TALL: A Research Proposal for Enhancing LLM Performance in Low-Resource Languages

Moshe Ofer

August 25, 2024

Abstract

Large Language Models (LLMs) have made significant strides in natural language processing (NLP). Yet, their effectiveness in low-resource languages remains limited due to challenges such as insufficient annotated data and complex linguistic features. This research proposes *TALL* (Trainable Architecture for Enhancing LLM Performance), a novel approach designed to improve LLM performance in low-resource languages by leveraging dual translation and custom transformers. By transforming the linguistic representations between Hebrew and English through custom, trainable transformers, TALL enables the model to leverage the strengths of high-resource language processing while continually adapting to capture the nuanced translations needed for accurate low-resource language output. By doing so, we aim to reduce the perplexity—a measure of uncertainty—in the LLM's performance on Hebrew tasks. The approach is designed to be adaptable, making it potentially useful for other low-resource languages as well.

1 Introduction

Large Language Models (LLMs) have significantly advanced the field of natural language processing (NLP), particularly for high-resource languages like English. However, their performance in low-resource languages, such as Hebrew, remains limited due to challenges like sparse data and linguistic complexity. These challenges often result in higher perplexity, reflecting greater uncertainty in the model's predictions and reduced accuracy.

Low-resource languages, which lack extensive annotated datasets and robust pre-trained models, present unique hurdles for LLMs. Hebrew, with its complex morphology and syntactic structure, exemplifies these challenges. Existing methods, such as transfer learning and data augmentation, have attempted to bridge the gap, but they often fall short of fully leveraging the potential of LLMs.

This proposal addresses this gap by introducing *TALL* (Trainable Architecture for Enhancing LLM Performance), a novel architecture designed to enhance LLM performance in low-resource languages.

The primary objective of this research is to demonstrate how *TALL* can effectively improve LLM performance in Hebrew, with the potential for broader application to other low-resource languages. The remainder of this proposal is structured as follows: Section 2 reviews related work, Section 3 details the proposed methodology, Section 4 discusses the proposed experiments.

2 Related Work

Several approaches have been explored to Improve the performance of Large Language Models (LLMs) in low-resource languages, ranging from fine-tuning techniques to in-context learning strategies.

One prominent approach is few-shot in-context learning (ICL), which allows LLMs to perform tasks in low-resource languages by utilizing a few examples provided in the input context. Recent work has shown that cross-lingual in-context learning (X-ICL)

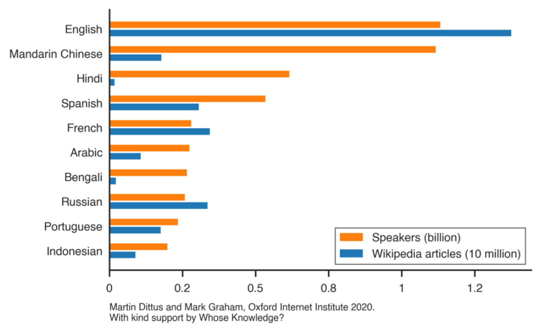

Figure 1: A comparison of the number of speakers (in billions) and the number of Wikipedia articles (in tens of millions) for various languages. This figure highlights the significant disparity between the number of native speakers and the amount of digital content available in different languages. This imbalance underscores the challenges faced by low-resource languages in the digital space and emphasizes the need for advanced architectures like *TALL*. *TALL* leverages high-resource languages to enhance the performance of language models in low-resource settings, addressing the critical gap in NLP resources for these underrepresented languages.

can leverage examples from high-resource languages to improve performance in low-resource languages. This method demonstrates that while ICL can be effective, its success is closely tied to the quality of alignment between the languages involved [1].

Another approach involves fine-tuning multilingual models specifically for low-resource languages, as explored in works like adaptMLLM. This method tailors the model to the target language pair, offering a more direct adaptation compared to general in-context learning strategies [2].

A recent approach introduced in PolyLM [3] explores the use of curriculum learning to enhance the multilingual capabilities of LLMs. This method involves a two-stage training process where the model first learns general language patterns from a large dataset dominated by high-resource languages, and then fine-tunes on a curated subset with an increased proportion of low-resource languages. This strategy facilitates knowledge transfer from high-resource languages to low-resource ones, resulting in improved performance across a wide range of multilingual tasks.

These studies highlight the diversity of strategies being developed to enhance LLM performance in low-resource settings, whether through in-context learning, fine-tuning, or cross-lingual methods. The *TALL* architecture proposed in this research builds on these approaches by leveraging high-resource language capabilities through a dual translation process, aiming to refine the cross-linguistic processing abilities of LLMs and further improve their performance in low-resource languages.

3 Proposed Methodology

3.1 Overview

The *TALL* architecture will be composed of a sequence of 5 components. The following sections describe each of these components in detail and illustrate how they will interact with each other.

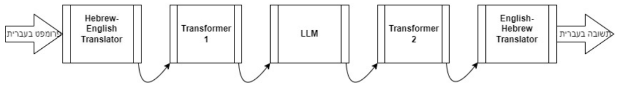

Figure 2: Overview of the *TALL* architecture, showing the dual translation process and the integration of custom transformers with the LLM.

3.2 Dual Translation and Hidden State Injection

The first step in the *TALL* architecture involves translating the input from the low-resource language (e.g., Hebrew) into a high-resource language (e.g., English) using a pre-trained translator. After this translation, the last hidden states from the translator will be fed into the first custom transformer, which will process them and generate output vectors. These vectors will then be "injected" into the LLM as its initial hidden states. This injection will essentially bypass the input embedding layer, of the LLM.

