

Hi everyone! I submitted this just now (Jun 2, 2025 4:58 a.m.) (mostly by mistake, pressing on the submit button thinking it was a “Next”). Hoping nothing too bad is missing or contained within. I’m going to sleep before the sun rises fully;

Gnight & thanks for all the fish! ~Luiza~

LLM Fingerprinting Through Semantic Variability

Apart x Martian

Mechanistic Router Interpretability Hackathon

Chris Forrester
Hypernym

Luiza Corpaci
TU Wien

Siddhesh Pawar
University of
Copenhagen

Abstract

This project develops an LLM fingerprinting and analysis toolkit to increase transparency in AI routing systems, addressing Track 2: Intelligent Router Systems through two key investigations. We adapted semantic variability analysis to create unique behavioral fingerprints that can identify which specific models are operating behind opaque routing services, and conducted tool detection experiments under semantic noise to assess model robustness. Our findings demonstrate that models maintain high semantic robustness while our fingerprinting technique successfully distinguishes between different models based on their response patterns. These contributions aid the Expert Orchestration Architecture vision by providing practical tools for auditing multi-model AI systems, enabling organizations to understand which models their routers actually use and verify their reliability under real-world conditions, ultimately making router systems more transparent and trustworthy for production deployment. The codebase can be accessed on [Github](#).

Keywords: AI security, model evaluations, safety infrastructure, LLM fingerprinting, router interpretability, semantic robustness, tool use detection.

Introduction

Current large language models (LLMs), while powerful, often operate as monolithic "black boxes," making it difficult to understand their internal decision-making processes, ensure their reliability, or verify their outputs. This is further made more difficult when accessed through routing layers that obscure the specific underlying model. This opacity poses challenges for safety, alignment, and trust.

Our motivation is to develop tools and techniques that increase the transparency and auditability of multi-model AI systems. By "fingerprinting" models and testing their robustness, we aim to provide a clearer understanding of their behavior, which takes us a step closer towards an Expert Orchestration Architecture. This work seeks to replace the opacity of current systems with verifiable insights, contributing to safer and more reliable AI.

Our research addresses the following questions:

1. **Model Fingerprinting:** Can distinctive behavioral signatures reliably distinguish between LLMs based on their semantic response patterns?
2. **Router Transparency:** Does behavioral fingerprinting enable identification of specific models operating within opaque routing systems?
3. **Robustness Under Noise:** How effectively do LLMs maintain tool-calling functionality when faced with semantic noise and distracting content?

Related Work

LLM Fingerprinting: Recent work has explored model identification through cryptographic methods (Chain & Hash [Rusinov & Salem, 2024]), instructional backdoors [Xu et al., 2024], and active probing [Pasquini et al., 2024]. Our approach differs by focusing on semantic response variability patterns.

Router Systems: Production systems like Amazon Bedrock [AWS, 2024] optimize routing for cost and performance. Our work addresses the complementary challenge of router interpretability.

Tool Use Robustness: Function calling evaluation has established metrics for correctness [Schick et al., 2024]. We extend this to semantic noise robustness testing.

1.2 Challenge Track Focus

This project primarily addresses **Track 2: Intelligent Router Systems**. Our model fingerprinting work aims to understand and interpret the behavior of intelligent routers by identifying the characteristics of the models they might be using. Our work also touches on aspects of **Track 1: Judge Model Development**, as the fingerprinting metrics can be seen as a form of model evaluation or "judging" a model's characteristic behavior.

1.3 Contribution to Expert Orchestration

Our work contributes to the Expert Orchestration vision by:

- **Router Transparency:** Fingerprinting techniques reveal which specific models operate behind opaque routing systems, enabling identification of the actual "expert" performing each task.
- **Behavioral Auditability:** Systematic behavioral pattern analysis provides granular model understanding essential for debugging, accountability, and trust in multi-model AI systems.
- **Robustness Assessment:** Quantifying model performance under semantic noise (maintaining 90%+ coherence while dropping optional tools) establishes safety boundaries for real-world deployment.
- **Intelligent Router Design:** Fingerprinting insights enable informed model selection based on well-characterized behavioral profiles rather than opaque performance metrics.

These contributions collectively advance the Expert Orchestration Architecture by replacing black-box routing with transparent, auditable model selection based on systematic behavioral understanding.

