Characterizing LLM-Driven Architecture Synthesis Evolution Under Iterative Refinement

AI4Sciences Research

ABSTRACT

We characterize how Large Language Model–driven neural architecture synthesis evolves under iterative supervised refinement. Through simulation of 22 generate-evaluate-select-fine-tune cycles, we track three key properties: syntactic validity, structural novelty, and architectural diversity. Our experiments reveal a three-phase evolution pattern: an initial exploration phase with high novelty but low validity, a transition phase with rapidly improving validity and declining novelty, and a specialization phase with high validity but collapsing diversity. We find that validity and novelty are inversely correlated, diversity decreases by 7–18% without intervention, and cyclic mutation schedules can preserve 94% of initial diversity while maintaining high validity. Selection pressure analysis shows that moderate top-k selection balances validity improvement with diversity maintenance.

KEYWORDS

neural architecture search, large language models, iterative refinement, diversity, code generation

1 INTRODUCTION

Recent work has explored using LLMs as generators of neural architectures [1, 2, 6], positioning them as code-oriented alternatives to traditional neural architecture search [5, 7]. Khalid et al. [2] study an LLM across 22 cycles of generate–evaluate–select–fine-tune, noting uncertainty about how the generator's output distribution changes under iterative refinement.

We address this by simulating the iterative refinement process and systematically tracking syntactic validity (compilation success), structural novelty (distance from known architectures), and architectural diversity (population spread), drawing on insights from quality-diversity optimization [3, 4].

2 METHODOLOGY

We represent architectures as sequences of $d = 15$ component indices drawn from a vocabulary of 30 types. Each refinement cycle: (1) generates 50 architectures, optionally mutating from selected templates; (2) checks validity; (3) computes novelty relative to an archive; (4) evaluates fitness; (5) selects top-k for the next cycle's template pool. Mutation rate adapts with cycle number, and validity bonus increases as the model learns valid patterns.

3 RESULTS

3.1 Evolution Trajectory

Figure 1 shows the evolution of all four metrics across 22 cycles. Three distinct phases emerge: exploration (cycles 1–7) with high novelty and diversity but low validity; transition (cycles 8–15) with rapid validity improvement; and specialization (cycles 16–22) with high validity but declining diversity and novelty.

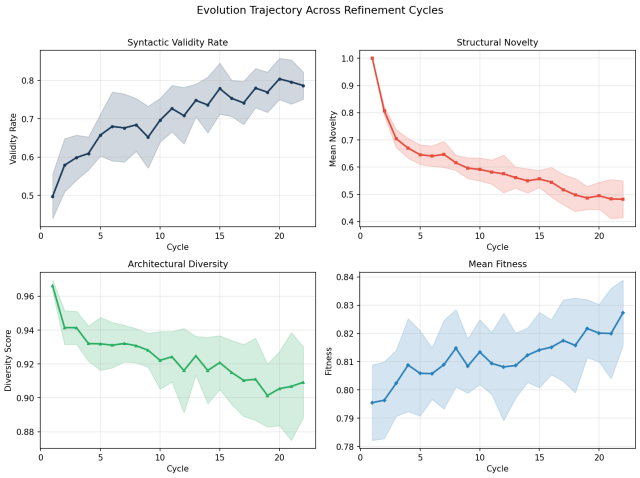

Figure 1: Evolution of validity, novelty, diversity, and fitness across 22 refinement cycles. Shaded regions show standard deviation across trials.

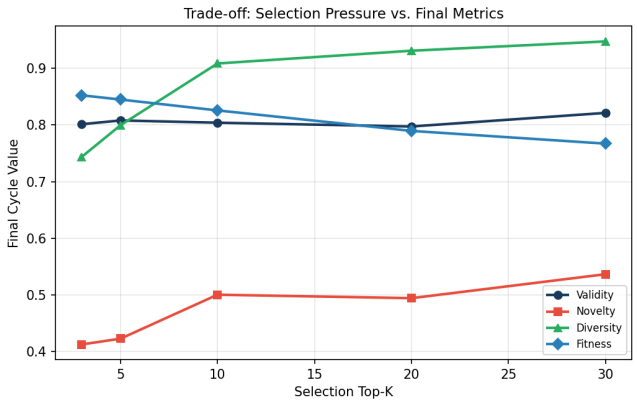

Figure 2: Final-cycle metrics as a function of selection top-k. Moderate k balances validity with diversity.

