	15.20	5.70	14.00	24.10	20.60	13.10

Table 9: The evaluation of the CORA model on the MKQA development data. Settings (a) and (b) show the re-evaluation of the mDPR and mGEN models provided for baselines 1 and 2 in the MIA 2022 Shared Task. Setting (c) shows the evaluation of the mDPR model we trained, using the DPR hyperparameters, and the mGEN model provided for baseline 1 in the MIA repository. Setting (d) shows the evaluation of the mDPR trained by us, using the hyperparameters from the training script in the MIA repository, and the mGEN model provided for baseline 1 in the MIA repository. Settings (f) and (g) show the results reported by the authors in the MIA repository for baselines 1 and 2.

	Test Loss	**Perplexity**
amBERT_{128} (batch size = 32)	0.8612	2.37
amBERT_{256} (batch size = 16)	0.8096	2.25
$\text{amBERT}_{128-512}$ (batch size = 8)	0.7555	2.13

Table 10: The MLM post-training results of different sequence lengths and batch size. amBERT_{128} is trained with a batch size of 32 and a sequence length of 128 (500,000 steps). amBERT_{256} is trained with a batch size of 16 and a sequence length of 256 (500,000 steps). $\text{amBERT}_{128-512}$ is amBERT_{128} additionally trained with a batch size of 8 and a sequence length of 512 (100,000 steps). For our research, we use $\text{amBERT}_{128-512}$.

Further, we assume that the retrieved passages can be in a different language than the question/answer, therefore the regex matching might not be effective. That is why we decided to translate the results. So, TL_EN and TL_ret use the same model as in NoTL_tr, but the evaluation is done after the translation of the passages and/or the answers. In TL_EN, we first translate the retrieved passages and answers into English before we evaluate the results. We chose English because Google Translate is most effective in translating language pairs involved with English. In TL_ret, we translated the answers into the languages of the corresponding passages retrieved by our models before the matching in the evaluation. We decided to use these translation methods because longer texts can provide more context, which might be easier for the translation API to translate. However, this might be less effective for low-resource languages as Google Translate is less trained in these languages. Therefore, we also use the second method where we only translate the answers into the language of the corresponding passage, and since the answers are often very short, it might be easier for the API to translate the answers in low-resource languages. Hereby, we also limit the impact of translation by

	Test Loss	Perplexity
Amharic		
amBERT_{ar}	1.0322	2.81
amBERT_{en}	0.7683	2.16
amBERT_{th}	0.7190	2.05
Khmer		
kmBERT_{ar}	0.8447	2.33
kmBERT_{en}	0.7568	2.13
kmBERT_{th}	0.5582	1.75

Table 11: The TLM post-training results of different language pairs. For example, amBERT_{ar} indicates the model post-trained on the language pair Amharic and Arabic. 'km' stands for Khmer, 'en' for English, and 'th' for Thai.

	XOR-TyDi Language F1								Average		
	ar	bn	fi	ja	ko	ru	te	th	**F1**	**EM**	**BLEU**
Baseline	48.43	26.36	41.91	37.23	26.71	34.48	39.04	28.28	35.30	26.51	25.35
Amharic											
Model_{am-en}	43.53	22.66	35.43	34.62	26.05	30.82	35.62	25.64	31.80	23.78	22.45
Model_{am-ar}	43.86	23.00	38.80	33.68	23.04	29.55	35.64	25.90	31.68	23.81	22.38
Model_{am-th}	42.62	23.49	37.21	35.32	21.43	32.59	36.91	28.60	32.27	24.23	23.20
Khmer											
Model_{km-en}	43.62	22.12	37.81	33.34	23.20	30.85	37.46	25.06	31.68	23.75	22.60
Model_{km-ar}	43.15	18.58	36.65	34.53	20.70	30.88	33.12	25.70	30.41	22.58	21.75
Model_{km-th}	44.21	22.88	36.27	33.02	23.70	32.44	36.05	29.01	32.20	24.04	23.13

Table 12: Evaluation of the mDPR+mGEN results for the different models on the XORQA development data (Thai extended).

keeping the passages in their original languages.

Recall results. The results in Table 14 show that for NoTL, all three Amharic models trained using MLM and TLM outperformed the baseline. Here, the Amharic-Thai model is slightly better than the other two language pairs according to answer-level Recall@10, and the Amharic-Arabic model is slightly better according to answer-level Recall@20. However, after training on the question-passage alignment data during fine-tuning, the best-performing model is the Amharic-English model in all settings. According to McNemar's test (Table 15), all results of the Amharic-English model are significantly different than that of the baseline model, although the improvement seems to be small. With the Amharic-Arabic model, the results of the NoTL_tr and Recall@10 TL_ret settings are not significantly different than those of the baseline. With the Amharic-Thai model, almost all results (except for the NoTL setting) are not significantly different than that of the baseline. However we noticed that all results are low, so it remains difficult for the model to learn the embeddings in Amharic in order to perform well on this QA task.

Then, for the Khmer models, we can see that for all settings, the best-performing model is the Khmer-English model. Overall, the worst-performing model is the Khmer-Arabic model.

	MKQA Language F1				Average		
	ar	en	km	th	**F1**	**EM**	**BLEU**
Baseline	8.62	26.77	3.04	11.54	12.49	8.50	11.05
Amharic							
Model_{am-en}	7.57	27.56	3.19	8.47	11.70	8.28	10.61
Model_{am-ar}	8.65	27.63	3.79	10.16	12.56	8.66	11.17
Model_{am-th}	7.68	26.54	2.85	13.22	12.57	8.43	11.04
Khmer							
Model_{km-en}	8.37	27.06	5.34	10.05	12.71	9.02	11.37
Model_{km-ar}	7.85	25.72	2.40	8.98	11.24	7.86	9.91
Model_{km-th}	7.46	27.03	4.06	13.44	13.00	9.07	11.74

Table 13: Evaluation of the mDPR+mGEN results for the different models on the MKQA development data (Thai extended).

However, the performance of the Khmer-Arabic and Khmer-Thai models is comparable, but that of the Khmer-English model is much better (a difference of around 10% points). This is possibly due to the fact that the Khmer QA data is sampled from the Natural Questions dataset that was originally given in English. According to McNemar's test, all results of the Khmer-English model are significantly different than that of the baseline. With the Khmer-Arabic model, almost all results are significantly different than that of the baseline except for the NoTL_tr setting. Also, with the Khmer-Thai model, most of the results are significantly different than that of the baseline.

Further, we also noticed that Khmer results are better than those of Amharic, this might be due to the fact that the Khmer QA data includes multiple answers (aliases) for some questions, and these answers might also be given in the high-resource language such as English. Whereas for Amharic, only a single answer is given per question (mostly given in Amharic), and therefore more difficult to find the correct answer in the retrieved passages compared to Khmer.

For both low-resource languages (leaving out the baseline model), we can see that the best-performing model is the model aligned with English, and the model with the related high-resource language is comparable to the model with the more distant high-resource language. Although the difference is small for Amharic, we can say that for both low-resource languages, the data size of the high-resource language is more useful than the relatedness between the language pair for improving the QA performance of these low-resource languages.

