Multi-Component ARCHON

Megan Mou, Andrew Park, Sherry Xie, Emily Zhang

CS 329A Final Project

Introduction

Problem: With the rising importance of using inference-time techniques to improve model capabilities, ARCHON proposes using a wide variety of LLMs and inference-time techniques to generate LLM systems more powerful than simply the combination of them. However, promising ARCHON performance in existing benchmarks almost all rely on LLMs with about 70B parameters and different models also achieve varied performance on sub-tasks within each query category.

Research Question: Can we improve ARCHON performance by adding more inference-time components and optimizing its architecture to enhance both closed-source and open-source models?

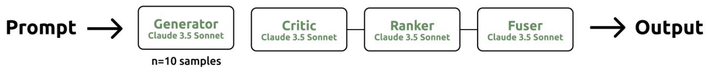


Figure 1: All-Source Archon Architecture for Instruction-Following from the Archon paper

Datasets

Humanity's Last Exam: a multi-modal benchmark at the frontier of human knowledge, designed to be the final closed-ended academic benchmark of its kind with broad subject coverage. → 2,700 challenging questions across over a hundred subjects. We use a subset of HLE (500 questions, limited by compute) to evaluate whether Archon's improvements enhance deep reasoning, problem-solving, and model calibration at the highest difficulty level.

AlpacaEval: an instruction-following benchmark designed to test LLMs on real-world tasks. It evaluates models using automated pairwise comparisons against top-performing LLMs like GPT-4 and Claude 3.5. Unlike factual benchmarks, AlpacaEval measures response quality, clarity, and adherence to instructions. We use it to ensure Archon's inference-time optimizations improve usability and human alignment while maintaining accuracy.

Method & Experiments

New Inference-Time Components:

1. **Planner**: Component at the beginning of all ARCHON layers which can rephrase the user prompt and list out step-by-step execution plan that is appended to future tool call and generations.
2. **Expander**: Enhance ARCHON existing generator responses by adding additional context and details to improve informativeness.
3. **Web Search Tool Call:** Generate multiple search queries for the Google API from planner trace to give the model additional context from live webpage results.

Ablations on our datasets:

1. Planner + Base ARCHON architecture
2. Planner + Expander + Base ARCHON architecture
3. Planner + Tool Call + Expander + Base ARCHON architecture

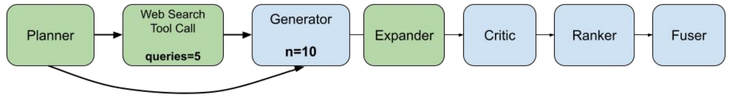
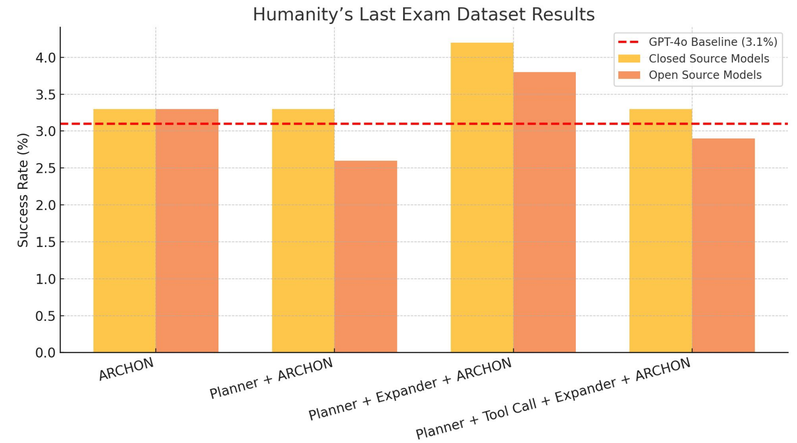


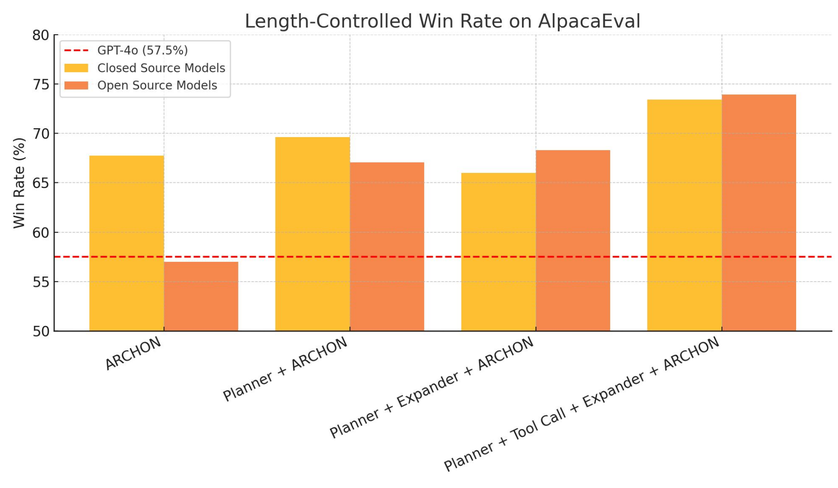
Figure 2: Our Complete Architecture (Ablation 3)

Future Directions

1. **Query-Adaptive Planner:** Component at the beginning of all ARCHON layers that dynamically generates an entire query-specific ARCHON architecture and executes the user prompt
2. **Additional Tool Calls:** External APIs and specialized databases may further improve knowledge retrieval
3. **Early Stopping:** Investigate ways to reduce inference costs by balancing number of LLM calls with overall response quality
4. **Model Size:** Investigate whether the planner, expander, and tool-call modules can compensate for reduced model size (e.g. 7B) while maintaining competitive accuracy

Results & Analysis





Expander effectiveness depends on benchmark: AlpacaEval's length-controlled scoring limited the impact of the Expander module, yet performed well on HLE due to extended reasoning time.

Open-source models rival closed-source performance: Planner and Tool Call brought open-source models to parity with closed-source models in instruction following and information retrieval tasks while maintaining efficiency.

Web search significantly improves instruction-following: Tool Call pushed AlpacaEval scores beyond 0.7, outperforming all original ARCHON architectures.

Reasoning remains difficult: While we attempted to hill-climb on HLE, adding external tool call was insufficient, suggesting that reasoning is the main bottleneck.