Neural Networks: Theory and Implementation
Term Project: Language modeling

Kai Wittenmayer
kawi00002@uni-saarland.de
7005859

Dhimitrios Duka
dhdu00001@uni-saarland.de
7059153

Abstract

Constructing a multilingual large language model (LLM) that performs equally well across different languages is a challenging task. One way of mitigating this issue is through the process of fine-tuning. In this report, we analyze the effect of various fine-tuning methods on language modeling and their hidden representations. We found out that the full fine-tuning resulted in the best performance, but PEFT methods like BitFit, LoRA, and IA^3 also showed promising results with only a slightly higher loss while enabling only up to 0.28% of the total model parameters.

1 Introduction

Due to the abundance of training data available in the English language, most models tend to perform exceptionally well in English but suffer from significant drops in performance when it comes to other languages, particularly the less common ones. In this report, we analyze the performance of a pre-trained LMM and explore fine-tuning methods to address performance issues in the quy_Latn language.

2 Running inference on a pre-trained LM

In this first language modeling task, we ran inference on two pre-trained models, XGLM-564M (Lin et al., 2021) and GPT-2 (Radford et al., 2019), and analyzed their performance on the Flores (NLLB Team, 2022) dataset.

As illustrated in Figure 1, the XGLM-564M model exhibits a more stable performance compared to GPT-2. However, the GPT-2 model outperforms it in all languages except on ita_Latn. A possible explanation for GPT-2's superior performance could be attributed to the difference in the tokenization process. To investigate this difference, we decided to analyze it in the context of the Albanian language. Both tokenizers are based on the Byte-Pair Encoding (Sennrich et al., 2015) scheme. However, they handle the tokenization process differently. These differences are summarized in Table 1. Taking a closer look at the output of the tokenizers, we concluded that at least for the Albanian language, each word is broken down into individual tokens. For example, the Albanian word "aeroportit" (airport in English), is broken down into ["Ġaer", "op", "ort", "it"] from the GPT-2's tokenizer. The same word is broken down into ["_aeroport", "it"] from the XGLM-564M's tokenizer. This hints that the vocabulary size of XGLM-564M's tokenizer is bigger than the GPT-2's as depicted in Table 1. This also explains the better performance of GPT-2. Breaking down a word into smaller/more tokens means that the model can better predict the next token because of the wider context. In the case of the aforementioned token, XGLM-564M needs to predict the entire token "_aeroport" at once while GPT-2 first predicts "aer", then "op", and then finally "ort".

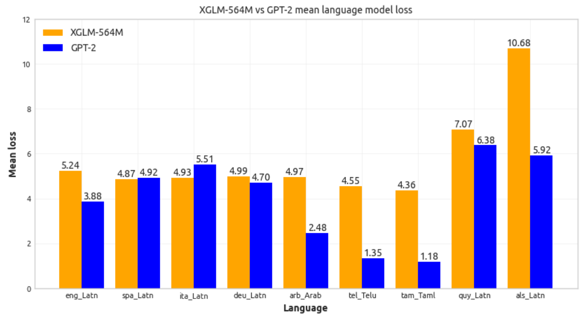


Figure 1: Language modeling loss of XGLM-564M and GPT-2.

Tokenizer	SOS	SOW	Vocab. size
GPT-2	-	Ġ	50,257
XGLM-564M	<s>	_	256,008

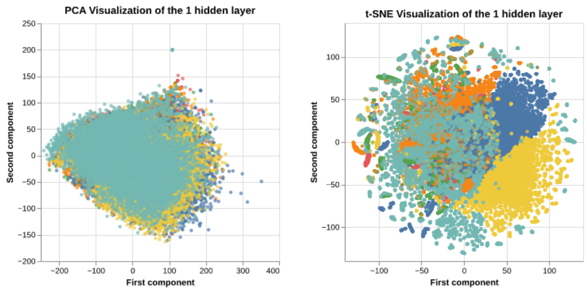
Table 1: A summarization of the tokenizer's properties. The "-" character indicates an absence of that property, SOS indicated the start of sentence token and SOW indicated the start of word token.