ROUGE results. In addition, we also applied the ROUGE-1 metric in the top 10 and top 20 retrieved passages. As we assume that some passages might not include the whole answer, but a part of the answer. For example, the answer to a question consists of two words, and if only one of the two words is found in the passage, this metric will give 0.5 as a result. When both words (whole answer) are found, this metric will give 1.0 as a result. As we look at the top 10 retrieved passages, we will see if any of these passages contains (part of) the answer, and then take the highest result among the passages as our final result for that question. After that, we take the average result among all questions as our final model performance for the corresponding language pair. The same is done for the top 20 retrieved passages. The results are shown in Table 16. In this table, we can see that for Amharic, the (overall) best-

	Recall@10				Recall@20			
	NoTL	NoTL_tr	TL_EN	TL_ret	NoTL	NoTL_tr	TL_EN	TL_ret
Amharic								
Baseline_{am}	0.46		2.52	2.90	0.72		3.93	4.42
Model_{am-en}	2.10	**1.41**	**4.96**	**5.45**	2.82	**1.87**	**6.60**	**7.44**
Model_{am-ar}	2.10	0.72	4.16	3.74	**3.28**	0.88	5.91	5.45
Model_{am-th}	**2.25**	0.50	2.90	2.40	3.09	0.69	4.35	3.74
Khmer								
Baseline_{km}	7.74		11.74	13.08	9.94		15.02	16.84
Model_{km-en}	6.76	**18.64**	**25.46**	**25.54**	8.56	**22.00**	**30.24**	**30.50**
Model_{km-ar}	5.62	8.30	15.76	14.82	7.18	10.54	20.00	19.06
Model_{km-th}	7.70	9.50	15.76	14.44	9.62	11.30	19.04	17.82

Table 14: Recall performance of different models. In NoTL and NoTL_tr, we directly evaluate the retrieval results without modifying the results. However, in NoTL, the corresponding mDPR is not trained on the translated data for the low-resource language during fine-tuning, whereas in NoTL_tr, the corresponding mDPR is trained on the translated data. TL_EN is the setting where we first translate the retrieved documents and answers for each question into English before the evaluation. TL_ret is the setting where we first translate the answers into the language of the corresponding retrieved documents before the matching.

performing model is still the Amharic-English model. The Amharic-Arabic model is in second place. Further, for Khmer, it seems that the Khmer-English model is the best-performing model (in most of the settings). The difference in ROUGE-1 performance for the Amharic and Khmer models seems to be larger than using the Recall metric. However, we also have to keep in mind that stopwords (or other less meaningful words) might occur in the answers as well (such as 'the') and this will affect the correctness of the given results.

5.3.2 Language distribution

In Table 17 and Figure 5, we also show the language distribution of the top 20 retrieved passages by the different models. NoTR indicates the models are not trained on the translated data during fine-tuning, whereas TR indicates the models are trained on the translated data during fine-tuning. If we look at the language distribution in the retrieved documents by the baseline model, we can see that 93.35% of the documents that are retrieved, are Thai. This is quite interesting as we assume that Amharic is not similar to Thai, but apparently according to the mBERT model, Amharic is similar to Thai, which is also observed during the post-training process. However, after we post-trained the models, the language distribution is more spread. For the NoTR Amharic-English model, the highest percentage belongs to English, which is expected. Whereas for the NoTR Amharic-Arabic model and the NoTR Amharic-Thai model, no retrieved documents appear in the corresponding high-resource language (Arabic or Thai). However, we do see that the amount of retrieved documents in Amharic is larger than in the Amharic-English model. This might indicate that the models have learned the representation of the Amharic language better but did not actually align Amharic to the corresponding high-resource language during post-training.

	P-values							
	Recall@10				**Recall@20**			
	NoTL	NoTL_tr	TL_EN	TL_ret	NoTL	NoTL_tr	TL_EN	TL_ret
Amharic								
Model_{am-en}	< 0.05	< 0.05	< 0.05	< 0.05	< 0.05	< 0.05	< 0.05	< 0.05
Model_{am-ar}	< 0.05	0.1185	< 0.05	0.0609	< 0.05	0.5034	< 0.05	< 0.05
Model_{am-th}	< 0.05	1.0	0.3634	0.2031	< 0.05	1.0	0.4031	0.1234
Khmer								
Model_{km-en}	< 0.05	< 0.05	< 0.05	< 0.05	< 0.05	< 0.05	< 0.05	< 0.05
Model_{km-ar}	< 0.05	0.1383	< 0.05	< 0.05	< 0.05	0.1367	< 0.05	< 0.05
Model_{km-th}	0.9558	< 0.05	< 0.05	< 0.05	0.4343	< 0.05	< 0.05	0.0666

Table 15: P-values of different models compared to the baseline model using McNemar's test. In NoTL and NoTL_tr, we directly evaluate the retrieval results without modifying the results. However, in NoTL, the corresponding mDPR is not trained on the translated data for the low-resource language during fine-tuning, whereas in NoTL_tr, the corresponding mDPR is trained on the translated data. TL_EN is the setting where we first translate the retrieved documents and answers for each question into English before the evaluation. TL_ret is the setting where we first translate the answers into the language of the corresponding retrieved documents before the matching.

After language alignment during fine-tuning, the language distribution for the Amharic-English model is still quite spread. Further, the language with the highest percentage is still English (45.96%), but this percentage is much higher than before the question-passage alignment. So, we can see that question-passage alignment improves the retrieval in the corresponding high-resource language. This is also applicable to Arabic and Thai. When we align Amharic with Arabic or Thai, the retrieved passages are mostly taken from these high-resource languages. For the Amharic-Arabic model, 98.26% of the passages are retrieved from Arabic. For the Amharic-Thai model, 99.53% of the passages are retrieved from Thai.

The same is done for the Khmer models. The language distribution is shown in Table 18 and Figure 6. As we can see, the language distribution is quite spread for the baseline model and all NoTR models. In the baseline model, 46.81% of the passages are retrieved from Japanese, which is quite interesting as we do not see any similarity between these two languages. Further, we can see that for the NoTR models, the percentage of the retrieved documents in Khmer is higher than that of the baseline model. Whereas, after language alignment, we can see that for all three language pairs, almost all documents are retrieved from the corresponding high-resource language. This might indicate that the models have learned the representation of the Khmer language during post-training, and the alignment between the language pair during fine-tuning (question-passage alignment).

5.3.3 Retrieval analysis

We also looked into the top 10 retrieved passages in the TL_EN setting for each language pair to see whether the retrieved passages are related to the questions. Here, we randomly took 20 questions and the top 10 passages retrieved for those questions where at least an answer is found in the passages for that question. For the Amharic baseline, we see that the retrieved

	Top10 ROUGE-1				Top20 ROUGE-1			
	NoTL	NoTL_tr	TL_EN	TL_ret	NoTL	NoTL_tr	TL_EN	TL_ret
Amharic								
Baseline_{am}	7.29		30.87	35.70	7.84		35.33	41.93
Model_{am-en}	10.17	**8.94**	**36.45**	**39.25**	11.61	**9.78**	**41.36**	**44.75**
Model_{am-ar}	**11.57**	7.80	36.03	29.52	**14.41**	8.39	40.77	34.90
Model_{am-th}	10.69	7.25	33.03	36.58	12.58	7.79	37.08	41.82
Khmer								
Baseline_{km}	18.17		24.48	29.64	22.51		30.51	35.97
Model_{km-en}	18.98	**34.13**	**43.11**	**44.01**	22.67	**39.05**	**49.86**	**51.01**
Model_{km-ar}	16.42	16.68	30.45	30.42	19.94	19.93	37.05	37.60
Model_{km-th}	**20.62**	18.27	29.52	38.39	**24.71**	21.25	34.85	44.48

Table 16: ROUGE-1 performance of different models. In NoTL and NoTL_tr, we directly evaluate the retrieval results without modifying the results. However, in NoTL, the corresponding mDPR is not trained on the translated data for the low-resource language during fine-tuning, whereas in NoTL_tr, the corresponding mDPR is trained on the translated data. TL_EN is the setting where we first translate the retrieved documents and answers for each question into English before the evaluation. TL_ret is the setting where we first translate the answers into the language of the corresponding retrieved documents before the matching.

passages are almost the same for each question, and not specifically related to the questions. For the Amharic-English model, in 8 out of 20 questions, the passages are (somewhat) related to the questions. For example, we have the following question: "How many times did athlete Haile win an Olympic gold medal?", the Amharic-English model gives passages all related to the Olympics. The corresponding titles of the retrieved passages for this example are shown in Table 19.

