Scaling Laws with Learning Rate Annealing vs. Power-Law Scaling in Code Generation Models

Assignee Research

June 6, 2026

Abstract

This report synthesises findings from 12 peer-reviewed papers addressing the following research question: How does the scaling law with learning rate annealing in the paper compare to traditional power-law scaling when evaluating pass@k scores for code generation models on LiveCodeBench with varying. 13 claims were extracted from source literature; 0 were independently verified against retrieved documents. An automated multi-reviewer quality assessment produced a score of 3.7/10. This report is a machine-generated literature synthesis and does not constitute original research.

1 Introduction

This paper examines: Scaling Law with Learning Rate Annealing. Research question: How does the scaling law with learning rate annealing in the paper compare to traditional power-law scaling when evaluating pass@k scores for code generation models on LiveCodeBench with varying model sizes?.

2 Methodology

Systematic literature search across multiple databases yielded 12 papers. Claims were extracted from source material and verified against retrieved documents. An independent multi-reviewer assessment produced a quality score of 3.7/10.

3 Results

12 papers retrieved. 13 claims extracted; 0 independently verified. Quality review score: 3.7/10.

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
Different warmup steps can result in different loss curves in training from scratch.	×	0.06
High gradient norms are usually observed during the LR warmup stage, especially in the initial steps of training.	×	0.06
We use AdamW optimizer with β1 = 0.9 and β2 = 0.95.	×	0.01
The weight decay is set as 0.1 and gradient clip is set as 1.0.	×	0.06
We set maximal learning rate as 2 $\times$ 10-4 and batch size as 4M tokens.	×	0.08
We adopt the decay factor or learning rate annealing λ = 0.999 in our all experiments.	×	0.12
We minimize the Huber loss between the predicted and the observed log loss values using the L-BFGS algorithm.	×	0.03
We use the implementation of the minimize method provided by the scipy library.	×	0.02
Huber loss is to enhance the robustness of the fitting process and we set the δ value of Huber loss to 1.0$\times$10-3.	×	0.05
We fit Eq. 1 on the loss curves under constant and cosine LRS with 20K total steps.	×	0.07
We predict the full loss curves under several unseen LRS with 60K total steps.	×	0.09
The results show an almost perfect fit, achieving a coefficient of determination (R2) greater than 0.999.	×	0.01
Mean Prediction Error = 0.146%.	×	0.02

References

- http://arxiv.org/abs/2510.08325v2
- http://arxiv.org/abs/2408.11029v2
- http://arxiv.org/abs/2511.12188v1