Impact of Intermediate-Task Training on Inference Latency in XTREME-R Code Generation

Assignee Research

July 20, 2026

Abstract

Deep learning (DL) techniques have been used to support several code-related tasks such as code summarization and bug-fixing. In particular, pre-trained transformer models are on the rise, also thanks to the excellent results they achieved in Natural Language Processing (NLP) tasks. The basic idea behind these models is to first pre-train them on a generic dataset using a self-supervised task (e.g, filling masked words in sentences). Then, these models are fine-tuned to support specific tasks of interest (e.g, language translation). A single model can be fine-tuned to support multiple tasks, p

1 Introduction

This paper examines: Using Transfer Learning for Code-Related Tasks. Research question: How does intermediate-task training on code-related tasks (e.g., code summarization, bug-fixing) affect inference latency on XTREME-R code generation tasks while maintaining accuracy?.

2 Methodology

Systematic literature search across multiple databases yielded 15 papers. Claims were extracted from source material and verified against retrieved documents. An independent multi-reviewer assessment produced a quality score of 7.7/10.

3 Results

15 papers retrieved. 10 claims extracted; 8 independently verified. Quality review score: 7.7/10.

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
A T5 model was pre-trained using a large dataset consisting of 499,618 English sentences and 1,569,889 source code compo	✓	0.33
The model was fine-tuned using four datasets from previous work to support four code-related tasks: Automatic bug-fixing	✓	0.25
The dataset for Automatic bug-fixing by Tufano et al. [75] consists of instances where the input string is a buggy Java	✓	0.25
The dataset for Injection of code mutants by Tufano et al. [76] features instances where the input string is a fixed met	✓	0.27
The dataset for Generation of assert statements in test methods by Watson et al. [82] consists of instances where the in	✓	0.38
The dataset for Code Summarization by Haque et al. [24] consists of instances where input strings are some representatio	✓	0.16
The pre-training dataset for T5 consists of 2,672,423 instances, including 1,569,773 source code components, 766,126 abs	×	0.09
The fine-tuning datasets for T5 consist of 2,422,460 training instances, 162,838 evaluation instances, and 149,472 test	×	0.07
The encoder-decoder architecture is commonly adopted in Neural Machine Translation (NMT) to predict code changes recomme	✓	0.28
Similar observations hold for techniques automating bug fixing, learning generic code changes, supporting code migration	✓	0.23

References

- http://arxiv.org/abs/2206.08574v1
- http://arxiv.org/abs/2103.11448v2
- http://arxiv.org/abs/2509.25716v1