For the Amharic-Arabic model, in 9 out of 20 questions, we have (somewhat) related passages, and for the Amharic-Thai model, 7 out of 20 questions. For the other questions, most passages share the same topic but are not related to the questions. This might be because the dense embeddings of these topics might be close to the dense embeddings of the questions, which confused the model for retrieving the right passages. Looking at the relatedness between the passages and the questions based on this small sample, we can say that the Amharic-Arabic model is the best-performing one. It seems that it is relatively difficult for the models to retrieve the correct passages. However, as we only looked at a small sample (4%), the observation might not be completely representative.

Further, for the Khmer baseline, in 7 out of 20 questions, the retrieved passages are (somewhat) related to the questions. For example, we have the question: "Who is the most subscribed YouTuber in the world?", the Khmer baseline gives passages all related to YouTube. The corresponding titles of the retrieved passages for this example are shown in Table 20. Then, for the Khmer-English model, in 15 out of 20 questions, we have (somewhat) related passages. For the Khmer-Arabic model, 13 out of 20 questions, and for the Khmer-Thai model, 11 out of 20 questions. For the other questions, most passages share the same topic but are not related to the questions.

	Baseline	**Model**$_{am-en}$		**Model**$_{am-ar}$		**Model**$_{am-th}$	
		NoTR	TR	NoTR	TR	NoTR	TR
am	-	8.38	-	19.56	-	11.24	-
ar	-	1.07	15.90	-	98.26	-	-
en	-	22.23	45.96	16.54	-	14.15	-
es	-	15.57	19.49	27.65	-	15.56	-
id	-	1.24	2.29	-	-	3.80	-
ja	6.28	15.28	1.55	2.25	-	7.53	-
km	-	1.27	-	1.41	-	5.41	-
ko	-	-	-	2.86	-	-	-
ms	-	1.26	2.08	-	-	3.25	-
sv	-	5.70	3.60	9.25	-	16.49	-
th	93.35	-	-	1.77	-	-	99.53
tr	-	10.36	-	9.05	-	2.85	-
zh	-	15.71	6.60	4.71	-	17.79	-

Table 17: Language distribution of the top 20 retrieved documents for the different Amharic models in percentage. NoTR implies the model is not trained on the translated training data for the low-resource language during fine-tuning. TR implies the model is trained on the translated training data for the low-resource language during fine-tuning. We only report languages where at least 1% of the retrieved documents appear in that language.

To get a better understanding of the models' performances for retrieving the correct passages, we looked into the top 100 passages and evaluated them against the ground truth answers (without any translation). As the ground truth answers are obtained from the Wikipedia articles, we have all Khmer Wikipedia context passages in the retrieval database as was done by Asai et al. [5], and we also have those Amharic Wikipedia passages from the articles used for the Amharic dataset by Abedissa et al. [1]. This evaluation with the top 100 passages is done for Amharic-English and Khmer-English models since these two language pairs give the best results among all language pairs. We looked at the answer-level Recall@100 and top100 ROUGE-1. Further, we also looked into the amount of passages in the top 100 that are in the corresponding low-resource language (Amharic or Khmer), and whether these passages contain the correct answer. Then, instead of having all passages in all languages available for retrieval, we only kept passages in Amharic/Khmer before retrieval. This will give us a sense of how well the model is performing in this single language without being confused by passages from other languages. Lastly, we also show the performance of only keeping the English passages for retrieval. The results are shown in Table 21.

As we can see in Table 21, the results of the top 100 for Amharic are slightly better than only having the top 20 passages, but the difference is not large, this means that for Amharic, it remains tough for the model to retrieve the correct passages. Also, the amount of retrieved passages in Amharic for each question is low. Only for 284 questions, we have Amharic passages retrieved in the top 100. Therefore, the results after filtering are also low. We assume that this low retrieval in Amharic is caused by the fine-tuning step on the question-passage alignment. Due to this alignment, the mapping between Amharic questions and Amharic pas-

	Baseline	**Model**$_{km-en}$		**Model**$_{km-ar}$		**Model**$_{km-th}$	
		NoTR	TR	NoTR	TR	NoTR	TR
am	-	-	-	-	-	-	-
ar	-	-	-	-	88.87	-	-
bn	-	-	-	-	-	-	-
en	13.26	21.94	87.82	11.25	4.10	51.26	9.20
es	3.58	19.49	6.18	8.88	1.79	6.46	3.07
fi	-	-	-	-	-	1.56	-
id	6.42	2.87	-	4.51	-	1.61	-
ja	46.81	-	-	1.36	-	1.73	-
km	4.21	30.79	-	10.47	-	8.63	-
ko	3.53	1.40	-	1.08	-	-	-
ms	-	2.08	-	3.63	-	1.48	-
ru	-	1.16	-	-	-	2.25	-
sv	5.00	13.25	1.85	22.66	2.30	16.33	-
th	5.25	-	-	1.78	-	-	83.64
te	7.66	-	-	-	-	-	-
tr	-	1.81	-	19.30	-	3.22	-
zh	2.33	2.45	-	12.06	-	4.01	-

Table 18: Language distribution of the top 20 retrieved documents for the different Khmer models in percentage. NoTR is the setting where the model is not trained on the translated training data for the low-resource language during fine-tuning. TR is the setting where the model is trained on the translated training data for the low-resource language during fine-tuning. We only report languages where at least 1% of the retrieved documents appear in that language.

sages is diminished, while mapping with the other (high-resource) languages is enlarged. This is also observed when looking at the language distribution mentioned earlier. We can see in Figure 5 and Figure 6 that there is no retrieval in the corresponding Amharic languages in the top 20 after the question-passage alignment. But if we only made the Amharic passages available for retrieval and no passages in other languages, we can see an improvement in performance, both the Recall and ROUGE-1 results. For Recall, we have a difference of around 8% points, and for ROUGE-1, 20% points compared to the top 20 results when all languages are considered. This means that the model can better find the correct passages when only Amharic passages are considered, which is as expected since there are fewer noises and the ground truth answers come from the Amharic Wikipedia articles. However, the results are still low, which means that it is still difficult for the model to find the correct passages and there is certainly room for improvement.

While, for Khmer, looking at the top 100 results, we can see that the performance is better than the top 20 retrievals. However, after filtering, thus only looking at the Khmer passages from the top 100, the results are very low, this is because there are almost no passages retrieved in Khmer in the top 100 (only for 7 questions). The same explanation as by Amharic can be applied here, the mapping between the Khmer questions and Khmer passages is diminished due to the question-passage alignment during finetuning. Further, the high results before filtering and low results after filtering for Khmer might also be caused by the Khmer QA dataset

Question	How many times did athlete Haile win an Olympic gold medal?
Titles	1. Argentina in the Olympic Games 2. Argentina in the Olympic Games 3. Chile in the Olympic Games 4. Australia at the Winter Paralympics 5. Argentina in the Olympic Games 6. 2016 Summer Olympics Chinese Table Tennis Team 7. Argentina in the Olympic Games 8. Argentina in the Olympic Games 9. Australia at the Winter Olympics 10. Australia at the Winter Olympics

Table 19: An example (question with the corresponding titles of the retrieved top 10 passages) given by the Amharic-English model where the passages are (somewhat) related to the question.

