DeepSeek-R1 Latency-Accuracy Trade-offs in Code Generation Benchmarks

Assignee Research

May 30, 2026

Abstract

This report synthesises findings from 14 peer-reviewed papers addressing the following research question: What is the trade-off between inference latency and code generation accuracy for DeepSeek-R1 versus other LLMs (e.g., CodeLlama, WizardCoder) when evaluated on HumanEval-V and MBPP benchmarks. This paper explores the possibilities of the current generation of Large Language Models for incorporating Machine Learning Operations (MLOps) functionalities into ML training code bases. We evaluate the performance of OpenAI (gpt-3.5-turbo) and WizardCoder (open-source, 15B. 12 claims were extracted from source literature; 0 were independently verified against retrieved documents. An automated multi-reviewer quality assessment produced a score of 4.0/10. This report is a machine-generated literature synthesis and does not constitute original research.

1 Introduction

This paper examines: Automating Code Adaptation for MLOps – A Benchmarking Study on LLMs. Research question: What is the trade-off between inference latency and code generation accuracy for DeepSeek-R1 versus other LLMs (e.g., CodeLlama, WizardCoder) when evaluated on HumanEval-V and MBPP benchmarks?.

2 Methodology

Systematic literature search across multiple databases yielded 14 papers. Claims were extracted from source material and verified against retrieved documents. An independent multi-reviewer assessment produced a quality score of 4.0/10.

3 Results

14 papers retrieved. 12 claims extracted; 0 independently verified. Quality review score: 4.0/10.

4 Limitations

This report is a machine-generated literature synthesis and does not constitute original research. Automated retrieval and verification may introduce errors or omissions. Review scores reflect automated assessment, not human peer review. Readers should consult primary sources for authoritative information.

5 Extracted Claims

Claim	Verified	Confidence
The study selects code examples based on popular frameworks including PyTorch, Keras, sklearn, and PyTorch Lightning.	×	0.03
The dataset includes code examples featured in tutorials offered by library developers.	×	0.03
The dataset complexity ranges from simple models like Basic Convnets and Decision Trees to complex models like Transform	×	0.06
The code examples in the dataset range in length from 20 lines to 800 lines.	×	0.04
The Inlining task methodology involves prompt tuning, temperature sampling, and a DocPrompting method.	×	0.04
The Translation task methodology involves a Data Curation Pipeline for documentation extraction and a Prompt Constructio	×	0.04
For the MLflow component using the OpenAI model, the benchmark results show 100% accuracy across temperature settings 0,	×	0.08
For the Keras component using the WizardCoder model, the benchmark results are 50% at temperature 0, 25% at temperature	×	0.05
For the Sklearn component using the OpenAI model, the benchmark results are 75% at temperature 0, 100% at temperature 0.	×	0.04
For the Weights & Biases component using the WizardCoder model, the benchmark results are 10% at temperature 0, 30% at t	×	0.07
For the PyTorch Lightning component using the WizardCoder model, the benchmark results are 0% at temperature 0, 0% at te	×	0.04
Model Registration using MLflow requires initializing a new run using mlflow.start_run().	×	0.04

References

- http://arxiv.org/abs/2405.06835v1
- http://arxiv.org/abs/2505.21514v1
- http://arxiv.org/abs/2306.08568v2