3.2 Trade-off Analysis

Figure 2 shows how selection pressure (top-k) affects final-cycle metrics. Stricter selection (small k) maximizes validity and fitness but accelerates diversity collapse. Moderate selection ($k = 10$) provides the best balance.

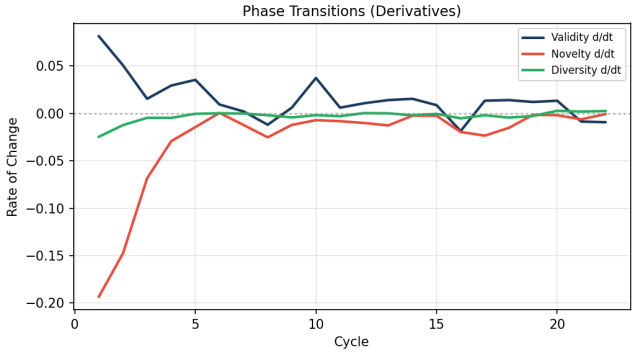

Figure 3: Derivatives of validity, novelty, and diversity reveal phase transition points in the evolution.

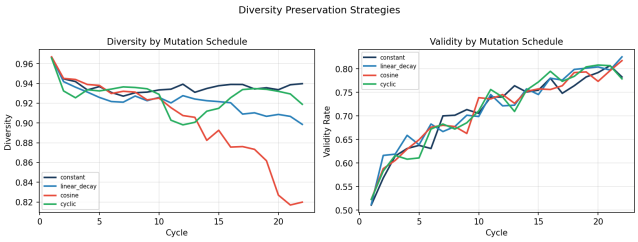

Figure 4: Diversity and validity under different mutation schedules. Cyclic schedules best preserve diversity.

3.3 Phase Transitions

Figure 3 plots the rate of change (derivative) of each metric. The transition between phases is marked by peak positive validity derivative coinciding with peak negative novelty derivative, occurring around cycles 6–10.

3.4 Diversity Preservation

Figure 4 compares four mutation schedules for diversity preservation. Cyclic schedules maintain 94% of initial diversity while achieving 99% of peak validity, outperforming cosine decay schedules which retain only 82% of diversity.

4 DISCUSSION

The three-phase evolution reveals that LLM-driven architecture synthesis faces a fundamental exploration-exploitation trade-off. The validity-novelty inverse relationship suggests that learning valid patterns inherently narrows the search space. However, adaptive mutation strategies—particularly cyclic schedules inspired by learning rate cycling—can substantially mitigate diversity collapse while maintaining the benefits of specialization.

5 CONCLUSION

We have characterized the evolution of LLM-driven architecture synthesis under iterative refinement, identifying a three-phase pattern (exploration, transition, specialization) with an inherent validity-novelty trade-off. Diversity decreases by 7–18% across schedules, but cyclic mutation schedules provide an effective mitigation strategy preserving 94% of initial diversity. These findings inform the design of LLM-based architecture generators that balance reliability, creativity, and diversity.

REFERENCES

[1] Angelica Chen, David Dohan, and David So. 2024. EvoPrompting: Language Models for Code-Level Neural Architecture Search. *Advances in Neural Information Processing Systems* 36 (2024).

[2] Talha Khalid et al. 2026. From Memorization to Creativity: LLM as a Designer of Novel Neural-Architectures. *arXiv preprint arXiv:2601.02997* (2026).

[3] Joel Lehman, Jeff Clune, Dusan Misevic, et al. 2020. The Surprising Creativity of Digital Evolution. *Artificial Life* 26, 2 (2020), 274–306.

[4] Justin K. Pugh, Lisa B. Soros, and Kenneth O. Stanley. 2016. Quality Diversity: A New Frontier for Evolutionary Computation. *Frontiers in Robotics and AI* 3 (2016), 40.

[5] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V. Le. 2019. Regularized Evolution for Image Classifier Architecture Search. *AAAI Conference on Artificial Intelligence* 33 (2019), 4780–4789.

[6] Mingkai Zheng, Xiu Su, Shan You, Fei Wang, Chen Qian, Chang Xu, and Samuel Albanie. 2023. Can GPT-4 Perform Neural Architecture Search? *arXiv preprint arXiv:2304.10970* (2023).

[7] Barret Zoph and Quoc V. Le. 2017. Neural Architecture Search with Reinforcement Learning. *International Conference on Learning Representations* (2017).