Question	Who is the Most Subscribed Youtuber in the World?
Titles	1. YouTube 2. YouTuber 3. YouTuber 4. YouTuber 5. YouTuber 6. YouTube 7. YouTube 8. YouTuber 9. YouTube channels with the most subscribers 10. YouTuber

Table 20: An example (question with the corresponding titles of the retrieved top 10 passages) given by the Khmer baseline model where the passages are (somewhat) related to the question.

that originates from the Natural Questions dataset that is originally in English. This means that the ground truth answers are more likely to be found in English (or other high-resource) Wikipedia articles instead of in Khmer Wikipedia articles. That is why we see higher Recall and ROUGE-1 results when considering all languages in the top 100 instead of only looking at the Khmer language since the answers are more likely to be found in Wikipedia passages in English or other high-resource languages. When only considering the Khmer passages for retrieval and no other languages, the performance is better than the filtered Khmer passages from the top 100, but still lower than the performance of considering all languages, which verifies our idea that the answers are less likely to be found in the Khmer passages. We also provide the results for Khmer with only English passages available for retrieval, and we indeed see that the performance is better than only having the Khmer passages and comparable to that of when all languages are considered.

	Recall	ROUGE-1
Amharic		
am-en$_{full}$ (top 20)	1.87	9.78
am-en$_{full}$ (top 100)	2.82	11.74
am-en$_{filtered}$ (top 100)	1.45	2.65
am-en$_{Amharic-only}$ (top 20)	10.60	29.97
am-en$_{English-only}$ (top 20)	1.14	8.73
Khmer		
km-en$_{full}$ (top 20)	22.00	39.05
km-en$_{full}$ (top 100)	29.88	49.09
km-en$_{filtered}$ (top 100)	0.02	0.02
km-en$_{Khmer-only}$ (top 20)	9.38	26.21
km-en$_{English-only}$ (top 20)	21.88	39.06

Table 21: Evaluation of the top 20 and top 100 passages (without translation) before and after filtering of the languages other than the target language (Amharic or Khmer) with the Amharic-English and Khmer-English models. We also show the evaluation of the top 20 passages with only passages of a single language available for retrieval (Amharic-only, Khmer-only, and English-only).

5.4 Ablation study

Besides our trained models and the baseline, we also did an ablation study where we left out the TLM part during post-training and only used the MLM post-trained model, and we further trained this model with the modified data during fine-tuning. This is only done for the language pair with English since this language pair gives the best-performing model among all language pairs. So, the difference between the MLM-only-trained models and MLM-TLM-trained models is that we left out the sentence alignment between the low-resource language and the high-resource languages with the MLM-only-trained models. The MLM-only-trained models are only post-trained on the low-resource language itself. However, the fine-tuning processes are the same, so both models are trained on question-passage alignment data. With this ablation study, we only measured the effect of having TLM or not.

The measured results are shown in Table 22 and Table 23. Based on the overall results using answer-level Recall@10, answer-level Recall@20, and ROUGE-1 metrics, the MLM-only-trained models seem to be worse than the models trained with MLM and TLM together, although, the differences are small in all settings, around 1% point (or less). So, we see that for all settings (except the Top20 ROUGE-1 results of the Khmer-English model in the TL_ret setting), the performance of the MLM-TLM-trained models is better than MLM-only-trained models. Therefore, we can say that TLM has a small effect in improving the models for these low-resource languages. However, the reason why the improvements are small might be that TLM is also used to learn the embeddings of these newly added Amharic/Khmer words with the help of the high-resource language, but these words are already learned during the MLM process. We think that the addition of TLM does not change the embeddings much, and therefore results in similar performance. Another reason might be that the language alignment is also done during the fine-tuning process, which reduces the effect of TLM and therefore results in similar performance between the models trained with or without TLM.

	Recall@10			Recall@20		
	NoTL_tr	TL_EN	TL_ret	NoTL_tr	TL_EN	TL_ret
Amharic						
$\text{Model}_{tlm-am-en}$	**1.41**	**4.96**	**5.45**	**1.87**	**6.60**	**7.44**
$\text{Model}_{mlm-am-en}$	0.61	4.00	4.23	1.07	5.99	6.18
Khmer						
$\text{Model}_{tlm-km-en}$	**18.64**	**25.46**	**25.54**	**22.00**	**30.24**	**30.50**
$\text{Model}_{mlm-km-en}$	18.42	25.22	25.32	21.92	30.08	30.46

Table 22: Recall performance of different models (MLM included and excluded) in the ablation study. In NoTL and NoTL_tr, we directly evaluate the retrieval results without modifying the results. However, in NoTL, the corresponding mDPR is not trained on the translated data for the low-resource language during fine-tuning, whereas in NoTL_tr, the corresponding mDPR is trained on the translated data. TL_EN is the setting where we first translate the retrieved documents and answers for each question into English before the evaluation. TL_ret is the setting where we first translate the answers into the language of the corresponding retrieved documents before the matching.

6 Discussion

In this section, we will discuss the outcomes of our research and describe the limitations and challenges of our research.

To improve QA performance for two low-resource languages, Amharic and Khmer, we fine-tuned the mBERT model using TLM during post-training and question-passage alignment (questions in the low-resource language aligned with the passages in the high-resource language) during fine-tuning. The outcomes are measured with the answer-level Recall and token-level ROUGE-1 metrics.

Based on the overall results, we found that language alignment can bring improvements compared to the baseline, especially looking at the alignment with English. However, the results remain low for these two low-resource languages (Amharic and Khmer) after language alignment, especially looking at the Recall performances. So, we can see that it remains difficult for the model to learn these two languages to perform well on the QA task. This is expected since these two low-resource languages did not exist in the mBERT model, and they are very different from the languages existing in the mBERT model, which makes it difficult for the model to learn these languages. However, we do see that the performance of the Khmer-English model is substantially higher than other language alignments, with a difference of at least 10% points. We believe that this is because the Khmer QA data is sampled from the Natural Questions dataset that is originally in English, therefore, the answers are more likely to be found in English Wikipedia articles. Since, with the Khmer-English model, the retrieved passages are mostly in English, the chance that the correct answers can be found is also larger compared to other models. Also, the Khmer QA dataset contains multiple answers (aliases) for some questions, which enlarges the possibility of a correct match, compared to the Amharic QA dataset which contains only a single answer for each question.

At the start of the research, we defined Amharic and Khmer as low-resource languages, and

	Top10 ROUGE-1			Top20 ROUGE-1		
	NoTL_tr	TL_EN	TL_ret	NoTL_tr	TL_EN	TL_ret
Amharic						
$\text{Model}_{tlm-am-en}$	**8.94**	**36.45**	**39.25**	**9.78**	**41.36**	**44.75**
$\text{Model}_{mlm-am-en}$	7.83	35.70	36.79	8.52	41.18	42.74
Khmer						
$\text{Model}_{tlm-km-en}$	**34.13**	**43.11**	**44.01**	**39.05**	**49.86**	51.01
$\text{Model}_{mlm-km-en}$	33.86	42.62	43.77	38.72	49.29	**51.05**

Table 23: ROUGE-1 performance of different models (MLM included and excluded) in the ablation study. In NoTL and NoTL_tr, we directly evaluate the retrieval results without modifying the results. However, in NoTL, the corresponding mDPR is not trained on the translated data for the low-resource language during fine-tuning, whereas in NoTL_tr, the corresponding mDPR is trained on the translated data. TL_EN is the setting where we first translate the retrieved documents and answers for each question into English before the evaluation. TL_ret is the setting where we first translate the answers into the language of the corresponding retrieved documents before the matching.

we expected that the QA performance could be improved by aligning these two languages with linguistically similar high-resource languages. However, after our research, we see that these two languages are not only low-resourced but also completely different from the high-resource languages, i.e., they use a very different script. This makes it extra hard for the model to connect the low-resource language with the related high-resource languages. So, we do not only have to deal with a low-resource language but a low-resource high-distance language.

