Unveiling *Code Region*: Identifying Critical Parameters for Coding Abilities in LLMs

Dongjun Kim°[1], Aiyanyo Imatitikua Danielle[2]
[1]Department of Computer Science and Engineering, Korea University [2]Human-Inspired AI Research
{junkim100,titi}@korea.ac.kr

Abstract

Large Language Models (LLMs) have demonstrated remarkable proficiency in code generation across various programming languages. This study introduces the concept of a "***Code Region***" - a set of parameters crucial for coding abilities in LLMs. We analyze the LLaMA 3.1 8B instruct model to identify these parameters using a novel approach combining parameter importance scoring and Taylor expansion approximation. Our methodology efficiently pinpoints parameters that consistently exhibit high importance across multiple programming languages. Experimental results show that removing just 1% of the identified ***Code Region*** leads to a significant 85.32% decrease in performance on the HumanEval benchmark, while having minimal impact on other language understanding tasks. This finding suggests the existence of a specialized subset of parameters responsible for coding abilities in LLMs. Our research provides valuable insights into the internal mechanisms of LLMs, highlighting the polysemantic nature of their parameters and opening new avenues for optimizing models for coding tasks.

Keywords: Large Language Models, Coding Models, Parameter Analysis, LLM Interpretability

1. Introduction

The rapid advancement of Large Language Models (LLMs) has significantly transformed automated code generation. Both code-specialized models (e.g., CodeLlama[1], StarCoder[2]) and instruction-tuned models (e.g., GPT-4 [3], Claude 3.5 Sonnet*) have demonstrated remarkable coding proficiency across diverse programming languages. This raises a pivotal research question: Are the coding abilities across different programming languages derived from a common set of parameters within these models? This study aims to uncover the ***Code Region***—the set of parameters that are consistently important across multiple programming languages. By analyzing the LLaMA 3.1 8B instruct model, we seek to identify these critical parameters and evaluate their impact on coding tasks, contributing to the broader understanding of LLM capabilities and their potential optimization.

*https://www.anthropic.com/news/claude-3-5-sonnet

2. Methodology

2.1 Parameter Importance Scoring and *Code Region* Identification

We introduce the concept of a ***Code Region***, which refers to parameters that are consistently important across multiple programming languages. To identify these parameters, we use parameter importance scoring, which measures how sensitive the model's performance is to the removal of specific parameters. Due to the computational challenges of calculating importance for each parameter in large models, we leverage the Taylor expansion [4] of the loss function. This approximation allows us to efficiently compute importance scores for all parameters using a single backward pass to obtain the gradients. The process for identifying the ***Code Region*** involves:

1. Performing a forward pass to compute the loss on a batch of code corpus data.
2. Conducting a backward pass to obtain gradients for all parameters.
3. Calculating importance scores using the Taylor expansion approximation.

4. Analyzing importance scores across different programming languages to identify consistently important parameters.

2.2 Further Fine-Tuning and Analysis

We further fine-tune an instruction-tuned model on code data for each target programming language. This process allows us to observe how the model adapts its existing knowledge to the specific domain of programming and track changes in parameter importance. By analyzing the distribution and evolution of importance scores across different layers and attention heads of the model, we aim to identify the regions of the network that are most critical for coding tasks across various programming languages.

3. Results and Discussion

We assessed the performance of both the original model and a model with 1% of the ***Code Region*** removed (damaged model) using several benchmarks. The findings provide critical insights into the function of the ***Code Region***:

1. HumanEval Benchmark (Python coding): The damaged model showed a significant decline in performance (85.32% decrease), highlighting the importance of the ***Code Region*** for general coding proficiency.
2. Other Benchmarks (HellaSwag, MMLU, WinoGrande): These showed relatively minor decreases in performance, (36.18% decrease) suggesting that the ***Code Region*** primarily influences coding abilities with minimal impact on other language understanding capabilities.

The substantial performance drop in the coding task suggests that the ***Code Region*** parameters contribute significantly to programming abilities. This finding underscores the specialized nature of the identified parameters and their crucial role in maintaining the model's coding capabilities. The relatively minor impact on other language understanding benchmarks further supports the hypothesis that the ***Code Region*** primarily affects coding abilities.

4. Conclusion

In this study, we explored the coding capabilities of Large Language Models (LLMs) by identifying a set of parameters, termed the "***Code Region***," that are crucial for coding tasks across multiple programming languages. By analyzing the LLaMA 3.1 8B instruct model, we found that these parameters are essential for maintaining coding abilities, as evidenced by the significant performance drop in the HumanEval benchmark when 1% of the ***Code Region*** was removed. The stark contrast between the substantial decrease in coding performance and the minimal impact on other language tasks suggests that LLMs may develop specialized parameter subsets for different types of cognitive processes. Our research provides valuable insights into the internal mechanisms of LLMs and their ability to handle diverse programming languages, highlighting the polysemantic nature of LLM parameters. By identifying and understanding the role of the ***Code Region***, we can better optimize LLMs for coding tasks, paving the way for more efficient and specialized models in the future. This work opens up new avenues for research into the internal representations of LLMs and their implications for model design and training strategies in the context of code generation and understanding.

Acknowledgment

This work was supported by Institute for Information communications Technology Promotion(IITP) grant funded by the Korea government(MSIT). (RS-2024-00398115, Research on the reliability and coherence of outcomes produced by Generative AI). This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(NRF-2021R1A6A1A03045425).

Reference

[1] B. Roziere, J. Gehring, F. Gloeckle, S. Sootla, I. Gat, X. E. Tan, Y. Adi, J. Liu, R. Sauvestre, T. Remez *et al.*, "Code llama: Open foundation models for code," *arXiv preprint arXiv:2308.12950*, 2023.

[2] R. Li, L. B. Allal, Y. Zi, N. Muennighoff, D. Kocetkov, C. Mou, M. Marone, C. Akiki, J. Li, J. Chim *et al.*, "Starcoder: may the source be with you!" *arXiv preprint arXiv:2305.06161*, 2023.

[3] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[4] P. Molchanov, A. Mallya, S. Tyree, I. Frosio, and J. Kautz, "Importance estimation for neural network

pruning," *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11 264–11 272, 2019.