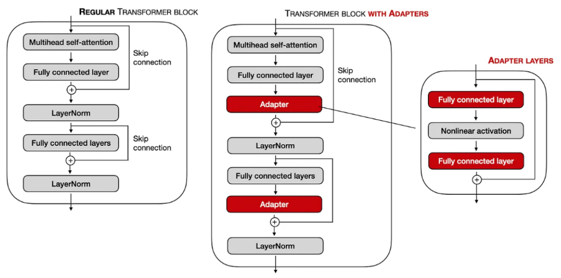
Multilingual large language models for everything

Jayveersinh Raj, j.raj@innoplis.university
Evgenii Evlampev, e.evlampev@innopolis.university

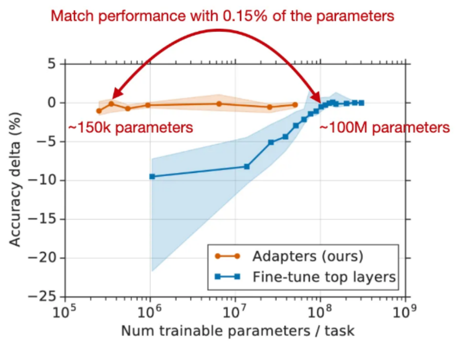
1. Introduction

Prompting adds some task-specific input text to assist direct language model behavior. The process of just training and updating freshly added prompt tokens to a pretrained model is known as prompt tuning. Instead of completely fine-tuning a distinct model, you can use a single pretrained model with frozen weights and train and update a smaller number of prompt parameters for each downstream task. Prompt adjustment can be more effective as models get bigger, and outcomes improve even more when model parameters increase [1]. Additionally by injecting trainable rank decomposition matrices into each layer of the Transformer architecture and freezing the pre-trained model weights, low-Rank Adaptation (LoRA) or Quantized Low-Rank Adaptation (QLoRA) significantly reduce the number of trainable parameters for downstream tasks [2]. The figure below is an illustration of the adapter layers from [3].



Comparison of the regular transformer blocks used in various LLMs and a transformer block modified via the adapter layers.

Moreover, these light weight adapters perform similar to an entire fine tuned model as described in the annotated figure below from [3].



Annotated figure from the adapter paper, https://arxiv.org/abs/1902.00751.

In this project we decide to showcase the use of such model **Bloom-3b** for 2 tasks:

1. Sentence correction
2. Question answering

2. Background on the task choices

2.1. *Sentence correction*

2.1.1. *Background*

Sentence correction is the task of correcting a misspelled word, the use of a word in the wrong context, or any mistake in the sentence. Ideally, the language model, because of its large knowledge base in multiple languages, should be able to do it for multiple languages. We trained on the Gujarati dataset, which was less than 1 % of the data, while the large language model was trained [6]. Yet, it should be able to correct the mistakes more proficiently for languages for which the data is larger.

2.1.2. *Use cases*

The solution has vast use cases; some of them are mentioned below:

- **Transcription model for Acoustic models**: Acoustic models are used for speech recognition, and due to the many similar words in the sound, the text produced has to be corrected by a sentence correction model. For example, the word **read** in its past tense and past particle are written the same (**read**), but are pronounced as **red**.
- **For search engines and grammar correction tools**: In search engines to correct a query, such models can be used; moreover, the very obvious use case is using them as a grammar correction tool.

2.2. *Question answering*

2.3. *Background*

A question-answering model is one that takes user queries and replies in a human way. All the models that are designed for question answering have to be prompt tuned if the model architecture is decoder type only. Otherwise, it has to be fine-tuned in a sequence-to-sequence way if the architecture contains both an encoder and a decoder [4].

2.3.1. *Use cases*

The use case of a question answering model is trivial, to generate responses based on the user input.

3. Data description

The HuggingFace datasets library was used to load the data that was prepared from raw data.

3.1. Sentence correction

Ideally, for the baseline, a dictionary-based approach was taken into account, which was formed by web scraping and cleaning. However, a corpus for the language is provided by [5]. We artificially generated the mistakes in the sentences with 20 % chance of having a sentence with mistakes and ended up with a dataset of size 1 million. A CSV file was created with corresponding correct and incorrect sentences. However, only 1000 samples were used for training.

