

How do Large Language Models Handle Multilingualism?

Yiran Zhao^{1,2†} Wenxuan Zhang^{2,3‡} Guizhen Chen^{2,4§} Kenji Kawaguchi¹ Lidong Bing^{2,3}

¹ National University of Singapore ² DAMO Academy, Alibaba Group, Singapore

³ Hupan Lab, 310023, Hangzhou, China ⁴ Nanyang Technological University, Singapore

Abstract

Large language models (LLMs) have demonstrated impressive capabilities across diverse languages. This study explores how LLMs handle multilingualism. Based on observed language ratio shifts among layers and the relationships between network structures and certain capabilities, we hypothesize the LLM’s multilingual workflow (**MWork**): LLMs initially understand the query, converting multilingual inputs into English for task-solving. In the intermediate layers, they employ English for reasoning and incorporate multilingual knowledge with self-attention and feed-forward structures, respectively. In the final layers, LLMs generate responses aligned with the original language of the query. To verify **MWork**, we introduce Parallel Language-specific Neuron Detection (**PLND**) to identify activated neurons for inputs in different languages without any labeled data. Using **PLND**, we validate **MWork** through extensive experiments involving the deactivation of language-specific neurons across various layers and structures. Moreover, **MWork** allows fine-tuning of language-specific neurons with a small dataset, enhancing multilingual abilities in a specific language without compromising others. This approach results in an average improvement of 3.6% for high-resource languages and 2.3% for low-resource languages across all tasks with just 400 documents.¹

1 Introduction

Recent advancements in large language models (LLMs) (OpenAI, 2023; Touvron et al., 2023; Team et al., 2023) have dramatically transformed the field of natural language processing (NLP). Thanks to the extensive pretraining on massive corpora mixed with different languages, these models demonstrate remarkable capabilities in understanding and generating text across multiple languages (Huang et al., 2023; Zhang et al., 2023a; Zhao et al., 2024a). Despite these advancements, the intricate mechanism of their multilingual processing behavior remains largely unclear, which leads to an important research question: *How do large language models handle multilingualism?*

To understand the working mechanism of LLMs, existing studies mainly focus on the relationship between model architectures and certain capabilities, with some investigating reasoning abilities with self-attention layers (Hou et al., 2023; Stolfo et al., 2023; Friedman et al., 2023), and others interpreting feed-forward layers as key-value memories for storing factual knowledge (Geva et al., 2021; Dai et al., 2022; Meng et al., 2022). However, these works solely center on English and neglect the multilingual features of LLMs in their interpretations.

[†]This work was done during the internship of Yiran Zhao at Alibaba DAMO Academy.

[‡]Wenxuan Zhang is the corresponding author: isakzhang@gmail.com

[§]Guizhen Chen is under the Joint Ph.D. Program between DAMO Academy and NTU.

¹Our code is available at https://github.com/DAMO-NLP-SG/multilingual_analysis

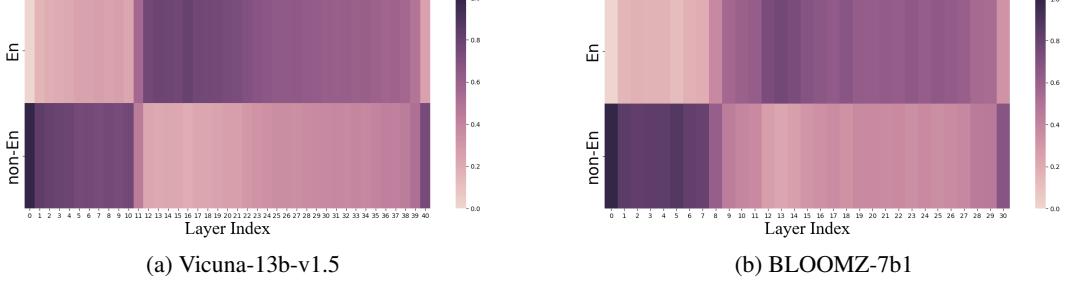


Figure 1: Ratio of English and non-English tokens among layers given non-English queries.

To gain an initial understanding of the multilingual mechanism of LLMs, we test LLMs with various non-English queries and decode the hidden embeddings of each layer to tokens within the LLM’s vocabulary. Subsequently, we classify these decoded tokens into either English or non-English, and analyze the ratio. Figure 1 illustrates the ratio of English and non-English tokens for each layer of two LLMs. We observe that non-English queries initially generate non-English embeddings as expected. However, as queries progress through the middle layers, the representations surprisingly become English-centric. In the final layers, there is a reversion to predominantly non-English embeddings, matching the non-English queries.

Motivated by the observed transformation above, we hypothesize a three-stage multilingual workflow: *understanding*, *task-solving*, and *generating*. This involves understanding the original non-English queries and interpreting them in English, solving tasks in English, and reverting outputs back to the original language. Furthermore, building upon previous studies that link self-attention structures to reasoning and feed-forward structures to factual knowledge storage (Hou et al., 2023; Geva et al., 2021), we further decouple the task-solving stage into reasoning with self-attention structures and extracting multilingual knowledge with feed-forward structures. Therefore, our hypothesized Multilingual Workflow (**MWork**) illustrated in Figure 2 outlines the three operational stages of LLMs in processing multilingual queries: Initially, LLMs *understand* queries by converting diverse linguistic features into a unified representation. In the *task-solving* phase, LLMs reason in English and incorporate multilingual knowledge to obtain factual content, using self-attention and feed-forward structures, respectively. Finally, models *generate* responses in the original language as the original query.

