Learning Rate Annealing Schedules and Their Impact on Code Generation Pass@k Scores

Assignee Research

June 6, 2026

Abstract

This report synthesises findings from 10 peer-reviewed papers addressing the following research question: How do different learning rate annealing schedules (e.g., linear, cosine, exponential) compare in terms of pass@k scores on LiveCodeBench when applied to code generation models like Code Llama or. 16 claims were extracted from source literature; 0 were independently verified against retrieved documents. An automated multi-reviewer quality assessment produced a score of 3.9/10. This report is a machine-generated literature synthesis and does not constitute original research.

1 Introduction

This paper examines: Beyond Pass@k: Breadth-Depth Metrics for Reasoning Boundaries. Research question: How do different learning rate annealing schedules (e.g., linear, cosine, exponential) compare in terms of pass@k scores on LiveCodeBench when applied to code generation models like Code Llama or StarCoder?.

2 Methodology

Systematic literature search across multiple databases yielded 10 papers. Claims were extracted from source material and verified against retrieved documents. An independent multi-reviewer assessment produced a quality score of 3.9/10.

3 Results

10 papers retrieved. 16 claims extracted; 0 independently verified. Quality review score: 3.9/10.

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
Models are usually evaluated via Pass@k, using k = 1 as the most common value.	×	0.07
Higher values of k (e.g., 16 or 64) are used to test the upper bound capabilities of models.	×	0.04
CoT-Pass@k accounts for the correctness of both the thinking tokens and the final answer.	×	0.03
maj@k counts a problem as solved if more than 50% of k completions are correct.	×	0.05
cons@k counts a problem as solved based on the mode (most frequent) of k completions.	×	0.03
Maj@k is equivalent to Cover@τ with τ = 0.5.	×	0.03
Cons@k does not correspond to a single Cover@τ because its effective τ varies per problem based on the mode's frequency.	×	0.02
G-Pass@kτ is a generalization of Pass@k that evaluates model capability under different consistency levels τ.	×	0.05
G-Pass@kτ focuses on higher reliability levels where τ is in the range [0.5, 1.0].	×	0.05
The Cover@τ metric evaluates performance at low (τ = 0.2) and high (τ = 0.8) reliability thresholds.	×	0.03
GRPO suffers from optimization biases and entropy collapse.	×	0.03
Entropy collapse reduces exploration and quickly saturates performance.	×	0.02
KL-cov suppresses high-covariance tokens that correlate with large decay.	×	0.02
In the provided benchmark table, the KL-Cov method achieved a score of 62.00 in the second metric column.	×	0.03
In the provided benchmark table, the Unlikeliness method achieved a score of 0.00 in the fourth metric column.	×	0.03
In the provided benchmark table, the GRPO method achieved a score of 59.66 in the first metric column.	×	0.03

References

- http://arxiv.org/abs/2510.08325v2
- http://arxiv.org/abs/2310.07831v2
- http://arxiv.org/abs/2403.07974v2