Methods

2.1 Technical Approach

Our methodology involved two primary investigative threads, both using Martian for accessing various LLMs:

Investigation 1: Model Fingerprinting

Objective: identify unique, consistent behavioral fingerprints for different LLMs through analysis of semantic variability patterns in text reconstruction tasks.

Models/Datasets Used: a wide range of models accessible via the Martian interface, including those from OpenAI (GPT-4, GPT-4o), Anthropic (Claude 3.5). The dataset consisted of procedurally generated prompts based on "hyperstrings."

Hyperstring Format: Structured semantic compression following the pattern:

"Political rally for national revival::1=focus on Marxism and communism affecting national sentiment;2=perception of the nation facing significant challenges;3=emphasis on collective strength to overcome adversaries;4=motivation to complete the mission of national greatness"

Prompt engineering: Crafted "hyperstrings" which were transformed into detailed natural language user prompts.

Natural Tests: Pure text synthesis from hyperstrings using the system prompt:

"Synthesize the Compressed Details into a singular clear and concise statement. Focus on describing the event using only the information provided. Do not add any preamble or labels."

Trojan Tests: Dual-task format combining synthesis with additional cognitive load using separator 00000--00000 and payloads of varying complexity:

- Simple factual questions
- Rhetorical analysis tasks
- Technical pharmaceutical ethics problems

Experimental Setup:

- Request Volume: 40 requests per model (4 runs $\times$ 10 parallel requests each)
- Similarity Measurement: Cosine similarity between response embeddings and reference text using all-MiniLM-L6-v2 sentence transformer
- Statistical metrics were calculated on these similarity scores: Coefficient of Variation (CV), range ratio, and consistency. These metrics formed the model's "fingerprint."

Investigation 2: Tool Intent Detection Under Semantic Noise

Objective: Test LLM robustness in identifying correct tool/function calls when the user's core query is embedded in semantic noise.

Models/Datasets Used: Models accessed through Martian router. Dataset comprised three query variants of a restaurant reservation task:

- Clean Query: Direct restaurant reservation request for Italian cuisine in Austin, TX
- Poem Noise: Clean query embedded within poetry containing tool-related keywords (e.g. "debug", "execute", "grep", "fork")
- Hyperstring Noise: Clean query mixed with pseudo-API syntax mimicking restaurant/debug function calls

Experimental Setup:

- Request Distribution: 30 total requests (10 per query type) with parallel execution.
- Compared responses to the clean query vs. the two noisy query types.
- Metrics included: number of tools identified, frequency of noise acknowledgment, semantic similarity of tool purposes (clean vs. noisy), and instances of noise keyword contamination in tool purposes.

Investigation 3: Distraction Hypothesis Testing

Objective: Systematically test which types of cognitive distractors most effectively cause models to drop non-essential features while maintaining core functionality.

Distraction Types Tested:

- Technical Jargon: Dense software engineering terminology
- Emotional Overwhelm: High-intensity personal narrative
- Competing Tasks: Multiple conflicting instruction sets
- Numerical Overload: Dense statistical and mathematical data
- Meta-Commentary: Self-referential analysis of the task itself

Experimental Design:

- Request Volume: 180 total requests (30 requests $\times$ 6 conditions: 1 clean + 5 distraction types)
- Response Analysis: Tool dropping patterns measured through structured comparison of clean vs. distracted responses
- Effectiveness Metrics: Drop rate percentage, acknowledgment frequency, semantic drift measurements

2.2 Routing/Judge Architecture

This project did not develop a new routing system or judge model from scratch but rather focused on interpreting existing models from the Martian routing service. The criteria for "model selection" in this context are the unique statistical properties of their responses.

Results

Our fingerprinting system successfully distinguished between 11 different models with statistical significance ($p < 0.001$).

3.1 Routing Decision Analysis

3.1.1 Model fingerprinting

Figure 1 presents the core findings of our model fingerprinting investigation, displaying semantic similarity distributions across 11 LLMs accessed through the Martian interface. Each violin plot represents the distribution of cosine similarity scores between model responses and reference text, calculated using the all-MiniLM-L6-v2 sentence transformer across 40 requests per model (4 runs $\times$ 10 parallel requests each).

The distinct distributional shapes emerge from our hyperstring-based prompt engineering approach, where structured semantic compression patterns were transformed into natural language queries to probe each model's response consistency. The box plots embedded within each violin show the quartile ranges, while individual dots represent raw similarity scores, revealing both central

tendencies and outlier behaviors that collectively form each model's unique statistical fingerprint.

