IndicQA

Abhishek Pratap Singh **Vignesh P** **Dibyakanti Kumar**
Indian Institute of Technology Guwahati
Group No: 1
{abhishek.pratap, vigneshp, dibyakan}@iitg.ac.in

Abstract

We pre-train BERT-class models for better understanding of Hindi and Tamil sentences which can in turn improve the model's performance in NLP tasks such as Question-Answering. However unlike English we cannot rely on the bulk of the data but rather try to find data similar to ours for pre-training. We will be exploring 3 methods for pre-training to assess which one suits our dataset the best. We will also be using atleast 3 datasets from differnt sources to see which one helps the model the most on pre-training.

1 Introduction

Multilingual Question Answering(MLQA) is an NLP tasks involving question classification, passage retrieval and ranking, and answer extraction in different languages. Our work mainly focuses on the final part of the question answering system, also called Multilingual machine comprehension(MMC). The main challenge of an MLQA system is understanding the syntactic and semantic structure of different languages. After the advent of Transformer-based models, the question answering systems achieved remarkable accuracy, but most of the work of question answering systems focused only on high resource languages like English. The Indian languages like Hindi and Tamil are underrepresented in public datasets and need a concentrated effort to develop suitable quality datasets and models for them. The multilingual-BERT(m-BERT) (Devlin et al., 2018) paved the way for the state-of-the-art models in various languages including Hindi and Tamil.

In recent years, we have seen that pre-training has improved performance in all spheres of NLP. (Eisenschlos et al., 2020) has shown that pre-training with synthetic/template based machine-generated data can improve understanding of semi-structured tables in NLI tasks, (Devlin et al., 2019) has shown how pre-training on a large corpus can help a model work better in general for all tasks. Since the introduction of BERT it has become a general practice to try and train larger models for more time to get better accuracy, and this practice seems to work great as seen in the transition from BERT to RoBERTa.

The chaii dataset contains 1,114 training examples which is very less for BERT-classs models hence the most sensible direction for us to go is find other Hindi data for pre-training. Having a dataset similar to ours is necessary for pre-training to work, since unlike English, Hindi/Tamil does not have a lot of training data to begin with. We are trying to pre-train our models using these 3 methods: (a) freeze the attention layers and train only the classification layers. (b) train the whole model. (c) Masked-LM pre-training which is a little power-hungry but if we stick with BERT and mBERT we can possibly improve the results with minimal training time. For the MLM pre-training we found some wikipedia text in Hindi and Tamil which we can use.

2 Method

Freezing We use the Transformer model as feature extractors. Similar to (Jain et al., 2020) we freeze the weights of the language model and use its output as feature-based embeddings to train different layers on top. We refer to the language model as base model and the trainable layers that we add on top as the head model. We will try to experiment with the following a few head types like Linear, Bi-linear etc.

Fine tuning In this setting, the entire language model is trained on our dataset.

Masked-LM pre-training In this method, we simply mask some percentage of the input tokens at random, and then predict those masked tokens.

Hence this method is called "masked" LM (MLM). Similar to (Devlin et al., 2019) the final hidden vectors corresponding to the mask tokens are fed into an output softmax over the vocabulary.

3 Datasets

We are using models pre-trained on mergedQUAD and xQUAD followed by fine-tuning. The methods used for training are mentioned in Section 2. The models gives as an output the starting point of the answer and ending point which are both integers, and the input is the tokenized sentence merged with the question.

Dataset	Train	Test
chaii	746	3
xQUAD	1190	-
mergedQUAD	6024	591

Table 1: This contains the train-test split of the Hindi datasets we have planned to use for pre-training or fine-tuning

Dataset	Train	Test
chaii	368	2

Table 2: This contains the train-test split of the Tamil datasets we have planned to use for pre-training or fine-tuning

4 Evaluation Metric

The evaluation metric used for this project is the word-level Jaccard score. In Jaccard similarity calculation, we will perform lemmatization to find the root words of the compared texts. Then Jaccard score is calculated as the number of common root words between the compared texts divided by the total number of unique root words.

$$jaccard(o_i, p_i) = \frac{count(O_i \cap P_i)}{count(O_i \cup P_i)} \quad (1)$$

where, O_i, P_i are the collections of root words in o_i, p_i respectively. The formula for overall metric is

$$score = \frac{1}{n}\sum_{i=1}^{n} jaccard(o_i, p_i) \quad (2)$$

5 Experiment and Results

mBERT was first introduced in (Devlin et al., 2019) where they trained BERT on a multilingual dataset (xQUAD). This has around 334M parameters. We first tested on the mBERT and obtained around 0.20 score (by our evaluation metric mentioned earlier) with fine-tuning[1] on chaii. To see how well the pre-training on xquad generalizes, we checked the accuracy without fine-tuning and it was very poor which made us infer that pre-training on xquad does not generalize well.

Model	No-finetuning	fine-tune on chaii
mBERT	0.0373	0.207
XLM-Roberta	0.2720	0.5837
indic-BERT	0.0016	0.0288

Table 3: This contains the accuracy score for mBERT, XLM-RoBERTa and Indic-BERT models tested using chaii test dataset.

ALBERT was introduced in (Lan et al., 2020). There main motivation was to try and reduce the number of trainable parameters, and hence ALBERT has around 12M parameters which reduce the training time. We used models from (Kakwani et al., 2020) which has been pretrained on mergedQUAD and did a test with and without fine-tuning. The results were pretty bad so we decided to not look any further into less power hungry models and moved to roberta-based models.

XLM-Roberta was introduced in (Conneau et al., 2020). It has around 550M parameters for the 'large' model. We used a model pre-trained with SQUAD2 dataset. Even without fine-tuning[2] we obtained a score of 0.27 which is higher than mBert and with fine-tuning we obtained an score of 0.58.

6 Conclusion

We started with the aim of achieving the best possible accuracy with the least power hungry model, but the results showed us that multilingual question answering works better of there are more parameters.

After ALBERT we had no doubt in our minds that the only way to move forward is towards more parameters. Further improvements could have been possible if more data was available since most of power hungry models will end up over-fitting on the small training set of chaii, hence running more

[1]Here the embedding layers are not frozen, we have tried both with and without freezing, but we got a lot better results without freezing and hence we have only kept the results without freezing

[2]without freezing the embedding layer

than 15-20 epochs reduced the score in the test set.

All codes for training these three models have been uploaded on github with proper instruction of how to run them. [3]

References

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *ACL*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Julian Martin Eisenschlos, Syrine Krichene, and Thomas Müller. 2020. Understanding tables with intermediate pre-training. In *FINDINGS*.

Kushal Jain, Adwait Deshpande, Kumar Shridhar, Felix Laumann, and Ayushman Dash. 2020. Indic-transformers: An analysis of transformer language models for indian languages. *ArXiv*, abs/2011.02323.

Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, Gokul N.C., Avik Bhattacharyya, Mitesh M. Khapra, and Pratyush Kumar. 2020. inlpsuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for indian languages. In *FINDINGS*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. *ArXiv*, abs/1909.11942.

[3]`https://github.com/Dibyakanti/IndicQA_CS565_course_project`