3.2. Question answering

A reading comprehension dataset called the Stanford Question Answering Dataset (SQuAD) is made up of questions crowdsourced from a collection of Wikipedia articles. Each question's response is either a passage of text or a span from the relevant reading passage, or it may be left unanswered. SQuAD2.0 combines the 100,000 questions from SQuAD1.1 with more than 50,000 adversarially crafted, seemingly answered questions that crowdworkers have contributed. In order for a system to perform effectively on SQuAD2.0, it must be able to distinguish between situations in which the paragraph does not support an answer and when it does and refrain from responding. SQuAD2.0 was used for the model training. The entire dataset was used.

4. Model description

4.1. Architecture

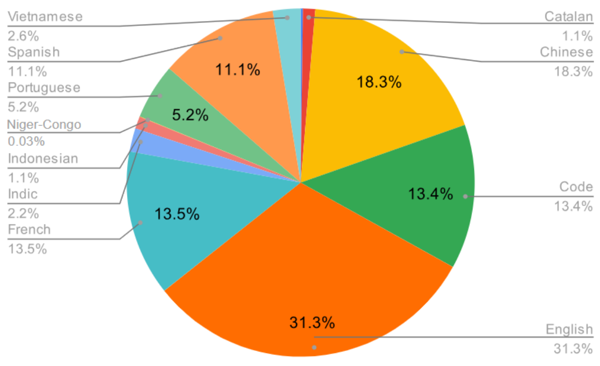
Bloom-3b which is a decoder-style Transformer-based language model developed by BigScience was used for prompt tuning. Due to computational limitations, 3 billion (8-bit quantize) is possibly the largest model that could be loaded on a free Google Colab T4 GPU (15GB). According to [4] by using low-precision data types, such as 8-bit integers (int8) rather than the more common 32-bit floating point data types (float32), to represent the weights and activations, quantization is a technique that can lower the computational and memory costs associated with executing inference.

In theory, a model with fewer bits uses less energy and memory storage, and operations like matrix multiplication can be completed considerably more quickly using integer arithmetic. Additionally, it enables the use of models on embedded systems, which occasionally can only handle integer data types. Nevertheless, the choice of such a model to bloom is because of its nature of open source, and multilingual capabilities. Below are the technical details of the model:

- **Architecture**: Modified from Megatron-LM GPT2 (decoder only architecture)
- **Parameters**: 3,002,557,440
- **Sequence length**: 2048 tokens (model can see up to previous 2048 tokens for next token generation)
- **Objective Function**: Cross Entropy with mean reduction
- **vocabulary size**: 250,680
- **tokenization algorithm**: byte-level Byte Pair Encoding (BPE)

4.2. Languages, and their data share

The pie chart shows the distribution of languages in the training data [6].



5. Training Process

By freezing most of the pretrained large language model's (LLM's) parameters and just fine-tuning a select few additional model parameters, Parameter efficient fine tunining (PEFT) techniques significantly reduce computing and storage expenses [7]. This also resolves the problem of catastrophic forgetting, which is a behavior that arises when large language models (LLMs) are fully fine-tuned [7]. Additionally, PEFT techniques have been demonstrated to generalize better to out-of-domain circumstances and outperform fine-tuning in low-data regimes [7]. In this solution, the parameter efficient fine tuning (PEFT) method used is low rank adaptation (LoRA). LoRA's method involves using low-rank decomposition to express the weight updates with two smaller matrices, or update matrices, in order to increase the efficiency of fine-tuning [8]. It is possible to train these new matrices to adjust to the new data while minimizing the total number of modifications [8]. There are no more changes made to the initial weight matrix; it stays fixed. Both the original and modified weights are summed to get the final values [8]. Below are the LoRA configuration details:

- Rank (r): 8
- LoRA alpha: 16
- target modules to apply the LoRA update matrices: **query_key_value**
- LoRA dropout: 0.05

In the configuration above,

- Rank (r): The rank of the update matrices, expressed in int. Lower rank results in smaller update matrices with fewer trainable parameters.
- LoRA alpha: LoRA scaling factor.
- LoRA dropout: dropout probability of the LoRA layers
- **query_key_value** from self attention was set as the target module to train which has the following attributes:
    - Linear layers(8 bit because of quantization)
    - input features: 2560
    - output featues: 7680
    - bias: True

Finally, after the above configurations, the parameters for training the model are as follows:

- trainable params: 2,457,600
- all params: 3,005,015,040
- trainable %: 0.0818

Additionally the important training arguments used for the trainer are as follows:

5.1. Sentence correction

- warm up steps: 100
- Total epochs: 1 (62 steps)
- learning rate: 1e-3
- fp16 precision: True