However, we think that the QA performance can be improved if we can provide more data (during post-training and fine-tuning) in these languages to train the model. Further, we see that the performance of the high-resource languages diminished due to the language alignment, known as the curse of multilinguality [28]. This also shows how difficult it is to improve the multilingual QA performance for low-resource languages.

Furthermore, we compared the models trained using three different language pair alignments for each low-resource language, namely alignment with a related high-resource language, alignment with a more distant high-resource language, and alignment with English. We found that the alignment with the English language is the best-performing one and the alignment with a more distant high-resource language is the worst-performing one (although it is comparable to the performance of the related high-resource language). We would expect that the alignment between the low-resource language and the related high-resource language should perform better than the alignment with English. However, if we look at Amharic and Arabic, although they belong to the same Semitic branch, they do have different scripts. So, for mBERT, it might still be difficult to take advantage of the relationship between these two languages. The same applies to Khmer. Khmer and Thai have similar scripts in the eyes of humans, but they are not exactly the same on the character code level, which makes it hard for the model to see the relatedness between the two. So, the Khmer script is visually similar to the Thai script, but both scripts consist of unique characters, and therefore a different unicode. For example, for us, the letter I (uppercase i) and the letter l (lowercase L) look similar, but for the machines, they are as different as between the letter k and the letter l. This means

that in our case, for the models, the alignment between the low-resource language and the related high-resource language is not much different from the alignment with a more distant high-resource language. However, we believe that when two languages share the same script, the results of the alignment would be better. So, in our case, the language with the most data, English, is better than the other two language alignments. This is expected since the mBERT model is extensively trained in English and the alignment with English can make the model better learn the low-resource languages compared to other alignments we used.

Moreover, the low performance of Amharic might also be due to the fact that the questions and relevant answers are taken from the Amharic Wikipedia articles, and there is a chance that the Wikipedia articles in other languages do not contain the correct answers for these Amharic questions. Therefore, having passages taken from Wikipedia articles in multiple languages for retrieval, and alignment with the high-resource languages can make the model more likely to retrieve passages in other languages instead of in Amharic. So, this might diminish the chance of retrieving the correct passages. This can be one of the reasons why we have such low performance for Amharic. Thus, we would expect that when more Amharic articles are retrieved, the performance would also be better. From our retrieval analysis, we can see that when we only kept the Amharic passages for retrieval, the performance indeed improved. However, when we have all passages in all languages, the model is more likely to retrieve passages from the high-resource languages instead of in Amharic. It remains a question for us whether the passages in the high-resource languages contain the correct answers since the Amharic articles are not guaranteed to have a translated Wikipedia version in other languages as well. However, the question-passage alignment can be viewed as successful in the sense that when giving an Amharic question, the model is more likely to map the Amharic question to the passages of the corresponding high-resource languages. But does this mean that the model is becoming worse at retrieving the correct passages in Amharic? We do see that the retrieval in Amharic is higher before the question-passage alignment, whereas the retrieval in the corresponding high-resource language is lower (or completely not) before the question-passage alignment. So, we think that the model might have learned the representation of the Amharic language with MLM+TLM (post-training), and the mapping between the low-resource and high-resource languages through question-passage alignment (fine-tuning). However, it remains difficult for the model to do the correct mapping while also becoming better in Amharic for retrieving the correct passages.

Further, the Khmer QA data are translated from the Natural Questions data that is originally in English. This means that the answers are originally found in English Wikipedia articles and not in Khmer Wikipedia articles. Therefore, when more English articles are retrieved, we would expect that the performance is also better. As the available Khmer Wikipedia articles are much fewer than the available English Wikipedia articles, it is less likely that the Khmer articles have the correct passages with the answers. This is also confirmed by the retrieval analysis we have done, where we showed that the performance of the Khmer model is worse when we only kept the Khmer passages for retrieval compared to only keeping the English passages for retrieval. Then, from the language distribution, we can see that MLM+TLM (post-training) improved the performance in retrieving the Khmer passages, whereas the question-passage alignment (fine-tuning) improved the performance in retrieving the passages in the corresponding high-resource languages. Although the alignment with English seems to improve the QA performance for Khmer, it is still limited as the questions are given in Khmer and most of the answers are

also given in Khmer, whereas, the right passages are actually in English (or probably in other high-resource languages). Therefore, it is still difficult for the model to perform the mapping between Khmer and English as Khmer is a low-resource language and very different from English.

Limitations. We faced in our research several limitations or challenges. The first limitation is the data quality. In our post-training alignment, we used the CCAligned dataset, although we used translation and similarity scores above a certain threshold, there is still a chance that a portion of the texts in each language pair is not actually aligned. When this is the case, the model can get confused by these not-aligned text pairs, which diminishes the performance of the model.

The second limitation is the translation quality. In our research, especially in the fine-tuning process, we trained the model on the translated questions, but the translation quality is not guaranteed. This is because the languages we work with are low-resource languages, and the translation API (Google Translate) we use is not well-trained in these languages. Thus, how well Google Translate can translate Amharic or Khmer remains a question for us. However, we believe that the performance of our model would improve if we can improve the translation quality.

The third limitation is the evaluation method. The evaluation method we used is based on answer-level Recall and token-level ROUGE-1 metrics. As we evaluated mDPR, which is a retrieval model, but we only have the ground truth answers for the extraction task (and no relevance labels), we can only evaluate our retrieval task with extraction labels. Since the Recall metric is an extraction metric, the ROUGE-1 metric is a generation metric, and our task is a retrieval task, there is some discrepancy between our task and the evaluation metrics. Therefore, we cannot 100% guarantee that the percentage we show is the percentage of correct answers since from our retrieval analysis, we can see that several retrieved passages are not related to the question, but the answer coincidentally appears in the passages. We can improve this in the future by providing relevance labels or transforming the task into an extraction or a generation task.

The last challenge is that it is very difficult for us to distinguish between the limitations that come from the model (the way we trained it), the data, and the evaluation method since they are all related. Therefore, it is hard for us only to choose one aspect to improve, we have to deal with all aspects in order to improve the QA performance for these low-resource languages.

7 Conclusion

In this research, we have addressed two main research questions. With the first research question, we investigated the impact of language alignment (during post-training and fine-tuning) for two (extremely) low-resource languages (Amharic and Khmer) compared to the baseline model (mBERT cased). With the second research question, we investigated which language pair alignment is most effective. We asked ourselves the following question: Is it easier for the model to learn the alignment when two languages are closer to each other, or when we have a high-resource language with the most training data (i.e., English)?

The answer to our first research question is that the language alignment in general (slightly) improves the QA performance compared to the baseline for both languages. Also, with the ablation study, we see that based on the overall results for the Amharic-English and Khmer-English models, the language alignment through TLM brings some improvements above MLM. The answer to our second research question is that the alignment with English performs better than the alignment with a related high-resource language and the alignment with a non-related high-resource language (the last two are comparable). This is probably because the similarity between the low-resource language and the related high-resource language is limited according to the mBERT model since the languages do not share the same script. Therefore the alignment with English performs better since English is higher-resource than the related high-resource language. We conclude that when a language is not only low-resourced but also very different from other (high-resource) languages (high-distance), it is difficult to improve the QA performance for that language through language alignment.