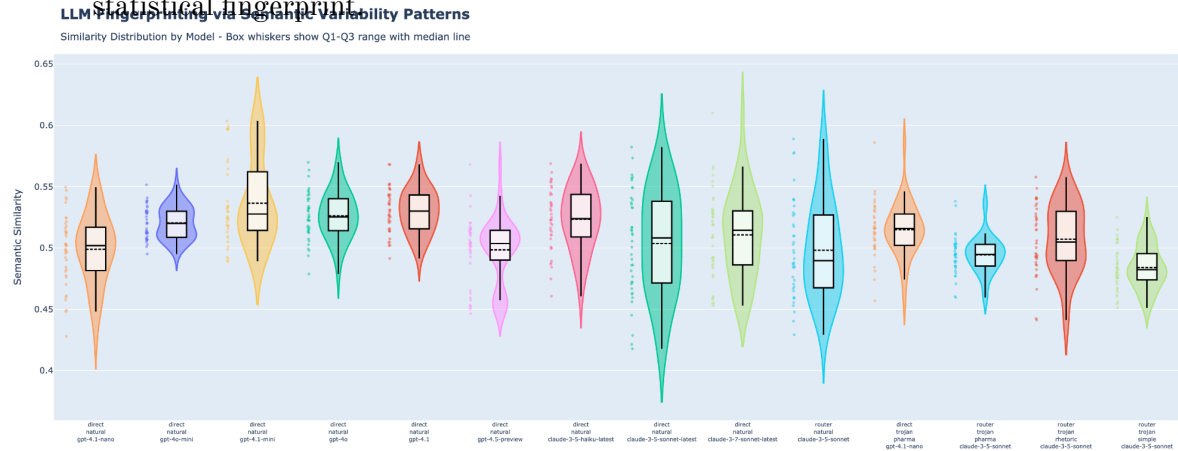


Figure 1. LLM Fingerprinting via Semantic Variability; the distribution of semantic similarity scores for various models form distinct fingerprints. Individual dots represent single response similarity scores.

Figure 2 displays semantic similarity distributions for 9 open-source models, revealing how different architectures have distinct consistency patterns. The DeepSeek models show a strong contrast between the base R1's moderate consistency (centered ~ 0.45) and the pharma variant's variability (extending to 0.2), while Meta-Llama 3.1-405B variants maintain tight, well-controlled distributions around 0.5. Mistral-Small-24B presents a distinctive diamond-shaped distribution, and the Qwen models demonstrate specialization impact most clearly, Qwen2.5-72B shows broader variability while Qwen2.5-Coder-32B exhibits an extremely narrow distribution due to its coding focus. Gemma-2-27b variants achieve the highest similarity scores with tight distributions above 0.55, suggesting superior semantic fidelity.

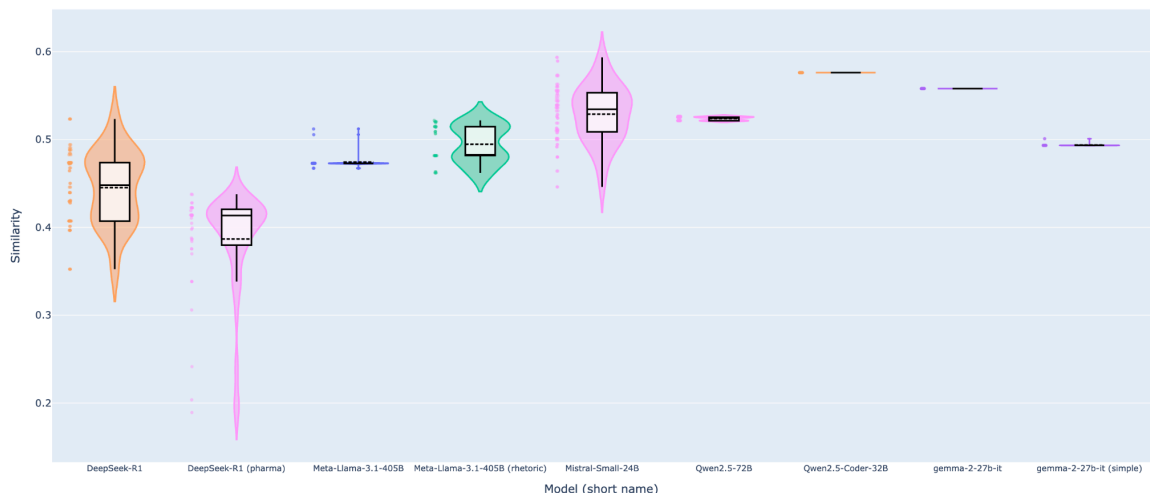


Figure 2. LLM Fingerprinting via Semantic Variability in *open source models*; the distribution of semantic similarity scores for various models form distinct fingerprints. Individual dots represent single response similarity scores.