3 Exploring multi-lingual representation spaces

For the second task, we randomly sampled 200 sentences per language. There were two approaches to this: either randomly sampling from one dataset and then using the same indices to sample from the other datasets, or randomly sampling each dataset independently. We thought that both methods would result in similar results, so we chose to go with the latter approach. Using the sampled sentences, we run XGLM-564M in inference mode and stored the hidden representation of each token, excluding the padding tokens, for each hidden layer, for each language in hdf5 files. We came up with a proprietary hierarchical template to store the hidden representations. The template is comprised of two main elements, a list of the sampled sentences and a dictionary of all the layers including the embedding layer, 25 in total. Each entry of the layer dictionary has an entry for each of the sentences that appear in the sentence list. Each sentence entry has a state, which is the mean-pooled hidden representation state across all tokens of that sentence at that particular layer, and a dictionary holding information of a token such as the state, encoded ID, and the string representation of it. Formatting the data according to this template enables support for duplicate tokens and storing the sentences only once per file, reducing the required storage to a minimum. The hidden representations lie in a 1024 dimensional space. To visualize the data, we had to project it down to two dimensions using either PCA (Maćkiewicz and Ratajczak, 1993) or t-SNE (Van der Maaten and Hinton, 2008). Using PCA, we could straight away visualize the hidden representations for both tokens and sentences. For t-SNE, we saw the best performance using a learning rate of $n/12$ (Belkina et al., 2019), where n is the number of points in the dataset (Belkina et al., 2019) and a perplexity of 30 (Gove et al., 2022).

We argue that tokens from languages that are similar or related tend to cluster closer together. Given this clustered nature of the representations, the t-SNE algorithm can more effectively define the boundaries between each language compared to PCA. This is because t-SNE preserves the local structure of the space (Abdullah et al., 2020). This can be seen in Figure 2.

Furthermore, it is noteworthy to observe this behavior not only on the first hidden layer but across all layers. The more we progress through the layers,

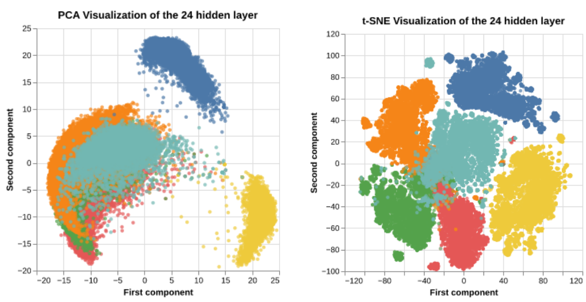


(a) PCA (b) t-SNE

Figure 2: Two-dimensional representations of the token hidden state for the first hidden layer: (a) PCA and (b) t-SNE. The color of the points represents one of the studied languages. (arb_Arab, blue), (deu_Latn, orange), (eng_Latn, red), (quy_Latn, teal), (spa_Latn, green), (tam_Taml, yellow).

the more the model is able to better define boundaries between different languages and cluster the ones that are similar to each other as seen in Figure 3. This is true not only for the token representation but also for the sentence representation as depicted in Figure **??**. This evolutionary process puts the learning process into perspective.

As expected, the Latin languages tend to cluster closely together within the representation space. This clustering reflects the linguistic affinity and common structural features of the Latin-based languages. On the other hand, the non-Latin languages show greater degree of separation, which indicates the distinct linguistic characteristics of these languages.



(a) PCA (b) t-SNE

Figure 3: Two-dimensional representations of the token hidden state for the last hidden layer: (a) PCA and (b) t-SNE.