For future research, we can consider similar language pairs using the same scripts where possible, to see whether alignment with English is still better or whether the alignment with a related high-resource language using the same script is better (this does not apply to Amharic or Khmer). Further, to ensure the evaluation process is more reliable, we can either use extractive QA or generative QA where we can better evaluate the results using our evaluation metrics. Lastly, we can improve the data/translation quality in the future to improve the model performance.

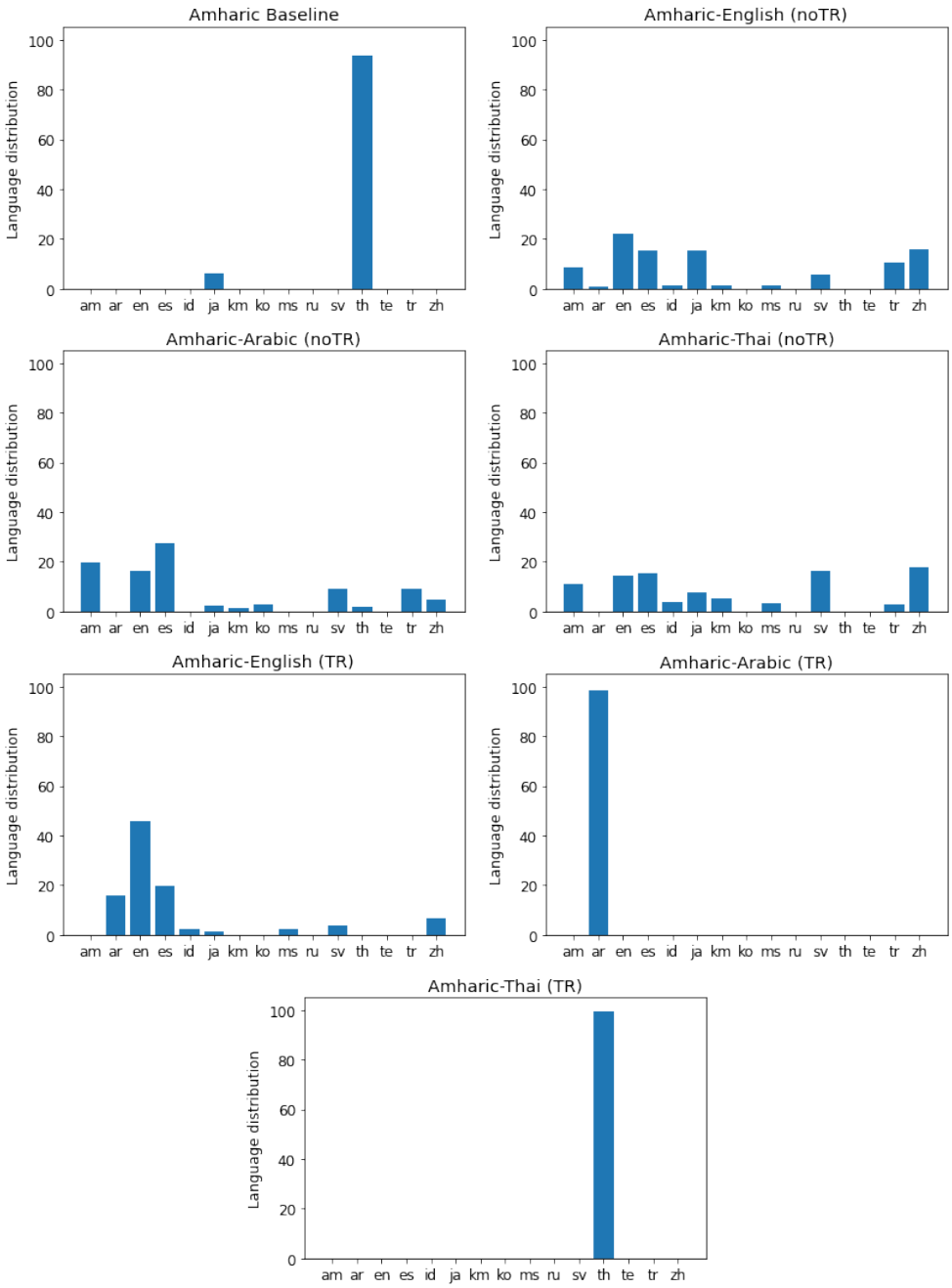

Figure 5: Language distribution of the top 20 retrieved documents for the different Amharic models. NoTR is the setting where the model is not trained on the translated training data for the low-resource language during fine-tuning. TR is the setting where the model is trained on the translated training data for the low-resource language during fine-tuning. We only report languages where at least 1% of the retrieved documents appear in that language.