After the LLM generates an output in English, the process will continue by passing the final layer of the LLM's hidden states into the second custom transformer. This transformer will process the LLM's output and convert it into the correct format needed for the first layer of the second pre-trained translator. This transformed input will then be "injected" into the second translator which will generate the final desired output in Hebrew.

This dual translation mechanism will hopefully allow the LLM to "think" in English while generating responses in Hebrew. Moreover, the entire *TALL* architecture will be fully trainable, allowing the custom transformers to adapt to the unique linguistic characteristics of the low-resource language, thereby enhancing overall performance. As the system is exposed to more data, it can refine the transformation of hidden states, optimize the translation processes, and, hopefully, capture the subtle nuances of the language. This continuous learning capability is expected to enable *TALL* to generate more accurate and contextually appropriate outputs over time.

3.2.1 Transformer 1: Seq2Seq Model

The first custom transformer in the *TALL* architecture will be a sequence-to-sequence (seq2seq) model trained on Hebrew sentences. To train this transformer, a specialized dataset will be created using Hebrew sentences from Wikipedia. The process will involve extracting hidden states from the Hebrew-to-English translation and using these as input to the seq2seq model. The corresponding output states will be derived from the LLM processing the translated English sentences. This dataset will allow the transformer to learn effective mappings between the hidden states, capturing the nuances of Hebrew and preparing the data for subsequent processing by the LLM.

3.2.2 Transformer 2: Neural Network Model

The second custom transformer will be a simple neural network that processes the last hidden states generated by the LLM. Its role will be to predict the most appropriate vectors to feed into the second translator, ensuring that the generated output is coherent and contextually appropriate in the target language.

4 Proposed Experiments

To evaluate the effectiveness of the *TALL* architecture, we will conduct baseline comparisons using a standard Large Language Model (LLM) capable of handling Hebrew natively. The primary metric for comparison will be perplexity, which measures how well a model predicts a sample. Perplexity P is mathematically defined as:

$$\text{Perplexity}(W) = \exp\left(-\frac{1}{N}\sum_{i=1}^{N}\log P(w_i)\right)$$

where N is the number of words in the sample W, and $P(w_i)$ is the probability assigned by the model to the i-th word.

Intuitively, perplexity reflects the model's uncertainty in predicting the next word in a sequence; lower perplexity indicates that the model is more confident in its predictions and is generally performing better. For instance, if a model has a perplexity of 10, it means that, on average, the model is as uncertain as if it had to choose among 10 equally probable outcomes for each word in the sequence.

The baseline LLM will be tested on a set of Hebrew sentences, measuring its perplexity when generating responses directly in Hebrew. This performance will serve as the benchmark against which the *TALL*-enhanced LLM will be compared.

By experimenting with different training strategies, we will also evaluate whether it is more effective to train the first and second transformers separately or together. We will also explore the impact of different architectural choices, such as the size of the transformers and the training data used, to identify the most effective configurations for improving performance in low-resource languages.

5 Expected Outcomes

We hypothesize that the *TALL* architecture will significantly reduce perplexity in Hebrew language tasks compared to baseline models. We expect that the use of dual-translation and custom transformers will enable the LLM to better capture the nuances of the Hebrew language, resulting in more accurate and contextually appropriate responses.

The successful implementation of *TALL* could have broader implications for improving LLM performance in other low-resource languages, providing a generalizable approach that can be adapted to different linguistic contexts.

6 Generalizability

The *TALL* architecture is designed with modularity in mind, allowing it to be extended and adapted to other low-resource languages beyond Hebrew. By structuring the codebase in a modular fashion, components such as the pre-trained translators, custom transformers, and LLM can be easily replaced or modified to handle different language pairs. This flexibility makes it straightforward to apply the *TALL* architecture to a wide range of low-resource languages, leveraging the same principles of dual translation and hidden state injection to improve language processing performance.

7 Future Work

While the current proposal focuses on Hebrew, there are several avenues for future work. First, we plan to extend the evaluation of *TALL* to other low-resource languages, to further validate its generalizability and effectiveness.

Another important area for future research is the development of more sophisticated techniques, like the addition of soft prompts and prompt tuning, to enhance their ability to capture cross-linguistic nuances. Through these future efforts, we hope to refine and expand the *TALL* architecture, making it a more powerful tool for addressing the challenges of low-resource language processing.

8 Conclusion

In this research proposal, we introduced the *TALL* architecture, a novel approach designed to improve the performance of Large Language Models (LLMs) in low-resource languages by leveraging dual translation and custom transformers. By allowing the LLM to "think" in a high-resource language like English and then translating the output back into the low-resource language, *TALL* aims to overcome the challenges posed by sparse data and linguistic complexity. Our modular design ensures that *TALL* can be easily adapted to other low-resource languages, offering a versatile solution for addressing the challenges inherent in these linguistic contexts. Future work will focus on extending the architecture's applicability and exploring advanced techniques to further refine its performance.

References

[1] Samuel Cahyawijaya, Holy Lovenia, and Pascale Fung. Llms are few-shot in-context low-resource language learners. *arXiv preprint arXiv:2403.16512*, 2024.

[2] Séamus Lankford, Haithem Afli, and Andy Way. adaptmllm: Fine-tuning multilingual language models on low-resource languages with integrated llm playgrounds. *Information*, 14(12):638, 2023.

[3] Xiangpeng Wei, Haoran Wei, Huan Lin, Tianhao Li, Pei Zhang, Xingzhang Ren, Mei Li, Yu Wan, Zhiwei Cao, Binbin Xie, et al. Polylm: An open source polyglot large language model. *arXiv preprint arXiv:2307.06018*, 2023.