4 Language model adaptation

4.1 Dataset acquisition and pre-processing

For the fine-tuning dataset, we evaluated six distinct datasets: OSCAR (Ortiz Su'arez et al., 2020), NLLB (Schwenk et al., 2021), CC100 (Conneau et al., 2020), Llamacha/monolingual-quechua-iic (Zevallos et al., 2022), wikimedia/wikipedia (Foundation) and hackathon-pln-es/spanish-to-quechua (Vilchez). We did discard OSCAR due to being too short and discarded CC100, LLamacha/monolingual-quy_Latn-iic, and wikimedia/wikipedia because of quality concerns. However, NLLB and spanish-to-quy_Latn showed good quality, despite an overly religious theme. We employed length-based filtering (Costa-jussà et al., 2022) to identify sentences that are not within a minimum and maximum length. We derived the interval from the Flores dataset quy_Latn sentences. Applying the pre-processing to our datasets, we reduced the spanish-to-quy_Latn size by 15.9% and the NLLB size by 21.3%.

4.2 Model adaptation for full Fine-Tuning

For the full fine-tuning process, we enabled gradient updates for all parameters throughout the architecture. We trained the model on a step-based approach on both datasets and kept track of the loss on all of the languages.

4.3 Model adaptation for BitFit

For BitFit-ing (Zaken et al., 2022), we only enabled gradient updates for the biases and the task-specific prediction head.

4.4 Model adaptation for LoRA

For the LoRA (Hu et al., 2021; Lor) implementation, we followed the approach of (Lor) and implemented a modified version of the XGLMAttention in which two new matrices, A and B, where introduced. We then substituted the XGLMAttention with our modified version. We implemented two versions of LoRA. In one version, we modified only the query and value projection, while in the other, we also modified the key and output projection. Before applying LoRA, all weights were frozen except for the introduced A and B matrices. Additionally, we assessed how enabling gradient updates of the task-specific prediction head would influence the performance.

4.5 Model adaptation for $(IA)^3$

We did follow the same approach for $(IA)^3$, this time incorporating three vectors per attention block: one for the key, one for the value, and one for the feed-forward layer. During the forward pass, we performed point-wise multiplication with the output vectors of the respective projections, initializing them to all ones to ensure consistent output at the beginning of fine-tuning. We inserted the l_{ff} after the activation function of the first feed-forward and before the second feed-forward for each attention block. Gradient updates were only enabled for the introduced vectors. Furthermore, we evaluated the impact of enabling gradient updates to the task-specific prediction head on the performance of this method.

5 Results

We trained the model on both datasets using a subset of 80.000 samples. The performance we got from the NLLB dataset consistently surpassed the performance of the spanish-to-quechua dataset. To provide some perspective the language modeling loss after full fine-tuning was 5.3 for spanish-to-quechua and 4.9 for NLLB. The discrepancy is approximately the same for the other methods. Therefore, the results presented below are based on the NLLB dataset trained model.

Furthermore, we used the same training hyperparameters for all of the methods, in order to get a fair comparison between them. It is important to note that some methods could perform better when trained with different hyperparameters. We used an AdamW Optimizer(Loshchilov and Hutter, 2017) with a learning rate of 5e-5. We implemented early stopping on the quy_Latn dev split as the validation set. We also made sure not to plot the loss of the quy_Latn devtest split during fine-tuning, so that we do not do any cherry-picking whatsoever.

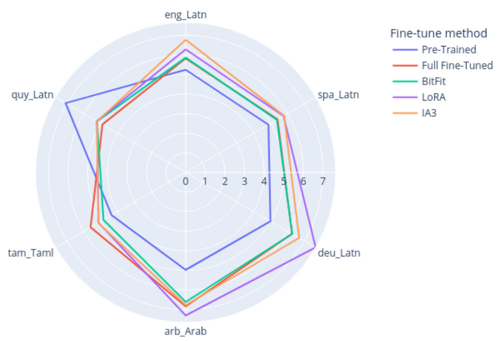
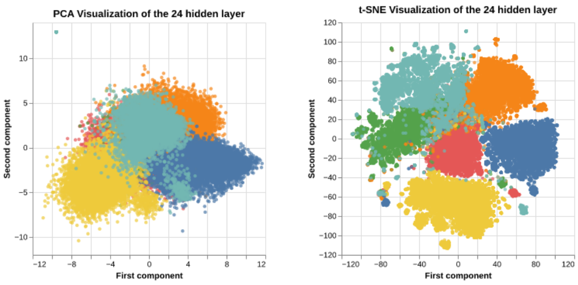