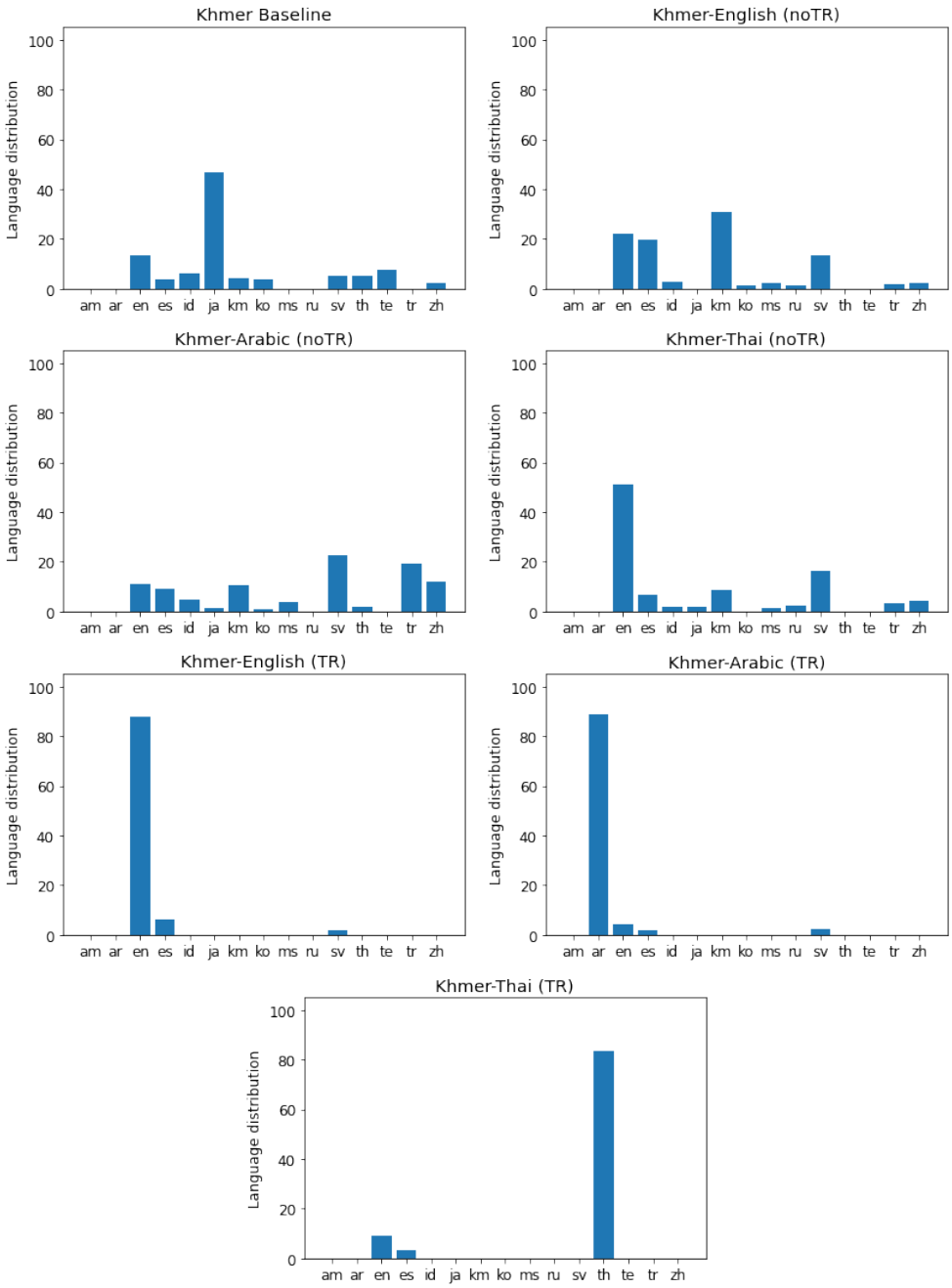

Figure 6: Language distribution of the top 20 retrieved documents for the different Khmer models. NoTR is the setting where the model is not trained on the translated training data for the low-resource language during fine-tuning. TR is the setting where the model is trained on the translated training data for the low-resource language during fine-tuning. We only report languages where at least 1% of the retrieved documents appear in that language.

References

1. Abedissa, T., Usbeck, R. & Assabie, Y. AmQA: Amharic Question Answering Dataset. https://arxiv.org/abs/2303.03290 (2023).
2. Antony, B. & Paul, N. R. *Question Answering System for Tamil Using Deep Learning* in *Speech and Language Technologies for Low-Resource Languages* (eds M, A. K. *et al.*) (Springer International Publishing, Cham, 2023), 244–252. ISBN: 978-3-031-33231-9. https://link.springer.com/chapter/10.1007/978-3-031-33231-9_17.
3. Artetxe, M., Ruder, S. & Yogatama, D. *On the Cross-lingual Transferability of Monolingual Representations* in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (Association for Computational Linguistics, Online, July 2020), 4623–4637. https://aclanthology.org/2020.acl-main.421.
4. Asai, A., Yu, X., Kasai, J. & Hajishirzi, H. *One Question Answering Model for Many Languages with Cross-lingual Dense Passage Retrieval* in *Advances in Neural Information Processing Systems* (eds Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P. & Vaughan, J. W.) **34** (Curran Associates, Inc., 2021), 7547–7560.
5. Asai, A. *et al. MIA 2022 Shared Task: Evaluating Cross-lingual Open-Retrieval Question Answering for 16 Diverse Languages* in *Proceedings of the Workshop on Multilingual Information Access (MIA)* (Association for Computational Linguistics, Seattle, USA, July 2022), 108–120. https://aclanthology.org/2022.mia-1.11.
6. Banerjee, S., Naskar, S. K. & Bandyopadhyay, S. *BFQA: A Bengali Factoid Question Answering System* in *Text, Speech and Dialogue* (eds Sojka, P., Horák, A., Kopeček, I. & Pala, K.) (Springer International Publishing, Cham, 2014), 217–224. ISBN: 978-3-319-10816-2. https://link.springer.com/chapter/10.1007/978-3-319-10816-2_27.
7. Bañón, M. *et al. ParaCrawl: Web-Scale Acquisition of Parallel Corpora* in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (Association for Computational Linguistics, Online, July 2020), 4555–4567. https://aclanthology.org/2020.acl-main.417.
8. Clark, J. H. *et al.* TyDi QA: A Benchmark for Information-Seeking Question Answering in Typologically Diverse Languages. *Transactions of the Association for Computational Linguistics* **8,** 454–470. https://aclanthology.org/2020.tacl-1.30 (2020).
9. Conneau, A. & Lample, G. *Cross-lingual Language Model Pretraining* in *Advances in Neural Information Processing Systems* (eds Wallach, H. *et al.*) **32** (Curran Associates, Inc., 2019). https://proceedings.neurips.cc/paper_files/paper/2019/file/c04c19c2c2474dbf5f7ac4372c5b9af1-Paper.pdf.
10. Darvishi, K., Shahbodaghkhan, N., Abbasiantaeb, Z. & Momtazi, S. PQuAD: A Persian question answering dataset. *Computer Speech & Language* **80,** 101486. ISSN: 0885-2308. https://doi.org/10.1016/j.csl.2023.101486 (2023).

11. Das, A., Mandal, J., Danial, Z., Pal, A. R. & Saha, D. An improvement of Bengali factoid question answering system using unsupervised statistical methods. *Sādhanā* **47,** 2. https://link.springer.com/article/10.1007/s12046-021-01765-3 (2022).

12. Das, A. & Saha, D. Question Answering System Using Deep Learning in the Low Resource Language Bengali. *Convergence of Deep Learning In Cyber-IoT Systems and Security,* 207–230. https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119857686.ch10 (2022).

13. Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding* in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (Association for Computational Linguistics, Minneapolis, Minnesota, June 2019), 4171–4186. https://aclanthology.org/N19-1423.

14. Do, P., Phan, T. H. V. & Gupta, B. B. Developing a Vietnamese Tourism Question Answering System Using Knowledge Graph and Deep Learning. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* **20.** ISSN: 2375-4699. https://doi.org/10.1145/3453651 (June 2021).

15. Karpukhin, V. *et al. Dense Passage Retrieval for Open-Domain Question Answering* in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (Association for Computational Linguistics, Online, Nov. 2020), 6769–6781. https://aclanthology.org/2020.emnlp-main.550.

16. El-Kishky, A., Chaudhary, V., Guzmán, F. & Koehn, P. *CCAligned: A Massive Collection of Cross-Lingual Web-Document Pairs* in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (Association for Computational Linguistics, Online, Nov. 2020), 5960–5969. https://aclanthology.org/2020.emnlp-main.480.

17. Kumar, G. K., Gehlot, A., Mullappilly, S. S. & Nandakumar, K. *MuCoT: Multilingual Contrastive Training for Question-Answering in Low-resource Languages* in *Proceedings of the Second Workshop on Speech and Language Technologies for Dravidian Languages* (Association for Computational Linguistics, Dublin, Ireland, May 2022), 15–24. https://aclanthology.org/2022.dravidianlangtech-1.3.

18. Kwiatkowski, T. *et al.* Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics* **7,** 452–466. https://aclanthology.org/Q19-1026 (2019).

19. Le, K., Nguyen, H., Le Thanh, T. & Nguyen, M. *VIMQA: A Vietnamese Dataset for Advanced Reasoning and Explainable Multi-hop Question Answering* in *Proceedings of the Thirteenth Language Resources and Evaluation Conference* (European Language Resources Association, Marseille, France, June 2022), 6521–6529. https://aclanthology.org/2022.lrec-1.700.

20. Liu, J. *et al. Enhancing Multilingual Document-Grounded Dialogue Using Cascaded Prompt-Based Post-Training Models* in *Proceedings of the Third DialDoc Workshop on Document-grounded Dialogue and Conversational Question Answering* (eds Muresan, S. *et al.*) (Association for Computational Linguistics, Toronto, Canada, July 2023), 44–51. https://aclanthology.org/2023.dialdoc-1.5.