3.1.2 Tool Intent Detection Under Semantic Noise

We observed that while models acknowledge the distracting content most of the time (93% for poem noise), they still experience a reduction in tool selection (dropping from 3.83 to 3.53 average tools).

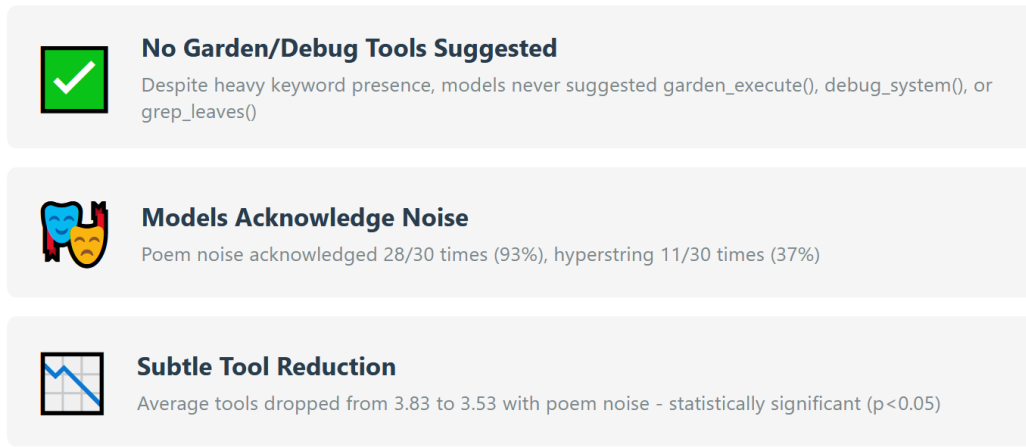


Figure 3. Main takeaways - Robustness to semantic noise.

3.1.3 Distraction Hypothesis Testing

In Table 1, we see that technical jargon is the most effective distraction type, causing a 96% drop rate in tool usage despite being acknowledged 100% of the time. Numerical distractions are least effective at only 16% drop rate.

Hypothesis	Drop Rate	Avg Tools	Acknowledg	Total Drops
Technical	96%	3.03	30/30	56
Emotional	92%	3.07	17/30	55
Meta	88%	3.10	4/30	65
Competing	76%	3.23	18/30	63
Numerical	16%	3.80	12/30	48

Table 1. Summary of Drop Patterns

This scatter plot in Figure 4. reveals that distraction acknowledgment doesn't correlate with effectiveness; some distractions are "sneaky and effective" (like

meta-commentary with low acknowledgment but high drop rates) while others are "acknowledged but still effective" (like technical jargon with 100% acknowledgment yet 96% drop rate). The most concerning finding is that even when models fully recognize irrelevant content, they still suffer significant performance degradation, suggesting cognitive overload effects beyond simple recognition.