5.2. Question answering

- warm up steps: 100
- Max steps: 100 (0.01 epochs)
- learning rate: 1e-3
- fp16 precision: True

6. Evaluation

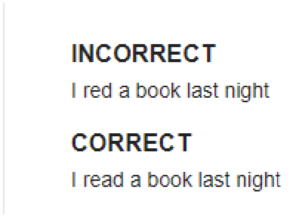
For evaluation of the model due to hardware constraints we could only test for the sentence correction model with 100 samples from **jfleg** dataset [9] whose numbers are as follow:

- Average ROUGE-1: 0.792
- Average ROUGE-2: 0.687
- Average ROUGE-L: 0.782
- BERT embedding similarity: 0.88

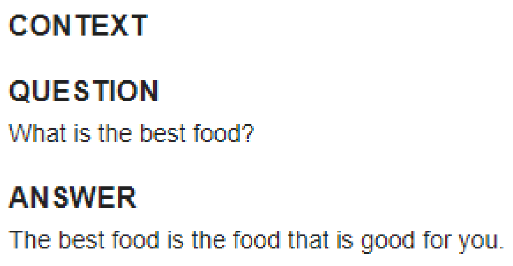
7. Results

The above evaluation results show that the model performs significantly despite being trained on a language which has less than 1 % of training data in the base model. Unfortunately, due aforementioned hardware constrains the question answering model was difficult to be evaluated, but below is an example output for both of the models.

7.1. Sentence correction


INCORRECT
I red a book last night
CORRECT
I read a book last night

7.2. Question answering


CONTEXT
QUESTION
What is the best food?
ANSWER
The best food is the food that is good for you.

8. Timeline

The project roughly was a round a month with a week or two extra. Mainly, some weeks were spent on deciding the data, a week for training and evaluation, and approximately 2–3 weeks for the research.

9. Contributions

Jayveersinh Raj: The entire work on the dataset, exploration, and research on Gujarati language, and sentence correction.
Evgenii Evlampev: The work on question answering and research Contributions in evaluation for the English dataset.

10. Conclusion

The project marks the vast potential of a large language pre-trained model fine-tuned efficiently on a few parameters with quantization. Moreover, the ability to have lightweight, streamlined adapter models for dedicated tasks gives room for a base model to perform several downstream tasks by swapping the light-weight adapters. Moreover, the model performs tasks in multiple languages, hence, marking cross lingual capabilities.

11. Future directions

It can be seen that prompt tuning a language model yields better results. Because of the huge knowledge base of the language, large language models can be prompt tuned to achieve tasks with high performance. However, due to computational limitations the model was 8-bit quantized, a model with higher parameters, and more precision would yield better results. According to [7] the potential of the method used is vast, the pretrained large language model can be topped with the tiny trained weights from parameter efficient fine tuning (PEFT) techniques. Thus, by adding tiny weights, the same large language model can be used for numerous tasks without requiring a complete model replacement.

12. Links for project and checkpoints

12.1. ***Project***

https://github.com/Jayveersinh-Raj/cross_lingual_LLM_for_everything

12.2. ***Sentence correction model***

https://huggingface.co/Jayveersinh-Raj/bloom-sentence-correction

12.3. ***Question Answering model***

https://huggingface.co/Jayveersinh-Raj/bloom-que-ans

12.4. ***Sentence completion dataset***

https://huggingface.co/datasets/Jayveersinh-Raj/Gujarati-correct-incorrect-sent

12.5. ***Question Answering dataset***

https://huggingface.co/datasets/squad_v2

13. References links

1. https://huggingface.co/docs/peft/task_guides/clm-prompt-tuning
2. https://towardsdatascience.com/leveraging-qlora-for-fine-tuning-of-task-fine-tuned-models-without-catastrophic-forgetting-d9bcd594cff4#:~:text=Approaches%20Low%2DRank%20Adaptation %20(LoRA,trainable%20parameters%20for%20downstream%20tasks.
3. https://magazine.sebastianraschka.com/p/finetuning-llms-with-adapters
4. https://aclanthology.org/2023.findings-acl.27.pdf
5. https://huggingface.co/ai4bharat/indic-bert
6. https://bigscience.huggingface.co/blog/building-a-tb-scale-multilingual-dataset-for-language-modeling
7. https://huggingface.co/blog/peft
8. https://huggingface.co/docs/peft/conceptual_guides/lora
9. https://aclanthology.org/E17-2037/