21. Longpre, S., Lu, Y. & Daiber, J. MKQA: A Linguistically Diverse Benchmark for Multilingual Open Domain Question Answering. *Transactions of the Association for Computational Linguistics* **9,** 1389–1406. ISSN: 2307-387X. https://doi.org/10.1162/tacl%5C_a%5C_00433 (Dec. 2021).

22. Lopes, É. P. *et al.* Exploring BERT for Aspect-based Sentiment Analysis in Portuguese Language. **35.** https://journals.flvc.org/FLAIRS/article/view/130601 (May 2022).

23. Mozafari, J., Kazemi, A., Moradi, P., Nematbakhsh, M. A. & Jafari, S. PerAnSel: A Novel Deep Neural Network-Based System for Persian Question Answering. *Intell. Neuroscience* **2022.** ISSN: 1687-5265. https://doi.org/10.1155/2022/3661286 (Jan. 2022).

24. Nogueira, R. & Cho, K. Passage Re-ranking with BERT. https://arxiv.org/abs/1901.04085 (2020).

25. Pan, L. *et al. Multilingual BERT Post-Pretraining Alignment* in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (Association for Computational Linguistics, Online, June 2021), 210–219. https://aclanthology.org/2021.naacl-main.20.

26. Papineni, K., Roukos, S., Ward, T. & Zhu, W.-J. *Bleu: a method for automatic evaluation of machine translation* in *Proceedings of the 40th annual meeting of the Association for Computational Linguistics* (2002), 311–318. https://aclanthology.org/P02-1040.pdf.

27. Park, G.-M., Hong, S.-E. & Park, S.-B. *Post-Training with Interrogative Sentences for Enhancing BART-based Korean Question Generator* in *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)* (eds He, Y., Ji, H., Li, S., Liu, Y. & Chang, C.-H.) (Association for Computational Linguistics, Online only, Nov. 2022), 202–209. https://aclanthology.org/2022.aacl-short.26.

28. Pfeiffer, J. *et al. Lifting the Curse of Multilinguality by Pre-training Modular Transformers* in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (eds Carpuat, M., de Marneffe, M.-C. & Meza Ruiz, I. V.) (Association for Computational Linguistics, Seattle, United States, July 2022), 3479–3495. https://aclanthology.org/2022.naacl-main.255.

29. Phan, T. & Do, P. Building a Vietnamese Question Answering System Based on Knowledge Graph and Distributed CNN. *Neural Comput. Appl.* **33,** 14887–14907. ISSN: 0941-0643. https://doi.org/10.1007/s00521-021-06126-z (Nov. 2021).

30. Pramana, R. & Prasojo, R. E. *Improving Low-Resource Question Answering with Cross-Lingual Data Augmentation Strategies* in *2022 10th International Conference on Information and Communication Technology (ICoICT)* (2022), 110–115. https://ieeexplore.ieee.org/document/9914847.

31. Rajpurkar, P., Zhang, J., Lopyrev, K. & Liang, P. *SQuAD: 100,000+ Questions for Machine Comprehension of Text* in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing* (Association for Computational Linguistics, Austin, Texas, Nov. 2016), 2383–2392. https://aclanthology.org/D16-1264.

32. Ravva, P., Urlana, A. & Shrivastava, M. *AVADHAN: System for Open-Domain Telugu Question Answering* in *Proceedings of the 7th ACM IKDD CoDS and 25th COMAD* (Association for Computing Machinery, Hyderabad, India, 2020), 234–238. ISBN: 9781450377386. https://doi.org/10.1145/3371158.3371193.

33. Reimers, N. & Gurevych, I. *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks* in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing* (Association for Computational Linguistics, Nov. 2019). https://arxiv.org/abs/1908.10084.

34. Riabi, A. *et al. Synthetic Data Augmentation for Zero-Shot Cross-Lingual Question Answering* in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing* (Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, Nov. 2021), 7016–7030. https://aclanthology.org/2021.emnlp-main.562.

35. Schwenk, H., Chaudhary, V., Sun, S., Gong, H. & Guzmán, F. *WikiMatrix: Mining 135M Parallel Sentences in 1620 Language Pairs from Wikipedia* in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume* (Association for Computational Linguistics, Online, Apr. 2021), 1351–1361. https://aclanthology.org/2021.eacl-main.115.

36. Shen, X. *et al.* Low-Resource Dense Retrieval for Open-Domain Question Answering: A Comprehensive Survey. https://arxiv.org/abs/2208.03197 (2022).

37. Singh, J., McCann, B., Keskar, N. S., Xiong, C. & Socher, R. XLDA: Cross-Lingual Data Augmentation for Natural Language Inference and Question Answering. https://arxiv.org/abs/1905.11471 (2019).

38. Tchakaloff, B., Saudrais, S. & Babau, J.-P. *ORQA: Modeling Energy and Quality of Service within AUTOSAR Models* in *Proceedings of the 9th International ACM Sigsoft Conference on Quality of Software Architectures* (Association for Computing Machinery, Vancouver, British Columbia, Canada, 2013), 3–12. ISBN: 9781450321266. https://doi.org/10.1145/2465478.2465488.

39. Thompson, M. *Thai Vs. Cambodian: How Similar Are These 2 Incredible Languages? - Ling App* 2023. https://ling-app.com/th/thai-vs-cambodian/.

40. Toraman, C., Yilmaz, E. H., Şahinuç, F. & Ozcelik, O. Impact of tokenization on language models: An analysis for turkish. *ACM Transactions on Asian and Low-Resource Language Information Processing* **22,** 1–21. https://doi.org/10.1145/3578707 (2023).

41. Vemula, R., Nuthi, M. & Srivastava, M. *TeQuAD:Telugu Question Answering Dataset* in *Proceedings of the 19th International Conference on Natural Language Processing (ICON)* (Association for Computational Linguistics, New Delhi, India, Dec. 2022), 300–307. https://aclanthology.org/2022.icon-main.36.

42. Viriyayudhakorn, K. & Polpanumas, C. *iapp_wiki_qa_squad* version 1 (Zenodo, Feb. 2021). https://doi.org/10.5281/zenodo.4539916.

43. Vuong, T.-H.-Y., Nguyen, H.-T., Nguyen, Q.-H., Nguyen, L.-M. & Phan, X.-H. Improving Vietnamese Legal Question–Answering System based on Automatic Data Enrichment. https://arxiv.org/abs/2306.04841 (2023).

44. Wang, Z., Mayhew, S., Roth, D., *et al.* Extending multilingual BERT to low-resource languages. *arXiv preprint arXiv:2004.13640.* https://doi.org/10.48550/arXiv.2004.13640 (2020).

45. Wanjawa, B. W. *et al.* KenSwQuAD—A Question Answering Dataset for Swahili Low-Resource Language. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* **22.** ISSN: 2375-4699. https://doi.org/10.1145/3578553 (Apr. 2023).

46. Wu, L., Zhu, J., Zhang, X., Zhuang, Z. & Feng, Z. *Enhancing Low-Resource Languages Question Answering with Syntactic Graph* in *Database Systems for Advanced Applications. DASFAA 2022 International Workshops* (eds Rage, U. K., Goyal, V. & Reddy, P. K.) (Springer International Publishing, Cham, 2022), 175–188. ISBN: 978-3-031-11217-1. https://link.springer.com/chapter/10.1007/978-3-031-11217-1_13.

47. Xu, H., Liu, B., Shu, L. & Yu, P. *BERT Post-Training for Review Reading Comprehension and Aspect-based Sentiment Analysis* in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (Association for Computational Linguistics, Minneapolis, Minnesota, June 2019), 2324–2335. https://aclanthology.org/N19-1242.