Figure 4: Language modeling loss of the pre-trained model and PEFT methods being applied to it.

As shown in Figure 4, applying the full fine-tuning method to the pre-trained model resulted in a decrease of the loss for quy_Latn from 7.04 to 4.90, a 30.4% improvement. However, all other losses exhibited an increase, with the most significant rise observed in the arb_Arab loss. This outcome was anticipated, considering that the model was fine-tuned within a Latin language context, while arb_Arab is a non-Latin language. This observation is further supported by the hidden state visualization of the full fine-tuned model shown in Figure 5. At first, we were expecting that the plot of the hidden representations after the fine-tuning would result in the quy_Latn language being more separated from the Latin languages, but as we found out from the plots, this is not the case. The quy_Latn language remains close to the Latin languages but it is further away from the non-Latin ones. This clearly reflects the linguistic affinity and common structural features of the quy_Latn language with the Latin-based languages and, it highlights the differences between quy_Latn and non-Latin languages.



(a) PCA (b) t-SNE

Figure 5: Two-dimensional representations of the token hidden state for the last hidden layer: (a) PCA and (b) t-SNE. (Figures best viewed in color.)

The language modeling loss of BitFit on quy_Latn is 5.22, which is slightly higher than the full fine-tuning. However, this was anticipated because applying BitFit-ing lowers the capacity of the model to 0.048% of the original one as we are only training the biases.

The language modeling loss of LoRA for quy_Latn is 5.23, which is approximately the same as BitFit's performance. However, LoRA had significantly higher losses for all other languages except tam_Latn. But one of the advantages of LoRA is that the original weights remain the same after fine-tuning. This means that when using the fine-tuned model for languages other than quy_Latn, one can remove the introduced A and B matrices and essentially have the pre-trained model again. Additionally one can compute A and B matrices for several tasks and efficiently switch between them because only a small fraction of the weights has to be loaded. We also considered modifying the query, key, value, and output projection, instead of only query and value, but observed only a slight decrease in loss, which aligns with the findings in the original paper.

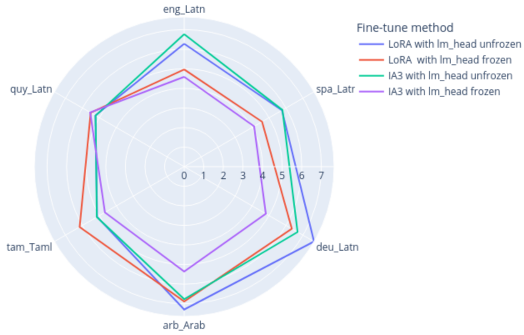


Figure 6: Impact on language modeling loss of freezing and unfreezing the lm_head of LoRA and IA^3.

The final language modeling loss that we considered was the loss of $(IA)^3$, which resulted in 5.22. This loss is on par with the other PEFT methods mentioned before. In the original paper the authors showed comparable performance of $(IA)^3$ to full fine-tuning, however, we did experience a slightly higher test loss. Furthermore, the same advantage of being lightweight as for LoRA does also apply to $(IA)^3$.

We further investigated how enabling or disabling gradient updates on the lm_head influences the language modeling loss after fine-tuning. Figure 6 neatly shows the trade-off between these two approaches. Enabling the lm_head produces a lower language modeling test loss on quy_Latn than the approach of not considering the lm_head. But this is at the expense of a higher language modeling test loss on other languages. Note that Figure 4 depicts the performance of LoRa and IA^3 with the unfrozen lm_head.

