Test-Time Compute Scaling and Adaptive Token-Level Reasoning in Language Models

Assignee Research

June 6, 2026

Abstract

This report synthesises findings from 15 peer-reviewed papers addressing the following research question: How does test-time compute scaling improve language model performance on reasoning benchmarks v10. 19 claims were extracted from source literature; 5 were independently verified against retrieved documents. An automated multi-reviewer quality assessment produced a score of 5.0/10. This report is a machine-generated literature synthesis and does not constitute original research.

1 Introduction

This paper examines: Understanding Dynamic Compute Allocation in Recurrent Transformers. Research question: How does test-time compute scaling improve language model performance on reasoning benchmarks v10.

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 5.0/10.

3 Results

15 papers retrieved. 19 claims extracted; 5 independently verified. Quality review score: 5.0/10.

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
ANIRA follows the Prelude–Recurrent–Coda architecture, akin to Geiping et al. (2025).	×	0.03
The Prelude is a small stack of causal Transformer layers (Vaswani et al., 2017) producing contextual token representati	×	0.03
The Recurrent core is a Transformer block applied for up to D iterations.	×	0.05
The Coda is a small stack of causal Transformer layers that maps the recurrent output to next-token logits.	×	0.03
ANIRA is token-level adaptive.	✓	0.20
Adaptivity is learned through a separate decider module that predicts per-token exit depth, i.e., the number of recurren	×	0.03
We study two variants of this decider module, Depth Decider that makes an early decision from the Prelude representation	×	0.06
We train the ANIRA models on MANO instances spanning difficulty levels L $\in${3, . . . , 16} and separately on BREVO instan	×	0.03
On CLRS, we train multitask ANIRA models across all problems in the CLRS training dataset.	×	0.03
On LANO, we train ANIRA models on the PCFG 3b6 described in Allen-Zhu (2025).	×	0.02
We set the maximum number of recurrent iterations to D = 6 for all tasks, except MANO where we use D = 14.	×	0.03
Figure 2 shows that the ANIRA models learn to allocate compute consistent with task complexity on the MANO and the BREVO	×	0.07
Figure 3 shows the same for a representative selection of tasks from the CLRS dataset.	×	0.02
Table 1 shows that the ANIRA models' compute allocation is consistent with the token level prediction complexity on synt	✓	0.17
The ANIRA models allocate compute consistent with complexity even in the absence of explicit complexity signals during t	×	0.06
Compute allocation aligned with task complexity can emerge without explicit difficulty supervision.	✓	0.29
Models fail to extrapolate to unseen input sizes despite allocating additional computation.	✓	0.26
Early compute decisions rely on static structural cues, whereas online halting more closely tracks algorithmic execution	✓	0.32
We identify a consistent two-phase regime—learning followed by compute reduction—providing insight into how adaptive com	×	0.04

References

- http://arxiv.org/abs/2504.00869v2
- http://arxiv.org/abs/2510.00071v2
- http://arxiv.org/abs/2602.08864v1