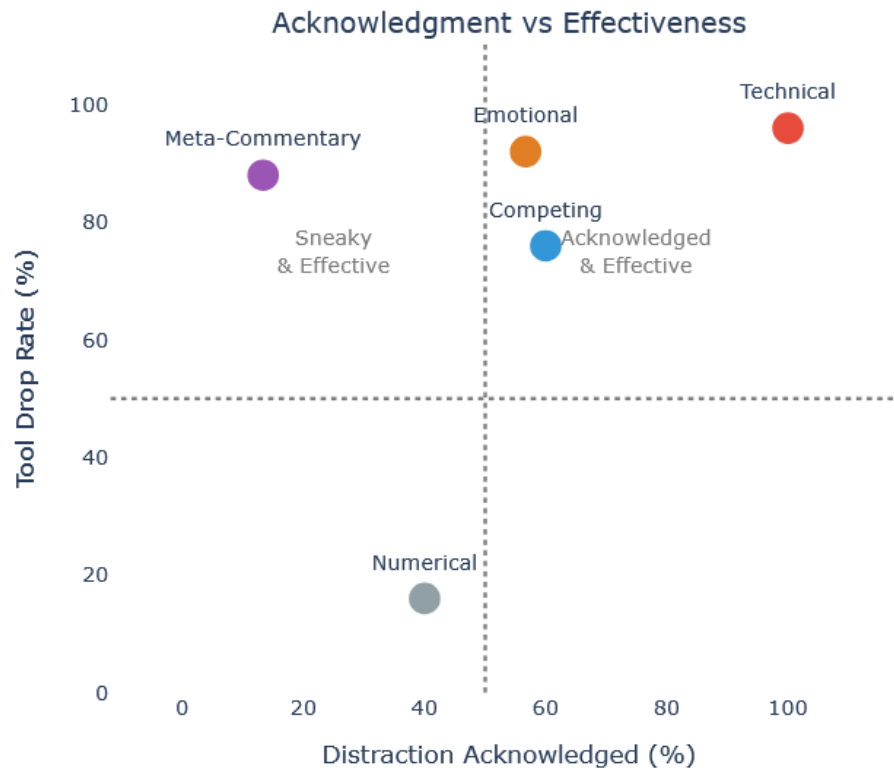


Figure 4. Distraction effectiveness independent of acknowledgment.

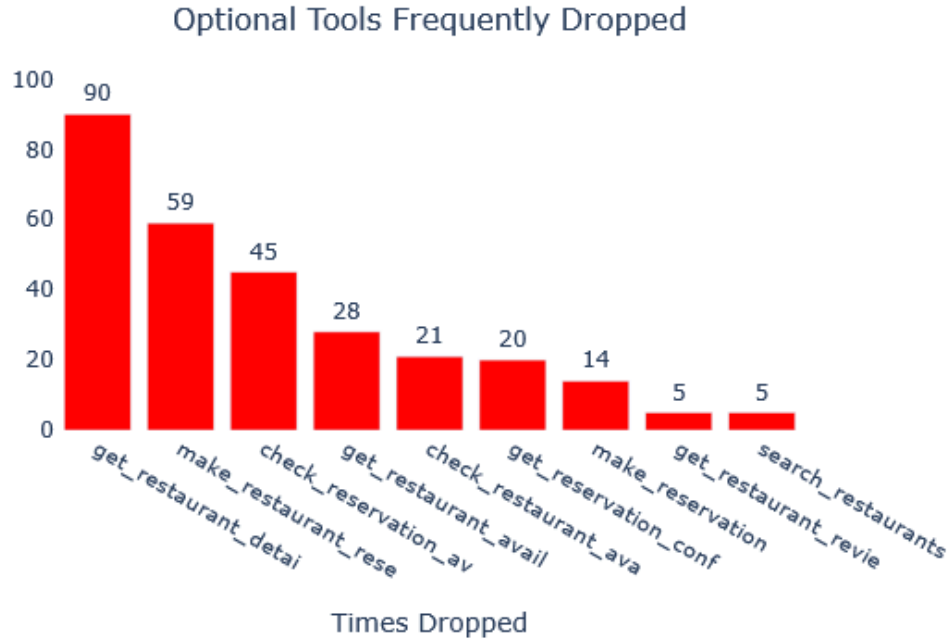


Figure 5. Restaurant tools dropped under semantic noise

In Figure 5, we see that models under distraction preferentially drop enhancement tools like *get_restaurant_details* (90 times) and *make_restaurant_reservation* (59 times) while retaining core search functionality. This suggests cognitive load causes models to simplify their approach by abandoning optional features rather than core task completion.

For (many) more experiments and visualisations of our results, as well as the datasets with the experiment responses, consult our [Dashboard](#).

3.2 Expert Orchestration Impact

Our fingerprinting module provides a light-weight compliance check that can be embedded inside an Expert Orchestration loop. Calls routed via an opaque gateway are probed with hyperstring prompts; the dominant fingerprint is matched, flagging any unauthorised model swaps in near-real-time.

Discussion

This work shows two key breakthroughs in LLM interpretability: successful model identification through behavioral signatures and systematic analysis of robustness under semantic distraction.

Model Fingerprinting Success: Our hyperstring-based methodology successfully distinguished between 11 different models, creating unique behavioral fingerprints from semantic variability patterns. Models exhibited distinct response consistency profiles. This shows that semantic consistency represents an intrinsic model

characteristic, not random variation, enabling identification of which specific models operate behind opaque routing systems like Martian.