To verify the proposed **MWork**, we could extract language-specific parameters, selectively deactivate them within different structures, and observe their corresponding effects, thereby assessing the functionality of corresponding structures and validating our hypothesis. To identify the parameters to be activated, we develop a novel approach called Parallel Language-specific Neuron Detection (**PLND**). Unlike existing methods that rely on fine-tuning(Frankle and Carbin, 2018; Zhang et al., 2023b), labeled data (Tang et al., 2024; Liu et al., 2024), or parallel corpora (Libovicky et al., 2020; Tanti et al., 2021; Zhang et al., 2024) to detect activated parameters, **PLND** measures the significance of individual neurons with respect to the input in both attention and feed-forward structures without any labeled data or parameter adjustments. Using **PLND**, we identify language-specific neurons by inputting a free text corpus of that language and isolating consistently activated neurons. We find that by deactivating language-specific neurons which account for only 0.13% of all neurons, LLMs’ performance on a multilingual summarization task could drop by 99%.

We then extensively verify the hypothesized **MWork** framework using the proposed **PLND** method. Employing various benchmark tasks, including XQuAD (Artetxe et al., 2020) for understanding,

MGSM (Shi et al., 2022) for reasoning, X-CSQA (Lin et al., 2021) for knowledge extraction, and XLSum for generation (Hasan et al., 2021), we selectively deactivate language-specific neurons in each component and verify the functionality of the component by observing a significant decline in performance on the corresponding task. For example, when deactivating the language-specific neurons in the understanding layer, the performance on the multilingual understanding task XQuAD remains stable in English, while experiencing a decrease of 14% in non-English languages. Other tasks exhibit similar pattern when deactivating corresponding neurons. More importantly, with the verified **MWork** framework, enhancing the multilingual capabilities of LLMs can thus be achieved through the fine-tuning of language-specific neurons for certain capabilities. With a remarkable reduction in the training corpus size to a mere few hundred documents, this fine-tuning procedure enhances the multilingual capabilities of LLMs for both high-resource and low-resource languages by an average of 3.6% and 2.3% across all tasks, respectively. Notably, even without an English training corpus, there is a noticeable improvement in English performance, as the enhancement of language-specific neurons yields greater accuracy in enhancing specific languages, while simultaneously ensuring a clear division of parameters among different languages. In summary, the verified **MWork** reveals how LLMs handle multilingual tasks and offers an effective approach for conducting language-specific enhancements without compromising performance in other languages.

2 Parallel Language-specific Neuron Detection (PLND)

To verify the hypothesized workflow, we propose PLND that effectively detects language-specific neurons without relying on any labeled data. In essence, PLND identifies neurons crucial for handling individual documents, with language-specific neurons being those that consistently show high importance when processing documents in a particular language.

2.1 Sequential Neuron Detection

We define a neuron as a single row or column of a parameter matrix of a language model. To identify neurons responsible for a specific language, it is crucial to discern the significance of a neuron with respect to the inference of a given input. Specifically, when processing the input c in the model, we denote the hidden embedding before the i -th layer in Transformer (Vaswani et al., 2017) as h_i , and the hidden embedding after the i -th layer as $h_{i+1} = T_i(h_i)$, where T_i represents the parameters of the i -th layer. For a specific neuron within the i -th layer, denoted as $N^{(i)}$, either located in the attention or feed-forward network, we quantify its importance in processing the input c by measuring the difference in the hidden embedding after the i -th layer, i.e., h_{i+1} , when $N^{(i)}$ is activated or deactivated. Formally, the impact of neuron $N^{(i)}$ for input c is defined as

$$\text{Imp}(N^{(i)}|c) = \|T_i \setminus N^{(i)}(h_i) - T_i(h_i)\|_2, \quad (1)$$

where $T_i \setminus N^{(i)}(\cdot)$ denotes deactivating $N^{(i)}$ in T_i , i.e., setting all parameters of the neuron $N^{(i)}$ to zero. With a set of n corpus in a specific language, denoted as $\mathcal{C} = \{c_1, \dots, c_l, \dots, c_n\}$, we calculate the importance of each neuron in each layer to each corpus. Furthermore, we can obtain language-specific neurons that are important to all corpus in that language, i.e.,

$$\{N^{(i)} \mid \text{Imp}(N^{(i)}|c_l) \geq \epsilon, \forall c_l \in \mathcal{C}\}, \quad (2)$$

where ϵ is the pre-defined threshold.

2.2 Parallel Neuron Detection

The sequential neuron detection requires traversal of all neurons and inputs sequentially and thus is time-consuming. To address this, we further propose a parallel algorithm for accelerating the process.

Feed-Forward Network (FFN) In the latest open-source models, when processing input c , the feed-forward network in a certain layer is defined as

$$\text{FFN}(x) = (\text{SiLU}(W_{\text{gate}}(x)) \cdot W_{\text{up}}(x)) W_{\text{down}}, \quad (3)$$