Overall, we found out that the full fine-tuning resulted in the best performance, but BitFit, LoRA, and IA^3 had only slightly higher losses while only enabling gradients up to 0.28% of the total model parameters.

References

Implementing lora from scratch. https://towardsdatascience.com/implementing-lora-from-scratch-20f838b046f1. Accessed: 2024-03-12.

Badr M. Abdullah, Jacek Kudera, Tania Avgustinova, Bernd Möbius, and Dietrich Klakow. 2020. Rediscovering the Slavic continuum in representations emerging from neural models of spoken language identification. In *Proceedings of the 7th Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 128–139, Barcelona, Spain (Online). International Committee on Computational Linguistics (ICCL).

Anna C Belkina, Christopher O Ciccolella, Rina Anno, Richard Halpert, Josef Spidlen, and Jennifer E Snyder-Cappione. 2019. Automated optimized parameters for t-distributed stochastic neighbor embedding improve visualization and analysis of large datasets. *Nature communications*, 10(1):5415.

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 *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Marta R Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, et al. 2022. No language left behind: Scaling human-centered machine translation. *arXiv preprint arXiv:2207.04672*.

Wikimedia Foundation. Wikimedia downloads.

Robert Gove, Lucas Cadalzo, Nicholas Leiby, Jedediah M Singer, and Alexander Zaitzeff. 2022. New guidance for using t-sne: Alternative defaults, hyperparameter selection automation, and comparative evaluation. *Visual Informatics*, 6(2):87–97.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, Tianlu Wang, Shuohui Chen, Daniel Simig, Myle Ott, Naman Goyal, Shruti Bhosale, Jingfei Du, et al. 2021. Few-shot learning with multilingual language models. *arXiv preprint arXiv:2112.10668*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Andrzej Maćkiewicz and Waldemar Ratajczak. 1993. Principal components analysis (pca). *Computers & Geosciences*, 19(3):303–342.

James Cross Onur Çelebi Maha Elbayad Kenneth Heafield Kevin Heffernan Elahe Kalbassi Janice Lam Daniel Licht Jean Maillard Anna Sun Skyler Wang Guillaume Wenzek Al Youngblood Bapi Akula Loic Barrault Gabriel Mejia Gonzalez Prangthip Hansanti John Hoffman Semarley Jarrett Kaushik Ram Sadagopan Dirk Rowe Shannon Spruit Chau Tran Pierre Andrews Necip Fazil Ayan Shruti Bhosale Sergey Edunov Angela Fan Cynthia Gao Vedanuj Goswami Francisco Guzmán Philipp Koehn Alexandre Mourachko Christophe Ropers Safiyyah Saleem Holger Schwenk Jeff Wang NLLB Team, Marta R. Costa-jussà. 2022. No language left behind: Scaling human-centered machine translation.

Pedro Javier Ortiz Su'arez, Laurent Romary, and Benoit Sagot. 2020. A monolingual approach to contextualized word embeddings for mid-resource languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1703–1714, Online. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Holger Schwenk, Guillaume Wenzek, Sergey Edunov, Edouard Grave, Armand Joulin, and Angela Fan. 2021. CCMatrix: Mining billions of high-quality parallel sentences on the web. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6490–6500, Online. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *CoRR*, abs/1508.07909.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Sara Benel Jose Vilchez. spanish-to-quechua.

Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. 2022. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models.

Rodolfo Zevallos, John Ortega, William Chen, Richard Castro, Nuria Bel, Cesar Toshio, Renzo Venturas, Hilario Aradiel, and Nelsi Melgarejo. 2022. Introducing qubert: A large monolingual corpus and bert model for southern quechua. In *Proceedings of the Third Workshop on Deep Learning for Low-Resource Natural Language Processing*, pages 1–13.