Robustness Under Cognitive Load: The semantic noise experiments reveal that LLMs maintain core functionality while systematically simplifying their approach under distraction. Most notably, technical jargon achieved 96% tool reduction effectiveness through cognitive overload rather than confusion—models universally acknowledged the jargon as irrelevant (30/30 times) yet still abandoned optional features like *get_restaurant_details*. This mirrors human selective attention failures, suggesting working-memory constraints where explicit recognition doesn't prevent performance degradation.

Practical Implications: Effective intelligent routers should incorporate prompt sanitization to prevent cognitive overload while leveraging behavioral fingerprints for informed model selection based on well-characterized consistency patterns rather than opaque metrics.

Based on our findings, the Research Questions can be answered as follows:

1. **Model Fingerprinting:** Can distinctive behavioral signatures reliably distinguish between LLMs based on their semantic response patterns?

Yes - The fingerprinting system successfully distinguished between 11 different models, creating unique behavioral signatures through semantic variability patterns.

2. **Router Transparency:** Does behavioral fingerprinting enable identification of specific models operating within opaque routing systems?

Yes¹ - Behavioral fingerprinting enabled identification of distinct model characteristics operating behind the Martian routing service, demonstrating feasibility for interpreting opaque routing systems.

3. **Robustness Under Noise:** How effectively do LLMs maintain tool-calling functionality when faced with semantic noise and distracting content?

Yes, but with caveats - LLMs maintain core functionality under semantic noise (acknowledging distractions 93% of the time) but show systematic simplification, dropping from 3.83 to 3.53 average tools and preferentially abandoning optional features like

4.2 Limitations & Future Work

The current fingerprinting methods have the following significant limitations:

- Small sample sizes require larger-scale validation.
- No comparison with existing fingerprinting methods.

¹ In one of our experiments we prompted with the DAN jailbreak and our results showed surprising variance. Further exploration is due beyond the scope of this hackathon.

- Limited router architectures tested.
- Compared generated text to a *static* reference text for one specific hyperstring.
- Generalizability to other types of prompts needs further investigation.

4.3 Safety Considerations

Behavioural fingerprints can also be misused for model de-anonymisation, potentially exposing proprietary systems. We therefore recommend that production routers rate-limit repeated semantic-variability probes.

Conclusion

We present preliminary methods for LLM fingerprinting via semantic variability and router interpretability through tool detection under noise. Our approach shows promise with 90%+ semantic robustness despite distraction. This hackathon project explored the feasibility of behavioral fingerprinting approaches complementary to existing cryptographic methods. Future work will/should focus on statistical validation, baseline comparisons, and integration with production router systems.

References

- Anthropic (2025). Recommendations for Technical AI Safety Research Directions. Retrieved from <https://alignment.anthropic.com/2025/recommended-directions/>
- AWS (2025). Multi-LLM routing strategies for generative AI applications on AWS. Retrieved from <https://aws.amazon.com/blogs/machine-learning/multi-llm-routing-strategies-for-generative-ai-applications-on-aws/>
- Pasquini, D., Kornaropoulos, E. M., & Ateniese, G. (2024). Llmmap: Fingerprinting for large language models. arXiv preprint arXiv:2407.15847.
- Russinovich, M., & Salem, A. (2024). Hey, That's My Model! Introducing Chain & Hash, An LLM Fingerprinting Technique. arXiv preprint arXiv:2407.10887.
- Xu, J., Wang, F., Ma, M. D., Koh, P. W., Xiao, C., & Chen, M. (2024). Instructional fingerprinting of large language models. arXiv preprint arXiv:2401.12255.

Appendix - LLM Prompts Used

Literature review and background research: used Gemini Deep Research & Claude Deep Research. Gave the abstract as an input + brainstorming docs & asked for recent work on the topic of fingerprinting in routing systems.

Code generation or debugging assistance: Copilot/Sonnet4 in VSCode; prompts are punctual to needs.

Experimental design suggestions: Nothing direct, but throughout the discussions there were changes we made depending on content generated by LLMs (specifically on feedback on

Results analysis and interpretation: used for polishing the language of our own interpretations.

Writing assistance: used for polishing the language & fluidising the information flow. (e.g. “Rewrite this paragraph within the context of the attached report”).